ДВНЗ «УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова
праця на правах рукопису

ПОВХАН ІГОР ФЕДОРОВИЧ

УДК 004.8: 004.89: 519.7

ДИСЕРТАЦІЯ

МЕТОДИ ТА ПРИНЦИПИ ПОБУДОВИ ДЕРЕВ КЛАСИФІКАЦІЇ ДИСКРЕТНИХ ОБ'ЄКТІВ ДЛЯ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

05.13.23 – системи та засоби штучного інтелекту

Подається на здобуття наукового ступеня доктора технічних наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Повхан І.Ф.

Науковий консультант Гече Федір Елемирович, доктор технічних наук, професор

Львів – 2021

АНОТАЦІЯ

Повхан І.Ф. **Методи та принципи побудови дерев класифікації дискретних об‘єктів для інтелектуального аналізу даних.** – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора технічних наук за спеціальністю 05.13.23 – «Системи та засоби штучного інтелекту» (126 «Інформаційні системи та технології») – ДВНЗ «Ужгородський національний університет», Національний університет «Львівська політехніка», 2021 р.

Задачі, об’єднані проблематикою розпізнавання та класифікації відрізняються надзвичайною різноманітністю, а також тим, що в даний час не існує універсального підходу до їх вирішення – запропоновано декілька досить загальних теорій які дозволяють вирішувати багато прикладних, спеціалізованих задач, набір теоретичних результатів отримано для спеціальних випадків та підзадач. Усі ці підходи в теорії розпізнавання мають свої переваги і недоліки та утворюють єдиний базис та інструментарій розв‘язку прикладних задач теорії ШІ. Вузьким місцем реальних систем класифікації залишається необхідність виконання величезного об‘єму обчислень, проте, значна кількість прикладних задач в різних областях природознавства, наприклад: в геології, геофізиці, геохімії, медицині, соціології, археології, біології та інших галузях, де вирішуються задачі класифікації, визначає інтенсивність та актуальність такого напрямку досліджень.

Дисертаційна робота присвячена дослідженню особливостей логічних та алгоритмічних дерев класифікації, деяких питань їх застосування в задачах інтелектуального аналізу даних, розпізнавання образів шляхом створення ефективних моделей класифікації та аналізу великих та надвеликих масивів даних, універсальних систем розпізнавання дискретних об‘єктів. У роботі досліджено та вирішено актуальну науково-прикладну проблему – розвитку теорії аналізу та синтезу дерев рішень, розроблення моделей, методів, прикладного інструментарію інтелектуального аналізу даних на основі

логічних та алгоритмічних дерев класифікації з більшою точністю, зменшеною складністю моделей та підвищеною ефективністю класифікації дискретних об‘єктів.

Перший розділ представленої роботи присвячено всебічному аналізу та дослідженню деревоподібних моделей класифікації та розпізнавання – причому представлення масивів даних (дискретної інформації) великого об‘єму у вигляді структури логічного дерева має свої суттєві переваги щодо економічного опису даних та ефективних механізмів роботи з ними, а галузь застосування концепції дерев класифікації може бути зведена до задач опису структур даних, задач розпізнання та класифікації, задач регресії. Так, при побудові структур дерев класифікації центральними питаннями залишаються питання вибору критерію розгалуження, критерію зупинки навчання та критерію відкидання гілок логічного дерева – тим не менше, не дивлячись на певну проблематику, яка виникає при побудові та використанні концепції дерев класифікації в загальному сенсі, вона залишається потужним інструментом теорії штучного інтелекту. Наведено загальну постановку задачі навчання, виділено основні питання, які виникають на етапі навчання СР. Проаналізовано сучасний стан досліджень концепції дерев рішень, висвітлено проблематику даної концепції, визначено шляхи подальших вдосконалень та перспективних досліджень моделей та методів дерев класифікації. Представлено структурно-логічну схему досліджень.

У другому розділі представлено загальну концепцію дерев класифікації, яка забезпечує покриття масиву даних навчальної інформації за рахунок фіксації об‘єктів вибірки в своїй структурі, причому такий підхід зберігання інформації забезпечує як механізм донавчання (розширення структури), так і виправлення помилок, а застосування ЛДК в задачах РО дозволяє просто та компактно описувати ФР. Також зафіксовано відсутність актуальних методів та алгоритмів, які би дозволили будувати ефективні конструкції логічних дерев на основі масивів даних (НВ) великої та надвеликої розмірності, розглянуто окремий клас дерев класифікації – випадкові ЛДК, які мають як

суттєві переваги (програмна простота побудови дерева, зменшення часу загальної генерації (відбору) вершин ЛДК, можливість оцінки та вибору найвідповіднішого ЛДК із множини побудованих), так і певні недоліки (неоптимальність структури, апаратні витрати на генерацію та зберігання неоптимального ЛДК). Розроблено алгоритмічну схему виправлення помилок ЛДК (алгоритм УПР) шляхом корекції структури дерева класифікації. Дано числову оцінку для фіксованого шляху структури довільного ЛДК максимальної кількості помилок класифікації всіх типів у процедурі розпізнавання. Досліджено питання можливості побудови ЛДК мінімальної складності на основі мінімального тесту та структурної складності максимального дерева класифікації, побудованого за даними НВ.

У третьому розділі розроблено метод Т – опорних множин, який полягає у відборі (фіксації) певного набору ознак разом зі своїми значеннями на основі інформації деякої початкової НВ, з можливістю наступної оцінки цих опорних множин за допомогою відповідних функціоналів, також запропоновано основні способи задання систем Т – опорних множин. На основі концепції Т – опорних множин запропоновано нове формальне визначення алгоритму розпізнавання (класифікації), що дало змогу ввести нові означення інформативності Т – опорних множин щодо довільного об‘єкту та класу початкової НВ, дозволило дослідити основні підмоделі алгоритму розпізнавання (на основі Т – опорних множин). Зафіксовано, що найбільший інтерес викликають саме прості системи Т – опорних множин, алгоритми розпізнавання, що їх визначають, не дають помилок розпізнавання на даних початкової навчальної інформації та відмов класифікації на всій множині допустимих об‘єктів. Уведено дерева моделей для задачі класифікації, які представляють собою зв‘язаний граф без циклів, в некінцевих вершинах якого знаходяться фіксовані моделі, а ребра нумеруються значеннями предикатів (узагальнених ознак) з цих моделей, причому в кінцевих вершинах знаходяться значення функції, що задає розбиття початкової НВ на класи, а на фіксованому шляху ДМ одна й та сама модель може зустрічатися тільки один

раз. Показано, що деяка множина моделей, що представляють алгебраїчні системи розпізнавання (у вигляді структур дерев класифікації) при фіксованих значеннях параметрів перетворюється на конкретну модель класифікації (деревоподібну логічну структуру), причому правилом розпізнавання (класифікації) ДМ являється деякий оператор, що відносить довільний допустимий об‘єкт класифікації (за його початковим інформаційним описом) до фіксованої кінцевої вершини дерева моделей. Запропоновано використання концепції Т – опорних множин для опису дискретних образів, на основі чого представлено нові означення інформативності відносно класу та об‘єкту класифікації, причому важливим моментом є те, що 2Т – опорні множини можна використовувати в якості ознак, що характеризують деякий образ, а відповідно для побудови алгоритмів розпізнавання або класифікації образів доречно використовувати дерева моделей класифікації (алгебраїчних систем), у вершинах яких знаходяться 2Т – опорні множини.

У четвертому розділі дослідження показано, що за підмножиною будь-яких значень набору змінних можна побудувати повне логічне дерево, яке представляє функцію розпізнавання, визначену на всіх наборах, що забезпечує можливість застосовування логічних дерев у розпізнаванні образів, причому одержання тієї чи іншої логічної функції залежить від порядку проходження змінних у логічному дереві, що безпосередньо впливає на кінцеву структуру логічного дерева, а принциповою задачею залишається питання мінімальної (оптимальної) структури ЛДК відносно початкової НВ. Розроблено простий метод мінімізації логічних функцій на основі концепції логічних дерев (процедури перестановки ярусів в його структурі), причому даний підхід характеризується простотою роботи за вказаною схемою на відповідній графічній структурі (конструкції логічного дерева), значною наочністю порівняно з іншими методами (є прямим або опосередкованим узагальненням двійкових (бінарних) методик для багатозначного випадку), можливістю нескладної програмної реалізації, що значно розширює сферу дій у випадку

багатозначної логіки. Дано числову оцінку впливу процедури перестановки ярусів структури регулярного логічного дерева на його складність для бінарного випадку, причому ефект перестановки ярусів є значним. Розроблено важливий механізм мінімізації логічних дерев – процедуру перестановки ярусів (блоків) у структурі дерева, яка забезпечує досягнення помітного ефекту зменшення складності логічного дерева, причому зі зростанням структурної складності логічного дерева ефективність перестановки ярусів швидко зростає.

У п‘ятому розділі досліджено питання побудови найскладнішого логічного дерева (на основі критерію структурної складності), яке містить у своїй конструкції максимальну кількість різних міток (атрибутів, вершин, функцій). Дано загальну числову оцінку (структурної складності) кількості різних міток (функцій, атрибутів) найскладнішого логічного дерева в залежності від випадку розташування ярусу злому в його структурі. Розроблено методи знаходження величини подібності для конструкції логічних дерев у задачах мінімізації їх структур, та на основі цього досліджено питання критерію оптимальності регулярного логічного дерева. Розв‘язок даного питання має принципову важливість з точки зору як пошуку ефективних перестановок в процедурі мінімізації структур регулярних логічних дерев, так і оцінки загального ефекту результату такої оптимізації логічних дерев. Розроблені схеми оптимального розташування змінних у структурі логічного дерева, які в більшості випадків дають оптимальне логічне дерево (або близьке до оптимального відносно його структури). Розроблено на основі структури логічного дерева мінімальної складності схему мінімізації логічної функції, яка базується на побудові мінімального логічного дерева та мінімізації функції із застосуванням відповідних співвідношень для отримання простих дужкових виразів функцій багатозначної логіки, причому можна привести й інші співвідношення, специфічні для багатозначної логіки, які дозволяють проводити подальше спрощення дужкової форми довільної функції, що отримана за допомогою концепції логічного дерева.

У шостому розділі розроблено комплексний метод покрокової апроксимації масиву початкових даних НВ набором відібраних та оцінених незалежних алгоритмів класифікації та розпізнавання, яка дає можливість будувати різнотипні моделі АДК. Досліджено питання оцінки якості (ефективності, інформативності) набору алгоритмів класифікації (вершин структури дерева класифікації) в схемі методів АДК – відбору критерію розгалуження конструкції дерева класифікації. Розроблено методи побудови моделей АДК двох типів, причому отримані дерева класифікації складаються з різних алгоритмів та методів розпізнавання в свою чергу представляють собою нові алгоритми (схеми) класифікації. Розроблено обмежений метод побудови АДК, який спрямований на добудову лише тих шляхів (ярусів) структури дерева класифікації, де є найбільша кількість помилок (усіх типів) класифікації. Такий підхід синтезу моделі розпізнавання дає можливість досить ефективно регулювати складність (точність) моделі дерева класифікації, що будується, причому доцільним є його застосування в ситуаціях з обмеженнями щодо апаратних ресурсів інформаційної системи, обмеженнями точності та структурної складності моделі, обмеженнями на структуру, послідовність та глибину розпізнавання масиву даних НВ. Досліджено питання збіжності процедури побудови моделей дерев класифікації представлених в роботі методів ЛДК/АДК для умов слабкого та сильного розділення класів початкової НВ, наведено відповідні числові оцінки, причому загальна кількість усіх кінцевих вершин логічної структури (листів дерева розпізнавання) побудованої схеми класифікації буде однозначно визначати кінцеву потужність схеми методу дерева класифікації (моделей ЛДК/АДК).

У сьомому розділі розроблено програмну схему обчислення важливості ознак (груп ознак) за допомогою введеного в роботі функціонала, який дає змогу вирішувати широке коло наведених практичних задач. Розглядається питання кодування початкової інформації НВ, один із шляхів вирішення якої полягає в схемі розбиття ознакового простору задачі, на $n-$вимірні

гіперпаралелепіпеди так, щоби елементи (об‘єкти) з різних класів не попадали в один і той самий паралелепіпед (задача геометричного обмеження) з присвоєнням кожному з даних геометричних об‘єктів фіксованого двійкового коду. Розроблено алгоритмічну схему побудови дерев класифікації, яка забезпечує ефективний механізм програмної побудови фіксованого ЛДК за набором деяких початкових даних (НВ), причому загальну структуру довільного ЛДК можна досить просто представити в програмному форматі у вигляді трьох базових елементів (набору трьох масивів). Побудовані моделі дерев класифікації (структури АДК) можна застосовувати для оцінки загального стану басейну річки Уж (на ділянці спостереження) та виявлення ситуації червоної (паводкової) зони на основі поточних замірів постів спостережень. Відмітимо, що проведені практичні випробовування моделей АДК підтвердили працездатність математичного забезпечення та запропонованих методів й алгоритмів побудови дерев класифікації, розробленого програмного забезпечення. Це дає можливість рекомендувати використання даного підходу (концепції моделей АДК) та його програмної реалізації для широкого спектру прикладних задач класифікації та розпізнавання в практичній площині.

Набір результатів з вище приведених розділів застосовано для розв‘язку задач класифікації, розроблено програмний інструментарій для розв‘язку задач розпізнавання образів. Розроблений в дисертаційній роботі підхід дає можливість методично одноманітно вирішувати багато задач розпізнавання та класифікації з використанням накопиченого досвіду українських та зарубіжних вчених.

Ключові слова: аналіз даних, системи розпізнавання, класифікації об‘єктів, дерева рішень, набір атрибутів, критерій розгалуження, алгоритмічне дерево класифікації.

SUMMARY

I.F. Povkhan. **Methods and principles of constructing discrete object classification trees for data mining.** – Qualification scientific work as a manuscript.

Doctor of technical science thesis, specialty: 05.13.23 – "Artificial intelligence systems and means". – State Higher Education Institution Uzhhorod National University, 2021.

Problems being combined by a common task of recognition and classification are distinguished by extreme diversity and the lack of a unified approach to their solution. Several quite general theories have been suggested allowing one to solve a number of applied specialized problems, whereas for particular cases and subtasks a set of theoretical results has been obtained. All the above approaches of recognition theory have their advantages and shortcomings and form a single basis and toolbox for solving applied artificial intelligence theory tasks. The bottleneck of real classification systems is the necessity to perform a huge amount of calculations, however, a large number of applied problems in different areas of natural studies, e.g. in geology, geophysics, geochemistry, medicine, sociology, archeology, biology and other fields, where the classification problems are being solved define the intensity and relevance of such trend of research.

Present thesis is devoted to studying the specific features of both logical and algorithmic classification trees, some issues of their use in the problems of intelligence data analysis and image recognition by creating the efficient models of classification and analysis of large and super-large data arrays and universal systems of discrete object recognition. A hot scientific and applied problem of developing a new theoretical ground of constructing models, algorithms and practical instruments of recognition in a form of classification trees (the LCT/ACT structures) and their use in automating the development of real image recognition systems related to processing the big arrays of different-type information has been studied and solved in this work.

The first chapter of this work is devoted to the comprehensive analysis and studies of the tree-like classification and recognition models, where presentation of the large-volume data arrays (discrete information) in a form of the logical tree structure has its own advantages in economic description of the data and efficient mechanisms of their processing, whereas the field of application of the classification tree concept could be reduced to the problems of the data structure description, recognition and classification and regression problems. So, the issues of choosing the branching criterion, training stopping criterion and logical tree branch rejection criterion remain the central issues of the classification tree structure construction. Nevertheless, besides a certain agenda that appears at constructing and using the classification tree concept, it remains a powerful tool of the artificial intelligence theory. The general setting of the training problem has been described, the principal questions that arise at the recognition system training stage have been distinguished. The up to date state of the decision tree concept studying has been analyzed, the circle of problems related to the above concept, the ways of further improvements and studies of the classification tree models and methods have been determined. The structural and logical scheme of research has been presented.

The general concept of the classification trees that provides covering the training information data array due to fixing the selection objects in their structure is presented in the chapter 2. Such approach of information conservation ensures both the complete training (structure extension) mechanism and that of error correction, while the LCT use in the image recognition tasks allows one to describe simply and compactly the recognition function. The lack of the relevant methods and algorithms that may construct efficient logical tree constructions on the basis of the training selection data arrays of large and super-large dimension has also been stated. A particular classification tree class, i.e. the random LCTs having both essential advantages (tree construction program simplicity, reduction of the time of general generation (selection) of the LCT vertices), possibility of estimating and selecting the most appropriate LCT from a set of constructed ones) and certain shortcomings (structural non-optimality, hardware expenses for non-optimal LCT

generation and conservation) has been considered. The algorithmic scheme of the LCT error correction (the recognition error elimination mechanism) by the classification tree structure correction has been suggested. The numerical estimate of a maximal number of any type classification errors in the recognition procedure was given for a fixed structural path of arbitrary LCT. The issues of the possibility to construct LCT of minimal complexity have been studied on the basis of the minimal test and structural complexity of the maximal classification tree constructed using the training selection data, the matrix (geometrical) interpretation of classification algorithm models (the LCT models) has been suggested allowing the simple program and instrumental estimation of the principal qualitative parameters of a given algorithm (logical tree structure) operation to be realized.

In chapter 3, the T-reference manifold concept based on the selection/fixation of a certain set of attributes with their values on the basis of certain initial training selection information with the possibility of further estimation of the reference manifold data by means of the relevant functionals has been suggested together with the principal methods of the T-reference manifold setting. On the T-reference manifold basis, a new formal definition of the recognition/classification algorithm has been suggested and this allowed us to introduce new definitions of the T-reference manifold informativity with respect to the arbitrary object and the initial training selection class and to study the basic submodels of the recognition algorithm (on the basis of the T-reference manifolds). It has been fixed that just the simple T-reference manifold systems attract the maximal interest, because the recognition algorithms that define them give no recognition errors on the initial training information data and no classification failures on the entire manifold of acceptable objects. The tree models for the classification problems have been introduced being the connected graph with no cycles containing the fixed models at its non-terminal vertices. Its ribs/edges are numerated by the predicate (generalized attribute) values from the above models, and the terminal vertices contain the values of the function that define the initial training selection partition into classes, while on the fixed model tree path, the same model may be met only once.

It has been shown that certain model manifold representing the algebraic recognition systems (in a form of the classification tree structures) at fixed parameter values transforms into the particular classification model (the tree-like logical structure), and the rule of the model tree recognition (classification) is a certain operator that relates the arbitrary acceptable classification object (by its initial informational description) to the fixed terminal model tree vertex. It has been suggested to use the T – reference manifold concept to describe the discrete images and this allowed the new definitions of informativity with respect to the classification object and class to be presented. The important moment here is that the 2T – reference manifolds could be used as the attributes that characterize a certain image, and, respectively, it is reasonable to use the classification model trees (algebraic systems) with the 2T – reference manifolds located at their vertices to construct the image recognition/classification algorithms. The new qualitative functional have been suggested to calculate a certain selective possibility (informativity) of particular discrete attributes with respect to the recognition function by the initial training selection. Here this set of functional of the discrete image quality estimation allows one to construct the minimal (by the arbitrary structure) image attribute description. It has been emphasized that the problem of minimizing the initial image description is closely related to the functional estimate of the importance of discrete attributes (that characterize this image) allowing the classification object to be described in a maximally compact way with high quality.

It has been shown in chapter 4 that, having the submanifold of different values of variable set, one may construct a complete logical tree that represents the recognition function defined at any set. This allows one to use the logical trees in the image recognition, and obtaining of particular logical function depends on the order of variable passing in the logical tree that affects directly the final structure of the logical tree. The problem of the minimal (optimal) LCT structure with respect to the initial training selection remains here the principal task. A simple method of logical function minimization on the basis of the logical tree concept (the procedure of layer rearrangement in the tree structure) has been suggested. This approach is

characterized by the simplicity of operation according to the above scheme at the relevant graphical structure (the logical tree construction), considerable clearness as compared to other methods (it is the direct or mediated generalization of the binary methods for the multiple-valued case), possibility of a simple program realization that extends considerably the field of actions in the case of the multiple-valued logic. A numerical estimate of the influence of the procedure of the regular logical tree layer rearrangement on its complexity has been given for the binary case, and the effect of the layer rearrangement appeared to be substantial. An important mechanism of the logical tree minimization has been suggested, i.e. the procedure of the layer (block) rearrangement in the tree structure that allows the visible effect of the logical tree complexity reduction to be achieved. Moreover, the higher is the structural complexity of the logical tree, the faster the layer rearrangement efficiency increases. The problem of the fixed-type logical tree automorphism has also been studied.

The question of constructing the most complex logical tree (on the basis of the structural complexity criterion) is studied in chapter 5. This tree contains in its construction the maximal number of different marks (attributes, vertices, functions). The general numerical estimate of the structural complexity/number of various marks (functions, attributes) of the most complex logical tree dependent of the break layer in its structure has been given. The methods of finding the value of the logical tree similarity in the tasks of their structure minimization have been developed. On this basis, the question of optimality criterion for the regular logical tree has been investigated. Solving this problem is of principal importance from the viewpoint of both searching the efficient rearrangements in the regular logical tree structure minimization procedure and estimating the total effect of the above logical tree optimization result. The algorithmic schemes of the optimal location of variables in the logical tree structure have been suggested that, in most cases, provide the optimal logical tree (or that close to the optimal one with respect to its structure). The logical function minimization scheme has been proposed on the basis of the minimal complexity logical tree structure that constructs the minimal logical tree and

minimizes the function by using the relevant relations to obtain simple bracketed expressions for the multiple-valued logical functions. Furthermore, one may use another relations specific for the multiple-valued logic that allow one to perform further simplification of the arbitrary function bracketed form obtained using the logical tree concept. The numerical estimates and the general scheme of the regular logical tree structure complexity estimation have been presented.

A concept of a step-by-step approximation of the initial training selection data by a set of selected and estimated independent classification/recognition algorithms is suggested in chapter 6 that allows the different-type ACT models to be constructed. The problem of the classification algorithms (classification tree structure vertices) quality (efficiency, informativity) estimation, i.e. the question of choosing the classification tree construction branching criterion, has been studied in the ACT method scheme. The methods of constructing the two types of the ACT models have been developed, and the classification trees obtained consist of various recognition methods and algorithms that, in turn, are the new classification algorithms/schemes. The constrained method of the ACT construction has been developed being directed to the complete construction of only those paths (layers) of the classification tree structure that have the maximal number of any-type classification errors. Such method of recognition model synthesis allows the complexity (accuracy) of the classification tree model to be regulated, whereas it seems expedient to use this method for the situations with the restrictions placed onto the information system hardware resources, model accuracy, structural complexity, training selection data array recognition sequence and depth. The problem of general complexity of the classification tree model construction has been investigated for the LCT/ACT methods presented in this work. The adequate numerical estimates have been given, and the total number of all terminal logical structure vertices (recognition tree leaves) of the constructed classification scheme will unambiguously define the ultimate capacity of the classification tree method scheme (i.e. the LCT/ACT models).

In chapter 7, the algorithm of unambiguous coverage of the discrete image by a set of rectangles is suggested, and the algorithmic scheme of the relevant problem solution has been developed. The algorithmic scheme of calculating the attribute (attribute group) importance has been elaborated using the introduced functional that allows a wide circle of applied tasks to be solved. The issue of coding the initial training selection information has been considered. One of the methods of solving this problem is to partition the attribute space into the n-dimensional hyperparallelepipeds in such a way that the elements (objects) from different classes do not enter the same parallelepiped (the geometric restriction problem), and a fixed binary code is assigned to each of the given geometric objects. The algorithmic scheme of constructing the classification trees has been developed enabling the effective mechanism of the program construction of a fixed LCT according to the set of some initial training selection data to be provided. Note that the general structure of arbitrary LCT can be easily presented in the program format in a form of three basic elements (a set of three arrays). The constructed models of classification tree (the ACT structure) could be used to estimate the general condition of the Uzh river basin (at the observation area) and detect the situation of the 'red' (flood) zone on the basis of the current measurements at the observation points. It should be noted that the practical trials of the ACT models have confirmed the performance of mathware, suggested methods and algorithms of classification tree construction and developed software allowing one to give recommendations on the use of this approach (the ACT model concept) and its program realization for a wide spectrum of applied classification/recognition.

A set of results from the above chapters has been utilized to solve classification problems, the matware and software have been developed to solve the problems of image recognition. The approach elaborated in this research allows many recognition/classification problems to be solved routinely and uniformly with the use of experience gained by Ukrainian and foreign scientists.

Keywords: data analysis, object classification/recognition systems, decision trees, attribute set, branching criterion, algorithmic classification tree.

Список публікацій здобувача в яких опубліковані основні наукові результати дисертації:

1. Повхан І. Ф. Логічні дерева класифікації в задачах штучного інтелекту : монографія. Ужгород : Поліграфцентр - Ліра, 2019. 276 с.

2. Повхан І. Ф. Питання структурної складності для випадку регулярного логічного дерева. *Scientific and technical progress in European countries and the contribution of higher education institutions* : collective monograph. Riga : Izdevnieciba «Baltija Publisher», 2020. 308 p.

3. Povhan I. F. Logical recognition tree construction on the basis a step-to-step elementary attribute selection. *Radio Electronics, Computer Science, Control*. 2020. № 2. P. 95–106.

4. Povkhan I. F. The general concept of the methods of algorithmic classification trees. *Radio Electronics, Computer Science, Control*. 2020. № 3. P. 108–121.

5. Povhan I. F. Limited method for the case of algorithmic classification tree. *Radio Electronics, Computer Science, Control*. 2020. № 4. P. 106–118.

6. Povhan I. Designing of recognition system of discrete objects. *Proceedings of the «2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP)»*, August 23–27, 2016, Lviv, Ukraine. Lviv, 2016. P. 226–231.

7. Povkhan I., Lupei M. The algorithmic classification trees. *Proceedings of the «2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP)»*, August 21–25 2020, Lviv, Ukraine. Lviv, 2020. P. 37–44.

8. Lupei M., Mitsa A., Povkhan I., Sharkan V. Determining the eligibility of candidates for a vacancy using artificial neural networks. *Proceedings of the «2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP)»*, August 21–25 2020, Lviv, Ukraine. Lviv, 2020. P. 18–23.

9. Povkhan I., Lupei M., Kliap M., Laver V. The issue of efficient generation of generalized features in algorithmic classification tree methods. *Data Stream Mining and Processing (DSMP 2020)*: Proceedings of the Third International Conference, Lviv, Ukraine, August 21–25, 2020. Cham: Springer, 2020. P. 98–113.

10. Povkhan I. Classification models of flood-related events based on algorithmic trees. *Eastern-European Journal of Enterprise Technologies.* 2020. Vol. 6, № 4 (108). P. 58–68. DOI: https://doi.org/10.15587/1729-4061.2020.219525

11. Povhan I. General scheme for constructing the most complex logical tree of classification in pattern recognition discrete objects. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2019. Vol. 11. С. 73–80.

12. Povhan I. Generation of elementary signs in the general scheme of the recognition system based on the logical tree. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2019. Vol. 12. С. 20–29.

13. Povhan I. Question of the optimality criterion of a regular logical tree based on the concept of similarity. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2020. Vol. 13. С. 19–27. DOI: https://doi.org/10.30970/eli.13.2

14. Повхан І. Ф. Особливості синтезу узагальнених ознак при побудові систем розпізнавання за методом логічного дерева. *Інформаційні технології та комп'ютерне моделювання ІТКМ-2019* : матеріали міжнар. наук.-практ. конф., Івано-Франківськ – Яремче, 20–25 трав. 2019. Івано-Франківськ, 2019. С. 169–174.

15. Povhan I. Generation of classification schemes in the form of logical trees in discrete object recognition problems. *Modern Problems of Mathematical Modeling, Automated Control and Information Technologies MCIT-2019* : Proceedings of the International Scientific and Practical Conference, Rivne, November 14–16 2019. Рівне, 2019. P. 195–200.

16. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф., Поліщук В. В. Групова та індивідуальна оцінка важливості бульових аргументів. *Вісник Національного технічного університету «Харківський політехнічний інститут»*: зб. наук. пр. 2011. № 53. С. 57–64.

17. Повхан І. Ф. Задача загальної оцінки складності максимального побудованого логічного дерева класифікації. *Вісник Національного технічного університету «Харківський політехнічний інститут»* : зб. наук. пр. Серія: Інформатика та моделювання. 2019. № 13 (1338). С. 104–117.

18. Повхан І. Ф. Питання побудови деревоподібних моделей розпізнавання образів. *Вісник Національного технічного університету «Харківський політехнічний інститут»* : зб. наук. пр. Серія: Інформатика та моделювання. 2019. № 28 (1353). С. 39–57.

19. Povhan I. Logical classification trees in recognition problems. *Kwartalnik Naukowo-Techniczny: Informatyka Automatyka Pomiary w gospodarce o ochronie srodowiska*. Krakow, 2020. № 2. P. 12–16. DOI: http://doi.org/10.35784/iapgos.927

20. Повхан І. Ф. Питання гнучкості логічних дерев класифікації в задачах розпізнавання образів. *Технічні науки та технології*. 2019. № 3 (17). С. 131–140.

21. Повхан І. Ф. Метод побудови алгоритмічного дерева другого типу на основі апроксимації навчальної вибірки набором алгоритмів класифікації. *Технічні науки та технології* : наук. журн. Чернігів, 2020. № 2 (20). С. 126–139.

22. Повхан І. Ф. Питання синтезу дискретних зображень в задачах розпізнавання образів. *Вісник Вінницького політехнічного інституту* : наук. журн. Вінниця, 2020. № 4 (151). С. 50 – 57.

23. Повхан І. Ф. Моделі алгоритмів розпізнавання у вигляді логічних дерев класифікації. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2019. № 1 (475). С. 156–163.

24. Повхан І. Ф. Питання оцінки ефекту перестановки ярусів логічного дерева максимальної складності для бінарного випадку. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2020. № 2 (480). С. 99–107.

25. Повхан І. Ф. Питання оптимізації логічного дерева шляхом перестановки структурних елементів. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2020. № 3 (481). С. 91–101.

26. Повхан І. Ф. Проблема функціональної оцінки навчальної вибірки в задачах розпізнавання дискретних об‘єктів. *Вчені записки Таврійського*

національного університету імені Вернадського. Серія: Технічні науки. 2018. Т. 29 (68), № 6. С. 217–222.

27. Повхан І. Ф. Поняття функції та алгебраїчної схеми розпізнавання в задачах класифікації дискретних об‘єктів. *Вчені записки Таврійського національного університету імені Вернадського*. Серія: Технічні науки. 2019. Т. 30 (69), №2. С. 171–177.

28. Повхан І. Ф. Задача апроксимації вибірки дискретних наборів геометричними об‘єктами. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), № 3. С. 136–142.

29. Повхан І. Ф., Лавер В. О. Алгоритми побудови логічних дерев класифікації в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), № 4. С. 100–106.

30. Повхан І. Ф. Особливості випадкових логічних дерев класифікації в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Серія: Технічні науки. 2019. Т. 30 (69), № 5. С. 138–143.

31. Повхан І. Ф. Питання представлення дискретних зображень в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), № 6. С. 154–159.

32. Повхан І. Ф. Питання однозначного покриття зображень прямокутниками в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2020. Т. 31 (70), № 1. С. 119-124.

33. Повхан І. Ф. Моделі дерев класифікації паводкових явищ річки Уж Закарпатського регіону. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2020. Т. 31 (70), № 5. С. 100-107.

34. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф., Повхан Л. С. Вычисление важности дискретных признаков (анализ некоторых подходов). *Восточно-европейский журнал передовых технологий*. 2010. № 5/4 (47). С. 71–75.

35. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Проблема оцінки складності логічних дерев розпізнавання та загальний метод їх оптимізації. *Восточно-европейский журнал передовых технологий*. 2011. № 6/4 (54). С. 24–28.

36. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Загальна оцінка мінімізації деревоподібних логічних структур. *Восточно-европейский журнал передовых технологий*. 2012. № 1/4 (55). С. 29–32.

37. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Дослідження стійкості максимального логічного дерева відносно перестановки ярусів. *Восточно-европейский журнал передовых технологий*. 2012. № 2/4 (56). С. 15–22.

38. Василенко Ю. А., Мошкола В. П., Повхан І. Ф. Модульний синтез схем класифікації та розпізнавання. *Сучасні тенденції інформаційних технологій* : матеріали IV міжнар. наук. конф., Прага, 15–31 берез. 2008 р. Прага, 2008. С. 32–34.

39. Василенко Ю. А., Повхан І. Ф. Автоматизація побудови схем класифікації. *Теорія прийняття рішень :* праці IV Міжнародної школи – семінару, Ужгород, 29 вересня – 4 жовтня 2008 р. Ужгород, 2008. С. 137-138.

40. Василенко Ю. А., Василенко Е. Ю., Бунда В. В., Бунда С. О., Лавер О. Г., Повхан І. Ф. Универсальный подход к анализу и обработке специальной информации. *Інновації в навчальному процесі вищих навчальних закладів: міжнародний та національний досвід* : зб. наук. ст. за матеріалами XV міжнар. наук.-практ. конф., Сніна (Словацька Республіка), 6–9 листоп. 2007 р. Ужгород : Ліра, 2008. С. 99–106.

41. Василенко Ю. А., Василенко Е. Ю., Бунда В. В., Повхан І. Ф., Чедреки Я. Н. Математическое конструирование распознающих систем для дискретных объектов. *Лісабонська стратегія як визначальний чинник європейської інтеграції в галузі освіти і науки* : матеріали XVI Міжнар. наук.-практ. конф. Ужгород – Гирляни, 6–9 трав. 2008 р. Ужгород : Ліра, 2008. С. 79–83.

42. Василенко Ю. А., Завілопуло А. М., Повхан І. Ф., Повхан Л. С. Синтез систем розпізнавання на основі схем-агентів. *Інститут електронної фізики*

Національної академії наук України – 2011: матеріали міжнар. конф. молодих учених, Ужгород, 24–27 трав. 2011 р. Ужгород : Мистецька Лінія, 2011. С. 187–188.

43. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. The importance of discrete signs. *Перспективні шляхи й напрями вдосконалення освітньої системи у світлі Болонського процесу* : зб. наук. доп. за матеріалами XX Міжнар. наук.-практ. конф., Ужгород (Україна) – Кошице (Словаччина) – Мішкольц (Угорщина), 16–19 листоп. 2010 р. Ужгород : ЗакДУ, 2011. Вип. 2 (21), Ч. 1. С. 46–56.

44. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Автоматизация построения систем классификации на основе схем-агентов. *Математическое моделирование, оптимизация и информационные технологии* : материалы 3-й междунар. конф., Кишинев, 19–23 марта 2012 г. Кишинев : Академия транспорта, информатики и коммуникаций, 2012. С. 444–446.

45. Повхан І. Ф. Побудова систем прогнозування економічних явищ на основі концепції логічного дерева класифікації. *Математичні методи, моделі та інформаційні технології в економіці :* матеріали VI міжнародної науково-методичної конференції, Чернівці, 18-19 квітня 2019 р. Чернівці : Друк –Арт, 2019. С. 134-136.

46. Повхан І. Ф. Загальна схема алгоритму усунення помилок розпізнавання в логічних деревах класифікації. *Technical Sciences: History, The Present Time, The Future, Eu Experience (informatics and cybernetics, electronics, radio engineering and communications, automation and computer technology, mechanic engineering, transport*: Proceedings of the international scientific and practical conference, Wloclawek (Republic of Poland), September 27–28 2019. Wloclawek, 2019. P. 42–45.

47. Повхан І. Ф. Проблематика методів логічних дерев класифікації в задачах розпізнавання. *Science, engineering and technology: global and current trends*: Proceedings of the international scientific and practical conference, Prague (Czech Republic), December 27–28 2019. Prague, 2019. P. 28–32.

48. Повхан І. Ф. Особливості реалізацій моделей дерев класифікації на основі селекції ознак. *Science, engineering and technology: global trends, problems and solutions*: Proceedings of the international scientific and practical conference, Prague (Czech Republic), September 25–26 2020. Prague, 2020. Part 1. P. 74-79.

49. Повхан І. Ф. Модульна концепція побудови дерев класифікації. *Цифрова економіка та інформаційні технології* : матеріали міжнар. наук.-практ. конф., Київ, 15–16 квіт. 2020 р. Київ, 2020. С. 87–90.

50. Повхан І. Ф. Методи логічних дерев класифікації в задачах штучного інтелекту. *Priority directions of science development*: Abstracts of II International Scientific and Practical Conference, Lviv (Ukraine), 25–26 November 2019. Lviv, 2019. P. 213–218.

51. Повхан І. Ф. Питання представлення неповністю визначених багатозначних логічних функцій у вигляді логічних дерев в задачах класифікації. *Topical issues of the development of modern science*: Abstracts of IV International Scientific and Practical Conference, Sofia (Bulgaria), 11–13 December 2019. Sofia, Bulgaria, 2019. P. 390–396.

52. Повхан І. Ф. Модифікований метод побудови дерев класифікації. *Комплексне забезпечення якості технологічних процесів та систем* : матеріали X міжнар. наук.-практ. конф., Чернігів, 29–30 квіт. 2020 р. Чернігів, 2020. Т. 2. С. 167–169.

53. Пастор Н. Е., Повхан І. Ф. Генерація схем розпізнавання дискретних об‘єктів на основі апроксимації навчаючої вибірки. *Радіоелектроніка та молодь в XXI столітті* : матеріали XXIII міжнар. молодіжного форуму, Харків, 16–18 квіт. 2019 р. Харків, 2019. Т. 5. С. 140–142.

54. Повхан І. Ф. Загальна концепція алгоритмічного дерева класифікації в задачах розпізнавання образів. *Інформаційні технології у житті молодих науковців Закарпаття* : матеріали V наук.-практ. конф., Ужгород, 7–8 листоп. 2019 р. Ужгород : УжНУ, 2019. С. 58–62.

55. Повхан І. Ф. Питання загальної складності процедури побудови логічного дерева. *Проблеми інформатики та моделювання* : матеріали XX наук.-техн. конф., Харків-Одеса, 16–21 верес. 2020 р. Харків : НТУ «ХПІ», 2020. С. 68–74.

56. Povkhan I. A constrained method of constructing the logic classification trees on the basis of elementary attribute selection. *CEUR Workshop Proceedings*: Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020), Zaporizhzhia, Ukraine, April 15–19, 2020. Zaporizhzhia, 2020. Vol. 2608. P. 843–857. URL: CEUR-WS.org.

57. Повхан І. Метод дерева алгоритмів для задачі класифікації геологічних даних. *Перспективні напрямки сучасної електроніки, інформаційних і комп'ютерних систем*: Тези доповідей на V Всеукраїнській науково-практичній конференції MEICS – 2020, Дніпро, 25-27 листопада 2020 р. Дніпро, 2020. С. 20–22.

58. Повхан І. Моделі класифікації паводкових явищ в Закарпатському регіоні. *Комп'ютерні технології, інновації, проблеми, рішення*: Тези доповідей III Всеукраїнської науково-технічної конференції, Житомир, 26-27 листопада 2020 р. Житомир, 2020. С. 75-79.

ЗМІСТ

СПИСОК ПРИЙНЯТИХ СКОРОЧЕНЬ

ДМ – дерево моделей

НВ – навчальна вибірка

ТВ – тестова вибірка

ПС – програмна система

РО – розпізнавання образів

СР – система розпізнавання

ФР – функція розпізнавання

УО – узагальнена ознака

РВО – метод розгалужений вибір ознак

УПР – алгоритм усунення помилок розпізнавання

ДМК – дерево моделей класифікації

ДВП – алгоритм донавчання та виправлення помилок

ДУО – дерево узагальнених ознак

ДНФ – диз‘юнктивна нормальна форма

КНФ – кон’юнктивна нормальна форма

ЛДК – логічне дерево класифікації

ВДК – випадкове дерево класифікації

АДК – алгоритмічне дерево класифікації

ООБ – об‘єктно-орієнтована бібліотека

ПЛДК – повне логічне дерево класифікації

ДДНФ – досконала диз‘юнктивна нормальна форма

ВСТУП

Актуальність проблеми. Задачі, які об’єднуються тематикою розпізнавання образів, дуже різноманітні та виникають у сучасному світі в усіх сферах економіки та соціального контенту діяльності людини, що приводить до необхідності побудови та дослідження математичних моделей відповідних систем. Станом на тепер не існує універсального підходу до їх розв‘язання, запропоновано декілька досить загальних теорій та підходів, що дають змогу вирішувати багато типів (класів) задач, але їх прикладні застосування відрізняються досить великою чутливістю до специфіки самої задачі або предметної області застосування. Багато теоретичних результатів отримано для спеціальних випадків та підзадач, причому слід відмітити, що вузьким місцем вдалих реальних систем розпізнавання залишається необхідність виконання величезного об‘єму обчислень та орієнтація на потужний апаратний інструментарій. Проте, велика кількість прикладних задач у різних галузях природознавства, наприклад у геології, геофізиці, геохімії, медицині, соціології, археології, біології та інше, де вирішуються задачі класифікації з використанням програмних та апаратних систем, визначають інтенсивність та актуальність такого напрямку досліджень [1-10].

Нейромережева концепція розпізнавання, не зважаючи на значні переваги, має, тим не менше, істотні недоліки, які обмежують галузь її застосування. Нейронні мережі дозволяють знаходити відповідні субоптимальні розв‘язки, що є проблемою в задачах із вимогою на високу точність моделі, що будується. Загальна схема функціонування зводиться до принципу чорної (сірої в окремих випадках) скрині, що недопустимо при умові аналізу причини прийняття того чи іншого рішення (в умовах правила класифікації). Значні апаратні та часові витрати інформаційної системи на процес навчання моделі не компенсуються швидкістю фінальної класифікації. Обмеження на формат вхідних даних, яке зводиться до числової шкали для методів нейромереж, накладає додаткове принципове обмеження щодо спектру можливих прикладних задач. Тому слід визнати, що клас реальних задач, які підпадають під ці обмеження достатньо широкий [11-40].

Зокрема, концепція дерев класифікації (дерев рішень) позбавлена значної частини наведених вище недоліків та дає можливість ефективно працювати в задачах із даними довільних шкал (де інформація задається в природній формі). На сьогоднішній день актуальні різні підходи до побудови СР у вигляді дерев класифікації (ЛДК), причому інтерес до методів розпізнавання, які використовують ЛДК, викликаний рядом корисних властивостей, якими вони володіють. З одного боку, складність класу ФР у вигляді моделей ЛДК, при визначених умовах, не перевищує складності класу лінійних функцій розпізнавання (простішого з відомих). З іншого боку, ФР у вигляді дерев класифікації дозволяють виділити в процесі класифікації як причинно-наслідкові зв’язки (та однозначно врахувати їх у подальшому), так і фактори випадковості або невизначеності, тобто врахувати одночасно і функціональні, і стохастичні відношення між властивостями та поведінкою всієї системи, причому відомо, що процес класифікації нових, таких що до сих пір не зустрічалися об’єктів світу багатьох тварин і людей (за виключенням об’єктів, інформація про які передається генетичним шляхом (наслідковим), а також у деяких інших випадках), відбувається за так званим логічним деревом рішень (в зв‘язку з нейромережевою концепцією) [4, 41-50].

Зафіксуємо, що в більшості задач прогнозування та класифікації, які використовують неструктуровані дані (наприклад набори дискретних зображень або текстові масиви), штучна нейронна мережа (підібраного типу) перевершує за продуктивністю всі інші типи алгоритмів або фреймворків дерев рішень. У протилежному разі (у випадку структурованих масивів дискретних даних великого об‘єму) значною мірою перевагу мають методи та алгоритми концепції дерев рішень.

У даний час існує декілька незалежних загальних підходів (концепцій) для вирішення задачі класифікації в загальній постановці, причому розробку різних концепцій, підходів, методів, моделей, які охоплюють загальну проблематику теорії штучного інтелекту та інформаційних систем проводили і проводять відомі як вітчизняні так і закордонні вчені, науковці, галузеві

спеціалісти, серед яких можна виділити наступних: М. А. Айзерман, Ю. І. Журавльов, А. Г. Аркадьев, А. Ш. Блох, В. Н. Вапник, А. Н. Червоненкіс, В. І. Васильєв, Б. А. Головкин, І. Б. Гурєвич, Б. А. Головкін, Ю. А. Зуєв, А. І. Кондратьєв, Г. С. Лбов, В. А. Орлов, А. Б. Глаз, А. А. Алєксанян, В. А. Євстєгнєєв, В. Є. Котов, Н. Г. Загоруйко, Л. А. Растрігін, Р. Х. Еренштейн, А. А. Барсегян, Н.А. Лагуновский, С. В. Абламейко, Д.М. Абрамов, Є. В. Бодянський, В. П. Машталір, Ю. А. Василенко, С. А. Суботин, Д. Квінлан, Г. Денг, Б. Камінскі, К. Карімі, Т. Хасті, Д. Стоун, Р. Олсен, Р. Ріверс, Л. Хайфіл, П. Гертс, Т. Хортон, Л. Брейман, М. Міакава, Д. Фрідман, К. Верхаген, Р. Дейн, Ф. Грун, П. Вебек, А. Ахо, Н. Вірт, В. Ліпскі, А. Фор, П. Вогоф, Г. Віттен, Є. Франк, Д. Макленнен, Л. Шапіро, Д. Стокман, Д. Форсайт, Р. Дуда, П. Харт, Д. Кнут, У. Претт, М. Хетч, Ш. Чен, Д. Аміт, Д. Геман, К. Вілдер, П. Лавракс, А. Фішер, Р. Тамасия, Р. Хантер та інші.

Усі ці підходи в теорії розпізнавання мають свої переваги і недоліки та утворюють єдиний інструментарій розв‘язку прикладних задач теорії ШІ. Зокрема цілісно пропрацьованим з математичної точки зору є класичний алгебраїчний підхід, розроблений Ю. І. Журавльовим [51-56]. Цей напрямок розвитку теорії розпізнавання зв’язаний з побудовою моделей алгоритмів класифікації та вибором у рамках моделі оптимального за якістю алгоритму розпізнавання, причому значна увага в межах даного дослідження буде приділена алгоритмам обчислення оцінок, на основі чого вводиться поняття Т – опорної множини, демонструється зв‘язок Т – опорних множини та дерев класифікації. Центральну увагу в даному дослідженні буде приділено актуальній концепції дерев рішень. Зокрема з робіт [57-60] відомо, що схема класифікації, яка задається довільним підходом, методом, алгоритмом дерева класифікації має деревоподібну логічну структуру, причому структура логічного дерева складається з вершин (ознак), які групуються за ярусами та побудовані (відібрані) на певному кроці (етапі) побудови моделі дерева класифікації [61-64], а головна особливість деревоподібних СР полягає в тому, що важливість окремих ознак (групи ознак чи їх наборів) визначається

відносно функції, яка задає розбиття об‘єктів на класи [65]. Важливим напрямком досліджень структур ЛДК залишаються питання стосовно генерації дерев рішень для випадку малоінформативних ознак [66] та актуальне питання теорії дерев класифікації – питання можливої побудови всіх варіантів логічних дерев, які відповідають початковій НВ та відбору мінімального за глибиною, структурною складністю (кількістю ярусів) дерева класифікації [67-75].

Таким чином, виникає принципова необхідність у розробці ефективних методів та моделей (на основі концепції дерев класифікації) інтелектуального аналізу великих (надвеликих) масивів даних, заданих у природній формі. Отже, актуальною науково-прикладною проблемою є розвиток теорії аналізу та синтезу дерев рішень, розроблення моделей, методів, прикладного інструментарію інтелектуального аналізу даних на основі логічних та алгоритмічних дерев класифікації з більшою точністю, зменшеною складністю моделей та підвищеною ефективністю класифікації дискретних об‘єктів. [75-81].

Зв‘язок роботи з науковими програмами, планами, темами. Дисертаційне дослідження виконано у Державному вищому начальному закладі «Ужгородський національний університет» на кафедрах факультету ІТ відповідно до плану наукових досліджень у рамках таких науково-дослідних програм (тем, проєктів):

1) кафедри інформатики та фізико-математичних дисциплін ДВНЗ «Ужгородський національний університет»: «Обробка великих масивів інформації за допомогою логіко-математичних методів» (номер державної реєстрації 0119U100733);

2) кафедри програмного забезпечення систем ДВНЗ «Ужгородський національний університет»: «Методи та засоби програмної інженерії реалізації процесів аналітики великих масивів даних на базі інформаційно-технічних платформ електронної науки» (номер державної реєстрації 0119U100703);

3) Науково-дослідної роботи «Моделювання та передбачення надзвичайних ситуацій в Карпатському регіоні та країнах Центрально-Східної Європи», номер державної реєстрації роботи – 0106V00285, категорія роботи – фундаментальні дослідження (КПКВ 2201020), 01 Фундаментальні дослідження з найважливіших проблем природничих, суспільних і гуманітарних наук;

4) «Інноваційні методи навчання на підтримку партнерських відносин – InovEduc (2015 – 2017)» – грантового проекту № CBC01008 Норвезького державного фонду із солідарним бюджетом Словацької Республіки в рамках програми SK08 транскордонне співробітництво.

Мета і задачі дослідження. Метою дисертаційної роботи є розроблення нової теоретичної основи побудови моделей розпізнавання у вигляді алгоритмічних дерев класифікацій (структур АДК) та їх застосування для автоматизації розробки систем розпізнавання образів.

Досягнення поставленої мети передбачало розв‘язання таких задач:

- аналіз властивостей різних методів, алгоритмів розпізнавання (випадкових логічних дерев у тому числі), заснованих на концепції логічних дерев класифікації;
- аналіз та класифікація моделей розпізнавання образів у вигляді дерев рішень – різнотипних структур логічних та алгоритмічних дерев класифікації;
- виявлення та встановлення зв‘язків опорних множин з структурами дерев класифікації в задачі представлення дискретних об‘єктів;
- розроблення методів знаходження подібності конструкції логічних дерев класифікації в задачах мінімізації їх структур та на основі цього дослідження питання критерію оптимальності регулярного логічного дерева;
- розроблення схем оптимального розташування змінних (атрибутів) у структурі логічного дерева, які найчастіше дають оптимальне логічне дерево відносно його структури;
- розроблення загальної схеми обчислення складності структур класу регулярних логічних дерев для бінарного випадку;

- розроблення методів побудови моделей дерев алгоритмів на основі покрокової апроксимації масиву початкових даних навчальної вибірки набором оцінених незалежних алгоритмів класифікації та розпізнавання;
- встановлення питання збіжності процедури побудови моделей дерев класифікації методів логічних та алгоритмічних дерев для умов слабкого та сильного розділення класів початкової навчальної вибірки;
- розроблення програмного інструментарію побудови структур логічних та алгоритмічних дерев класифікації з метою розв‘язку прикладних задач класифікації образів.

Об‘єкт дослідження. Процеси синтезу структур логічних дерев класифікації різних типів та схем (дерев рішень).

Предмет дослідження. Методи, алгоритми, моделі та інструментальні засоби побудови логічних дерев класифікації.

Методи дослідження. При проведенні досліджень та вирішенні вище зазначених задач використовувались наступні методи: методи та математичний апарат теорії розпізнавання образів, розпізнавання зображень, концепція організації процесу розпізнавання на основі дерева класифікації (дерева рішень), методи математичної статистики, нечітких множин, теорії графів, алгебри – логіки, багатозначної логіки, логіко-комбінаторні методи, методи теорії алгоритмів, методи теорії дискретних функцій, методи структурного та об’єктно-орієнтованого програмування для розробки програмного забезпечення інформаційних систем, використовувались програмні системи для розв‘язку прикладних задач – ОРІОН ІІІ, відкриті ООБ дерев рішень LightGBM, XGBoost.

Наукова новизна одержаних результатів. Результати дисертаційного дослідження формують новий науковий напрямок у теорії аналізу та синтезу дерев рішень, інтегруючим елементом якого є загальний принцип покрокової апроксимації масиву початкових даних набором автономних алгоритмів класифікації та представлення побудованої моделі у вигляді структури АДК.

Наукову новизну дисертаційного дослідження становлять такі основні результати:

вперше:

- розроблено комплексний метод побудови деревоподібних моделей класифікації, який за рахунок поетапної апроксимації масиву початкових даних набором різноманітних відібраних алгоритмів розпізнавання, забезпечує побудову різнотипних моделей класифікації, їх універсальність та можливість роботи з великими масивами різнотипних даних;

- розроблено метод Т – опорних множин, який шляхом фіксації набору ознак разом зі своїми значеннями на основі начальної вибірки з можливістю оцінки даних опорних множин за допомогою відповідних функціоналів, забезпечує ефективний механізм представлення дискретних об‘єктів для структур логічних дерев класифікації;

- розроблено метод побудови структур алгоритмічних дерев класифікації, які відрізняються модульним принципом побудови моделей, що забезпечує розширення прикладної області застосування, побудову моделей з регульованою точністю класифікації;

- розроблено метод побудови обмежених за складністю структур алгоритмічних дерев класифікації, який за рахунок добудови шляхів конструкції дерева класифікації з найбільшою кількістю помилок забезпечує регулювання складності моделей дерев класифікації;

- розроблено метод знаходження подібності конструкцій логічних дерев, який за рахунок подібних вершин в структурі дерева класифікації забезпечує механізм фінальної обрізки побудованої структури дерева класифікації;

- розроблено метод оцінки впливу процедури обрізки логічного дерева класифікації за рахунок перестановки ярусів, рівнів у конструкції регулярного логічного дерева, що забезпечує зменшення складності його структури;

набули подальшого розвитку:

- методи структур логічних дерев класифікації (випадкових дерев класифікації), які за рахунок схеми виправлення помилок в конструкції логічних дерев шляхом корекції (донавчання) структури дерева класифікації, забезпечують побудову якісно кращих структур дерев класифікації;

- метод оцінки збіжності процедури побудови моделей дерев класифікації (структур дерев алгоритмів) для умов слабкого та сильного розділення класів навчальної вибірки за рахунок схем потужності структур дерев класифікації, забезпечує отримання параметрів моделі дерева алгоритмів максимальної складності;

удосконалено:

- схеми дерев моделей класифікації, які є зв‘язаними графами без циклів, у некінцевих вершинах яких знаходяться фіксовані моделі, ребра нумеруються значеннями предикатів цих моделей, що забезпечує спрощення побудови нових класифікаторів за рахунок використання модульного принципу.

Практичне значення одержаних результатів. За допомогою пакетів прикладних програм, що представляють основні результати дисертації, вирішувалися різні задачі розпізнавання образів (з геології, геохімії, геофізики, метрології, медицини, соціології). Розроблені в дисертаційному дослідженні концепції, методи та моделі синтезу СР доведено до практичної реалізації в інформаційних системах, зокрема для аналізу та класифікації екологічних ситуацій у басейнах річок Уж та Тиса Закарпатського регіону.

Практичне значення одержаних результатів полягає в тому, що на основі запропонованих принципів, моделей і методів дерев класифікації (ЛДК/АДК) розроблено алгоритмічно-програмний інструментарій для вирішення широкого кола проблем класифікації дискретних об‘єктів прикладного характеру. Значна увага направлена на дослідження процесу обчислення оцінок над опорними множинами спеціального вигляду (Т – опорні множини) та дослідження повноти моделі алгоритмів типу логічних дерев класифікації, що дозволяють значно скоротити об‘єм обчислень при вирішенні конкретних практичних задач.

Ряд побудованих моделей і схем АДК у межах даного дослідження використано ТОВ «ІНФОСФЕРА» та ТОВ «Медіа - Сервіс» (м. Ужгород), Управлінням економічного розвитку міста Ужгородської міської ради, Закарпатською обласною громадською організацією «Патріотичний

оборонний – спортивний центр Вітязь». Додатково окремі положення дисертаційного дослідження використовуються в різних сферах навчального процесу факультету інформаційних технологій ДВНЗ «Ужгородський національний університет» при викладанні дисциплін: «Методи та засоби штучного інтелекту», «Програмування в системах абстрактних об’єктів і задачі штучного інтелекту», «Теорія алгоритмів», «Технології програмування та створення програмних продуктів», «Пакети прикладних програм», «Організація баз даних і знань».

Особистий внесок здобувача. Дисертаційна робота є самостійно виконаним закінченим науково-прикладним дослідженням, в якому висвітлено набори власних ідей, думок та розробок, що дозволило розв’язати поставлені на початку роботи автором завдання, причому всі результати дисертаційної роботи теоретичного та прикладного характеру отримано здобувачем самостійно. Робота містить теоретичні, методичні положення та висновки, які сформульовано дисертантом особисто. З набору наукових праць, опублікованих автором у співавторстві, в дисертаційному дослідженні використано ідеї та положення, які є результатом індивідуальної праці автора. У наукових роботах, написаних у співавторстві, автору належать: загальна постановка задачі, вибір або розробка методів та алгоритмічних схем їх розв‘язку, ідеї використання концепції апроксимації даних навчальної вибірки набором незалежних автономних алгоритмів класифікації. Зокрема, у публікаціях здобувачу належать: [82-84] – постановка задачі, методики та алгоритмічні схеми обрахунку важливості дискретних ознак, наборів та їх сполучень; [72-74] – схеми оцінки складності структур логічних дерев розпізнавання, метод мінімізації деревоподібних логічних структур та схема оцінки ефекту такої мінімізації, оцінка стійкості максимального за структурною складністю логічного дерева відносно процедури перестановки ярусів; [85-87] – модульний метод побудови схем (структур дерев класифікації) класифікації та розпізнавання, підхід представлення великих масивів даних за допомогою відповідних структур (ЛДК/АДК), загальний метод побудови моделей АДК; [88, 89] – методика представлення правил

класифікації в задачах РО у вигляді схем – агентів; [90] – алгоритми та загальні схеми побудови моделей дерев класифікації, алгоритм донавчання та виправлення помилок класифікації в структурі ЛДК; [91] – метод та схема алгоритму генерації наборів УЗ в структурах АДК, набір синтезованих моделей АДК; [92] – загальна концепція алгоритмічних дерев, метод побудови АДК (типу І), набір побудованих моделей АДК; [93] – побудова та порівняльний аналіз запропонованих моделей.

Програмні інструментальні засоби, які представляють реалізацію конкретних методів, моделей і алгоритмічних схем, у дисертаційній роботі для набору актуальних прикладних задач, розроблено безпосередньо автором цього дослідження.

Апробація результатів дисертації. Основні теоретичні та практичні аспекти та результати дисертаційної роботи було представлено та обговорено на таких науково-практичних конференціях та семінарах:

IV Міжнародній науковій конференції «Сучасні тенденції інформаційних технологій», м. Прага (Чеська Республіка), 2008;

XV Міжнародній науково-практичній конференції «Інновації в навчальному процесі вищих навчальних закладів», м. Сніна (Словацька Республіка), 2008;

IV Міжнародній школі – семінарі «Теорія прийняття рішень», м. Ужгород, 2008;

II Міжнародній науковій конференції молодих вчених Інституту електронної фізики Національної академії наук України «ІЕФ-2011», м. Ужгород, 2011;

III Міжнародній науковій конференції «Математическое моделирование, оптимизация и информационные технологи», м. Кішеневу (Республіка Молдова), 2012;

Міжнародному науковому семінарі «Innovative Methods in Education and Research – InovEduc», м. Прага (Чеська Республіка), 2015;

Міжнародній науково-практичній конференції «Data Stream Mining & Processing – DSMP», м. Львів, 2016, 2020;

Міжнародній школі – семінарі «Communications in Computer and Information Science – CCIS», м. Львів, 2000;

III Міжнародній науково-практичній конференції «Computer Modeling and Intelligent Systems – CMIS», м. Запоріжжя, 2020;

III Міжнародній науково-практичній конференції «Сучасні проблеми математичного моделювання, автоматизованого керування та інформаційних технологій – MCIT», м. Рівне, 2019;

II Міжнародній науково-практичній конференції «Priority directions of science development», м. Львів, 2019;

Міжнародній науково-практичній конференції «Інформаційні технології та комп'ютерне моделювання ІТКМ», м. Івано-Франківськ, 2019;

Міжнародній науково-практичній конференції «Science, Engineering and Technology: Global and Current Trends», м. Прага (Чеська Республіка), 2019, 2020;

X Міжнародній науково-практичній конференції «Математичні методи, моделі та інформаційні технології в економіці», м. Чернівці, 2019;

Міжнародній науково-практичній конференції «Цифрова економіка та інформаційні технології», м. Київ, 2020;

X Міжнародній науково-практичній конференції «Комплексне забезпечення якості технологічних процесів та систем», м. Чернігів, 2020;

XX Міжнародній науково-технічній конференції «Проблеми інформатики та моделювання», м. Харків – м. Одеса, 2020;

V Всеукраїнській науково-практичній конференції «Перспективні напрямки сучасної електроніки, інформаційних і комп'ютерних систем", м. Дніпро, 2020;

III Всеукраїнській науково-технічній конференції «Комп'ютерні технології: інновації, проблеми, рішення», м. Житомир, 2020.

Публікації за темою дисертації. Результати дисертаційного дослідження повною мірою відображені у 58 наукових працях, у тому числі 2 монографіях, 27 статтях у фахових наукових виданнях України з технічних

наук, 8 із них індексуються у міжнародних наукометричних базах, 1 авторське свідоцтво, 28 публікацій в збірниках матеріалів конференцій.

Структура та обсяг дисертації. Дисертаційна робота має класичну структуру та складається зі вступу, семи розділів, висновків, списку використаних джерел із 317 найменувань та 3 додатків на 19 сторінках. Загальний обсяг дисертації становить 382 сторінки, у тому числі 276 сторінок основного тексту, робота містить 67 рисунків та 58 таблиць.

РОЗДІЛ 1
РОЗПІЗНАВАННЯ, ЗАСНОВАНЕ НА МОДЕЛЯХ ЛОГІЧНИХ ДЕРЕВ КЛАСИФІКАЦІЇ

1.1. Постановка задачі машинного навчання в довільних шкалах

На початку дослідження розглянемо принципові моменти задачі дослідження в даній дисертаційній роботі. Нехай задано множину M об‘єктів w та на ній існує розбиття R на кінцеве число підмножин (класів, образів) Ω_i, $(i = 1, \dots m)$, $M = \bigcup_{i=1}^{m} \Omega_i$. Припустимо, що розбиття M визначено неповністю. Задано тільки деяку інформацію I про класи Ω_i. Об’єкти w задаються значеннями деяких ознак x_j, $j = 1, \dots, n$ (цей набір один і той самий для всіх об’єктів, тобто однакова розмірність об‘єктів). Деяку скінчено-значну функцію $f_R(w)$, яка задає розбиття R, задана на множині об‘єктів M, та дає на виході номер класу i, будемо називати функцією розпізнавання (ФР). Зауважимо, що кожний образ (клас) характеризується певною спільністю деяких властивостей його елементів (об‘єктів), а елементи з різних образів не мають цієї спільності. Загальна задача розпізнавання полягає в тому, щоб для довільного об‘єкта w встановити його належність певному класу (образу). Множини Ω_i також називаються компонентами розбиття множини M.

Сукупність значень ознак x_j визначає опис (інформацію) $I(w)$ об’єкта w. Кожна з ознак може приймати значення з різних множин допустимих значень ознак, наприклад, з наступних:

- $\{0; 1\}$ – бінарна ознака, FALSE або TRUE відповідно, фіксує наявність або відсутність певної властивості;
- $\{1; 2; \Delta\}$, тут Δ – інформація про ознаку відсутня;
- $\{0; 1; \dots; d-1\}$ – ступінь визначення ознаки має різні градації, $d > 2$;
- $\{a_1; \dots; a_k\}$ – ознака приймає скінчене число значень, $k > 2$;
- $[a; b], (a; b), [a; b), (a, b]$, тут a, b – довільні числа або символи;
- значеннями деякої ознаки x_j являється функція деякого класу;

•значеннями деякої ознаки x_j являється функція розподілу деякої випадкової величини.

Опис об’єкта $I(w) = (x_1(\mathrm{w}), \dots, x_n(\mathrm{w}))$ будемо називається стандартним, якщо $x_j(w)$ приймає значення лише із множини допустимих значень.

1.1.1. Задача розпізнавання із стандартною інформацією

Задача розпізнавання із стандартною інформацією полягає в тому, щоб для фіксованого об’єкту w та набору класів $\Omega_1, \dots, \Omega_m$ за допомогою навчальної інформації $I(\Omega_1, \dots, \Omega_m)$ та опису $I(w)$ розрахувати значення деяких предикатів $P_i(w), (w \in \Omega_i; i = 1, \dots, m)$.

Одним із можливих варіантів початкового задання навчальної інформації є табличне представлення навчальної вибірки $T_{N,M}$ (табл. 1.1).

Таблиця 1.1. Табличне представлення навчальної інформації.

Об‘єкти	Ознаки та їх значення	Класи
	x_1 x_2 x_j x_N	
w_1 w_1 … w_{r_1}	$a_{1,1}$ $a_{1,2}$ … $a_{1,j}$ … $a_{1,N}$ $a_{1,1}$ $a_{1,2}$ … $a_{1,j}$ … $a_{2,N}$ … … … … … … … … … … .. $a_{r_1,1}$ $a_{r_1,2}$ … $a_{r_1,j}$ … $a_{r_1,N}$	Ω_1
… …		
$w_{r_{i-1}+1}$ $w_{r_{i-1}+2}$ … w_{r_j}	$a_{r_{i-1}+1,1}$ … $a_{r_{i-1}+1,j}$ … $a_{r_{i-1}+1,N}$ $a_{r_{i-1}+2,1}$ … $a_{r_{i-1}+2,j}$ … $a_{r_{i-1}+2,N}$ … … … … … … … … … … .. $a_{r_i,1}$ $a_{r_i,2}$ … $a_{r_i,j}$ … $a_{r_i,N}$	Ω_i
… …		
$w_{r_{m-1}+1}$ $w_{r_{m-1}+2}$ … w_{r_m}	$a_{r_{m-1}+1,1}$ … $a_{r_{m-1}+1,j}$ … $a_{r_{m-1}+1,N}$ $a_{r_{m-1}+2,1}$ … $a_{r_{m-1}+2,j}$ … $a_{r_{m-1}+2,N}$ … … … … … … … … … … .. $a_{r_m,1}$ $a_{r_m,2}$ … $a_{r_m,j}$ … $a_{r_m,N}$	Ω_m
w	b_1 b_2 b_j b_N	$\Omega_?$

Очевидно, що об’єкти $w_1, \dots, w_{r_1}$ належать класу Ω_1, об’єкти $w_{r_{i-1}}, \dots, w_{r_i}$ належать класу Ω_i, а об’єкти $w_{r_{m-1}+1}, \dots, w_{r_m}$ класу Ω_m.

Нехай є розбиття R та деяка система розпізнавання Q. В якості системи Q може бути людина або програмно-апаратна система (система операцій або логічних елементів). Задача розпізнавання образів буде зводитися до навчання системи Q обчислювати функцію $f_R(x)$. Тобто система має реагувати при подачі на вхід деякого сигналу (об‘єкту) x, сигналом $f_R(x)$ (фактичним номером класу належності). Основною інформацією при навчанні системи Q є значення функції $f_R(x)$ в деяких точках $n-$ вимірного простору (розмірністю в кількість ознак об‘єктів множини M). Останнє означає, що при навчанні системи Q їй подаються пари сигналів $\left(x_i, f_R(x_i)\right)$. На основі цієї інформації (апріорної інформації) система Q будує схему обчислення $f_R(x)$.

Отже, можна зафіксувати задачу машинного навчання розпізнавання в межах даного дослідження в наступній постановці: необхідно знайти (побудувати) таку функцію $f_R(x)$, яка би по можливості мала мінімальну форму (складність) та забезпечувала максимальну точність (якість) апроксимації початкової навчальної інформації (навчальної вибірки) [94-100].

1.1.2. Схема побудови ФР у межах методів дерев класифікації

У межах даного дисертаційного дослідження буде розглядатись схема, коли система Q будує послідовність функцій f_1, f_2, ..., які за кожним кроком наближаються до шуканої $f_R(x)$. Підкреслимо, що дана методика покладена в основу абсолютної більшості методів дерев класифікації (методів структур ЛДК/АДК). Відмітимо, що дана схема побудови набору функцій f_1, f_2, ..., з поступовим наближенням до ФР $f_R(x)$ має принципові переваги, які полягають у наступному:

1) Система класифікації Q знаходиться в постійній готовності до розпізнавання (за умови виконання першого кроку – побудови функції f_1). Звичайно, що якість класифікації зростає зі зростанням кількості кроків (індексу функції f_i).

2) Принципова перевага базується на тому факті, що система класифікації Q не має запам‘ятовувати всі навчальні пари (навчання призводить до зміни структури дерева класифікації), за рахунок такої властивості на основі структур ЛДК/АДК відбувається стиск та перетворення структур даних.

3) Побудова фінальної ФР (класифікатора), який реалізується $f_R(x)$, проходить у вигляді послідовності кроків шляхом побудови послідовності функцій $f_1, f_2, \ldots$.

Відмітимо, що завдяки другій властивості система Q на вхід може отримувати необмежену кількість навчальних пар (у разі великого та надвеликого об‘єму НВ), але це не загрожує переповненню пам‘яті інформаційної системи.

Тобто система Q запам‘ятовує (змінює свою структуру дерева класифікації) відповідно лише тим навчальним парам, які враховуються істотними в межах поточної задачі. Проте, така схема генерації структур дерев класифікації не позбавлена і певних недоліків, про які слід згадати:

1) Система класифікації Q може забувати деякі навчальні пари, які поступали на вхід раніше, або взагалі не реагувати на певні набори (відповідні структурні блоки вже є в конструкції даного дерева класифікації).

2) Зрозуміло, що загальна якість класифікації буде тим вища чим довша послідовність $f_1, f_2, \ldots$ побудована (відповідно перші функції $f_1, f_2, \ldots$ можуть бути дуже далекими від шуканої $f_R(x)$). У межах конкретних прикладних задач може виникнути ситуація, коли треба будувати дуже довгу послідовність $f_1, f_2, \ldots, f_i$, доки f_i не наблизиться до ФР $f_R(x)$.

Підкреслимо, що така схема побудови (поетапного наближення) ФР не є єдиною в межах розпізнавання, але в абсолютній більшості випадків притаманна саме методам та алгоритмам концепції дерев класифікації [101-121].

1.1.3. Міра близькості між функціями в задачі розпізнавання

Нехай у спрощеному випадку будемо мати ситуацію, коли розбиття R задає на множині M набор із двох класів Ω_0, Ω_1. Задача буде полягати в тому, щоб на основі початкового набору навчальних пар побудувати фіксовану послідовність функцій $f_1, f_2, \dots, f_e$, які з кожним кроком будуть наближатися до ФР $f_R(x)$. Принциповим питанням, яке виникає з цього факту, є питання міри близькості між функціями $f_e(x)$ та $f_R(x)$.

Отже, нехай маємо дві довільні функції $\phi(x)$ та $\varphi(x)$, які задані на множині початкових сигналів M та приймають значення з множини $\{0,1\}$. Поставимо питання вводу міри близькості між ними.

На першому етапі розглянемо випадок, коли M має скінченну, дискретну природу. Введемо величину $P -$ кількість всіх елементів (потужність) початкової множини M та додатково величину $U(\phi, \varphi) -$ кількість всіх елементів M, на яких функції $\phi(x)$ та $\varphi(x)$ відрізняються. Отже, близькість між функціями $\phi(x)$ та $\varphi(x)$ можна визначити наступною величиною:

$$r(\phi, \varphi) = U(\phi, \varphi)/P. \tag{1.1}$$

На наступному етапі припустимо, що поява об‘єктів x початкової множини M відбувається на основі деякого ймовірнісного розподілу $Q(x)$, де $Q(x)$ відповідно ймовірність появи об‘єкта x з множини M. Відмітимо, що в даному випадку поява одного об‘єкта з M не залежить від появи іншого. Отже, в цьому разі доречно ввести близькість між функціями $\phi(x)$ та $\varphi(x)$ у наступному вигляді:

$$l(\phi, \varphi) = \sum_{x \in M} |\varphi(x) - \phi(x)| * Q(x). \tag{1.2}$$

Зрозуміло, що для випадку, коли поява об‘єкта x має випадковий характер, величина $|\varphi(x) - \phi(x)|$ також випадкова. Введемо математичне сподівання $M_x|\varphi - \phi|$ величини $|\varphi(x) - \phi(x)|$, причому будемо мати наступне:

$$M_x|\varphi - \phi| = \sum_{x \in M} |\varphi(x) - \phi(x)| * Q(x). \tag{1.3}$$

Отже, на основі (1.2) та (1.3) можна зробити наступний логічний висновок:

$$l(\phi, \varphi) = M_x|\varphi - \phi|. \quad (1.4)$$

Зауважимо, що у випадку (1.2) припускається, що функції $\phi(x)$ та $\varphi(x)$ мають детермінований характер. Тому варто на наступному етапі розглянути випадок, коли функції $\phi(x)$ та $\varphi(x)$ є випадковими відносно свого аргументу x, причому $|\varphi(x) - \phi(x)|$ є також випадковою функцією. Тоді введемо математичне сподівання значення випадкової функції $|\varphi(x) - \phi(x)|$ для об‘єкта x – $M_x(x)$. Відповідно до прийнятих припущень близькість між функціями $\phi(x)$ та $\varphi(x)$ доцільно представляти в наступній формі:

$$\rho(\phi, \varphi) = \sum_{x \in M} M_x(x) * Q(x). \quad (1.5)$$

Відмітимо, що випадковість функцій $\phi(x)$ та $\varphi(x)$ у цьому разі означає, що для кожного об‘єкта з множини M набір функцій $\phi_0(x), \phi_1(x), \varphi_0(x), \varphi_1(x)$, де відповідно $\phi_i(x)$ або $\varphi_i(x)$,$(i = 0,1)$, визначає ймовірність того, що $\varphi_i(x)$ або $\varphi(x)$ приймає значення i. Зважаючи на факт того, що $\phi(x)$ та $\varphi(x)$ приймають значення незалежно одна від іншої, можна зробити принциповий висновок, що представлена різниця $|\varphi(x) - \phi(x)|$ приймає значення нуль для об‘єкта x з ймовірністю:

$$X_0(x) = \phi_0(x)\varphi_0(x) + \varphi_1(x)\phi_1(x).$$

Аналогічно різниця $|\varphi(x) - \phi(x)|$ приймає значення одиниці для об‘єкта x з ймовірністю:

$$X_1(x) = \phi_0(x)\varphi_1(x) + \phi_1(x)\varphi_0(x).$$

Отже, зважаючи на все вищезазначене, будемо мати наступну ситуацію:

$$M_x(x) = 1 * X_1(x) + 0 * X_0(x) = \phi_0(x) * \varphi_1(x) + \phi_1(x) * \varphi_0(x). \quad (1.6)$$

Тоді, виходячи з (1.5) та (1.6), будемо мати наступне представлення:

$$\rho(\phi, \varphi) = \sum_{x \in M}(\phi_0(x) * \varphi_1(x) + \phi_1(x) * \varphi_0(x)) * Q(x). \quad (1.7)$$

Зауважимо, що для випадку неперервного простору множини M міри близькості (1.1), (1.2) та (1.5) можна представити в наступній формі:

1) $r(\phi, \varphi) = \sigma(M(\phi, \varphi))/\sigma M;$

2) $l(\phi,\varphi)=\int_M|\varphi(x)-\phi(x)|*p(x)dx;$

3) $\rho(\phi,\varphi)=\int_M M_x(x)*p(x)dx.$ (1.8)

Тут величина σ задає деяку міру для M, причому σM – міра всієї множини M, а відповідно $\sigma(M(\phi,\varphi))$ – міра множини $M(\phi,\varphi)$. Множина $M(\phi,\varphi)$ представляє собою загальну сукупність усіх об‘єктів, для яких функція ϕ відмінна від функції φ, а $p(x)$ задає щільність ймовірності появи об‘єкта з множини M.

Відмітимо, що в неперервному випадку замість відстані (1.8.2) доцільніше використовувати відстань:

$$\int_M((\varphi(x)-\phi(x))^2*p(x)dx. \quad (1.9)$$

Таке представлення має більш аналітичний вигляд.

1.1.4. Міра близькості побудованої ФР та початкової навчальної вибірки в методах ЛДК/АДК

На першому етапі нехай маємо набір $z_1,\dots,z_h$ усіх різних об‘єктів (точок $n-$ мірного простору) початкової послідовності $w_1,\dots,w_s$. Введемо величину $S_i,(i=1,\dots,h)$, яка представляє собою кількість входжень фіксованого об‘єкта z_i до початкової послідовності $w_1,\dots,w_s$ навчальної вибірки.

Отже, тоді на основі S_i можна ввести величину $\delta_i=S_i/S,(i=1,\dots,S)$, яка буде представляти собою частоту входження об‘єкта z_i до початкової послідовності $w_1,\dots,w_s$.

Аналогічним чином вводимо $O_j^i,(i=1,\dots,h;j=1,\dots,m)$, яка представляє кількість пар (z_i,j) (об‘єктів відомої класифікації) в початковій навчальній виборці.

Розмірковуючи в цьому напрямі, отримаємо величину $v_i^j=O_j^i/S_i$, яка представляє собою частоту належності об‘єкта z_i фіксованому класу Ω_j. Тобто припускається, що функція апроксимації $f_S(x)$ має ймовірнісний характер і задається розподілом загального вигляду $p(x/\Omega_0),\dots,p(x/\Omega_m)$.

Отже, виходячи з вищезазначеного, міру близькості між початковою навчальною вибіркою та побудованою $f_S(x)$ (послідовністю функцій апроксимації) можна представити у вигляді наступної величини:

$$\rho_S = \sum_{i=1}^{h} \delta_i \left(\sum_{j=1}^{m} \left| v_i^j - p(z_i/\Omega_j) \right|\right). \quad (1.10)$$

Так у разі, коли послідовність класів $\Omega_1, \dots, \Omega_m$ деякого початкового розбиття R є детермінованою, де даний набір класів представляється у вигляді підмножин, які не перетинаються з початковою множиною M. Тоді $f_S(x)$ можна представляти у вигляді деякої детермінованою функції, де аргумент x приймає значення з початкової множини сигналів M, причому сама $f_S(x)$ приймає значення класу належності (значення з множини $\{1, \dots, m\}$).

Отже, для даного детермінованого випадку величину близькості між початковою навчальною вибіркою та побудованою функцією апроксимації можна представити в наступному, прощеному вигляді:

$$\rho_S = \sum_{i=1}^{h} \delta_i \psi |f_R(z) - f_S(z)|). \quad (1.11)$$

Відмітимо, що в даному виразі $\psi(x) = 0$ у випадку $x = 0$ та відповідно $\psi(x) = 1$ при $x \neq 0$. Підкреслимо, що вимога близької апроксимації початкової навчальної вибірки функція $f_S(x)$ фактично означає зменшення величини ρ_S з наведених вище фунціоналів.

Слід відмітити, що набір класів $\Omega_1, \dots, \Omega_m$ початкової множини сигналів M характеризується не тільки розподілом $p(x/\Omega_i), (i = 1, \dots, m)$, але й деяким ймовірнісним розподілом $p(x)$, де $p(x) -$ ймовірність появи об‘єкта x з множини M, причому в початковій навчальній виборці об‘єкти $x_1, \dots, x_S$ з‘являються незалежно один від одного та відповідно до розподілу $p(x)$. Тоді можна зробити висновок, що представлена вище величина δ_i представляє собою ефективну оцінку ймовірностей $p(z_i), (1, \dots, h)$.

Зауважимо, що на функцію апроксимації начальної послідовності $f_S(x)$ накладається наступна вимога: при збільшенні навчальної послідовності (потужності НВ) функція $f_S(x)$ буде переходити у функцію $f_{S+k}(x)$, де величина $k -$ кількість навчальних пар, що додається до даних НВ.

Підкреслимо, що такий перехід значною мірою впливає на складність результуючого алгоритму класифікації.

Зрозуміло, що дана схема може бути застосована і у разі наявності деякої апріорної інформації відносно початкового розбиття R, наприклад, коли набір класів $\Omega_1, \dots, \Omega_m$ задається деякою функцією $f_R(x)$ про яку відомо, що вона належить класу T. Тоді на функцію апроксимації $f_S(x)$ крім початкових вимог накладається умова належності $f_S(x) \in T$ або якась інша зв‘язана з нею умова.

Резюмуючи все вищезазначене, приходимо до наступних положень:

1) Задача розпізнавання образів зводиться до того, щоб навчити систему Q обраховувати деяку функцію $f_R(x)$, яка визначена на множині M та приймає скінчену кількість значень. Функція $f_R(x)$ задає однозначне розбиття R. Причому дві функції $f_R(x)$ та $h_R(x)$ будуть вважатись однаковими, якщо вони представляють одне і теж саме розбиття.

2) Базовий етап у розпізнаванні образів – навчання (фактична обробка великих масивів інформації). При навчанні система Q приймає послідовність навчальних пар $(x_1, f_R(x_1)), \big(x_2, f_R(x_2)\big), \dots$ та на основі цієї інформації будує схему обчислення $f_R(x)$ або її наближення.

3) На етапі навчання системи виникають наступні питання: економія пам‘яті системи Q, швидкодія навчання системи Q, побудова такої схеми для обчислення $f_R(x)$, щоб вона була за деякими важливими параметрами оптимальною (об‘єм пам‘яті СР, швидкодія, надійність).

4) У межах дисертаційного дослідження буде розглядатись схема, коли система Q будує послідовність функцій $f_1, f_2, \dots$, які з кожним кроком наближаються до шуканої $f_R(x)$. Підкреслимо, що дана методика покладена в основу абсолютної більшості методів структур ЛДК/АДК).

6) У дисертаційній роботі ставиться задача дослідження та розробки таких методів та моделей розпізнавання, які б давали можливість у процесі навчання побудувати по-можливості просту деревоподібну схему розпізнавання, яка забезпечує необхідну ефективність та складність системи розпізнавання Q.

1.2. Поточний стан концепції дерев класифікації в задачах штучного інтелекту

Аналізуючи проблематику деревоподібних моделей класифікації та розпізнавання, можна побачити певний брак поточних досліджень у цьому напрямку, коли головна увага зміщена в бік концепції нейромережевого розпізнавання [122-134]. Як буде підкреслено в подальшому, значною мірою це пояснюється особливостями самих моделей ЛДК, складнощами реалізаційних моментів концепції алгоритмічного дерева класифікації (найвищого рівня абстракції концепції ЛДК), набором жорстких правил та обмежень щодо практичної роботи з такими структурами даних [135-140]. Проте, представлення навчальних вибірок (дискретної інформації) великого об‘єму у вигляді структур логічних дерев має свої суттєві переваги для економічного опису даних та ефективних механізмів роботи з ними [141-145]. Тобто процедура покриття навчальної вибірки набором елементарних ознак у випадку ЛДК або покриття навчальної вибірки фіксованим набором автономних алгоритмів розпізнавання та класифікації у випадку АДК породжує фіксовану деревоподібну структуру даних, яка певною мірою забезпечує навіть стиск та перетворення початкових даних НВ – а, отже, дозволяє суттєву оптимізацію та економію апаратних ресурсів системи [146, 147].

Зауважимо, що галузь застосування концепції ЛДК нині надзвичайно об‘ємна, а множина задач та проблем, які розв‘язуються даним апаратом може бути зведена до наступних трьох базових сегментів:

1) Задача опису структур даних. Схеми ЛДК дають можливість накопичувати та зберігати інформацію про набори даних у компактній формі (допускається робота з надвеликими масивами даних), замість них можна зберігати логічне дерево (або його певну часткову конструкцію), яке містить точний опис початкового набору об'єктів.

2) Задача розпізнавання та класифікації. Схема ЛДК дозволяє ефективно забезпечувати класифікацію та розпізнавання наборів дискретних

об‘єктів, тобто відносити об'єкти до одного з відомих класів за даними деякої початкової НВ (яка фіксується в структурі логічного дерева). У цьому разі цільова змінна повинна мати дискретні значення.

3) Задача регресії. Тут слід відмітити, що якщо цільова змінна має неперервні значення, то логічні дерева дозволяють встановити деяку залежність цільової змінної від незалежних (початкових, вхідних) наборів змінних. Наприклад, сюди відносяться задачі чисельного прогнозування (тобто передбачення значень цільової змінної).

1.2.1. Загальна концепція дерев рішень

Важливим сегментом області застосувань концепції логічних дерев залишають методи дерев рішень (дерева класифікації, регресійні дерева), які активно використовуються як для задач теорії штучного інтелекту засобом підтримки прийняття рішень, так і в суміжних практичних галузях економіки, управління тощо [148-159]. Дерева рішень є одним із базових методів автоматичного аналізу даних. Слід підкреслити, що перші дослідження та концепції ідеї застосування дерев рішень (класифікації) ведуть початок від робіт С. Ховленда і Е. Ханта. Однак, зауважимо – вважається, що основоположною роботою, яка забезпечила ініціалізацію та розвиток даного напрямку є робота Е. Ханта, Д. Меріна, П. Стоуна [160].

Відмітимо, що важливою особливістю дерев класифікації є гнучкість, тобто здатність ЛДК послідовно враховувати та досліджувати ефект впливу окремих змінних, атрибутів структури. Відповідно, є ще цілий ряд причин, що забезпечують структурам ЛДК більшу гнучкість, ніж традиційні методи та інструменти аналізу даних. Так, здатність ЛДК виконувати одномірне розгалуження для аналізу впливу (важливості, якості) окремих змінних дає можливість працювати зі змінними різних типів у вигляді предикатів (у випадку АДК – відповідними автономними алгоритмами класифікації та розпізнавання) [161-163]. У цьому разі структура логічного дерева представлена у вигляді гілок та вузлів, причому на гілках дерева розташовуються деякі мітки (атрибути, значення), від яких залежить цільова

функція (у випадку ЛДК – функція розпізнавання), а у вузлах (вершинах) знаходяться значення ФР або розширені атрибути переходів (рис. 1.1). Така концепція логічних дерев активно використовується в інтелектуальному аналізі даних, де кінцева мета полягає в синтезі моделі, яка прогнозує значення цільової змінної на основі деякого набору початкових даних на вході інформаційної системи [164-171].

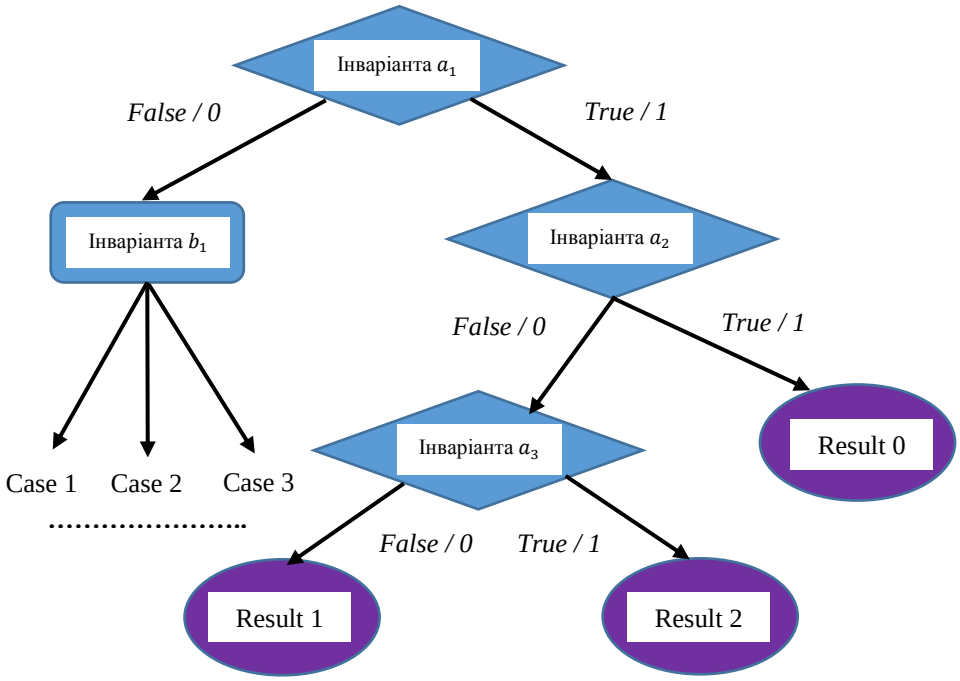


Рис. 1.1. Загальна схема логічного дерева рішень.

На даному етапі приведемо загальну схему побудови дерева рішень у класичній інтерпретації Р. Куінлена, яка полягає в наступному: нехай на початку в якості актуальної начальної інформації маємо деяку навчальну множину M, яка містить об‘єкти відомої класифікації з ознаковим простором розмірності n (n – кількість ознак, атрибутів об‘єктів) [172]. Зауважимо, що це – об‘єкти відомої класифікації, а, отже, останній атрибут відповідає за значення функції розпізнавання (вказує на належність об‘єкта до відповідного класу). Нехай множина $\{H_1, H_2, \dots, H_k\}$ є множиною деяких класів, тоді необхідно розглянути три можливі варіанти:

1) Варіант, коли початкова навчальна множина M містить один або більше об‘єктів, які відносять до деякого класу H_k. Тоді фактично остаточне дерево рішень для M буде гілкою, що визначає клас H_k.

2) Варіант, коли початкова навчальна множина M не містить жодного об‘єкту відомої класифікації (є пустою множиною). У цьому разі це буде знову гілка логічного дерева, а клас, який асоціюється з гілкою, береться з іншої множини відмінної від M.

3) Варіант, коли початкова навчальна множина M містить об‘єкти, які належать різним класам H_i. У цьому випадку початкову множину M розбивають на набір деяких підмножин, для цього вибирається одна з ознак, яка має від двох і більше різних значень $q_1, q_2, \dots, q_s$. Тоді початкова навчальна множина M розбивається на підмножини $M_1, M_2, \dots, M_s$, причому кожна з M_i містить усі навчальні об‘єкти, які мають значення q_j для вибраної ознаки. Відмітимо, що дана схема має рекурсивно працювати до тих пір, доки кінцева множина не буде містити об‘єкти, які належать одному й тому ж фіксованому класу.

1.2.2. Проблеми сучасних методів і схем дерев класифікації

Отже, запропонована вище схема (рис 1.1) фактично об‘єднує більшість сучасних алгоритмів та методів побудови логічних дерев рішень, а сама схема відома із спеціалізованої літератури під назвою «розділення та захоплення» (divide and conquer). Відмітимо лише, що при застосуванні цієї схеми, побудова ЛДК буде здійснюватися за напрямом зверху до низу [173-177].

Оскільки на початку роботи схеми маємо справу з фактичною навчальною вибіркою (набором об‘єктів відомої класифікації), то такий процес побудови ЛДК відносимо до схеми навчання з вчителем (supervised learning), у літературі також називають індуктивним навчанням або індукцією логічних дерев (trees induction). Зауважимо, що на сьогоднішній день існує значна кількість алгоритмів, які реалізують концепцію дерев рішень: CART, C4.5/C5.0, Sparc, NewID, ITrule, CHAID, CN2, Oris та інші, але найбільшого вживання та розповсюдження отримали наступні два представники [178-180]:

1) Алгоритм класифікації регресійного дерева (CART – Classification and Regression Tree) – це алгоритм побудови деякого бінарного дерева розв‘язків (класифікаційна модель на основі дихотомії). Причому кожна

вершина (вузол логічного дерева) при розбитті має лише два ребра (гілки або нащадків). Головна галузь застосування цього алгоритму – задачі класифікації та регресії.

2) Алгоритм дерева рішень C4.5/C5.0 (універсальний алгоритм класифікації) не накладає обмежень на кількість гілок із вершини (вузла) логічного дерева (небінарний випадок), але не вміє працювати з неперервним цільовим полем – тобто може бути застосованим лише для задач класифікації. Зауважимо лише, що алгоритм C4.5/C5.0 є подальшим концептуальним розвитком алгоритму ID3 (Interactive Dichotomizer). Відмітимо, що крайній білд C5.0 є останньою модифікацією схеми C4.5 (2018 року) та має певні принципові відмінності порівняно зі своїм початковим стандартом.

Слід відмітити, що абсолютна більшість відомих алгоритмів побудови дерев рішень відносяться до класу «жадібних алгоритмів». Тобто якщо на якомусь етапі була відібрана деяка вершина (атрибут, вузол) та по ній проведено розбиття на підмножини початкової вибірки – алгоритм не може на наступному кроці повернутися назад для вибору іншої вершини (вузла) з якіснішими показниками розбиття. Це фактично означає, що на етапі побудови логічного дерева неможливо визначити, чи дасть відібрана вершина розгалуження, а в кінцевому етапі – деяке оптимальне розбиття.

При синтезі дерев рішень центральними питаннями залишаються питання вибору критерію атрибуту (вершини ЛДК), за яким відбудеться розбиття початкової НВ, критерію зупинки навчання (побудови структури ЛДК) та критерію відкидання гілок логічного дерева (піддерев ЛДК).

Так, при побудові структури логічного дерева на кожній вершині (вузлі ЛДК) необхідно знайти таку умову, яка би забезпечувала розбиття множини, що відповідає даній вершині, на підмножини. В якості такої умови (інваріанти) можна вибирати один з атрибутів (ознак об‘єктів з НВ). Причому загальне правило відбору ознак можна представити в наступному вигляді: відібрана ознака має розбивати множину так, щоб отримані підмножини складалися з об‘єктів належності одному фіксованому класу (у випадку, якщо це не

можливо – максимально наближатись до цього, тобто кількість об‘єктів належності іншим класам була б мінімальною) [181-190].

Відмітимо, що згаданий вище алгоритм логічного дерева C4.5/C5.0 як критерій відбору вузла (вершини) використовує так званий теоретико-інформаційний критерій, а алгоритм CART базується на розрахунку індексу Gini, який враховує відносні відстані між розподілами класів.

Базовим питанням у схемі побудови логічного дерева залишається проблема відбору правила зупинки розгалуження (критерій зупинки побудови ЛДК). Аналізуючи існуючи методи та алгоритми побудови логічних дерев, можна виділити три наступні базові підходи в цьому напрямку:

1) При побудові ЛДК використовувати статистичні методи оцінки необхідності подальшого розбиття (розгалуження) множини (це так звана рання зупинка – prepruning). Зауважимо, що такий підхід ефективний щодо економії загального часу навчання. Однак зрозуміло, що є і негативні моменти, які пов‘язані з впливом такого підходу на точність результуючої класифікаційної моделі, а, отже, завчасна зупинка розгалуження в структурі ЛДК є вкрай небажана. Так, наприклад, у згаданих вище роботах Р. Куінлен та Л. Брейман пропонують замість зупинки розгалуження використовувати процедуру відсікання.

2) При побудові ЛДК використовують схему обмеження глибини (кількості ярусів) логічного дерева. Тобто обмежується побудова логічного дерева, якщо поточне розгалуження (вибір вершини, вузла) призводить до структури дерева з глибиною, яка перевищує задане значення. Зрозуміло, що такий підхід має як позитивні, так і негативні моменти, які впливають на результуючу модель розпізнавання (класифікації).

Зауважимо, що далі в межах даного дисертаційного дослідження буде також представлений модифікований (обмежений) метод АДК, який як раз і базується на цій ідеї.

3) При побудові ЛДК розбиття (розгалуження) має бути не тривіальним, тобто в структурі дерева вершини мають містити не менше заданої кількості початкових прикладів (об‘єктів).

Слід відмітити, що в практичній площині досить часто алгоритми та методи побудови ЛДК на виході дають структурно складні логічні дерева (щодо кількості вершин, кількості розгалужень, належності до класу нерегулярних дерев), які нерівномірно заповнені даними, мають різну кількість розгалужень. Такі складні деревоподібні структури досить важко сприймаються для зовнішнього аналізу за рахунок великої кількості вузлів (вершин) та великої кількості покрокових розбиттів початкової НВ, якщо містять мінімальну кількість об‘єктів (можливо навіть одиничні об‘єкти в найгіршому випадку). Зрозуміло, що значно краще мати деяке ЛДК з мінімальної кількістю вершин (вузлів), яким би відповідала абсолютна більшість об‘єктів початкової НВ. Саме на цьому етапі виникає принципове питання теорії ЛДК – питання можливої побудови всіх варіантів логічних дерев, які відповідають початковій НВ та відбору мінімального за глибиною (загальною кількістю ярусів) логічного дерева [70]. Тут слід відмітити, що дана задача є $NP-$ повною (це було зафіксовано ще Л. Хайфілем та Р. Рівестом), а, отже, не має простих та ефективних методів розв‘язку [191].

Однак, для розв‘язку вище описаної проблеми часто застосовують так званий механізм відсікання гілок *(pruning)*. Нехай під точністю розпізнавання побудованого ЛДК береться відношення правильно класифікованих об‘єктів при навчанні до загальної потужності НВ, а відповідно кількість помилок класифікації буде представлятися параметром помилки. Отже, припустимо, що відомий ефективний спосіб оцінки помилки ЛДК. Тоді можна запропонувати наступне просте та раціональне правило:

1) На першому етапі побудувати ЛДК за даними початкової НВ.

2) На другому етапі відсікти або провести заміну піддеревом тих гілок, які не призводять до збільшення параметру помилки.

Варто відмітити, що на відміну від процесу побудови ЛДК, відсікання гілок відбувається знизу догори, рухаючись від кінцевих вершин (листів), відмічаючи вершини як листи або замінюючи їх піддеревами. Проте, процедура відсікання не завжди дозволяє вирішити проблему складності, але в більшості практичних задач дає непогані результати, що дає змогу зробити висновок про необхідність такої методики.

Зауважимо, що в практичній площині, наприклад у разі великих масивів даних можливі випадки, коли навіть усічені логічні дерева все одно відрізняються великою складністю, зокрема складністю аналізу та сприйняття. Тоді зазвичай використовується процедура виносу (вибору) правил (шляхів) ЛДК із наступним етапом створення множин правил опису класів. Для цього проводиться загальне дослідження всіх шляхів ЛДК від початкової вершини до кінцевого ярусу, причому кожний такий шлях (або $T-$ опорна множина з даного дослідження) генерує деяке правило, де інваріантами (умовами перевірки) будуть атрибути з вершин, які належать даному шляху в ЛДК. Відмітимо лише, що визначення $T-$ опорної множини (в сенсі логічних дерев), області її застосування та зв‘язку з ЛДК буде введено в даному дисертаційному дослідженні в наступних розділах.

Отже, попри певну проблематику, яка виникає при побудові та використанні дерев рішень (концепції ЛДК у загальному сенсі) слід зафіксувати наступні їх переваги:

1) Деревоподібні моделі відрізняють принципово швидкий етап навчання системи розпізнавання.

2) Можливість синтезу множини правил рішень у межах області, де навіть кваліфікованому експерту важко сформувати набір рекомендацій.

3) Синтез правил (класифікаційних правил, правил рішень) на природній мові.

4) Отримана результуюча деревоподібна модель класифікації є інтуїтивно зрозумілою.

5) Побудований прогноз, який отримується на останньому етапі роботи моделі, характеризується високою точністю навіть у порівняні з статистичними та нейромережевими моделями.

6) Можливість побудови непараметричних моделей.

Отже, зважаючи на все вищезазначене, можна зробити висновок, що моделі дерев рішень є важливим та актуальним інструментом широкого аналізу да представлення структур даних [192-200].

1.2.3. Моделі дерев класифікації на основі схеми C4.5/C5.0

Зауважимо що ПС, які базуються на алгоритмах схеми C5.0 (за авторством J. Ross Quinlan) використовують в якості критерію чистоти підмножин початкової НВ параметр ентропії. Використовуючи ентропію в якості міри чистоти (однорідності) класів (підмножин початкової НВ), які є результатом процедури розбиття (розгалуження структури дерева), алгоритм може зафіксувати (відібрати) ту ознаку (атрибут), розбиття за якою дає найчистішу (однорідну) підмножину початкової НВ (тобто підмножину початкової НВ з найменшою ентропією). Така схема в літературі позначається «*information gain»* (схема підсилення інформації), причому, якщо для відібраної ознаки x_i величина *information gain* є нульовою, то це фактично означає безперспективність (неможливість) розбиття НВ на підмножини – не приводить до зменшення коефіцієнту ентропії. Підкреслимо, що максимально можливе значення величини *information gain* дорівнює величині ентропії до розбиття, а це в свою чергу означає, що ентропія після поточного розбиття частини НВ буде дорівнювати нулю для повністю чистих (однорідних) підмножин початкової НВ.

У зв‘язку з тим, що структури ЛДК після побудови за вибірками реальних даних великого об‘єму мають здебільшого складну для аналізу та неоднорідну за рівнями (ярусами) структуру, то принциповою проблемою залишається питання організації процедури оптимізації або обрізки (*pruning*) таких конструкції. Під складністю структури ЛДК розуміється загальна кількість вершин конструкції дерева (вузлів розгалуження), в такому випадку зазвичай

мається на увазі, що модель ЛДК перевизначена (*is overfitted*). Важливою особливістю схеми C5.0 в плані корекції структури побудованого дерева є можливість використання механізму *post-pruning*, коли відкидаються ті вузли, блоки конструкції, піддерева, які мало (відповідно деякого заданого критерію) впливають на результат загальної класифікації (допустима помилка), причому допускається не лише просте відсікання структур дерева, але і їх перенесення в іншу частину структури ЛДК або заміну на іншу конструкцію з меншою структурною складністю (меншої розгалуженості). Ці схеми оптимізації (обрізки) структур ЛДК – *subtree raising* (підняття піддерева) та *subtree replacement* (заміна піддерева) – в процедурі *pruning* використовуються в C5.0 та в небагатьох інших методах побудови дерев класифікації, причому абсолютна більшість інших методів та схем базуються на процедурі попередньої обрізки структури ЛДК, що будуються – *pre-pruning*, яка має суттєві недоліки щодо можливості пропуску (відсікання) важливих даних, які важко виявити. Отже, зважаючи на зазначене вище, можна зафіксувати наступні особливості схеми C5.0 щодо побудови структур ЛДК:

1) Високий рівень універсальності та адаптивності – дозволяє роботу з широким спектром прикладних задач різноманітних галузей практичної діяльності (обмеження щодо структури та природи початкової НВ не накладаються).

2) Універсальність щодо типів початкових масивів даних – дає можливість працювати не тільки з дискретними вибірками, але також з масивами номінальних даних (дозволяє коректну обробку випадків пропущених даних).

3) Організація розгалуження в структурі ЛДК за принципом селекції елементарних ознак – враховує тільки найважливіші (інформативні) ознаки (атрибути) дискретних об‘єктів, тобто такі, які мають найбільший вплив на остаточну класифікацію.

4) Незалежність відносно об‘єму та структури початкової НВ – дає можливість працювати з початковими НВ як відносно невеликого об‘єму, так і з надвеликими масивами даних.

5) Висока наочність та простота інтерпретації роботи побудованої моделі (структури ЛДК) – яка не вимагає спеціалізованої математичної підготовки.

6) Висока ефективність побудованих моделей ЛДК – навіть в порівнянні з аналогічними структурами (моделями) побудованими за класичними схемами C4.5 та CART.

Відмітимо, що незважаючи на високу ефективність у практичній площині, наявність якісного механізму оптимізації (обрізки, *post-pruning*) побудованих структур ЛДК схема C5.0 не позбавлена і певних системних недоліків, які обов‘язково потрібно враховувати як при реалізації так і при роботі з побудованими моделями ЛДК:

1) Визначальною особливістю є те, що будуються дерева високої структурної складності з великою кількістю вершин, рівнів (ярусів) та високою неоднорідністю побудованої структури. Така особливість схеми C5.0 накладає високі вимоги на ефективність роботи процедури обрізки побудованої структури ЛДК та негативно впливає на інтерпретабельність – можливість доступного аналізу моделі та простого сприйняття побудованих конструкцій дерев класифікації. Так за структурою, рівнем неоднорідності та складністю побудовані моделі ЛДК дуже схожі на дерева класифікації синтезовані на основі методів РВО з роботи.

2) Модель ЛДК, яка побудована на основі схеми C5.0 може бути як недовизначеною (*overfit*) так і перевизначеною (*underfit*).

3) У роботі моделі ЛДК, яка побудована на основі схеми C5.0, можливі певні неточності (помилки) класифікації в зв‘язку з використанням лише прямого розбиття на підмножини (*axis-parallel split*).

4) Принциповою особливістю схеми C5.0 є її дуже висока чутливість щодо корекції в структурі та об‘єму початкової НВ, причому навість її

відносно невеликі зміни можуть приводити до дуже різких змін структурної складності (радикального збільшення вершин, ярусів конструкції дерева класифікації) та ефективності процедури кінцевої оптимізації (обрізки) побудованих моделей ЛДК.

5) Моделі ЛДК, які побудовані на основі схеми C5.0 в абсолютній більшості випадків відрізняються великою складністю, а їх аналіз можливий лише за рахунок автоматичного або зовнішнього виділення правил класифікації конструкції ЛДК.

1.2.4. Моделі дерев класифікації на основі градієнтного бустингу

Іншим варіантом щодо хороших показників точності побудованих моделей (структур ЛДК) та швидкості генерації конструкцій дерев класифікації може бути платформа машинного навчання XGBoost (*eXtreme Gradient Boosting*), яка в концептуальному плані схожа з LightGBM. Відмітимо, що XGBoost це платформа (набір алгоритмів) машинного навчання, яка заснована на загальній концепції дерева рішень та використовує фреймворк градієнтного бустингу.

Отже концепція фреймворку XGBoost базується на вдосконаленій реалізації алгоритмічної схеми градієнтного бустингу. Так, при реалізації моделі на платформі XGBoost необхідно враховувати набір різноманітних параметрів та їх атрибутів, які задаються шляхом прямої ініціалізації, причому модель XGBoost по аналогії LightGBM потребує правильного початкового налаштування параметрів для побудови ефективних структур ЛДК та повного використання своїх переваг в порівнянні з іншими подібними платформами машинного навчання. Серед принципових особливостей платформи XGBoost слід зафіксувати наступні:

1) Стандартна реалізація LightGBM не має схеми нормалізації моделі, яка присутня в XGBoost та дозволяє достатньо ефективно вирішувати задачу переускладнення побудованої структури дерева класифікації.

2) Підкреслимо, що на платформі XGBoost реалізована схема так званого нормалізованого бустингу, яка показує хороші результати щодо структурної складності результуючої моделі дерева класифікації.

3) Відмітимо, що платформа XGBoost також реалізує механізм паралельної обробки даних та в абсолютній більшості прикладних задач працює швидше в порівнянні з LightGBM. Звернемо увагу, що процедура прямого бустингу є суто послідовною схемою генерації набору нових класифікаторів, тим не менше є механізми, які дозволяють розбити даний процес на блоки та конвеєри.

4) Важливою особливістю XGBoost є те що вона дозволяє в процесі побудови моделі визначати набори власних цілей процедури оптимізації та критерії її фінальної оцінки. Зрозуміло, що це є дуже гнучким інструментом при побудові моделі дерева та наборів класифікаторів.

5) По аналогії з LightGBM, XGBoost має в своєму складі вбудовану процедуру для обробки пропущених значень (ознак та атрибутів). Використовуючи інтерактивний режим можливо вказати інше значення атрибуту, відмінне від інших спостережень (наборів), і передати його в якості параметра.

6) Схема LightGBM на відміну від XGBoost припинить розділення вузла (поділ підмножини початкової НВ), коли зіткнеться з негативною втратою на етапі поділу. Таким чином, таку схему можна скоріше віднести до класу жадібних алгоритмів.

7) У бібліотеці XGBoost реалізована схема, в якій при побудові дерева класифікації проводиться процедура розгалуження відповідно до значення параметру *max_depth* і лише потім відбувається перехід до фінального етапу мінімізації (обрізки структури) дерева, причому видаляються саме ті розгалуження (відповідні блоки конструкції дерева) за межами яких немає позитивного посилення якості моделі.

8) Важливою особливістю XGBoost на відміну від LightGBM є те, що тут реалізована схема глибинного аналізу структури розгалужень, яка

дозволяє добитися позитивного покращення якості моделі навіть у випадку, коли загальний коефіцієнт якості розгалужень, які розташовані вище в блоці структури дерева, дають загальний менший негативний ефект.

9) Механізм XGBoost дозволяє виконувати кінцеву перехресну перевірку (*cross-validation*) на кожній ітерації процесу бустингу, а, отже, остаточно отримати максимально точну – оптимальну – кількість бустингових ітерацій за один запуск. Зрозуміло, що процедура перехресної перевірки є досить ресурсоємною щодо апаратних витрат і потребує значних витрат часу на остаточну побудову та перевірку моделі дерева класифікації.

10) Важливою особливістю в XGBoost є те, що дозволяється схема поетапного навчання моделі, причому експерт може продовжити навчання моделі XGBoost з її останньої ітерації попереднього запуску. Зрозуміло, що це є суттєвою перевагою для деяких прикладних задач. Подібний механізм донавчання моделі в іншій формі реалізується також і в LightGBM.

Загальний еволюційний шлях концепції дерев рішень можна схематично представити у вигляді, поданому на рис. 1.2. Причому алгоритми класичних методів дерев рішень спрямовані на роботу з відібраними та сформованими за тими чи іншими правилами наборами базових критеріїв. Відповідно методи бегінгу базуються на схемі зваженого голосування при прийнятті відповідних рішень щодо класової належності того чи іншого об‘єкту класифікації. Схеми випадкового лісу базуються на ідеї випадкового голосування щодо того чи іншого рішення класифікатора, причому атрибути (ознаки) в процедурі розгалуження вибираються також випадковим чином. Алгоритми бустингу базуються на схемі поетапної оцінки класифікаторів щодо прийняття того чи іншого рішення (такий підхід в залежності від реалізації може в деяких випадках значно прискорити процедуру побудови фінальної моделі класифікації та позитивно вплинути на її точність). У схемі градієнтного бустингу (як часткового випадку прямого бустингу) модель класифікації будується на основі алгоритму градієнтного спуску. Схема XGBoost (найбільш сучасне переосмислення загальної концепції бустингу) базується на так званій

схемі екстремального градієнтного бустингу та спрямована на значну програмну та апаратну оптимізацію обчислень при побудові моделей дерев класифікації.

Відмітимо, що XGBoost та Gradient Boosting Machines (платформа LightGBM) – набір методів та алгоритмів концепції дерев рішень, які використовують принцип послідовного бустингу (в найпростішому випадку – схему побудови бінарного дерева рішень) на основі алгоритму градієнтного спуску, причому бібліотека XGBoost розглядається як вдосконалена версія фреймворку LightGBM через системну оптимізацію та вдосконалення алгоритмів.

Принциповим моментом в XGBoost є те, що побудова моделей дерев заснована на схемі розпаралелювання, причому цей механізм реалізований завдяки взаємозамінній природі циклів (ітерацій), що використовуються для побудови початкової бази навчання – зовнішній цикл проходить усі вузли структури дерева, внутрішній цикл призначений для обчислення наборів ознак (атрибутів). Така організація вкладення одного циклу всередину іншого заважає забезпечити ефективне розпаралелювання алгоритму. Для подолання цього принципового обмеження та покращення роботи алгоритму, порядок циклів змінюється за наступною схемою: ініціалізація проходить при зчитуванні (введенні) початкових даних, далі слідує етап сортування, який відповідно можна розпаралелити (розбити на потоки та конвеєри).

Зрозуміло, що така оптимізація значною мірою покращує продуктивність алгоритму, причому важливою особливістю фреймворку LightGBM є те що, критерій зупинки розгалуження структури дерева (розбиття дерева класифікації) залежить від параметру втрати у вузлі, де проводиться розгалуження (поточне розбиття на підмножини).

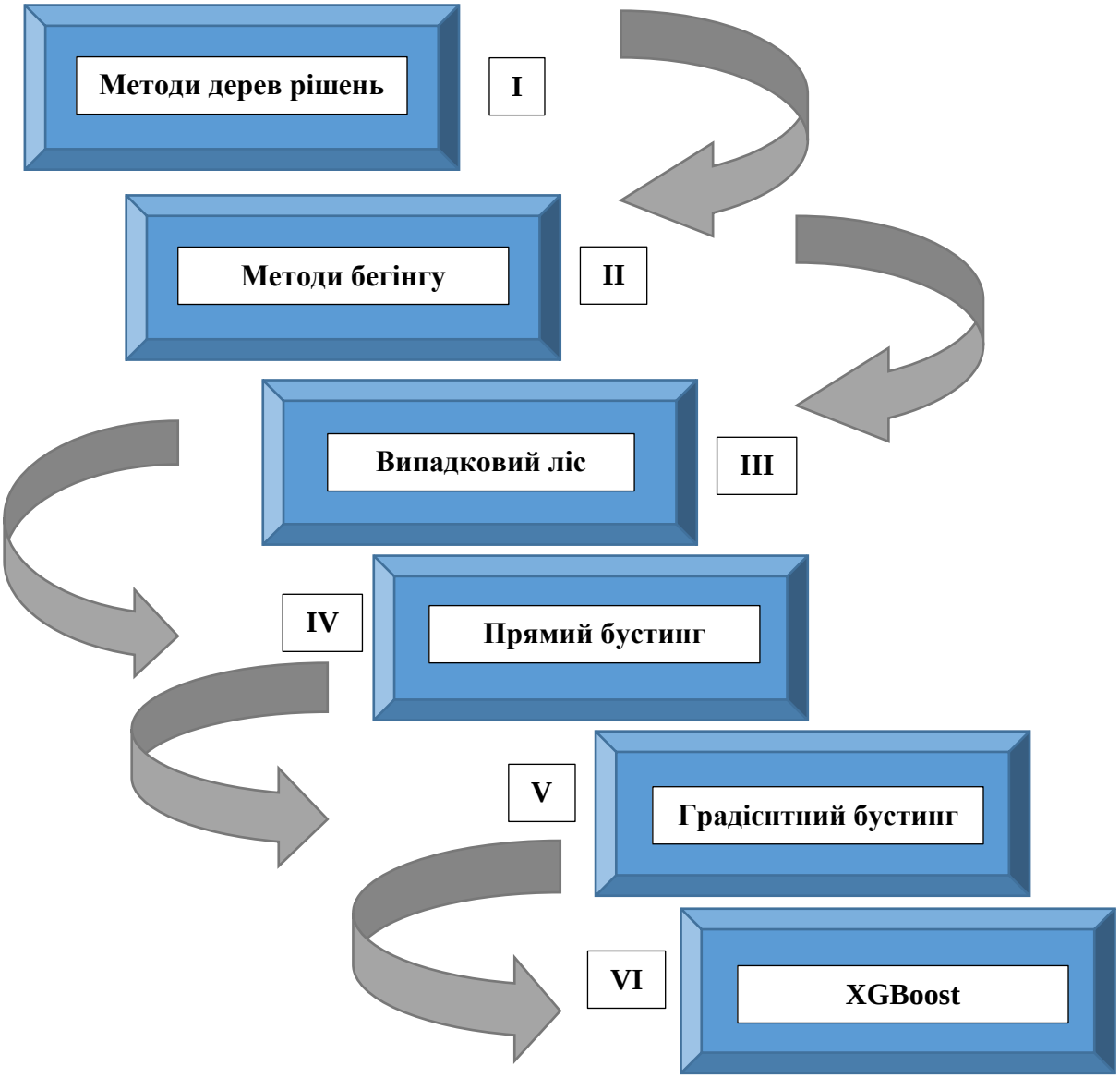


Рис. 1.2. Еволюція методів та алгоритмів концепції дерев рішень.

На відміну від цієї схеми XGBoost використовує параметр максимальної глибини структури дерева *max_depth* (замість критерію розгалуження LightGBM) та проводить так зване зворотне відсікання, причому такий підхід, заснований на загальній глибині структури дерева значною мірою покращує обчислювальну продуктивність схеми.

1.2.5. Метод розгалуженого вибору ознак

Відмітимо, що проблематика концепції ЛДК піднімалася в роботах українських науковців, серед яких слід виділити роботи Василенка Ю. А., присвячені загальним алгоритмам методу розгалуженого вибору ознак (РВО) [201-203]. Важливими елементами методів РВО є публікації [204-206], де

пропонується та аналізується схема побудови ЛДК на основі алгоритму логічного дерева з покроковою оцінкою важливості дискретних ознак за даними НВ. У роботі [207] пропонується модифікований алгоритм РВО з одноразовою оцінкою набору ознак, а в [112] – алгоритм генерації набору (множини) випадкових логічних дерев класифікації з фінальним етапом відбору оптимального. Зрозуміло, що результуюче правило класифікації, яке побудоване певним способом (на основі методу або алгоритмом РВО), має структуру ЛДК. Таке дерево складається з вершин (елементарних ознак об‘єктів), які групуються за ярусами (рівнями) і отримані на певному кроці (етапі) побудови ЛДК (алгоритму РВО).

Так, принциповим моментом є те, що на відміну від існуючих методів класифікації, головною особливістю моделей розпізнавання (які побудовані за методикою РВО) є те, що важливість окремих ознак (груп ознак та їх наборів) розраховується відносно функції, яка задає розбиття об‘єктів на класи (ФР), а числова величина важливості (інформативності) ознак та їх наборів характеризує собою помилку класифікації об‘єктів за класами.

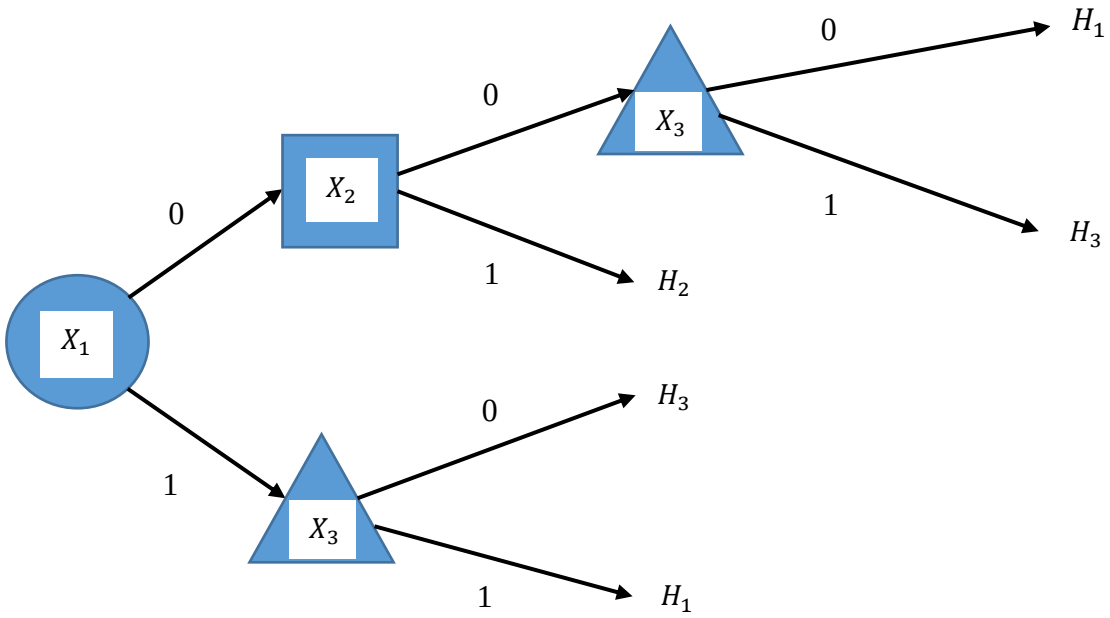


Рис. 1.3. Загальна схема логічного дерева побудованого за алгоритмом з покроковою оцінкою важливості ознак (метод РВО).

Оскільки головну ідею методів та алгоритмів РВО можна визначити як оптимальну апроксимацію деякої початкової НВ набором елементарних ознак (атрибутів об‘єкту), то на перший план виходить їх центральна проблема – питання вибору ефективного критерію розгалуження (відбору вершин, атрибутів, ознак дискретних об‘єктів). Саме цим принциповим задачам присвячені роботи [83,100], де піднімаються питання якісної оцінки окремих дискретних ознак, їх наборів та фіксованих сполучень, що дає можливість запровадити ефективний механізм реалізації розгалуження.

Приклад ЛДК, яке побудовано за даними НВ на основі методу РВО, зображено на рис. 1.3 (алгоритм з покроковою оцінкою важливості ознак). Зауважимо, що дане дерево не є регулярним, і на різних ярусах розташовані різні мітки залежно від їх оцінки інформативності на кожному кроці. Зокрема, на першому кроці з найбільшою інформативністю обирається атрибут x_1, на другому – x_2, на третьому – x_3, на четвертому– знову x_3 (послідовність – зліва направо, зверху вниз). Причому на кожному кроці відбувається фактичне розбиття початкової НВ на підмножини з наступним етапом відбору найбільш якісної, інформативної ознаки (генерації вершини), а загальна схема побудови ЛДК актуальна не тільки для бінарного випадку атрибутів. Відмітимо, що така особливість методу РВО щодо роботи з початкової НВ (багатокроковим розбиттям масиву даних) є досить ресурсоємною та затратною за часом генерації результуючої схеми процедурою. Одним із можливих варіантів подолання цих недоліків є модифікація методу щодо одноразової оцінки вибірки та побудови структури ЛДК, виходячи лише з неї (без покрокової оцінки розбиттів множини НВ).

Отримана структура ЛДК характеризується компактністю з одного боку та нерівномірністю заповнення (розрядженістю) ярусів з іншого боку порівняно з регулярними деревами (алгоритмом з одноразовою оцінкою важливості ознак) [67]. Відмітимо, що важливими питаннями залишаються питання збіжності процесу побудови ЛДК за методами РВО та питання вибору критерію зупинки процесу синтезу логічного дерева (наприклад, обмеження за

глибиною або складністю дерева, обмеження за точністю або кількістю помилок структури, що будується). Вирішення даного питання в зрізі обмеженого (модифікованого) методу побудови АДК буде представлено в цьому дослідженні далі.

Звернемо увагу, що концепції логічних дерев не суперечить можливість в якості ознак (вершин) ЛДК використовувати не тільки окремі атрибути (ознаки) об‘єктів їх сполучення (ідея узагальненої ознаки розглядалась в роботі [208]) та набори, якщо не розглядати в якості розгалужень атрибути об‘єктів (ознаки), а відбирати окремі незалежні алгоритми розпізнавання (оцінені за даними НВ), то на виході буде отримано нову структуру – АДК. Саме ідеї методу побудови алгоритмічних дерев класифікації і присвячено дане дослідження, а відповідний метод побудови АДК буде представлений у наступних розділах.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) Дослідження деревоподібних моделей класифікації та розпізнавання залишається актуальною задачею теорії штучного інтелекту, хоча головна увага зараз приділяється теорії нейромережевого розпізнавання.

2) Представлення масивів даних (дискретної інформації) великого об‘єму у вигляді структури логічного дерева має свої суттєві переваги щодо економічного опису даних та ефективних механізмів роботи з ними.

3) Галузь застосування концепції ЛДК може бути зведена до задач опису структур даних, задач розпізнання та класифікації, задач регресії.

4) Важливою особливістю ЛДК є гнучкість, тобто здатність ЛДК послідовно враховувати та досліджувати ефект впливу окремих змінних, атрибутів структури.

5) Структура логічного дерева являє собою зв‘язану множину гілок та вузлів, причому на гілках дерева розташовуються деякі мітки (атрибути, значення ознак), від яких залежить цільова функція (у випадку ЛДК – функція

розпізнавання), а у вузлах (вершинах) знаходяться значення ФР або розширені атрибути переходів.

6) На сьогоднішній день існує значна кількість алгоритмів, які реалізують концепцію дерев рішень, але найбільшого вживання та розповсюдження отримали алгоритми CART та C4.5/C5.0.

7) При побудові ЛДК центральними питаннями залишаються питання вибору критерію розгалуження, критерію зупинки навчання та критерію відкидання гілок логічного дерева.

8) Можливим раціональним розвитком методів та підходів ЛДК може бути використання ідеї алгоритмічних дерев (саме цьому питанню буде приділено значну увагу в даному дослідженні).

9) Незважаючи на певну проблематику, яка виникає при побудові та використанні концепції ЛДК в загальному сенсі, вона залишається потужним інструментом теорії штучного інтелекту.

1.3. Вибір напрямків та структурно-логічна схема досліджень

Враховуючи принципові особливості загальної концепції, методів та схем дерев рішень (моделей дерев класифікації), особливостей постановки задачі розпізнавання образів, визначимо основні напрямки дослідження. Підкреслимо, що вихідною точкою досліджень є те, що існуючі підходи та методи аналізу, класифікації, представлення великих масивів різнотипних даних мають істотні недоліки щодо універсальності, великих апаратних витрат на етап навчання, структурної складності результуючих моделей, відсутності простих механізмів ефективного регулювання точності побудованих класифікаторів [209-223].

На основі проведеного аналізу поточного стану концепції дерев рішень, основної проблематики методів та схем синтезу дерев класифікації, постановки задачі розпізнавання на основі поетапної апроксимації масивів початкових даних набором функції класифікації випливає науково-прикладна проблема, яка полягає в суттєвих обмеженнях існуючих підходів та методів щодо прикладної універсальності, можливості роботи в довільних шкалах початкових даних, значних апаратних витратах на процедуру навчання, обмежень на точність побудованих моделей, неможливості комплексного (модульного) використання накопиченого потенціалу методів та алгоритмів теорії розпізнавання. В зв‘язку з цим виникає необхідність розробки простих та ефективних моделей, методів, практичного (програмного) інструментарію розпізнавання дискретних об‘єктів на основі принципів концепції АДК.

Представлене дослідження спрямовано на отримання обґрунтованих моделей структур дерев рішень на основі ідей, принципів та концепцій, моделей та методів, програмного інструментарію, які мають актуальне практичне застосування. Зокрема, в дисертаційному дослідженні показано, що моделі дерев класифікації можна будувати на основі різнотипних алгоритмів розпізнавання, які можуть навіть не бути пов‘язані спільною концепцією, використовуючи їх в якості вершин структури АДК, об‘єднувати в яруси та

рівні дерева класифікації, використовуючи модульну концепцію, будувати новий алгоритм розпізнавання на основі відомих.

Зважаючи на вище зазначене можна запропонувати наступну структурно-логічну схему досліджень у вигляді послідовності таких етапів:

Етап I. Представлення та обґрунтування напрямку досліджень.

1) Базовий аналіз аспектів задачі розпізнавання дискретних об‘єктів у межах концепції методів дерев класифікації (дерев рішень).

2) Дослідження сучасного стану методів дерев рішень у розрізі їх проблематики.

3) Обґрунтування доцільності застосування методів дерев класифікації для розв‘язку широкого спектру задач аналізу великих даних.

4) Загальний аналіз моделей, методів та схем концепції дерев рішень, які складають базис досліджень.

Етап II. Загальне формування мети та набору задач дослідження.

1) Дослідження структур дерев класифікації (конструкцій ЛДК) як складової концепції дерев рішень.

2) Формалізація задачі побудови класифікатора у вигляді структури дерева розпізнавання (конструкції ЛДК).

3) Визначення етапів процесу побудови структур дерев класифікації різних типів.

Етап III. Визначення основних аспектів набору задач дослідження.

1) Формування основних проблемних компонентів задач класифікації на основі концепції дерев рішень.

2) Визначення та аналіз процесу побудови структур ЛДК/АДК в методах дерев класифікації.

3) Аналіз моделей систем розпізнавання на основі граф – схемного представлення класифікаторів.

4) Обґрунтування використання концепцій дерев класифікації для широкого спектру задач теорії штучного інтелекту.

Етап IV. Аналіз структур логічних дерев та задач, які випливають з нього.

1) Аналіз моделей (конструкцій) ЛДК/АДК щодо їх структурної складності.

2) Визначення загальної схеми оптимізації (обрізки) побудованих моделей дерев класифікації шляхом перестановки їх структурних елементів.

3) Розробка оптимальних схем розташування атрибутів (змінних) у структурах логічних дерев класифікації.

4) Класифікація типів конструкцій структур логічних дерев для задачі їх фінальної оптимізації.

Етап V. Розробка методів дерев класифікації у вигляді структур ЛДК/АДК.

1) Розробка методу побудови моделі ЛДК на основі покрокової селекції ранжованих наборів елементарних ознак.

2) Розробка загальної концепції побудови структур АДК на основі апроксимації даних НВ набором незалежних алгоритмів розпізнавання та представлення побудованого класифікатора у вигляді деревоподібної схеми.

3) Розробка методів побудови моделей АДК різних типів.

Етап VI. Розробка обмежених методів побудови дерев класифікації.

1) Розробка методів побудови дерев класифікації в умовах обмеження апаратних ресурсів інформаційної системи.

2) Розробка методу побудови моделей ЛДК/АДК для випадку обмеженого методу дерева класифікації.

3) Оцінка максимальної складності структур ЛДК/АДК для випадку слабкого та сильного розділення класів початкової НВ.

Етап VII. Розробка програмного забезпечення, прикладне застосування та верифікація результатів.

1) Розробка загальної схеми програмної побудови структури ЛДК.

2) Особливості програмної реалізації моделей ЛДК на основі селекції наборів елементарних ознак.

3) Розробка програмної системи побудови дерев класифікації DeTree.

4) Модель АДК для задачі класифікації гідрографічних даних.

Висновки до розділу 1

1. Всебічний аналіз та дослідження деревоподібних моделей класифікації та розпізнавання залишається актуальною задачею теорії штучного інтелекту, хоча головна увага зараз приділяється теорії нейромережевого розпізнавання. Зокрема представлення масивів даних (дискретної інформації) великого об‘єму у вигляді структури логічного дерева має свої суттєві переваги щодо економічного опису даних та ефективних механізмів роботи з ними, а галузь застосування концепції дерев класифікації може бути зведена до задач опису структур даних, задач розпізнання та класифікації, задач регресії. Відмітимо, що при побудові структур дерев класифікації центральними питаннями залишаються питання вибору критерію розгалуження, критерію зупинки навчання та критерію відкидання гілок логічного дерева – попри певну проблематику, яка виникає при побудові та використанні концепції дерев класифікації, в загальному сенсі вона залишається потужним інструментом теорії штучного інтелекту.

2. У розділі наведено загальну постановку задачі машинного навчання в довільних шкалах, яка зводиться до навчання СР обраховувати деяку функцію ФР (ФР визначена на початковій множині та приймає скінчену кількість значень), причому на даному етапі СР приймає послідовність навчальних пар НВ та на основі цієї інформації будує схему обчислення ФР або її наближення. Виділено основні питання, які виникають на етапі навчання СР: економія пам‘яті, швидкодія навчання, питання побудови схеми для обчислення ФР, оптимальної за деякими важливими параметрами (об‘єм пам‘яті, швидкодія, надійність і так далі).

3. Проаналізовано сучасний стан досліджень концепції дерев рішень, висвітлено проблематику даної концепції, визначено шляхи подальшого вдосконалення та перспективних досліджень моделей та методів дерев класифікації.

РОЗДІЛ 2
БАЗОВІ МЕТОДИ ГРАФ – СХЕМНОГО ПРЕДСТАВЛЕННЯ КЛАСИФІКАТОРІВ

2.1. Схеми синтезу логічних дерев класифікації

На сьогодні відомі різні схеми побудови ЛДК [90, 119, 224-230]. Проте всі вони, як правило, зводяться до побудови одного дерева класифікації за даними НВ. Відмітимо також, що в літературі дуже мало алгоритмів побудови ЛДК для НВ великого об’єму. Зрозуміло, що це має під собою об‘єктивні фактори, пов‘язані з особливостями генерації таких складних структур, методиками роботи з ними та зберіганням [231-243]. Навіть використовуючи інструментарій Java або C#, необхідно забезпечити реалізацію спеціальних структур даних для роботи з логічними деревами, а готові бібліотеки класів (LightGBM, XGBoost) [244-270] хоча і близькі ідеологічно (схема логічного дерева), але не дозволяють реалізувати концепцію алгоритмічного дерева класифікації (АДК), яке складається з набору вершин – різнотипних автономних алгоритмів класифікації. Проте, основним недоліком при побудові ЛДК є відсутність алгоритмів та методів, котрі б давали змогу одноманітно описувати різні алгоритми у вигляді ЛДК.

У даному розділі роботи зупинимось на описі алгоритму побудови ЛДК для НВ великого об’єму та покажемо шлях щодо можливості одноманітного опису одного класу ЛДК. Основні переваги даного алгоритму побудови ЛДК полягають в наступному:

1. Метод побудови ЛДК не потребує одночасного запам‘ятовування всіх даних НВ: можливе поетапне введення даних (по векторам) і навіть окремих значень ознак об‘єктів. Після введення вектор (об‘єкт, який подається на вхід системі) використовується для побудови ЛДК і надалі в пам’яті комп‘ютера не зберігається. Таке введення інформації та специфіка алгоритму значно економлять пам’ять машини і дають можливість розв’язувати задачі розпізнавання з НВ великої потужності (об‘єму).

2. Алгоритм забезпечує безпомилкову класифікацію НВ у разі початкової відмінності між об’єктами різних класів.

3. Робота алгоритму не залежить від кількості образів (класів) у НВ. Зокрема алгоритм можна застосовувати при наявності в НВ тільки одного образу. Відсутність такої можливості, в деяких випадках, представляє собою негативний фактор.

4. Кількість помилок, що допускаються ЛДК на фіксованій НВ не збільшується, якщо збільшується об’єм НВ. Ця вимога дозволяє визначити збіжність даного алгоритму.

5. Алгоритм включає в себе просту схему донавчання та усунення знайдених помилок розпізнавання.

6. Алгоритм дає можливість сформулювати підхід, що дозволяє одноманітно описувати достатньо широкий клас алгоритмів у вигляді ЛДК.

2.1.1. Загальна схема побудови структури ЛДК

Нехай НВ задана у вигляді деякої таблиці (матриці):

$$\begin{matrix} x_{1,1}, x_{1,2}, \dots \dots \dots \dots \dots \dots \dots \dots \dots, x_{1,n} \\ \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ x_{m,1}, x_{m,2}, \dots \dots \dots \dots \dots \dots \dots \dots \dots, x_{m,n} \\ x_{m+1,1}, x_{m+1,2}, \dots \dots \dots \dots \dots \dots, x_{m+1,n} \\ \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ x_{k,1}, x_{k,2}, \dots \dots \dots \dots \dots \dots \dots \dots \dots, x_{k,n} \end{matrix} \qquad (2.1)$$

$\{x_{i,j}\}$ – значення j – вої ознаки i – го об’єкта НВ ($i = 1, \dots k; j = 1, \dots, n$). Для спрощення викладання вважаємо, що $x_{i,j} \in \{0,1\}$, m – рядків НВ характеризує клас Ω_1, а інші – Ω_2.

Побудуємо за НВ вигляду (2.1) ЛДК. У першу вершину логічного дерева ставимо ознаку x_1 і від неї будуємо повний шлях, який відповідає набору $x_{1,1}, x_{1,2}, \dots, x_{1,n}$, тобто з вершини з ознакою x_j виходить стрілка, що входить у вершину з ознакою x_{j+1} (номер стрілки залежить від значення, яке приймає $x_{1,j}$, у кінцевій вершині ставимо значення f_R (функції розпізнавання).

Для наступного об’єкта $w_2 = (x_{2,1}, \dots, x_{2,n})$ при $w_1 \neq w_2$ ЛДК змінюється наступним чином: при $x_{2,j} \neq x_{1,j}$ з вершини з ознакою x_j проводимо другу

стрілку, яка відповідає значенню $x_{2,j}$ (для зручності стрілки з номером 0 розташовуємо з лівого боку, а 1 – з правого), та в кінці стрілки ставимо вершину з ознакою x_{j+1}. На наступному кроці з цієї вершини проводимо стрілку, відповідну значенню ознаки x_{j+1} і так далі. Зрозуміло, що в кінцеву вершину ставимо значення $f_R(w_2)$ (рис. 2.1).

Аналогічним чином добудовуємо ЛДК для всіх інших наборів (2.1). Якщо на одному ЛДК зустрічаються два або декілька однакових наборів із різних класів (різних значень функції розпізнавання), то кількість значень функції f_R (відповідних цим наборам) фіксуємо в кінцевій вершині і остаточно записуємо те значення функції f_R, для якої кількість наборів буде максимальною.

Якщо кількість значень функції f_R однакове в деякій кінцевій вершині, то перевагу віддаємо образу, потужність якого менше. Після цього побудову ЛДК за даними НВ будемо рахувати закінченим.

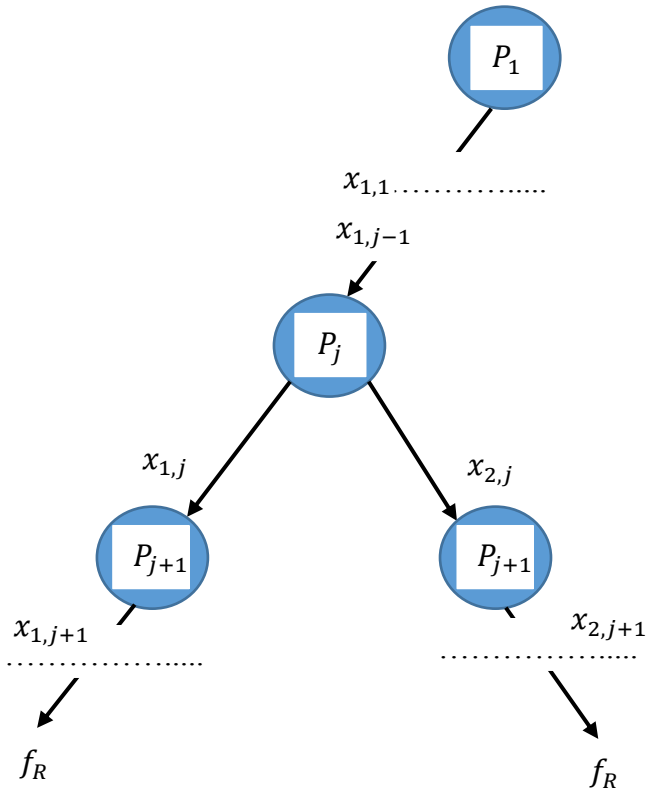


Рис. 2.1. Загальна схема побудови ЛДК.

Задача розпізнавання образів полягає в тому, щоб побудоване ЛДК класифікувало всі набори ТВ, які не входять в НВ (зауважимо, що набори

об‘єктів НВ, логічне дерево розпізнає без помилок за рахунок їх фіксації в структурі самого дерева). Після побудови ЛДК за даним алгоритмом будемо мати структуру, де не з усіх вершин виходять дві стрілки. Це дозволяє провести просту мінімізацію – якщо з вершини виходить одна стрілка, то її разом зі стрілкою видаляємо зі структури ЛДК. Мінімізоване таким чином ЛДК позначимо через ЛДК*. Відмітимо наступні загальні властивості ЛДК:

1) складність (під складністю дерева розуміємо загальну кількість вершин у ЛДК [67,70-72,75]) ЛДК* менше ніж складність початкового ЛДК (при мінімізації відбувається відсів найменш важливої інформації);

2) ЛДК*, так само як і початкове ЛДК, безпомилково класифікує об’єкти НВ та забезпечує екстраполяцію наборів ТВ, котрі не зустрічались в масиві початкових даних НВ.

Кожному набору Δ з ТВ відповідає своя $f_R(\tau_\Delta), \Delta \in (0,1,\dots,n)$, кінцевої вершини на ЛДК, яка визначає приналежність даного набору до того чи іншого класу. Зрозуміло, що різні алгоритми дають різні результати на фіксованій ТВ – можлива ситуація, коли деякий об’єкт буде віднесений до різних образів (класів). Причина цього полягає в тому, що задача розпізнавання не має єдиного розв’язку і кожний з алгоритмів дає свій. Задача про існування єдиного розв’язку розглядалася, наприклад у роботі [271]. Також слід відмітити наступний момент – структура логічного дерева, яке побудоване деяким алгоритмом розпізнавання (наприклад алгоритмом із покроковою оцінкою важливості ознак) буде відрізнятись від ЛДК, побудованого іншим алгоритмом або методом (наприклад алгоритмом із одноразовою оцінкою важливості ознак) за даними однієї фіксованої НВ [206].

Розглянемо якість розв’язку задачі РО на простому прикладі. Нехай, маємо справу з НВ, яка представлена в табл. 2.1 об‘ємом 5 наборів. Значення $f_R(P_1, P_2, P_3)$ вказує на приналежність наборів НВ до того чи іншому класу. Потрібно знайти функцію $f_R^*(P_1, P_2, P_3)$, яка забезпечить апроксимацію $f_R(P_1, P_2, P_3)$. Очевидно, що існує 8 таких різних функцій. Якщо $P_j \in \{0,1\}, j = 1, \dots n, n$ – кількість ознак, m – кількість наборів НВ, то кількість

різних розв‘язків, які не входять до наборів НВ дорівнює $L = 2^{2^n - m}$. Якщо $P_j \in \{0,1, \dots, k-1\}$, то $L = k^{k^n - m}$.

Таблиця 2.1. Таблична форма не повністю визначеної $f(P_1, P_2, P_3)$.

Номер набору	P_1	P_2	P_3	$f_R(P_1, P_2, P_3)$
1	0	0	0	0
2	0	0	1	0
3	0	1	0	1
4	0	1	1	1
5	1	0	0	1
6	1	0	1	?
7	1	1	0	?
8	1	1	1	?

Відмітимо, що якщо задано m різних наборів НВ, то кількість помилок не може бути більше ніж $2^n - (m-1)$. Отже, лише за даними НВ неможливо зробити висновок, який з алгоритмів класифікації дає кращий результат для фіксованої задачі розпізнавання. Беручи до уваги вищезазначене, неможливо класифікувати алгоритми РО лише за вузькою специфікою області застосування. Враховуючи даний факт, кожний метод розпізнавання повинен включати в себе механізм (алгоритм), котрий усуває знайдені помилки, та забезпечувати необхідну гнучкість щодо області застосування. Запропонуємо простий та ефективний алгоритм донавчання та виправлення помилок (алгоритм ДВП) в ЛДК.

2.1.2. Загальна схема донавчання та виправлення помилок в ЛДК

Припустимо, що маємо деяке ЛДК*, яке побудовано на основі фіксованої НВ (розмірність об‘єктів якої – n ознак). Нехай на деякому наборі - $\tau = (x_{i,1}, \dots, x_{i,n})$ відбувається помилка. Тоді результуюче ЛДК будемо міняти наступним чином. У кінцеву вершину, що знаходиться на шляху $(x_{i,1}, \dots, x_{i,n})$, ставимо одну з цих ознак та добудовуємо ЛДК так, щоби гілка цих ознак відповідала шляху, в кінцеву вершину котрого ставимо номер класу відповідний набору $(x_{i,1}, \dots, x_{i,n})$.

Для вершин ЛДК з однією стрілкою на шляху $(x_{i,1}, \dots, x_{i,n})$ додаємо стрілки, яких бракує, та в кінці їх записуємо те значення $f_R(\tau_\Delta)$, котре було в

кінцевій вершині до виявлення помилки. Зрозуміло, що при такому виправленні помилок кількість вершин результуючого ЛДК дещо збільшується, але всі набори НВ розпізнаються безпомилково (на наборі $(x_{i,1}, \dots, x_{i,n})$ помилок не виникає). Зокрема, слід зауважити, що така схема виправлення помилок актуальна для ситуації, коли ми не маємо справу з помилками першого роду в ЛДК – існування певних об‘єктів у НВ та ТВ, які співпадають за ознаками, але відрізняються класами належності. В такому випадку слід знову ж таки застосувати схему з голосуванням за кількістю належності класам та оцінкою їх потужностей. Крім того, слід зауважити, що помилки першого роду зазвичай пов‘язані з помилками, невдалого кодуванням $I(l)$ та мають усуватися саме на цьому етапі (за рахунок вибору правильного та ефективного кодування, правильного вибору простору ознак) або на етапі початкової верифікації НВ та ТВ при перевірці на несуперечливість. Розглянемо простий приклад застосування описаного вище алгоритму. Нехай НВ задана у вигляді наборів об‘єктів $w(P_1, \dots, P_n)$ загальною кількістю $m = 5$ та розмірністю ознак $n = 3$ (табл. 2.2), а відповідно ТВ – (табл. 2.3). Побудуємо за даними НВ результуюче ЛДК та перевіримо його роботу (включаючи алгоритм донавчання та виправлення помилок) на ТВ. Усі етапи побудови результуючого ЛДК за даними НВ показані на рис. 2.2. Зауважимо, що остаточне мінімізоване ЛДК буде отримано на останньому етапі (етап 7). Причому сама побудова ЛДК за даними НВ продемонстрована з першого по п‘ятий етап (послідовно з надходженням об‘єктів НВ). На етапі 6 показано вже мінімізоване дерево після роботи за даними НВ, де відкинуті непотрібні вершини.

Таблиця 2.2. Таблична форма НВ об‘ємом $m = 5$.

Номер набору	P_1	P_2	P_3	$f_R(P_1, P_2, P_3)$
1	0	0	0	1
2	0	0	1	0
3	0	1	1	0
4	1	0	1	1
5	1	1	0	0

На наступному кроці перевіримо роботу ЛДК на даних ТВ:

1) 1-й і 2-й набори ТВ ЛДК розпізнає правильно, оскільки вони входили (співпадали) в дані НВ;

2) 3-й набір відноситься до класу 0 (насправді він з класу 1). Після застосування алгоритму донавчання та виправлення помилок ЛДК буде мати вигляд, показаний на (Рис 2.4 – етап 7). Отже, відбулося донавчання на n −вому наборі ТВ. Інші набори ТВ ЛДК розпізнає без помилок.

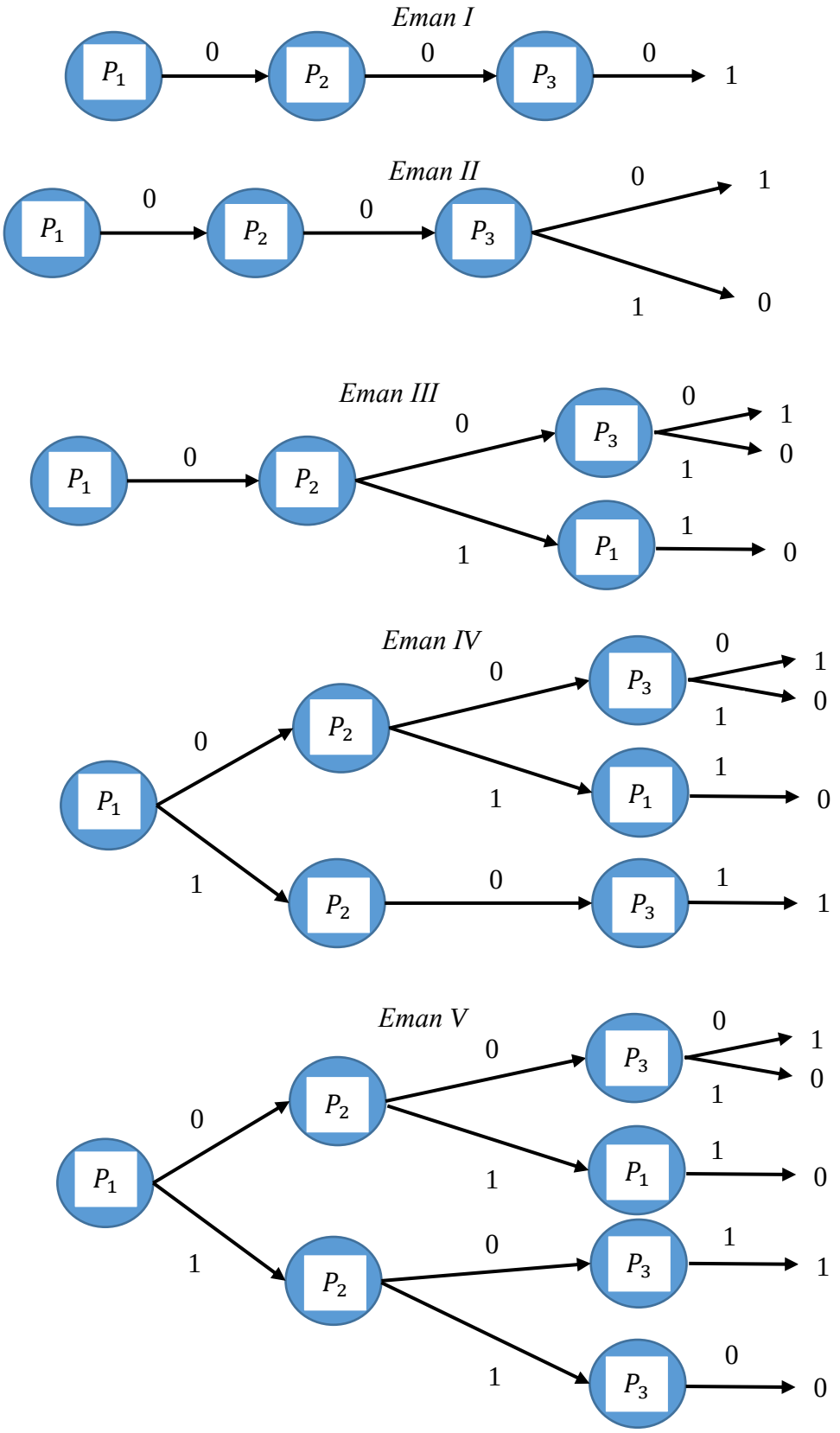
Зауважимо лише, що на етапі 7 показано все остаточне ЛДК після закінчення роботи на першому етапі алгоритму синтезу ЛДК, а на другому етапі алгоритму донавчання та виправлення помилок. У певному сенсі дане ЛДК буде забезпечувати просту та ефективну апроксимацію даних як навчальної, так і тестової вибірок.

Очевидно, що за допомогою даного алгоритму, при розв'язуванні практичних задач розпізнавання, не завжди досягаються прийнятні результати. Даний алгоритм можна звести до вигляду, що дозволяє описувати достатньо широкий клас методів розпізнавання у вигляді ЛДК. Зауважимо, що ознаки в процесі побудови ЛДК вибиралися послідовно та впорядковано –одна за одною. Якщо відмовитися від цієї умови та вибрати ознаки в процесі побудови ЛДК довільним чином (наприклад за допомогою генератора псевдовипадкових чисел), то в результаті кожен раз при застосуванні цього алгоритму буде одержуватись інше за структурою ЛДК.

Таблиця 2.3. Таблична форма ТВ об'ємом $m = 8$.

Номер набору	P_1	P_2	P_3	$f_R(P_1, P_2, P_3)$
1	0	0	1	0
2	0	1	1	0
3	0	1	0	1
4	0	1	0	1
5	1	0	0	1
6	1	1	1	0
7	1	0	0	1
8	1	1	1	0

Вибір того чи іншого ЛДК при розв'язуванні практичних задач може бути різноманітним. Основні дослідження та практичні реалізації з цього питання будуть викладені в наступних розділах даного дослідження.



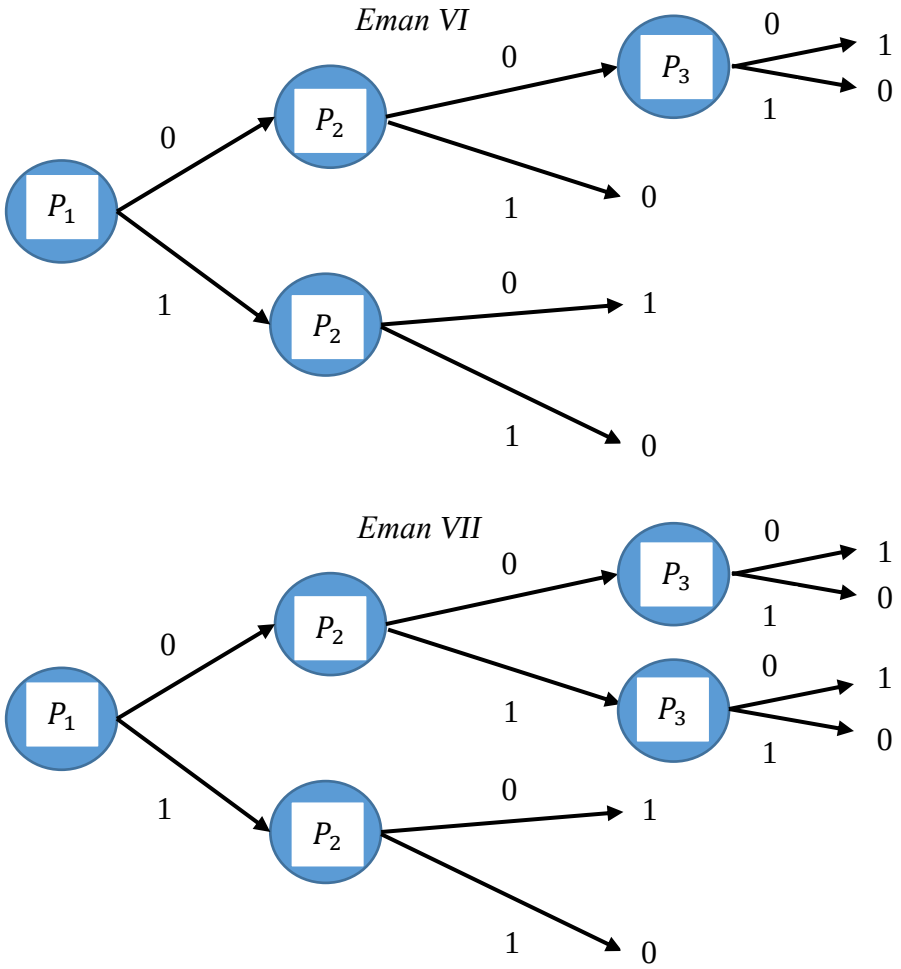


Рис. 2.2. Етапи побудови результуючого ЛДК.

Результати застосування описаного вище алгоритму та порівняння його ефективності відносно інших алгоритмів на реальних задачах за масивами даних геології, геохімії та геофізики відображені (для спрощення візьмемо добре відомі тестові приклади з [272]) у табл. 2.4. Для кожного з 14 алгоритмів (на кожній з 3-х задач) приведено кількість правильних відповідей (у %), одержаних на відповідних НВ.

Зауважимо, що для порівняльної таблиці частина алгоритмічних реалізацій була взята з програмного комплексу «Оріон ІІІ» [94] – наприклад алгоритм потенціальних функцій та алгоритм розгалуженого вибору ознак з покроковою оцінкою їх важливості [205]. Зважаючи на отримані дані відносно проценту успішної класифікації, можна зробити висновок про в цілому доволі високу ефективність у цьому плані алгоритму ДВП логічного дерева (не

враховується економічність, інформаційна ємність логічного дерева та швидкодія результуючої схеми класифікації).

Таблиця 2.4. Порівняльна таблиця алгоритмів.

АЛГОРИТМ		ЗАДАЧА		
		№1	№2	№3
№1	Лінійна розв‘язуюча функція «Кульбаба»	64	64	55
№2	Нелінійна розв‘язуюча функція «Едельвейс»	76	68	55
№3	Алгоритм «Кора-3» (алгоритм М. Вайнцвайга та М. Бонгарда) [273]	32	68	65
№4	Алгоритм «Тест-2»	72	65	70
№5	Алгоритм «Ентропія–1»	73	64	72
№6	Алгоритм «Ентропія–2»	56	56	25
№7	Алгоритм «Ентропія–3»	56	76	15
№8	Алгоритм «Голотип»	44	52	50
№9	Алгоритм на основі потенціальних функцій. (Для тесту взята реалізація ПК «Оріон ІІІ») [94]	44	60	50
№10	Алгоритм «Геолог-1»	72	36	55
№11	Алгоритм «Південь»	76	68	55
№12	Алгоритм розгалуженого вибору ознак з покроковою оцінкою важливості ознак (алгоритм методу РВО в редакції Ю. Василенка) [205]	56	68	70
№13	Алгоритм побудови повного ЛДК [67]	74	70	65
№14	Запропонований вище алгоритм ДВП у ЛДК [90]	74	80	72

Як підсумок, можна зробити наступні висновки щодо побудови ЛДК:

1) ЛДК забезпечує покриття масиву навчальної інформації за рахунок фіксації об‘єктів вибірки в своїй структурі. Причому такий підхід зберігання інформації забезпечує як механізм донавчання (розширення), так і виправлення помилок.

2) На сьогодні у спеціалізованій літературі відсутній опис алгоритмів та методів, які би дозволили будувати логічні дерева на основі масивів даних великої розмірності.

3) Застосування ЛДК у задачах РО дозволяє просто та компактно описувати ФР, що позитивно впливає на складність та швидкодію результуючої СР.

2.2. Питання особливостей випадкових логічних дерев класифікації

У дисертаційному дослідженні пропонується підхід, що дозволяє одноманітно, за допомогою єдиною методики, описувати та автоматично створювати (забезпечувати програмну генерацію) достатньо широкі класи алгоритмів розпізнавання дискретних об‘єктів на основі ЛДК. Розглядаються питання формування колективу алгоритмів розпізнавання дискретних об‘єктів та оцінки їх компетентності у процесі прийняття рішень. Розглянемо особливості випадкових дерев класифікації, тобто ЛДК, в яких вибір (генерація) вершин на довільному етапі побудови дерева відбувається випадковим чином.

Запропонуємо відповідну постановку задачі розпізнавання, яка буде полягати в наступному: нехай задано набори об‘єктів НВ, ТВ, якісну оцінку кожної ознаки (ціна, якість), мінімальний час, необхідний для проведення класифікації, максимально допустиму складність правил класифікації (колективу правил класифікації). У цих умовах потрібно знайти (синтезувати) таке правило класифікації (колективу правил класифікації) у вигляді ЛДК, побудованого за даними НВ, котре дає мінімальну кількість помилок на НВ в умовах дії зовнішнього середовища та здатне швидко адаптуватися до цих умов. Відмітимо, що в цій задачі ніякі обмеження на НВ та ТВ не накладається, тобто ознаки можуть мати довільну природу (бути різнотипними), об’єм НВ та ТВ буде довільний і так далі.

2.2.1. Питання побудови випадкових дерев класифікації

Далі, залежно від природи (типу) ознак, розглядаються два шляхи розв’язку даної задачі:

1) випадок, коли ознаки приймають бінарні значення;

2) випадок, коли ознаки мають різнотипну природу.

Відмітимо, що другий випадок можна звести до першого, застосувавши спеціальний алгоритм кодування. Зауважимо, що при цьому часто виникають досить великі втрати інформації та інформаційні спотворення розбиття класів НВ, що негативно впливає на якість розпізнавання. Зрозуміло, що взагалі

якісне та правильне кодування початкової НВ є нетривіальною задачею, від якої безпосередньо залежать наступні етапи побудови результуючого ЛДК.

Тому розглянемо новий тип алгоритмів кодування початкової інформації $I(l)$, що дають змогу закодувати НВ у такий спосіб: якщо класи у вихідному описі не перетинаються, то і після кодування вони перетинатися не будуть (умова несуперечливості при кодуванні). При цьому відбувається часткове виявлення певних властивостей (логічних закономірностей) об'єктів початкової вибірки.

Запропонований тип алгоритмів кодування базується на наступній ідеї: оскільки об'єкти НВ є деякими точками $n-$ вимірного простору, то завжди, при умові вихідної відмінності між класами, ми можемо ввести $n-$ вимірний простір, обмежений можливими значеннями ознак, площинами, на $n-$ вимірні гіперпаралелепіпеди (гіперкуби) так, щоб точки (об'єкти) з різних класів не попадали в один і той самий гіперпаралелепіпед (тобто провести геометричне розділення класів) [229]. На наступному етапі кожному гіперпаралелепіпеду в $n-$ вимірному просторі можна поставити у відповідність певний двозначний код. Ці двозначні коди, залежно від того, куди попадають об'єкти НВ та ТВ, визначають набір бінарних ознак задачі розпізнавання образів. Замість гіперпаралелепіпедів можна взяти гіпереліпси або гіперсфери – відповідні ефективні алгоритмічні реалізації можна взяти з роботи [94], де вони пройшли необхідну прикладну апробацію.

Розглянемо загальну методику побудови різних випадкових ЛДК та їхні переваги перед відомими алгоритмами класифікації у вигляді ЛДК.

При побудові правил класифікації (схем) у вигляді ЛДК за допомогою відомих алгоритмів, зрештою, як правило, буде отримано одне фіксоване ЛДК. Вибір ознак в ту або іншу вершину ЛДК відбувається цілеспрямовано, наприклад використовуються ті або інші критерії (функціональні оцінки) важливості ознак [83]. Проте, найважливіші ознаки, визначені за даними НВ, можуть виявитися не такими важливими в реальному процесі та навпаки.

Зокрема в роботі [205] розглянуто дві алгоритмічні реалізації побудови ЛДК на основі методу розгалуженого вибору ознак. Причому вибір ознак на кожному кроці здійснювався в першому випадку на основі функціональної оцінки якості ознак на кожному кроці, а в другому випаду – для економії ресурсів та зменшення часу генерації ЛДК, лише на початку роботи алгоритму з відбором ознак від найбільш до найменш інформативних. У нашому випадку пропонується алгоритм (схема) побудови випадкового ЛДК, тобто дерева, ознаки у вершинах якого в процесі його побудови відбираються випадковим способом (випадковим програмним генератором – PRG). Запропонуємо одну з можливих алгоритмічних реалізацій побудови випадкового ЛДК.

2.2.2. Загальна схема побудови випадкового ЛДК

Крок 1. Нехай задано НВ об‘ємом (потужністю) m об‘єктів розмірності (ознаковим простором) n відомої класифікації. Необхідно за цією початковою інформацією та за допомогою деякого випадкового програмного генератора PRG побувати ЛДК, яке дозволяє однозначно класифікувати об‘єкти НВ. На першому етапі, використовуючи PRG, вибирається початкова вершина ЛДК та будуються відповідні стрілки (дуги), які виходять із вершини даного дерева. Лічильнику об‘єктів НВ присвоюється одиниця. В змінну структури дерева – ідентифікатор ЛДК, що будується записується перша згенерована вершина.

Крок 2. Відповідно до значення лічильника НВ аналізується поточний об‘єкт – проставляються відповідні значення ознак (атрибутів) у гілках дерева за умови, що кількість ярусів не більша кількості ознак об‘єкта, в іншому разі в кінцевій вершині проставляється значення функції розпізнавання для поточного об‘єкту. Далі перевіряється значення лічильника НВ відносно m, якщо співпадають, то робота алгоритму передається на *Крок 4*.

Крок 3. За допомогою PRG генерується наступна вершина ЛДК за умови, що кількість ярусів не більша кількості ознак об‘єкта, яка також послідовно дописується в змінну – ідентифікатор дерева, та передається керування на *Крок 2*. Якщо кількість ярусів ЛДК дорівнює n, то проводиться інкремент лічильника НВ та передається керування на *Крок 2*.

Крок 4. Робота алгоритму завершено, ЛДК побудовано, а в ідентифікаторі дерева зберігається послідовність вершин, які однозначно характеризують дане синтезоване дерево.

Відмітимо, що цей алгоритм досить легко реалізується програмним шляхом та за його допомогою для кожної конкретної задачі розпізнавання ми можемо побудувати деяку множину ЛДК та серед них вибрати саме те дерево (колектив ЛДК), котре задовольнить поставленим вимогам.

Відмітимо, що в пам'яті комп'ютера ми не зберігаємо побудовані випадкові ЛДК, а тільки деякий вектор-характеристику (ідентифікатор) цього ЛДК, наприклад число, знаючи яке ми знов можемо побудувати це ЛДК та деякі важливі його характеристики (ефективність на ТВ, складність, максимальний час прийняття рішень у процесі експлуатації і так далі). Отже, побудувавши програмно деяку фіксовану множину випадкових ЛДК та сформувавши їх ідентифікатори, ми маємо деяку передісторію. Далі, на основі цієї передісторії, ми можемо застосувати ідею цілеспрямованого пошуку для наступної побудови ЛДК та знаходження таких, які задовольняють поставленим вимогам.

Відмітимо, що даний підхід можна застосувати для побудови випадкових $k-$значних ЛДК. До того ж, можна розглядати випадкові ЛДК, де у вершинах знаходяться не окремі ознаки, а довільні алгоритми розпізнавання (алгоритмічне дерево класифікації), але в цьому разі для успішного програмного моделювання необхідно мати модулі цих алгоритмів [94]. Водночас утворюється новий спосіб практичного використання вже відомих алгоритмів (методів) розпізнавання з автоматичним визначенням областей їх компетентності, що являється важливою умовою використання колективу правил класифікації [274-285].

Подібне застосування колективу алгоритмів розпізнавання використовується в програмній системі «ОРИОН III» [94], причому алгоритми розпізнавання у вершинах АДК вибиралися за деякими критеріями, визначеними на основі початкових даних НВ.

Отже, зважаючи на вищезазначене, можна зафіксувати наступне:

1) Випадкові ЛДК мають свої як суттєві переваги (програмна простота побудови дерева, зменшення часу загальної генерації ЛДК, можливість оцінки та вибору найбільш підходящого ЛДК з множини побудованих), так і суттєві недоліки (неоптимальність структури, апаратні витрати на генерацію неоптимального ЛДК, гарантована складність структури та велика інформаційна ємність, наявність додаткового етапу оцінки та відбору).

2) Алгоритм побудови випадкового ЛДК дозволяє генерувати цілі набори (множини) дерев класифікації різної структури (складності), серед яких можна вибирати оптимальне для даної задачі.

3) Відмітимо, що важливим питанням в застосуванні випадкових ЛДК є питання вибору (та можливої перебудови) найефективнішого дерева серед множини побудованих випадкових ЛДК.

4) Алгоритм побудови випадкового логічного дерева (який був описаний вище) разом із алгоритмом з покроковою оцінкою важливості дискретних ознак та алгоритмом одноразової оцінки важливості дискретних ознак утворюють основну трійку алгоритмів розпізнавання методу розгалуженого вибору ознак при побудові дерев класифікації [204-206].

5) Відмітимо також, що для зберігання фактичної структури побудованого випадкового ЛДК використовується лише програмний ідентифікатор (паспорт) логічного дерева, який містить лише фактичну послідовність вершин (змінних) у даному дереві, що є дуже ресурсно-економним способом представлення таких складних структур даних. Алгоритм генерації випадкових ЛДК отримав свою програмну реалізацію в бібліотеці алгоритмів та класифікаторів РО програмного комплексу системи «ОРИОН ІІІ».

2.3. Питання гнучкості логічних дерев класифікації в задачах розпізнавання

Нехай $H_0, H_1, \dots, H_{k-1}$ – система класів (образів), яка задана на множині G, що складається з об‘єктів $x_i, (i = 1, \dots, m)$. Характер розбиття множини G на відповідні класи задається за допомогою НВ такого вигляду:

$$\Big(\big(x_1, f_R(x_1)\big), \big(x_2, f_R(x_2)\big), \dots, \big(x_m, f_R(x_m)\big)\Big). \quad (2.2)$$

Зокрема $x_h \in G, f_R(x) \in \{0,1, \dots, k-1\}, \big(h = 1,2, \dots, m\big), k -$ кількість класів НВ, $m -$ загальна кількість об’єктів НВ, а $f_R(x)$ – деяка скінчено-значна функція, що задає розбиття множини G на відповідні образи. Співвідношення $f_R(x_h) = l,\ (l = 0,1, \dots, k-1)$ означає, що $x_h \in H_l$.

Отже, ЛДК – це зв’язний граф без циклів, у некінцевих вершинах якого знаходяться ознаки (тут ознаки – відповідні компоненти об’єктів $x_1, \dots, x_m$), тобто

$$x_1 = (x_1^1, x_2^1, \dots, x_n^1), \dots, x_m = (x_1^m, x_2^m, \dots, x_n^m);$$

$$P_i(x) = x_i^j, (i = 1, \dots, n), (j = 1, \dots, m).$$

Зауважимо, що n – кількість ознак об‘єктів НВ вигляду (2.2), а ребра нумеруються значеннями цих ознак. У кінцевих вершинах ЛДК знаходяться значення функції $f_R(x_h)$. Причому на довільному шляху ЛДК одна і та сама ознака зустрічається тільки один раз.

Відмітимо, що кожній НВ вигляду (2.2) можна поставити у відповідність (за допомогою деякого алгоритму або методу представлення) цілком визначене ЛДК, яке об‘єктам $x_h, (h = 1, \dots, m)$ НВ (2.2) ставить у відповідність значення функції $f_R(x_h)$, що задає розбиття R на множині G [204, 286, 287].

Нехай маємо ЛДК, побудоване за даними НВ (2.2) та набори такого вигляду:

$$\Big(\big(x_{m+1}, f_R(x_{m+1})\big), \big(x_{m+2}, f_R(x_{m+2})\big), \dots, \big(x_{m+\delta}, f_R(x_{m+\delta})\big)\Big). \quad (2.3)$$

Зокрема (2.3) – набір об’єктів тестової вибірки, на яких знайдено помилки розпізнавання, де $2 \le m \le K_1 * K_2 * \dots * K_n, \delta -$ кількість об’єктів ТВ, на яких

знайдено помилки розпізнавання, $K_i -$ кількість значень ознаки $P_i(x)$, $(i = 1, \dots, n)$, де $n -$ кількість ознак об‘єктів НВ.

Нехай маємо ситуацію, коли $x_h \in H_l,\ 0 \le l \le k-1$ у процесі розпізнавання з різним l, то будемо говорити, що виникла помилка розпізнавання $x_h \in H_l$. Якщо об’єкт зустрічається в ТВ декілька разів, то вважатимемо, що це одна і та сама помилка (помилка першого роду).

2.3.1. Загальна схема усунення помилок розпізнання в структурі ЛДК

Побудуємо таке ЛДК, яке безпомилково розпізнаватиме об’єкти з НВ (2.2) та об’єкти з (2.3), тобто опишемо наступний алгоритм усунення помилок розпізнавання (УПР). У парі $\left(x_{m+1}, f_R(x_{m+1})\right)$ об’єкту $x_{m+1} = (x_1^{m+1}, \dots, x_n^{m+1})$ відповідає свій шлях $a_1, a_2, \dots, a_r; (r \in \{1, \dots, n\})$, який закінчується значенням $f_R(x_{m+1})$.

$$P_{i_1} = q(a_1), a_2 = (a_1)_{x_{i_1}^h}, P_{i_2} = q(a_2), a_3 = (a_2)_{x_{i_2}^h}, \dots$$

$$\dots, P_{i_{r-1}} = q(a_{r-1}), a_r = (a_{r-1})_{x_{i_{r-1}}^h}.$$

Зокрема $f_R(x_h) = q(a_r), (h = 1,2, \dots, m), 1 \le r \le n,$ відповідно $i_1, i_2, \dots, i_{r-1}$ номери ознак, що знаходяться у вершинах $a_1, a_2, \dots, a_{r-1}$ та $f_R(x_{m+1}) = q(a_r)$ (такий запис означає, що значення функції $f_R(x_{m+1})$ не співпадає зі значенням $f_R(x_h)$, що знаходиться в некінцевій вершині a_r, тобто виникла помилка розпізнавання). Запам’ятовуємо, що $f_R(x_h) = q(a_r)$, та після цього виконуємо таку процедуру: $P_{i_r} = q(a_r), a_{r+1} = (a_r)_{x_{i_r}^{m+1}}$ (це означає, що з вершини a_r виходить стрілка $x_{i_r}^{m+1}$, відповідна значенню ознаки P_{i_r} (об’єкта x_{m+1}) та входить у наступну за нею вершину a_{r+1}. Далі будемо мати:

$$P_{i_{r+1}} = q(a_{r+1}), a_{r+2} = (a_{r+1})_{x_{i_{r+1}}^{m+1}}, \dots, P_{i_t} = q(a_t), a_{t+1} = (a_t)_{x_{i_t}^{m+1}},$$

$$P_{i_{t+1}} = q(a_{t+1}), a_{t+2} = (a_{t+1})_{x_{i_{t+1}}^{m+1}}, \dots, P_{i_n} = q(a_n), a_{n+1} = (a_n)_{x_{i_n}^{m+1}}.$$

Тут $f_R(x) = q(a_{n+1}),\ i_1, i_{r+1}, \dots, i_t, i_{t+1}, \dots, i_n -$ номери ознак, що знаходяться у вершинах ЛДК $a_r, a_{r+1}, \dots, a_t, a_{t+1}, \dots, a_n,\ r \le t \le n-1.$

На наступному етапі будемо вважати, що:

$a_{r+1}^{k_r} = (a_r)_{k_r}$ та $f_R(x_h) = q\left(a_{r+1}^{k_r}\right), \forall k_r,$

$a_{r+2}^{k_{r+1}} = (a_{r+1})_{k_{r+1}}$ та $f_R(x_h) = q\left(a_{r+2}^{k_{r+1}}\right), \forall k_{r+1},$

… …

$a_t^{k_{t-1}} = (a_{t-1})_{k_{t-1}}$ та $f_R(x_h) = q\left(a_t^{k_{t-1}}\right), \forall k_{t-1},$

$a_{t+1}^{k_t} = (a_t)_{k_t}$ та $f_R(x_h) = q\left(a_{t+1}^{k_t}\right), \forall k_t,$

… …

$a_n^{k_{n-1}} = (a_{n-1})_{k_{n-1}}$ та $f_R(x_h) = q\left(a_n^{k_{n-1}}\right), \forall k_{n-1},$

$a_{n+1}^{k_n} = (a_n)_{k_n}$ та $f_R(x_h) = q\left(a_{n+1}^{k_n}\right), \forall k_n.$

Зауважимо, що $r \le t \le n-1$, $k_1, k_{r+1}, \dots, k_t, k_{t+1}, \dots, k_{n-1}, k_n -$ множина значень ознак $P_{i_r}, P_{i_{r+1}}, \dots, P_{i_t}, P_{i_{t+1}}, \dots, P_{i_{n-1}}, P_{i_n}$ відповідно.

Усі інші знайдені помилки розпізнавання у вибірці $\left(\left(x_{m+2}, f_R(x_{m+2})\right), \dots, \left(x_{m+\delta}, f_R(x_{m+\delta})\right)\right)$ усуваються аналогічно.

ЛДК*, побудоване за допомогою алгоритму УПР, буде безпомилково розпізнавати не тільки об’єкти основної НВ (2.2), але й об’єкти з (2.3). Це випливає із вищеописаної загальної процедури побудови алгоритму УПР.

Якщо описану вище схему УПР включити в довільний алгоритм РО, який пов‘язаний з концепцією ЛДК, наприклад [208], то це забезпечить велику гнучкість даних алгоритмів.

Отже, алгоритм УПР усуває знайдені помилки розпізнавання, а ЛДК, використовуючи при цьому тільки правило класифікації, побудовано за даними НВ та об’єктами з ТВ, для яких виникла помилка розпізнавання. При цьому правило класифікації міняється таким способом, що правильно розпізнає як об’єкти НВ, так і об’єкти ТВ, на яких раніше виникали помилки розпізнавання. Нижче приведемо ще ряд практичних застосувань алгоритму УПР.

Для цього спочатку з’ясуємо кількість можливих помилок розпізнавання на кожному шляху в ЛДК. При розпізнаванні дискретних наборів за допомогою ЛДК інколи доцільно знайти на дереві шляхи, на яких можливіше виникнення помилок розпізнавання. Залежно від ситуації, можна поступити

по різному – відмовитися розпізнавати об’єкти, розпізнавати об’єкти і так далі. Нехай НВ приведена до допустимого вигляду (виконується гіпотеза адекватності та несуперечливості). Відомо, що ЛДК об’єкти з початкової НВ розпізнає безпомилково. Загальна кількість усіх можливих об’єктів, що містять n ознак, не перевищує $K_1 * K_2 * ... * K_n$, де K_i – множина значень ознаки $P_i(x), (i = 1,2, ..., n)$.

Звідси слідує, що ЛДК у процесі розпізнавання може допустити не більше ніж $K_1 * K_2 * ... * K_n - m$ помилок, де $m -$ кількість об’єктів НВ.

Нехай $M^j_{\eta_{i_1} ... \eta_{i_\xi}}$ – кількість пар $\left(x_1, f_R(x_1)\right), (i = 1,2, ..., m)$, що входять до НВ, які задовольняють таким умовам:

$x_i \in G_{\eta_{i_1} ... \eta_{i_\xi}}, (1 \le \xi \le n)$ та $f_R(x) = j, (j = 0,1, ..., k-1)$,

де $G_{\eta_{i_1} ... \eta_{i_\xi}} = \{x \in G / P_{i_1}(x) = \eta_{i_1}, ..., P_{i_\xi}(x) = \eta_{i_\xi}\}$.

Твердження 2.1. Якщо $\max\limits_{0 \le j \le k-1} M^j_{\eta_{i_1} ... \eta_{i_\xi}} = S, (1 \le \xi \le n)$, то на шляху $\eta_{i_1} ... \eta_{i_\xi}$ ЛДК у процесі розпізнавання можливо не більше ніж $K_{n-\xi} - S$ помилок, де $K_{n-\xi} = K_1 * K_2 * ... * K_n / K_{i_1} * ... * K_{i_\xi}$.

Доведення. Розглянемо регулярне ЛДК (тобто дерево, в якого у вершинах ξ – го ярусу знаходяться тільки ознаки $P_{i_\xi}(x)$, $(i = 1,2, ... n)$ називається регулярним). Кількість вершин на ξ – вому ярусі регулярного ЛДК дорівнює $K_{i_1} * ... * K_{i_\xi}$. А кількість кінцевих вершин регулярного ЛДК дорівнює $K_1 * K_2 * ... * K_n$, де $K_{i_\xi} -$ кількість значень ознаки $P_{i_\xi}(x)$, $(i = 1,2, ... n)$, $n -$ кількість ознак НВ. Звідси, якщо кожну вершину ξ – ярусу регулярного ЛДК рахувати начальною, то таке регулярне ЛДК розіб’ється на $K_{i_1} * ... * K_{i_\xi}$ регулярних ЛДК, кожне з яких буде мати по $K_{n-\xi} = K_1 * K_2 * ... * K_n / K_{i_1} * ... * K_{i_\xi}$ кінцевих вершин. Звідси виходить, що, якщо кінцева вершина деякого ЛДК виявляється на $\xi -$ вому ярусі, то вона представляє $K_{n-\xi}$ кінцевих вершин регулярного ЛДК. Якщо задано шлях $\eta_{i_1} ... \eta_{i_\xi}, (1 \le \xi \le n)$, який приводить в кінцеву вершину, то ця вершина

представляє $K_{n-\xi}$ об'єктів. Але якщо в дану вершину вже попадає S об'єктів із НВ, то кількість помилок, які можуть виникнути на даному шляху, не перевищує $K_{n-\xi} - S$.

2.3.2. Основні аспекти задачі побудови структури ЛДК

Із даного твердження випливає, що якщо $\max\limits_{0\le j\le k-1} M^{j}_{\eta_{i_1}\dots\eta_{i_\xi}} = K_{i_1} * \dots * K_{i_\xi}$, то на шляху $\eta_{i_1} \dots \eta_{i_\xi}$ у ЛДК всі об'єкти будуть розпізнаватися правильно. Якщо $K_1 = K_2 = \dots = K_n = 2$, то твердження буде звучати наступним чином: якщо $\max\limits_{0\le j\le k-1} M^{j}_{\eta_{i_1}\dots\eta_{i_\xi}} = S$, то на шляху $\eta_{i_1} \dots \eta_{i_\xi}$ у ЛДК у процесі розпізнавання можливо не більше ніж $2^{n-\xi} - S$ помилок.

Задача побудови ЛДК за неповною НВ. Інколи доцільно мати алгоритми, за якими можна було би побудувати ЛДК за неповною НВ, за m ознаками $(m < n)$, $n -$ кількість ознак об'єктів у НВ або по l_1 об'єктам з l, заданих об'єктів НВ $(l_1 < l)$. У другому випадку, коли ЛДК будується за l_1 об'єктами з НВ, можна скористуватися будь-яким із відомих алгоритмів [94]. Побудоване таким чином ЛДК буде безпомилково розпізнавати об'єкти l_1 з НВ, а на інших об'єктах НВ можливі помилки розпізнавання. У цьому разі використовуємо алгоритм УПР, описаний вище. Побудоване таким способом ЛДК буде безпомилково розпізнавати об'єкти НВ. У ряді випадків описаний вище метод істотно прискорює процес побудови ЛДК за даними НВ, що дуже важливо для задач РО.

Для побудови ЛДК за m ознаками з n, якими задаються об'єкти НВ, впорядкуємо ознаки за важливістю (мірою інформативності або ціною). Для цієї задачі можна використовувати методи та підходи, описані в роботах [204-208]. Після цього береться m впорядкованих за важливістю ознак та будується ЛДК за допомогою будь-якого відомого алгоритму дерева та алгоритму УПР. Це суттєво прискорює процес побудови ЛДК, що дуже важливо для практичного застосування.

Задача відновлення значень ознак за допомогою ЛДК та алгоритм УПР. Спочатку побудуємо ЛДК для об'єктів НВ, для яких усі значення ознак визначені. Після цього за допомогою побудованого дерева розпізнаємо об'єкти НВ, для яких значення ознак визначені неповністю.

Крок 1. Зафіксуємо один із таких об'єктів тестової вибірки.

Крок 2. Додатково визначаємо невідомі значення ознак, $P_{i_1}, \dots, P_{i_\xi}$, $(1 \le \xi \le n)$ довільними значеннями. Проведемо розпізнавання цих об'єктів за побудованим ЛДК. Нехай об'єкт, для якого значення ознак $P_{i_1}, \dots, P_{i_\xi}$ невизначені на НВ, належить класу H_l, $(1 \le l \le k-1)$, k – кількість класів НВ. Якщо ЛДК віднесе цей об'єкт до класу H_l, то вважаємо, що ознаки $P_{i_1}, \dots, P_{i_\xi}$, визначені значеннями $x_{i_1}, \dots, x_{i_\xi}$, можуть приймати ці значення, а в протилежному випадку – ні.

Крок 3. Беремо (фіксуємо) наступний з об'єктів тестової вибірки та переходимо до кроку 1. Якщо таких об'єктів більше нема, то процес відновлення значень ознак закінчуємо.

Визначення 2.1 Інформацію, що вказує тільки на порядок розміщення ознак у логічному дереві, будемо називати схемою ЛДК, а кількість ознак, що входять в цю схему – її порядком (потужністю).

Теорема 2.1 Кількість схем ЛДК визначається рекурсивним відношенням: $N_{n+1} = N_n^2 * (n+1)$, де $N_2 = 2$, $n-$ кількість ознак (ознак об'єкта в НВ) задачі розпізнавання.

Доведення. Для $n = 2$ теорема очевидна. Для довільного n задача розбивається на прості випадки, при $n = 3$ маємо дві схеми порядку 2. Оскільки ці схеми незалежні, то маємо $N_3 = N_2^2 * 3$, при $n = 4$ схема ЛДК розбивається на дві незалежні схеми порядку 3, тому $N_4 = N_3^2 * 4$ і так далі.

Задача побудови мінімального ЛДК відносно кількості вершин. Запропонуємо алгоритм знаходження мінімальних тестів та використання їх для побудови ЛДК. Уведемо деякі позначення, необхідні для роботи.

$$G_{r_1 \dots r_n} = \{x \in G / P_1(x) = r_1, \dots, P_n(x) = r_n\}. \qquad (2.4)$$

Зокрема $P_1(x), \dots, P_n(x)$ – деяка система ознак, $r_i \in \{0,1, \dots, k-1\}$, $(i = 1, \dots, n)$, $M_{r_1 \dots r_n}$ – кількість тих пар НВ $(x_1, f_R(x_i))$, $(1 \le i \le m)$, які задовольняють умову: $x_i \in G_{r_1 \dots r_2}$.

$M^j_{r_1 \dots r_n}$ – кількість із (2.5) таких наборів, які задовольняють наступним умовам:

$$x_i \in G_{r_1 \dots r_n} \text{ та } f_R(x) = j; \tag{2.5}$$

$$t^j_{r_1 \dots r_n} = M^j_{r_1 \dots r_n} / M_{r_1 \dots r_n}. \tag{2.6}$$

$a_{r_1 \dots r_n} = \max\limits_{1 \le j \le k-1} t^j_{r_1 \dots r_n}$, де $r_1 \dots r_n \in \{0, \dots, k-1\}$, $(j = 0,1, \dots, k-1)$, $(i = 1, \dots, n)$ тоді:

Теорема 2.2. Набір ознак із номерами $i_1, \dots, i_\xi, (\xi \in 1, \dots, n)$ є тестом тоді і тільки тоді, коли для них виконується:

$$\sum_{r_{i_1} \dots r_{i_\xi}} \left(\frac{M_{r_{i_1} \dots r_{i_\xi}}}{m} \right) * a_{r_{i_1} \dots r_{i_\xi}} = 1. \tag{2.7}$$

Формула (2.7) дає можливість знаходити тести заданої довжини (під довжиною тесту будемо розуміти кількість ознак, що входять до нього). При $\xi = 1$ формула (2.7) дає можливість знаходити тести довжиною «1», при $\xi = 2$ – тести довжини «2», а при $\xi = \eta$ – тести довжиною «η» $(1 \le \eta \le n)$. Формула (2.7) дає можливість знаходити тести у випадку, коли в НВ присутня довільна кількість класів $f_R(x) \in \{0,1, \dots, k-1\}$, а ознаки $P_i(x)$ приймають довільні значення $P_i(x) \in \{0,1, \dots, k-1\}$.

Зауваження. Якщо хоча би один член виразу (2.7) не дорівнює одиниці, то набір ознак із номерами $r_{i_1} \dots r_{i_\xi}$ не є тестом. Це зауваження дозволяє ефективно знаходити тести за допомогою формули (2.7). Мінімальні тести за цією формулою знаходяться, виходячи з визначення поняття мінімального тесту. Виникає питання, яким чином побудувати ЛДК за мінімальним тестом та чи буде таке побудоване дерево мінімальним (ЛДК, побудоване за даними НВ, називається мінімальним, якщо не існує іншого ЛДК*, побудованого за цією ж НВ, у котрого кількість вершин менша за початкове дерево). Відповідь

на першу частину питання досить тривіальна. Якщо група (набір) ознак із номерами $r_{i_1} \ldots r_{i_\xi}$ є мінімальним тестом, то побудова ЛДК за даними ознаками не є важкою. Це можна зробити за допомогою алгоритмів побудови дерев класифікації, описаних наприклад у [90, 94]. Відповідь на другу частину питання не є простою. Виявляється, що ЛДК, побудоване за мінімальним тестом, не завжди є мінімально можливим. Це можна побачити з табл. 2.5.

Таблиця 2.5. Залежність складності логічного дерева від кількості ознак.

$KL\backslash D$	1	2	3	4	5	6	7	8	…	n
Max	3	7	15	31	63	127	255	511	…	$2^{n+1}-1$
Min	3	5	7	9	11	13	15	17	…	$2n+1$

Зауважимо, що D – довжина тупикового тесту, KL – кількість можливих вершин, Max вказує на максимальну кількість вершин ЛДК (складність) при заданій довжині тупикового тесту, Min вказує на мінімальну кількість вершин ЛДК, можливу при заданій довжині тупикового тесту. З табл. 2.5 бачимо, що можливий випадок, коли ЛДК, побудоване за мінімальним тестом, може мати більше вершин, ніж ЛДК, побудоване за тупиковим тестом більшої довжини. Наприклад, ЛДК, побудоване за мінімальним тестом довжини 4, може мати 31 – ну вершину, а ЛДК, побудоване за тупиковим тестом довжини 14, може мати 29 вершин, ЛДК, побудоване за мінімальним тестом довжини 10, може мати 2047 вершин, а ЛДК, побудоване за тупиковим тестом довжини 100, може мати 201 вершину (за тупиковим тестом довжиною 1000 – 2001 вершину). Із зазначеного вище випливає дуже важливе питання – як за заданою НВ побудувати ЛДК, яке мало би мінімальну кількість вершин. Особливої важливості це питання набуває в тих випадках, коли треба побудувати деяку технічну схему, модель, систему, яка би правильно класифікувала фіксовану кількість наборів. Вирішення такого питання дозволить у цих випадках побудувати найпростішу процедуру (схему, систему) розпізнавання. Складність ЛДК залежить від кількості кінцевих вершин у дереві. Якщо

кількість кінцевих вершин у ЛДК позначити через K, то загальна кількість вершин N у ЛДК можна розрахувати за формулою:

$$N = 2K - 1. \quad (2.8)$$

Це справедливо для випадку, коли $(P_i(x) \in \{0,1\})$, $2 \le K \le 2^n$, де n – кількість ознак об‘єктів НВ.

Кількість кінцевих вершин ЛДК залежить від того, скільки наборів НВ відповідають одній кінцевій вершині.

Виявляється, що можна знайти за заданою НВ таке K, яке відповідає мінімальній кількості вершин деякого можливого ЛДК. Зрозуміло, що таке ЛДК буде мінімальним за складністю (кількістю вершин). Це випливає з формули (2.8).

Нехай вже є набір тупикових тестів, які знайдені за формулою (2.7). Зафіксуємо деякий тупиковий тест $P_{i_1} \vee P_{i_2} \vee \ldots \vee P_{i_\xi}$. Для кожної ознаки, що входить у даний тупиковий тест знайдемо:

$\max\limits_{1 \le j \le k-1} M^j_{r_{i_\varepsilon}}$, де $(\varepsilon = 0,1, \ldots, \xi), (\xi = 1, \ldots, n)$, для кожної ознаки.

Розглянемо два можливі випадки:

1) Існує таке значення ознаки P_{i_ε}, що для неї $\max\limits_{1 \le j \le k-1} M^j_{r_{i_\varepsilon}} < M_{r_{i_\varepsilon}}$.

2) Не існує такого значення ознаки P_{i_ε}, що для нього $\max\limits_{1 \le j \le k-1} M^j_{r_{i_\varepsilon}} < M_{r_{i_\varepsilon}}$.

У першому випадку будемо говорити, що для даного значення ознаки P_{i_ε} існує деякий шлях у ЛДК (ЛДК побудовано за даним тупиковим тестом) довжиною один крок, і цей шлях приводить у кінцеву вершину.

У другому випадку в ЛДК не існує шляху довжиною в один крок для даного значення ознаки P_{i_ε} такого, який би приводив у кінцеву вершину дерева.

У першому випадку – це кількість наборів НВ, на яких ознака $P_{i_\varepsilon}(x)$ приймає фіксоване значення з множини $\{0, \ldots, k-1\}$, а функція $f_R(x)$ приймає на всіх цих наборах фіксоване значення з множини $\{0, \ldots, k-1\}$. Усі ці набори входять в одну кінцеву вершину ЛДК. Якщо у вершинах ЛДК стоїть ознака

P_{i_ε}, то далі знаходимо всі $\max\limits_{1\le j\le k-1} M^j_{r_{i_1}r_{i_2}}$ для даного тупикового тесту і так далі до тих пір, поки рівність $\max\limits_{1\le j\le k-1} M^j_{r_{i_1}\dots r_{i_\xi}} = M_{r_{i_1}\dots r_{i_\xi}}$ не виконається для всіх наборів $r_{i_1}\dots r_{i_\xi}$ з НВ. Після цього підраховуємо кількість ситуацій, коли рівність $\max\limits_{1\le j\le k-1} M^j_{r_{i_1}\dots r_{i_\varepsilon}} = M_{r_{i_1}\dots r_{i_\varepsilon}}$, $(\varepsilon = 1,\dots,\xi)$ виконувалась для деяких наборів $r_{i_1}\dots r_{i_\xi}$ з НВ. Позначимо це число через η, η – це кількість можливих кінцевих вершин ЛДК, яке може бути побудовано за даним тупиковим тестом. Проробимо цю процедуру для кожного тупикового тесту, знайденого за формулою (2.7), та порівняємо знайдене η для кожного тупикового тесту. Тест, для якого η буде мінімальним, і дозволяє побудувати мінімальне ЛДК.

Підсумовуючи, можна зафіксувати такі положення:

1) Навіть в ситуації виникнення помилок у ЛДК на етапі роботи з тестовою вибіркою є можливість виправлення помилок (алгоритм УПР) шляхом корекції структури дерева, таким способом правильно класифікується як НВ, так і ТВ.

2) У довільному ЛДК, побудованому за даними початкової НВ, на деякому фіксованому шляху $\eta_{i_1}\dots\eta_{i_\xi}$ (шляху в ЛДК) у процесі розпізнавання можливе не більше ніж $K_{n-\xi} - S$ помилок.

3) Інколи доцільно мати алгоритм, за допомогою якого можна було би побудувати ЛДК за неповною НВ (і за кількістю об‘єктів, і за кількістю ознак). Побудоване таким способом ЛДК буде безпомилково розпізнавати частину НВ, за якою побудоване дерево, а на інших наборах давати помилки (уникнути цього можна при застосуванні алгоритму УПР).

4) Довільне ЛДК, побудоване за даними НВ за мінімальним тестом, не завжди є мінімально можливим.

5) Загальну кількість вершин у ЛДК, яке побудоване за даними НВ, можна розрахувати за формулою: $N = 2K - 1$.

Висновки до розділу 2

1. У розділі представлено загальну концепцію дерев класифікації, яка забезпечує покриття масиву даних навчальної інформації за рахунок фіксації об‘єктів вибірки в своїй структурі, причому такий підхід зберігання інформації забезпечує як механізм донавчання (розширення структури), так і виправлення помилок, а застосування ЛДК в задачах РО дозволяє просто та компактно описувати ФР. Також зафіксовано відсутність актуальних методів та алгоритмів, які би забезпечили можливість будувати ефективні конструкції логічних дерев на основі масивів даних (НВ) великої та надвеликої розмірності.

2. Розглянуто окремий клас дерев класифікації – випадкові ЛДК, які мають суттєві переваги (програмна простота побудови дерева, зменшення часу загальної генерації (відбору) вершин ЛДК, можливість оцінки та вибору найбільш відповідного ЛДК із множини побудованих). Запропоновано алгоритмічну схему побудови випадкових ЛДК, яка дозволяє генерувати цілі набори (множини) дерев класифікації різної структури (складності), серед яких можна вибирати оптимальне для даної прикладної задачі.

3. Запропоновано просту алгоритмічну схему виправлення помилок ЛДК (алгоритм УПР) шляхом корекції структури дерева класифікації. Запропоновано метод числової оцінки для фіксованого шляху структури довільного ЛДК максимальної кількості помилок класифікації всіх типів в процедурі розпізнавання. Досліджено питання можливості побудови ЛДК мінімальної складності на основі мінімального тесту та структурної складності максимального дерева класифікації, побудованого за даними НВ.

РОЗДІЛ 3
МЕТОДИ ПОБУДОВИ ДЕРЕВОПОДІБНИХ МОДЕЛЕЙ РОЗПІЗНАВАННЯ ОБРАЗІВ НА ОСНОВІ Т – ОПОРНИХ МНОЖИН

3.1. Метод опорних множин у задачах розпізнавання

Як відомо, вибір системи опорних множин представляється першим і основним етапом у загальній схемі алгоритмів обчислення оцінок детально опрацьованих Ю. Журавльовим, який у свій час запропонував більш загальну формалізацію з різними способами вибору інформативних підсистем ознак та формулами обчислення оцінок. Зрозуміло, що вибір системи опорних множин Ω також визначає множину задач, для розв‘язку яких можна побудувати коректний алгоритм класифікації [288-290].

Відмітимо, що можливість ефективної реалізації алгоритмів типу обчислення оцінок істотно залежить від вибору системи опорних множин. Алгоритми обчислення оцінок мають загальну схему, яка складається з послідовності шести етапів, для кожного з яких існують різні варіанти безпосередньої реалізації. В даному дослідженні велика увагу приділено саме першому етапу в схемі даних алгоритмів (у розрізі концепції дерев класифікації) за рахунок введення нового поняття – поняття $T-$ опорної множини.

Слід також зауважити, що питання опорних множин безпосередньо пов‘язано з питанням ефективного пошуку тупикових тестів, яке вже піднімалося в даному дослідженні раніше (розділ II). Саме питання пошуку множини всіх тупикових тестів є обчислювально – складна комбінаторна задача, і, навіть при сучасному програмно-апаратному забезпеченні (зокрема з використанням механізму розпаралелювання), ефективно не може бути вирішена для відносно невеликих масивів НВ (декілька тисяч об‘єктів та ознак). Тому при розв’язуванні практичних задач обчислюється та використовується лише певна частка тупикових тестів.

У даному розділі вводиться модифікація поняття опорної множини та проводиться всебічне його вивчення. Істотну роль при цьому грають ЛДК, які

розглядалися раніше. Пропонується підхід, що дає змогу методично та одноманітно описувати (автоматично створювати) за допомогою програмних реалізацій достатньо широкі класи алгоритмів розпізнавання на основі ЛДК із використанням ідей алгебраїчного підходу. Проте, в даному дисертаційному дослідженні центр уваги зміщений на вибір систем T – опорних множин (головна ідея яких буде представлена нижче), що дозволяє значно зменшити час класифікації довільного допустимого об'єкта S за рахунок попередньої обробки начальної інформації (організації асоціативного пошуку). Представлення алгоритмів розпізнавання у вигляді ЛДК дає можливість економити апаратну пам'ять комп'ютера при їх практичній реалізації. Крім цього, велика увага направлена на створення ефективних моделей алгоритмів розпізнавання, призначених для класифікації дискретних об'єктів.

Розглянемо модель добре відомих алгоритмів розпізнавання та класифікації типу обчислення оцінок [51-56].

Нехай стоїть задача класифікації за l класами об'єктів деякої множини $M \in M_1 \cdot \ldots \cdot M_n$, де M_i – область ознак (метричний простір із метрикою p_i, $i = 1, \ldots, n$). За допомогою навчаючої інформації $I(l)$, об'єкта S, будуються оцінки $\Gamma_j(S)$, що належать j – вому класу, та за допомогою порогових функцій проводиться класифікація об'єкта. Кожний конкретний алгоритм класифікації A в моделі визначається вибором системи опорних множин Ω_A, функцією близькості $B(w, S, S')$, визначенням ваги об'єктів за навчаючою інформацією та правилом класифікації.

Оцінки $\Gamma_j(S)$, як правило, вираховуються за наступною формулою:

$$\Gamma_j(S) = \frac{1}{N * |W|} \sum_{S' \in W_j} \gamma(S') \sum_{w \in \Omega \in \Omega_A} P(w) B(w, S, S').$$

Зауважимо, що N – множник, що нормує; $|.|$ – потужність (вага) деякої множини; W_j – множина об'єктів навчаючої інформації, що належать j – вому класу $(j = 1, \ldots, l)$; $\gamma(S')$ – деяке число, що відповідає кожному об'єкту з навчаючої інформації $I(l)$ (характеризує ступінь важливості об'єкта S', його

інформативність); $P(w) = P_{i_1} + \cdots + P_{i_k}$ – вага опорної множини $w \leftrightarrow \Omega = \{i_1, \dots, i_k\}$; $S = (a_1, \dots, a_n)$, $S' = (b_1, \dots, b_n)$ – об'єкти з M; $B(w, S, S') = 1$ якщо в деякій системі нерівностей $p_{i_1}(a_{i_1}, b_{i_1}) \leq \varepsilon_{i_1}, \dots, p_{i_k}(a_{i_k}, b_{i_k}) \leq \varepsilon_{i_k}$ виконано не менше q_1 та не виконано не більше q_2 нерівностей, де q_1, q_2 – фіксовані додатні числа та вектор $(\varepsilon_1, \dots, \varepsilon_n)$, де $\varepsilon_i \leq 0, i = 1, \dots, n$, у протилежному випадку $B(w, S, S') = 0$.

Зрозуміло, що прямі розрахунки за даною формулою обчислення оцінок будуть не ефективні, якщо взагалі практично можуть бути здійснені. У даний час запропоновано ряд підходів, що забезпечують ефективніше обчислення оцінок, та інші, що залежать від вибору системи опорних множин. Зауважимо, що для часткового подолання цих обмежень введена модифікація поняття опорної множини ($T -$ опорної множини), що не зменшує загальності цього поняття. Пропонується підхід, що забезпечує ефективний вибір таких опорних множин, при яких обчислення оцінок належності об'єкта S до класу з номером j значно спрощується. Приводяться дослідження по розширенню даного класу алгоритмів.

Зважаючи на все зазначене вище, можна зафіксувати, що:

1) задача пошуку набору всіх тупикових тестів не є тривіальною та не дозволяє простого розв'язання шляхом прямого перебору навіть для відносно невеликих масивів НВ;

2) пряме обчислення оцінки $\Gamma_j(S)$, як правило, ускладнене або взагалі неможливе, хоча і запропоновані підходи, що значно спрощують цей процес за рахунок особливостей вибору набору опорних множин;

3) кожний фіксований алгоритм класифікації в моделі алгоритмів обчислення оцінок буде однозначно визначатися вибором системи опорних множин, функцією близькості, визначенням ваги об'єктів за навчаючою інформацією та правилом класифікації.

У наступній частині роботи перейдемо безпосередньо до визначення поняття Т – опорної множини.

3.2. Концепція Т – опорних множин та способи їх задання

Перейдемо безпосередньо до головної ідеї Т – опорної множини, яка полягає у відборі та фіксації деякого набору ознак разом зі своїми значеннями.

Визначення 3.1. Т – опорна множина – це фіксований набір ознак із фіксованими їх значеннями:

$$\binom{e_{i_1}}{X_{i_1}}, \dots, \binom{e_{i_k}}{X_{i_k}}, k = 1, \dots, n$$

Систему Т – опорних множин будемо позначати Ω^T.

Нехай зафіксовано деякі числа q_1, q_2 та вектор $(\varepsilon_1, \dots, \varepsilon_n) -$ де $\varepsilon_i \geq 0$, $i = 1, \dots, n$.

3.2.1. Визначення інформативності по відношенню до об‘єкта та класу на основі Т – опорних множин

Визначення 3.2. Т – опорна множина $\binom{e_{i_1}}{X_{i_1}}, \dots, \binom{e_{i_k}}{X_{i_k}}$ називається інформативною по відношенню до об‘єкта S, якщо в системі нерівностей $p_{i_1}(a_{i_1}, e_{i_1}) \leq \varepsilon_{i_1}, \dots, p_{i_k}(a_{i_k}, e_{i_k}) \leq \varepsilon_{i_k}$ виконано не менше q_1 та не виконано не більше q_2 нерівностей.

Визначення 3.3. Т – опорна множина $\binom{e_{i_1}}{X_{i_1}}, \dots, \binom{e_{i_k}}{X_{i_k}}$ називається інформативною по відношенню до класу j за інформацією $I(l)$, якщо в системі нерівностей $p_{i_1}(a_{i_1}, e_{i_1}) \leq \varepsilon_{i_1}, \dots, p_{i_k}(a_{i_k}, e_{i_k}) \leq \varepsilon_{i_k}$ виконано не менше q_1 та не виконано не більше q_2 нерівностей, для деяких об‘єктів тільки $j -$ тового класу. Відносно об‘єктів, що не входять у $I(l)$, ніяких обмежень не накладається.

Уведемо наступні позначення: $w^T = \{w_1, \dots, w_\xi\}$, де w_i буде $i -$ товою Т – опорною множиною, $\xi -$ кількість Т – опорних множин, $H_j -$ множина об’єктів S, що належать класу H_j. Той факт, що Т – опорна множина w інформативна до класу H_j, будемо позначати так: $w * H_j$, аналогічно $w * S -$ інформативність w до об‘єкту S. Запис $w * H_j$ означає, що існує хоча би один об‘єкт $S \in H_j$, що $w * H_j$.

Визначення 3.4. Систему Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ будемо називати інформативною, якщо:

1) $\forall i \exists j (w_i * H_j)$;

2) $\forall j \exists i (w_i * H_j)$;

3) $\forall S \exists i (w_i * S)$;

4) $\forall S' \exists i (w_i * S')$.

Зауважимо, що $i = 1, \dots, \xi, j = 1, \dots, l$.

Визначення 3.5. Систему Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ будемо називати несуперечливою, якщо для неї виконуються умови:

1) $\forall i \exists j (w_i * H_j)$;

2) $\forall S' \in H_j \ \exists i (w_i * S')$;

3) $\overline{\exists}(i_1, i_2), i_1 \neq i_2, \left(w_{i_1} * S \text{ та } w_{i_2} * S\right) \Rightarrow \left(w_{i_1} * H_{j_1} \text{та } w_{i_2} * H_{j_2}\right), i_1 \neq i$.

Зауважимо, що $i = 1, \dots, \xi, j = 1, \dots, l$.

Визначення 3.6. Інформативну несуперечливу систему Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ назвемо простою, якщо, для неї виконується умова $\forall S \exists! \, i (w_i * S)$.

У подальшому будемо розглядати більш складні системи Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ з використанням одномісних, k −місних предикатів, скінчено значних обчислюваних функцій, алгоритмів та відображень. Дослідження, проведені для найбільш простих систем T − опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$, будуть використані для вивчення більш складних систем опорних множин Ω^T.

3.2.2. Задання систем Т – опорних множин

На наступному етапі розглянемо способи задання систем Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$.

1) Найбільш природний спосіб задання систем Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ полягає у використанні їх основного визначення. Наприклад, випадковим чином фіксуються певні піднабори ознак та їх значення. Таким

способом, використовуючи програмний генератор псевдовипадкових чисел, можна сформулювати різні множини систем Т – опорних множин. Після цього можна провести процедуру селекції і виділити, і інформативні, і несуперечливі, і інформативно-несуперечливі системи Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$.

2) Задання систем Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ за допомогою тестів. Нехай ознаки з номерами $i_1, \dots, i_k$ утворюють тест, знайдений за деякою початковою інформацією $I(l)$. Якщо зафіксувати значення ознак, що входять у даний тест (звичайно, що з відповідними їх значеннями на $I(l)$), то дана множина буде утворювати несуперечливу систему Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$, але не завжди інформативну (знову же відповідно до базового визначення Т – опорної множини).

3) Зауважимо, що є можливість знаходити системи Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$, в тому числі інформативні, несуперечливі, прості використовуючи інші методи та алгоритми.

4) Використовуючи поняття допустимого розбиття та алгоритм його загальної побудови, можна знаходити системи несуперечливих Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ за інформацією $I(l)$ з числом Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ не більше деякого l, де l – кількість об’єктів у $I(l)$.

5) Знаходження простих систем Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ за допомогою $I(l)$ та дерев розпізнавання має найбільше перспектив, оскільки мало залежить від об‘єму та виду початкової інформації $I(l)$, тобто ЛДК у певному сенсі є стисненим структурним типом даних. Наприклад*:*

- Інформація $I(l)$ може бути повістю розміщена в оперативній пам‘яті комп‘ютера.
- Інформація $I(l)$ є масивом великого (надвеликого) об’єму, що значно більше об’єму оперативної пам’яті комп‘ютера.

- Інформація $I(l)$ являє собою дані, в якій деякі значення ознак на деяких наборах об‘єктів не визначені.
- Набори деяких ознак у початковій інформації $I(l)$ є різнотипними.

До того ж, як буде показано в роботі далі, за допомогою ЛДК можна проводити деякі операції над системами Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ та в результаті отримувати деякі інші прості системи Т – опорних множин Ω^T. Це дозволить (при виконанні деяких умов) знаходити базис Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ та визначати всі інші набори множин через цей базис.

Зважаючи на все вище зазначене можна зафіксувати наступні пункти:

1) Головною ідеєю Т – опорної множини, є відбір (фіксація) певного набору ознак разом зі своїми значеннями на основі інформації деякої початкової НВ, з можливістю наступної оцінки даних опорних множин за допомогою відповідних функціоналів [100].

2) Більший інтерес представляють більш складні системи Т – опорних множин з використанням одномісних, $k-$ місних предикатів, скінчено значних обчислюваних функцій, алгоритмів та відображень.

3) Відмітимо, що існує багато способів задання систем Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$, причому найпростіший і природний випливає безпосередньо із самого визначення даного поняття й полягає в фіксації деякого набору ознак та їх значень на основі програмного генератора PRG.

4) Зауважимо, що процес знаходження простих систем Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$ за допомогою масиву деякої початкової інформації та дерев розпізнавання мало залежить від об‘єму, структури та виду початкової інформації $I(l)$, тобто побудова логічного дерева представляє собою процес стиснення початкових даних НВ.

3.3. Формальне визначення алгоритму розпізнавання на основі Т – опорних множин

Отже, в наступній частині даного дослідження введемо формальне визначення алгоритму класифікації на основі Т – опорних множин.

Нехай $\Omega_A^T -$ деяка фіксована система Т – опорних множин $\Omega^T = \{w_1, \dots, w_\xi\}$, яка побудована за початковою інформацією $I(l)$, $S(a_1, \dots, a_n)$ – деякий об'єкт із множини об'єктів M. Уведемо також $B(w, S) -$ функцію близькості «інформативності» об'єкта S до Т – опорної множини w.

На наступному етапі за Ω_A^T та об'єктом S будуються оцінки $\Gamma_j(S)$ належності $j -$ вому класу об'єкту S наступним чином: $\Gamma_j(S) = \sum_{w \in \Omega_A^T} B(w, S)$.

Класифікація об'єкта S проводиться за допомогою деякої скінченозначної функції $y = f(\Gamma_j(S))$, що приймає значення $0,1, \dots, l$. Отже, можемо зафіксувати таку модель алгоритму розпізнавання:

$$\{A\} = (\Omega_A^T, B(w, S), \Gamma_j(S), y = f(\Gamma_j(S))).$$

На відміну від відомих алгоритмів обчислення оцінок [55], тут центр уваги зміщений саме на вибір систем Т – опорних множин за початковою інформацією $I(l)$. Це дозволяє зменшити час класифікації довільного об'єкта S за рахунок попередньої обробки масиву начальної інформації $I(l)$ (наприклад, організація асоціативного пошуку).

Дослідження даної моделі алгоритмів можна було би провести за традиційною схемою, але практична програмна реалізація такого підходу вимагає достатньо великих затрат оперативної пам'яті комп'ютера та процесорного часу. Дане дослідження направлено на з'ясування можливостей значного зменшення витрат апаратних ресурсів (оперативної пам'яті) при практичній реалізації алгоритмів за цією моделлю та її розширення.

3.3.1. Моделі алгоритму розпізнавання на основі Т – опорних множин

Приведемо приклади деяких підмоделей уведеної моделі алгоритму розпізнавання:

$(\Omega_A^T, B(w,S), \Gamma_j(S), y = f(\Gamma_j(S)))$, нехай $M_i \in \{0,1\}, i = 1, \dots, n$.

На першому етапі введемо наступні визначення.

Визначення 3.7. Т – опорна множина називається інформативною по відношенню до об'єкта S, якщо відповідна комбінація ознак Т – опорної множини співпадає з відповідними значеннями ознак даного об'єкта S.

Визначення 3.8. Т – опорна множина називається інформативною по відношенню до деякого класу j за початковою інформацією $I(l)$, якщо відповідна комбінація ознак (мається на увазі саме набір фіксованих ознак даної Т – опорної множини) зустрічається тільки на деяких наборах об'єктів цього класу (тобто класу j).

На наступному етапі зафіксуємо такі функції:

$$B(w,S) = \begin{cases} 1, & \text{якщо } w * S \\ 0, & \text{в протилежному випадку} \end{cases};$$

$$\Gamma_j(S) = \sum_{w \in \Omega_A^T} B(w,S);$$

$$y = \begin{cases} 1, & \text{якщо } \Gamma_j(S) > \Gamma_i(S), i = 1, \dots, j-1, j+1, \dots, l \\ 0, & \text{в протилежному випадку} \end{cases};$$

1) У якості Ω_A^T будемо вибирати тільки інформативні несуперечливі системи Т – опорних множин. Позначимо такі системи Т – опорних множин $\Omega_A^T(U,H)$, а модель: $R(U,H) = (\Omega_A^T(U,H), B(w,S), \Gamma_j(S), y = f(\Gamma_j(S)))$.

Теорема 3.1. Не існує деякого алгоритму $A \in R(U,H)$, який би видавав помилки на даних початкової навчальної інформації $I(l)$ та відмови класифікації на всій множині допустимих об'єктів.

Доведення. Зафіксуємо довільну систему $\Omega_A^T(U,H)$ Т – опорних множин з $\Omega_A^T(U,H)$. Візьмемо довільний об'єкт S з початкової навчальної інформації $I(l)$, причому $S \in H_j, j = 1, \dots, l$. Тоді відповідно до визначень інформативності та несуперечливості $\Omega_A^T(U,H)$ виходить, що $w * S$; $\overline{\exists} i_1 \neq i$,

що з $(w_{i_1} * S$ та $w_i * S) \to (w_{i_1} * H_{j_1}$ та $w_i * H_j), j_1 \neq j$. Отже $j_1 = j$, а це фактично означає, що не існує довільного алгоритму А, який би видавав помилки на масиві початкової навчальної інформації $I(l)$. Аналогічним чином доводиться друга частина теореми.

2) Якщо у якості Ω_A^T вибирати тільки інформативні Т – опорні множини, то системи Т – опорних множин позначаємо $\Omega_A^T(U,-)$, а модель – через $R(U,-) = (\Omega_A^T(U,-), B(w,S), \Gamma_j(S), y = f(\Gamma_j(S)))$.

Алгоритми цієї моделі можуть видавати помилки розпізнавання на даних початкової навчальної інформації $I(l)$ та відмови класифікації на деяких допустимих об‘єктах.

3) Якщо у якості Ω_A^T вибрати тільки несуперечливі Т – опорні множини, то системи Т – опорних множин позначаємо $\Omega_A^T(-,H)$, а модель відповідно через $R(-,H) = (\Omega_A^T(-,H), B(w,S), \Gamma_j(S), y = f(\Gamma_j(S)))$.

Алгоритми цієї моделі не дають помилок розпізнавання на даних початкової навчальної інформації $I(l)$, проте дають відмови класифікації на деяких допустимих об‘єктах.

4) Якщо у якості Ω_A^T вибрати просто довільні Т – опорні множини, то системи Т – опорних множин позначаємо $\Omega_A^T(-,-)$, а отриману модель відповідно через $R(-,-) = (\Omega_A^T(-,-), B(w,S), \Gamma_j(S), y = f(\Gamma_j(S)))$.

Теорема 3.2. Кількість усіх алгоритмів у моделі $R(-,-)$ буде дорівнювати 2^{3^n-1}, де $n-$ кількість ознак у задачі розпізнавання (в описі об‘єктів початкової НВ).

Доведення. При $n = 1$, маємо дві Т – опорні множини: $\binom{0}{X_1}, \binom{1}{X_1}$.

При $n = 2$ маємо вісім Т – опорних множин: $\binom{0}{X_1}, \binom{1}{X_1}, \binom{0}{X_2}, \binom{1}{X_2}$, $\binom{0\ \ 0}{X_1\ X_2}, \binom{0\ \ 1}{X_1\ X_2}, \binom{1\ \ 0}{X_1\ X_2}, \binom{1\ \ 1}{X_1\ X_2}$.

При $n = k$ маємо $2^1 c_n^1$ Т – опорних множин довжини 1, $2^2 c_n^2$ – довжини 2, і відповідно $2^k c_n^k$ – довжиною k.

Отже, для довільного n маємо $S_n = 2^1 c_n^1 + 2^2 c_n^2 + \cdots + 2^n c_n^n$. Звідси $S_n = 3^n - 1$. Це – кількість усіх можливих Т – опорних множин. Кількість усіх можливих систем $\Omega_A^T(-,-)$ при цьому визначається $2^{3^n-1} - 1$.

3.3.2. Прості системи Т – опорних множин

Відмітимо, що найбільший інтерес викликають саме прості системи Т – опорних множин. Алгоритми розпізнавання, що їх визначають, не дають помилок розпізнавання на даних початкової навчальної інформації $I(l)$ та відмов класифікації на всій множині допустимих об‘єктів. Крім того, як буде показано далі, складність прийняття рішень на довільно допустимому об‘єкті S, не перевищує n, простих значень ознак цього об‘єкта.

Теорема 3.3. Ємність усіх моделей $R(-,-)$, $R(-,H)$, $R(U,-)$, $R(U,H)$ є скінченою.

Для доведення теореми достатньо розглянути лише модель $R(-,-)$, оскільки інші являються її частиною.

Приклад. Нехай $S_1 = (0,0,0,0)$, $S_2 = (1,0,1,0)$, $S_3 = (1,1,1,1)$, $S_1 \in H_1$, $S_2 \in H_2$, $S_3 \in ?$.

Для цього прикладу не існує $\Omega_A^T(-,-)$, щоб алгоритм розпізнавання класифікував об‘єкт S_3 до класу H_1.

Наслідок. Введення різних параметрів при обчисленні оцінок $\Gamma_j(S)$ не розширяє моделі $R(-,H)$, $R(U,H)$.

Наведена теорема та її наслідок дають можливість розглядати й інші моделі. Дані моделі можна розширити за рахунок запропонованих інших визначень інформативності Т – опорної множини по відношенню до класу j та об’єкту S. Наведемо одне з них.

Визначення 3.9. Т – опорна множина w називається інформативною по відношенню до об’єкта S, якщо відповідна комбінація значень ознак знаходиться в деякому відношенні φ з відповідними значеннями ознак об’єкта S.

Визначення 3.10. Т – опорна множина w називається інформативною до класу j за початковою інформацією $I(l)$, якщо існує відповідна множина деяких відношень φ_i, значень ознак усіх об‘єктів класу j з комбінацією ознак множини w.

Зважаючи на все зазначене вище, можна зафіксувати наступні пункти:

1) На відміну від алгоритмів обчислення оцінок, при роботі з запропонованим алгоритмом розпізнавання, центр уваги зміщений на вибір систем Т – опорних множин за початковою інформацією $I(l)$, що дає змогу зменшити час класифікації довільного об’єкта S за рахунок попередньої обробки масиву начальної вибірки.

2) Формальне визначення алгоритму розпізнавання на основі Т – опорних множин дозволяє ввести нові означення інформативності Т – опорних множин щодо довільного об‘єкту S та класу H_j.

3) Важливим фактом є те, що не існує алгоритму розпізнавання з моделі $R(U,H)$, який би видавав помилки на даних початкової навчальної інформації $I(l)$ та відмови класифікації на всій множині допустимих об‘єктів.

4) Відмітимо, що алгоритми моделі $R(U,-)$ можуть видавати помилки розпізнавання на даних початкової навчальної інформації $I(l)$ та відмови класифікації на деяких допустимих об‘єктах.

5) Відповідно алгоритми моделі $R(-,H)$ не дають помилок розпізнавання на даних початкової навчальної інформації $I(l)$, проте дають відмови класифікації на деяких допустимих об‘єктах.

6) Загальна кількість усіх алгоритмів у моделі $R(-,-)$ (для довільних Т – опорних множин) буде дорівнювати 2^{3^n-1}, де $n-$ кількість ознак у задачі розпізнавання (інформаційному описі об‘єкта НВ).

7) Відмітимо, що ємність усіх моделей $R(-,-)$, $R(-,H)$, $R(U,-)$ та $R(U,H)$ є скінчена, це випливає з властивості самої моделі $R(-,-)$, оскільки всі інші являються її частиною.

8) Відмітимо також, що введення різних параметрів при обчисленні оцінок $\Gamma_j(S)$ не розширяє моделі $R(-,H)$, $R(U,H)$.

На наступному етапі даного дослідження розглянемо питання взаємозв‘язку та спільного використання моделей ЛДК та Т – опорних множин у задачах розпізнавання образів.

3.4. Взаємозв‘язок дерев класифікації та Т – опорних множин, розширення моделі алгоритмів розпізнавання, які засновані на концепції ЛДК

Розглянемо в даній частині роботи деякі прості можливості застосування логічних дерев для вирішення поставлених задач розпізнавання образів. Нехай маємо бінарний випадок, тобто $M_i \in \{0,1\}, i = 1, \dots, n$.

Визначення 3.11. ЛДК – це зв‘язаний граф без циклів, в некінцевих вершинах якого знаходяться ознаки (узагальнені ознаки, алгоритми, узагальнені алгоритми для випадку АДК), а ребра нумеруються значеннями цих ознак. У кінцевих вершинах (кінцевого ярусу) ЛДК знаходяться значення функції, що задає розбиття початкової інформації на класи або символ Δ невизначеності. При чому будуть виконуватись такі умови:

1) на фіксованому шляху в ЛДК одна і та сама ознака зустрічається тільки один раз (випадок регулярного дерева);

2) об’єктам із $I(l)$ відповідає значення функції, що задає розбиття при умові початкової відмінності між об’єктами різних класів (виконання гіпотези несуперечливості);

3) довільному об’єкту $I(l)$ відповідає одно із значень множини $\{0,1, \dots, l\}$.

3.4.1. Класифікатор для структури ЛДК

У даний час існують різні алгоритми та методи побудови ЛДК, що задовольняють даному визначенню та інші [94].

Визначення 3.12. Правилом розпізнавання (класифікації) являється деякий оператор П, що приводить довільний допустимий об‘єкт S за його інформаційним описом $I(S)$ в кінцеву вершину ЛДК (де знаходиться значення функції розпізнавання для даного об‘єкту).

Визначення 3.13. Деякий алгоритм A називається алгоритмом розпізнавання (класифікації), якщо він переводить початкову інформацію $I(l)$ та інформаційний опис довільного числа q допустимих об’єктів $I(q)$ у вектор a_{r+q}, складений з елементів $\{0,1, \dots, l\}$, де $r = |I(l)| -$ кількість об’єктів, що

складають початкову інформацію $I(l)$, при чому об'єктам із $I(l)$ відповідає значення функції, що задає розбиття, при умові початкової відмінності між об'єктами різних класів. Очевидно, що фіксоване ЛДК, побудоване за початковою інформацією $I(l)$ із заданим правилом розпізнавання (класифікації) задає деякий алгоритм розпізнавання A.

3.4.2. Т – опорна множина в структурі ЛДК

Теорема 3.4. Довільне ЛДК (за даними деякої початкової НВ) задає систему Т – опорних множин, причому кількість Т – опорних множин у цій системі дорівнює кількості кінцевих вершин ЛДК, а кількість інформативних Т – опорних множин до заданих класів дорівнює кількості визначених вершин ЛДК (рис. 3.1). Для даного прикладу загальна кількість Т – опорних множин складає 8, причому серед них будуть інформативними до класу H_1 – 2, інформативними до класу H_2 – 3, інформативними до класу H_3 – 3.

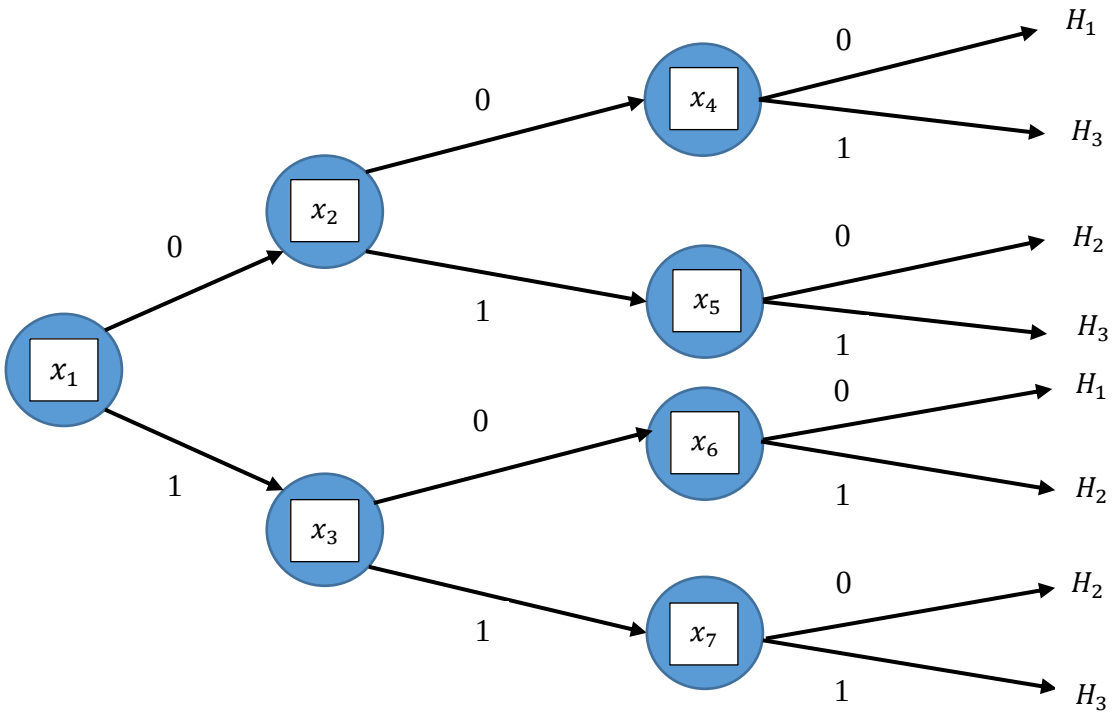


Рис. 3.1. Приклад ЛДК, яке задає вісім Т – опорних множин.

Зауваження. Під визначеною вершиною ЛДК розуміємо вершину, в якій знаходиться конкретна мітка (лист дерева) зі значенням ФР (число або

константа). Тобто визначена вершина зазвичай фіналізує деякий шлях (схему розпізнавання) в ЛДК.

Наслідок. Якщо в ЛДК всі кінцеві вершини визначені, то воно задає просту систему Т – опорних множин.

Доведення. Зафіксуємо в ЛДК деяку його визначену вершину. Зрозуміло, що цій вершині відповідає цілком визначений шлях у ЛДК, що зв'язує її з початковою вершиною ЛДК та характеризується деяким набором ознак і їх фіксованих значень. Це ніщо інше, як Т – опорна множина. Виконаємо аналогічну процедуру з всіма кінцевими вершинами (листами дерева) ЛДК. Очевидно, що кожна кінцева вершина в ЛДК задає деяку Т – опорну множину. Якщо кінцева вершина ЛДК визначена, то Т – опорна множина, відповідна цій вершині, буде інформативною до класу, номер якого знаходиться в цій вершині, так як об'єкти, що попадають у цю вершину, належать одному фіксованому класу.

Доведення наслідку. Оскільки всі кінцеві вершини ЛДК визначені, то:

1) дерево визначає тільки інформативні Т – опорні множини до відповідних класів;

2) кожний клас у ЛДК характеризується, щонайменше, однією кінцевою вершиною, яка в свою чергу визначає інформативну Т – опорну множину до цього класу;

3) довільний допустимий об'єкт класифікації S за допомогою деякого правила розпізнавання (класифікації) приводиться в одну з кінцевих вершин ЛДК, але ця кінцева вершина визначає інформативну Т – опорну множину по відношенню до нього (рис. 3.1).

3.4.3. Випадок структури ВДК

Визначимо формально алгоритм побудови деякого ЛДК – візьмемо за основу, наприклад, алгоритм, описаний у [95]. Як правило, всі відомі алгоритми побудови ЛДК засновані на цій ідеї [291-297]. Відмінність їх полягає у виборі оцінки інформативності ознак на відповідному кроці побудови ЛДК. Якщо опустити оцінку інформативності ознак при побудові

ЛДК, то можна ввести поняття випадкового дерева класифікації (ВДК), схему побудови та основні властивості якого вже розглядалися в даному дослідженні.

Визначення 3.14. ВДК називається ЛДК, ознаки у вершинах якого в процесі його побудови вибираються випадковим чином (програмним генератором PRG) [112]. Будемо розглядати тільки такі ВДК, на фіксованому шляху яких одна і та сама ознака зустрічається тільки один раз (ВДК першого типу). Очевидно, що довільне ЛДК можна звести до такого без втрати інформації. Множина алгоритмів, що визначаються простими системами Т – опорних множин, що задаються ВДК, являється неповною. Цей факт легко доводиться контр-прикладом. Для розширення цієї моделі введемо поняття оператора над ВДК. Нехай $D = \{D_1, \dots, D_\xi\}$ – деяка фіксована множина ВДК, що побудована для конкретної прикладної задачі розпізнавання (наприклад за алгоритмом із роботи [112]). Нехай $D_i \in D$ та C – довільна його вершина, що однозначно відповідає його піддереву (фіксує в його структурі) $D_i(C)$ та $D_j \in D, j \neq i$ (рис. 3.2). Уведемо операцію $\coprod$ над D_i та D_j, яка полягає в заміні $D_i(C)$ у D_i деревом розпізнавання D_j, тобто $D_i^j = (D_i(C))\coprod D_j$.

Теорема 3.5. $D_i^j = (D_i(C))\coprod D_j$ являється деревом розпізнавання, $i, j = 1, \dots, \xi$, де ξ– загальна кількість ЛДК (або ВДК), C – довільна вершина ЛДК D_i.

Визначення 3.15. Нехай D – деяка множина ВДК. Замиканням $[D]$ множини D називається сукупність всіх ЛДК з P_k, які одержані в результаті дії оператора підстановки над ВДК з D.

Операція отримання $[D]$ з D називається операцією замикання.

Визначення 3.16. Деяка множина ВДК – P , для яких виконується умова $[P] = D$ називається неприводимою системою, якщо замикання любої власної підмножини P' з P ($P' \subset P$) відмінно від замикання всієї підмножини P, тобто $[P'] \subset [P]$, $[P'] \neq [P]$.

Визначення 3.17. Неприводима, повна в замкнутому класі система ВДК називається базисом множини ВДК – D.

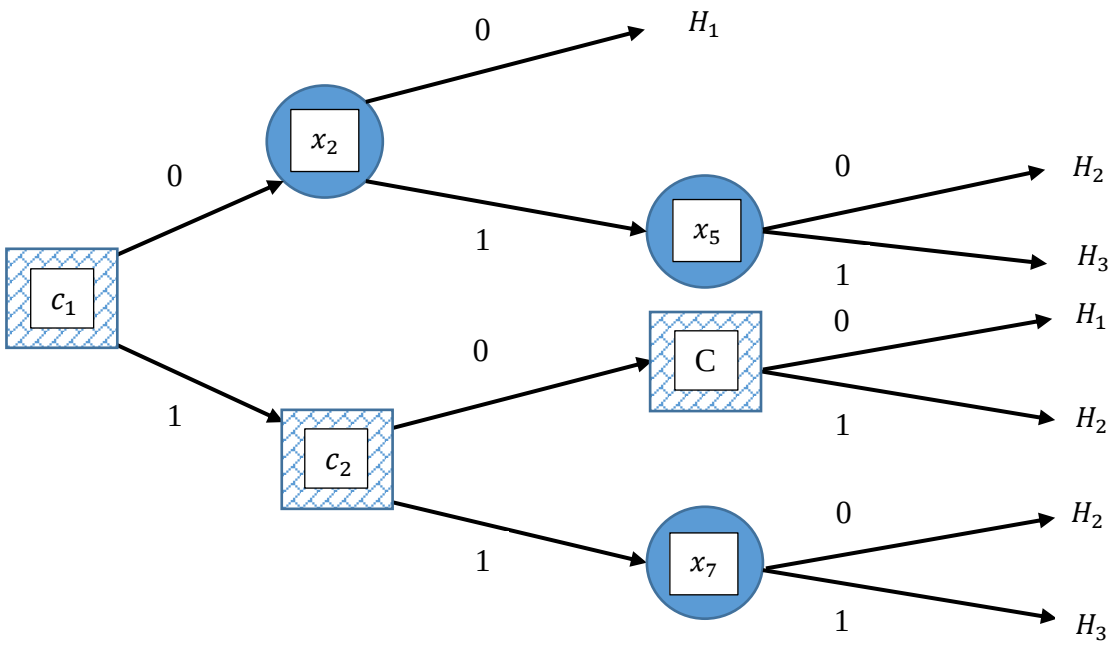


Рис. 3.2. Виділення вершиною C піддерева $D_i(C)$ в структурі ЛДК D_i.

Визначення 3.18. Правило розпізнавання, класифікації (оператор $\prod$) задіює ознаки $P_{i_1}, \dots, P_{i_\xi}$ (у вигляді вершин ЛДК) за допомогою інформації $I(S)$, якщо вони зустрічаються на шляху, що приводить цей об‘єкт у кінцеву вершину ЛДК.

Позначимо цей факт наступним чином: $\prod\big(I(S)\big) = \{P_{i_1}, \dots, P_{i_\xi}\}$. Запис $D_i\big(I(S)\big) = j$ означає віднесення об‘єкта S до класу j за допомогою деякого ЛДК – D_i. Через $\prod_i$ позначимо правило розпізнавання, класифікації (оператор) для фіксованого ЛДК D_i.

Теорема 3.6. Множина $D = \{D_1, \dots, D_\xi\}$ утворює базис задачі розпізнавання, якщо виконуються такі умови (визначеності та повноти):

1) $\forall S \in I(q), \forall j, j = 1, \dots, l, \exists i, i = 1, \dots, \xi, D_i\big(I(S)\big) = j$;

2) $\prod_1(I(S)) \cup \dots \cup \prod_\xi(I(S)) = \{P_1, \dots, P_n\}$.

Зауважимо, що n – загальна кількість ознак задачі розпізнавання (в інформаційному описі об‘єкта НВ), S – об’єкт із множини допустимих об’єктів $I(q), D_i\big(I(S)\big) = j$ – віднесення деякого об’єкта S до класу j за допомогою ЛДК D_i та його інформаційного опису $I(S)$.

Отже, зважаючи на все вище зазначене, для даної частини роботи можна зробити наступні висновки:

1) На сьогодні існують різноманітні методи побудови як ЛДК з одноразовим використанням ознак у структурі логічного дерева (алгоритми випадкових дерев, метод розгалуженого вибору ознак із початковою оцінкою інформативності), так і ЛДК із повторами різних ознак на ярусах логічного дерева (алгоритм побудови ЛДК з покроковою оцінкою важливості ознак).

2) Деяке правило розпізнавання (класифікації) приводить довільний допустимий об‘єкт S за його інформаційним описом $I(S)$ у кінцеву вершину ЛДК, тобто в деякій мірі задає фіксований шлях у структурі логічного дерева.

3) Якщо зафіксувати деяку вершину C у структурі довільного ЛДК – D, то вона буде однозначно виділяти певне піддерево D^* (фіксувати в його структурі).

4) Замиканням $[D]$ деякої множини ВДК – D називається сукупність усіх ЛДК, які одержано в результаті дії оператора підстановки над ВДК із множини логічних дерев D.

5) Фіксовану множину логічних дерев P, для яких виконується умова $[P] = D$ будемо називати неприводимою системою, якщо замикання любої власної підмножини P' з P ($P' \subset P$) відмінно від замикання всієї підмножини P ($[P'] \subset [P]$, $[P'] \neq [P]$).

6) Неприводима, повна в замкнутому класі система ВДК називається базисом множини ВДК – D.

7) Довільне правило розпізнавання, класифікації задіює ознаки $P_{i_1}, \dots, P_{i_\xi}$ (у вигляді вершин ЛДК) за допомогою інформації $I(S)$, якщо вони зустрічаються на шляху, що приводить цей об‘єкт у кінцеву вершину ЛДК.

8) Множина $D = \{D_1, \dots, D_\xi\}$ утворює базис задачі розпізнавання, якщо виконуються умови визначеності та повноти.

3.5. Особливості дерев моделей класифікації та розпізнавання

На даному етапі дослідження піднімемо важливе питання – особливостей дерев моделей класифікації та розпізнавання (ДМК). Нехай $\eta_k = \{i_1, \dots, i_k\}, k \in \{1, \dots, n\}, I\left(S_j^{\eta}\right) = \left(b_{i_1}^j, \dots, b_{i_k}^j\right) -$ частковий опис деякого допустимого об'єкта S_j, $b_i^j \in M_i$, де M_i $-$ ознакова область (метричний простір із метрикою $p_i, i = 1, \dots, n$), P_{η_k} – деякий $k -$ товий предикат $P_{\eta_k}(b_{i_1}^j, \dots, b_{i_k}^j)$, заданий на множині часткових (ознакових) описів допустимих об'єктів S_j, $\{I(_j^{\eta_k}), P_{\eta_k}\} -$ множина моделей, що представляють деякі алгебраїчні системи розпізнавання у вигляді логічних дерев. При фіксованому η_k та P_{η_k} будемо мати деяку конкретну модель $\{I(_j^{\eta_k}), P_{\eta_k}\}$.

3.5.1. Дерево моделей та дерево моделей класифікації

Визначення 3.19. По аналогії з логічним деревом, дерево моделей (ДМ) – це зв'язаний граф без циклів, в некінцевих вершинах якого знаходяться фіксовані моделі з $\{I(_j^{\eta_k}), P_{\eta_k}\}$, а ребра нумеруються значеннями предикатів із цих моделей. У кінцевих вершинах розташовані значення функції, що задає розбиття $I(l)$ на класи або символ Δ невизначеності. Причому на фіксованому шляху ДМ одна й та сама модель може зустрічатися тільки один раз.

Визначення 3.20. Аналогічно, правилом розпізнавання (класифікації) ДМ являється деякий оператор, що відносить довільний допустимий об'єкт S (за його початковим інформаційним описом $I(S)$) у фіксовану кінцеву вершину дерева моделей.

Визначення 3.21. Аналогічно, дерево моделей класифікації та розпізнавання (ДМК) – це деяке ДМ із заданим правилом розпізнавання (класифікації), для якого об'єктам із $I(l)$ відповідає значення функції розпізнавання, що задає розбиття $I(l)$ на класи.

Очевидно, що довільне ДМК являється алгоритмом розпізнавання та класифікації (у вигляді логічного дерева) деяких об'єктів за початковою інформацією НВ.

3.5.2. Визначення інформативності моделі по відношенню до об‘єкта та класу

Отже, відповідно до приведеного визначення ДМК можна запропонувати такі означення інформативності по відношенню до об‘єкту S та інформативності для деякого класу j за початковою інформацією $I(l)$.

Визначення 3.22. Деяка модель $\{I(_j^{\eta_k}), P_{\eta_k}\}$ називається інформативною по відношенню до об‘єкту S, якщо значення предиката P_{η_k} вірно на його частковому (ознаковому) описі.

Визначення 3.23. Модель $\{I(_j^{\eta_k}), P_{\eta_k}\}$ називається інформативною для деякого класу j за початковою інформацією $I(l)$, якщо вона інформативна тільки для деяких об‘єктів цього класу. Відносно об‘єктів, що не входять у $I(l)$, ніяких обмежень не накладається.

Визначення 3.24. Просте ДМК – це ДМК, на кожному ярусі якого, крім початкового, знаходяться по дві вершини.

Теорема 3.7. ДМК, у вершинах якого знаходяться моделі інформативні по відношенню до відповідних класів, є простим.

Нехай $V = \{V_1, \dots, V_\xi\}$ – деяка множина ДМК, побудованих для фіксованої задачі класифікації. Введемо наступний оператор підстановки над V. Нехай $V_i \in V$ та D_j полягає в заміні $D_i(C)$ у D_j ДМК $D_j, \rightarrow V_i^j = (V_i, C) \cup V_j$.

Теорема 3.8. $V_i^j = (V_i, C) \cup V_j$ представляє собою дерево.

Ця теорема дозволяє ввести операцію замикання над ДМК, таку же саму як над ЛДК, та дослідити можливість знаходження базису задачі над множиною ДМК. Відмітимо також принциповий момент, що в якості $\{I(_j^{\eta_k}), P_{\eta_k}\}$ (тобто вершин, вузлів логічного дерева) можна взяти любі відомі алгоритми розпізнавання (отже, певною мірою ми вперше прийшли до концепції АДК, у вершинах якого знаходяться окремі, автономні алгоритми розпізнавання та класифікації). Тоді можна зафіксувати, що таке ДМК буде представляти собою орієнтований граф без циклів (по аналогії з ЛДК), у

некінцевих вершинах якого знаходяться відомі алгоритми розпізнавання (вершини алгоритмічного дерева є окремими алгоритмами), стрілки нумеруються значеннями алгоритму на об‘єктах, а в кінцевих вершинах знаходяться деякі значення функції розпізнавання, що задає початкове розбиття інформації $I(l)$ на класи.

Отже, зважаючи на все вищезазначене, можна зробити наступні висновки:

1) Деяка множина моделей $\{I(_j^{\eta_k}), P_{\eta_k}\}$, що представляють алгебраїчні системи розпізнавання (у вигляді структур дерев класифікації) при фіксованому η_k та P_{η_k} перетворюється на конкретну модель $\{I(_j^{\eta_k}), P_{\eta_k}\}$ (деревоподібну логічну структуру).

2) ДМ представляє собою зв‘язаний граф без циклів, у некінцевих вершинах якого знаходяться фіксовані моделі з $\{I(_j^{\eta_k}), P_{\eta_k}\}$, а ребра нумеруються значеннями предикатів із цих моделей, у кінцевих вершинах знаходяться значення ФР, що задає розбиття $I(l)$ на класи, а на фіксованому шляху ДМ одна й та сама модель може зустрічатися тільки один раз.

3) Правилом розпізнавання (класифікації) ДМ являється деякий оператор, що відносить довільний допустимий об‘єкт S (за його початковим інформаційним описом) у фіксовану кінцеву вершину ДМ.

4) ДМК – це деяке ДМ із заданим правилом розпізнавання (класифікації), для якого об’єктам з НВ відповідає значення ФР, що задає розбиття початкової вибірки на класи.

5) Просте ДМК – це ДМК, на кожному ярусі якого, крім початкового, знаходяться по дві вершини, а ДМК, у вершинах якого знаходяться моделі інформативні по відношенню до відповідних класів, є простим.

6) ДМК, у вершинах якого знаходяться моделі інформативні по відношенню до відповідних класів – є простим ДМК.

3.6. Експертні системи в розрізі логічних дерев класифікації

У цій частині дослідження піднімемо важливе питання експертних систем у розрізі концепції дерев класифікації в задачах розпізнавання образів.

Нехай $H = H_1 \cup \ldots \cup H_l$, де $H_1, \ldots, H_l$ – класи об'єктів (образи), $I(l) = \{w_1, \ldots, w_m\}$ – деякі об'єкти з множини H, відносно яких відома приналежність до визначених класів, причому маємо наступну ситуацію:

$$\forall H_j, j = 1, \ldots, l;\ \exists t, t = 1, \ldots, m, w_t \in H_j.$$

Аналогічно $I(g) = \{w_1^{'}, \ldots, w_g^{'}\}$ – деяка множина об'єктів із H, відносно яких невідома приналежність до класів. Тобто в задачі розпізнавання задано деяку НВ та необхідно провести процес класифікації невідомих об‘єктів.

Отже, необхідно на основі цієї початкової інформації та деяких апріорних даних $\mu(H)$ побудувати алгоритм, який би довільний (але допустимий) об‘єкт $w \in H$ за його інформаційним (ознаковим) описом $I(w)$ відніс до визначеного класу H_l.

Відмітимо, що багато задач розпізнавання може бути зведено до такої постановки. Наприклад, якщо w – об‘єкт, то за допомогою деяких алгоритмів A_i, $i = 1, \ldots, k$ любому дискретному об‘єкту можна поставити у відповідність його опис $I(w)$, причому тут алгоритми виступають в якості ознак об‘єкта.

Алгоритми розпізнавання будемо представляти у вигляді деревоподібних схем (тобто ЛДК).

Нехай $f \in G_{|I(l)|}$ та $f^{'} \in G_{|I(l)|}$, де:

$$f^{'} = \begin{cases} f(a_1, \ldots, a_n), \text{якщо } (a_1, \ldots, a_n) \in I(l) \\ \bar{f}(a_1, \ldots, a_n), \text{якщо } (a_1, \ldots, a_n) \notin I(l) \end{cases}.$$

Як відомо, довільну функцію k – значної логіки можна представити у вигляді деякого ЛДК, і навпаки, довільному ЛДК можна поставити у відповідність функцію k – значної логіки (даному питанню буде приділено значну увагу в наступному розділі даного дослідження). Тоді, на основі цього факту, можна зробити наступний висновок.

Теорема 3.9. Деяке ЛДК, що реалізує функції $\{f, f', \dots, f'^{\dots l-1 \dots'}\}$ з $G_{|I(l)|}$, утворює базис задачі Z_k (ознаки приймають не більше, як k різних значень), зокрема l – кількість класів.

Доведення. Для доведення даної теореми на першому етапі зафіксуємо довільний набір, що не входить у $I(l)$ та покажемо, що за допомогою фіксованого ЛДК, яке реалізують функції $\{f, f', \dots, f'^{\dots l-1 \dots'}\}$ з $G_{|I(l)|}$, та операції ∪ над ЛДК можна побудувати такі дерева, які би цей набір віднесли до різних класів, тобто $-\{1,2,\dots,l\}$.

Відмітимо також, що ЛДК, які реалізують функції $\{f, f', \dots, f'^{\dots l-1 \dots'}\}$ відносять фіксований набір до різних класів. Це безпосередньо випливає із визначення функції f'.

З іншого боку слід відмітити, що ЛДК, яке відповідає довільній із функцій $\{f, f', \dots, f'^{\dots l-1 \dots'}\}$, можна поставити у відповідність повне ЛДК, яке еквівалентне даному логічному дереву. Тоді для довільного фіксованого набору можна зробити – фактично довільне значення шляхом виконання однієї операції ∪ у кінцевій вершині з ЛДК, яке дає потрібне значення. Відмітимо, що ця операція зводиться до звичайної заміни одного значення на інше, яке потрібне за умовами на даному етапі та має математичне обгрунтування.

Отже, визначення – утворює базис задачі Z_k – фактично означає, що за допомогою деякого ЛДК, яке реалізують функції $\{f, f', \dots, f'^{\dots l-1 \dots'}\}$ та операції ∪ над даними ЛДК, можна побудувати фіксоване логічне дерево, яке реалізує довільну функцію з Z_k, тобто функцію з $G_{|I(l)|}$.

Відмітимо, що довільному ЛДК (деякої прикладної задачі) можна поставити у відповідність повне логічне дерево класифікації (ПЛДК). Якщо в якості вершини піддерева вибрати кінцеву вершину ПЛДК та проводити операцію ∪ з довільним ЛДК із $G_{|I(l)|}$,–то результатом буде проста заміна одного значення іншим. Це повністю доводить теорему.

Найпростіше доведення, яке, проте, на практиці не використовується, полягає в наступній схемі: ЛДК $\rightarrow$ ПЛДК $\rightarrow$ заміна значень у кінцевих вершинах довільного ЛДК.

Нехай маємо деякий колектив експертів, який представляє апріорну інформацію $U(H)$. Найбільш трудомістким процесом при побудові експертних систем являються етап отримання та формалізація знань експертів про дану задачу.

Пропонується формалізований процес отримання цих знань. Тобто експертам пропонуються деякі фіксовані твердження, які сформульовані за допомогою деякого логічного дерева (структури ЛДК) на мові, що близька до природних програмних інваріант (IF, THEN, ELSE), які безпосередньо фіксуються структурою самого логічного дерева.

Отже, задачею експертів буде погодитися або ні з даним твердженням (представленим у вигляді ЛДК). Результати відповідей експертів ставляться в кінцеві вершини деякого ЛДК.

Фактично теорема (3.9) говорить про те, що при виконанні достатньо простих умов, що отримуються на основі інформації $I(l)$, можна побудувати довільний алгоритм розпізнавання для даної задачі. Завдання експертів – обмежити кількість розв'язків.

Отже $I(l)$, $I(w)$ будемо називати первинною базою даної задачі, $\{D_1^E, \dots, D_\eta^E\}$ – базою знань, а D_j^E – ЛДК з оцінками експертів. Тобто побудова експертної системи буде полягати в знаходженні найбільш прийнятного розв'язку задачі.

Підсумовуючи, можна навести наступні практичні висновки:

1) Деяке логічне дерево, що реалізує функції $\{f, f', \dots, f'^{\dots l-1 \dots'}\}$ з $G_{|I(l)|}$, утворює базис задачі Z_k, причому ознаки приймають не більше як k різних значень, а $l -$ кількість класів у задачі.

2) Для розв'язку задачі Z_2 достатньо побудувати одне ЛДК – f, далі, за допомогою алгоритму донавчання з ЛДК – f, будуємо ЛДК – $f^{'}$, тобто базис задачі Z_2.

3) Аналогічно, для задачі Z_k: достатньо побудувати одне ЛДК – f, далі, за допомогою алгоритму донавчання з ЛДК – f будуємо ЛДК – $\{f, f^{'}, \dots, f^{'\dots l-1\dots'}\}$, тобто базис задачі Z_k.

4) Складність прийняття рішень у задачі Z_2 (незалежно від кількості класів) не перевищує n бінарних порівнянь, де n – кількість ознак у задачі (розмірність ознакового простору задачі).

5) Складність прийняття рішень у деякій задачі Z_k не перевищує $(k-1)*n$ порівнянь.

6) Існує можливість ефективної побудови експертних систем на основі отриманих результатів.

7) Існує можливість ефективного використання до цього відомих алгоритмів (із різними евристиками) при побудові нових систем (моделей) розпізнавання, класифікації, в тому числі побудови експертних систем у розпізнаванні.

8) Відносна простота програмної реалізації даного підходу (розв'язувались тестові задачі з 3000 бінарних ознак та 10 – тьма класами).

9) Складність побудови деякого ЛДК за початковою інформацією $I(l)$ еквівалентна числу $Q*(n*r)$, зокрема $r=|I(l)|$, а $Q-$ деякий коефіцієнт.

3.7. Метод представлення дискретних об‘єктів на основі Т – опорних множин у задачах розпізнавання образів

При розв’язуванні задач розпізнавання (класифікації) дискретних об‘єктів зазвичай маємо наступну ситуацію: в даний час накопичено значну кількість програмних та апаратних інструментів, які вирішують деякі часткові задачі, підзадачі (опис або представлення) об‘єктів, виділення характерних ознак (структурних елементів) об‘єктів та інше.

Використання накопиченого досвіду при створенні програмного інструментарію для розв‘язку нових задач у цій області (в сенсі запозичення деяких програм або їх частин, модулів, елементів) пов’язано з великими труднощами (вивчення їх особливостей, декомпозиції, формалізація, інтеграція та інше). Тим більше, слід відмітити, що цей процес цілком залежить від досвіду розробника та потребує для його реалізації значних матеріальних, інтелектуальних, і що найкритичніше – часових витрат.

Подібна ситуація існувала при розв’язуванні практичних задач розпізнавання або класифікації образів до створення алгебраїчного підходу [55], який дав змогу формалізувати цей процес. Безпосереднє використання алгебраїчного підходу для вирішення широкого кола задач розпізнавання або класифікації, пов‘язано з деякими принциповими проблемами, головна з яких – адекватне представлення об‘єктів наборами значень ознак. Сьогодні є цілий ряд підходів, методів та алгоритмів для виділення ознак дискретних об‘єктів та пакети інструментальних програм для їх реалізації. Проте, залишається актуальною задача знаходження системи оптимальних (у певному сенсі) ознак, тобто пошук таких властивостей об‘єктів $P_1, \dots, P_n$ (визначення та фіксація ознакового простору), в просторі яких класифікація (розпізнавання) була би можливою та не дуже складною (економічно вигідною).

Використання існуючих програмних систем (ПС) з цією метою становиться можливим лише при наявності методів, які би за результатами роботи різних ПС дозволили виділяти системи ознак, що мають інтерес для розробника відносно умов поточної задачі. Причому, для кожної практичної

задачі класифікації об‘єктів системи ознак, що мають визначальне значення (важливі відносно фіксованої задачі або класу задач), як правило, різні і їх потрібно заново визначати.

Припустимо, що знайдено систему ознак (яка має важливість для фіксованої задачі з певної точки зору) для фіксованої задачі розпізнавання об‘єктів – тоді, застосовуючи алгебраїчний підхід, можна побудувати коректний алгоритм розв‘язку для даної поставленої задачі.

Звідси стає очевидним, наскільки важлива задача знаходження оптимальних (у певному сенсі) систем ознак. Часто задачу знаходження оптимальних систем ознак зводять до задачі мінімізації вихідного опису об‘єкту, проте, це стосується лише випадку, коли оптимальна система ознак є серед множин ознак, які задають опис об‘єктів, що являється як правило, тільки припущенням.

Звичайно, мінімізуючи вихідні описи деяких об‘єктів за результатами роботи різних ПП (причому вони можуть бути абсолютно незалежними та створеними різними розробниками), ми таким чином мінімізуємо вихідний набір ПП, алгоритмів у них. Коректний алгоритм розпізнавання при цьому вказує оптимальну послідовність їх роботи та загальну структуру програмного забезпечення для вирішення поставленої задачі. Такий підхід повинен забезпечити значне скорочення матеріальних, інтелектуальних та часових затрат при виникненні нових (таких, що раніше не зустрічалися) задач класифікації об‘єктів: від їх постановки до створення простого програмного забезпечення для їх вирішення.

Зауважимо, що немає сумнівів відносно певної складності даного підходу, але він значно простіший, ніж створення інтелектуальних програмних комплексів на основі інтеграції різних ПП (враховуючи, що немає ніяких гарантій, що дана задача може бути вирішена за допомогою цього комплексу, набору алгоритму). Відмітимо також, що алгебраїчний підхід достатньо складний у зв‘язку із значними, апаратними, обчислювальними затратами та з великим об‘ємом потрібної пам‘яті.

Розробка логіко-алгебраїчного підходу для розв‘язку задач класифікації та розпізнавання образів (на основі концепції ЛДК/АДК) ставить за мету одержання математичних методів та відповідного супровідного програмного забезпечення вирішення широкого класу практичних задач теорії штучного інтелекту.

3.7.1. Т – опорні множини як атрибути дискретних об‘єктів

Знову, нехай маємо, що $H = H_1 \cup ... \cup H_l$, де $H_1, ..., H_l$ – класи дискретних об‘єктів. Відповідно $I(l) = \{w_1, ..., w_m\}$ – деякі об‘єкти з множини H, відносно яких відома приналежність до визначених класів, причому: $\forall H_j, j = 1, ..., l;\ \exists t, t = 1, ..., m, w_t \in H_j$. Аналогічно $I(g) = \{w_1^{'}, ..., w_g^{'}\}$ – множина об‘єктів із H, відносно яких невідома їх класова належність.

Необхідно на основі цієї початкової інформації та деяких апріорних даних $U(I)$ побудувати алгоритм класифікації, який би довільне допустиме значення (об‘єкт класифікації) $w_i \in H$ відніс до визначеного класу.

Будемо вважати, що первинна інформація про деяке зображення (об‘єкт), яка доступна для обробки, задана у вигляді фіксованої матриці. Якщо кожному об‘єкту $w_i \in H$ можна поставити у відповідність деякий вектор $I(w_j) = (a_1^j, ..., a_n^j)$ за допомогою деяких алгоритмів $A_i, i = 1, ..., k$, то маємо класичну задачу розпізнавання образів, де алгоритми A_i виступають в якості інструменту синтезу ознак об‘єкта.

На першому етапі введемо деяку адресацію пікселів зображення – фіксовану однозначну адресацію пікселів зображення назвемо координатною сіткою. Для спрощення тепер і далі цю адресацію будемо представляти позиційними системами числення (двійковою, трійковою тощо), а відповідні розряди (позиції) будемо позначати $P_1, ..., P_k$.

Наприклад, нехай задано зображення розміром $2^n \times 2^m$. Якщо в якості позиційної системи числення вибрати двійкову систему, то позначення її розрядів (позицій) буде мати вигляд: $P_1, P_2, ..., P_n, P_{n+1}, ..., P_{n+m}$. Будемо

застосовувати два типи координатних сіток: зовнішню та внутрішню. На першому етапі зафіксуємо деяку систему числення K.

Твердження 3.1. Для фіксованої системи числення K зовнішню координатну сітку для довільного зображення розміром $N \times M$ можна вибрати $N! \times M!$ способом.

Твердження 3.2. Для фіксованої системи числення K внутрішню координатну сітку для довільного зображення розміром $N \times M$ можна вибрати $(N \times M)!$ способом.

Зауваження. Множина зовнішніх координатних сіток міститься у множині внутрішніх координатних сіток.

Нехай для деякого класу зображень зафіксовано координатну сітку та систему числення K.

Визначення 3.25. Т – опорна множина – це фіксований набір символів $P_{i_1}, \dots, P_{i_k}$ з фіксованими їх значеннями: $\binom{e_{i_1}}{P_{i_1}}, \dots, \binom{e_{i_k}}{P_{i_k}}$, $e_{i_k} \in \{0,1, \dots, k-1\}$, $j = 1, \dots, k$, $k \in \{1,2, \dots, R\}$, а $R -$ кількість позицій, необхідних для представлення координатної сітки.

Твердження 3.3. Т – опорна множина $\binom{e_{i_1}}{P_{i_1}}, \dots, \binom{e_{i_k}}{P_{i_k}}$, характеризує цілком визначену частину зображення з кількістю пікселів, що дорівнюють $\eta = K^{R-k}$.

3.7.2. Особливості 2Т – опорних множин

Визначення 3.26. Т – опорну множину $\binom{e_{i_1}}{P_{i_1}}, \dots, \binom{e_{i_k}}{P_{i_k}}$ з наперед заданими значеннями пікселів будемо називати 2Т – опорною множиною.

Нехай $S = (a_1, \dots, a_\eta)$ – множина пікселів зображення, що характеризується 2Т – опорною множиною $\binom{e_{i_1}}{P_{i_1}}, \dots, \binom{e_{i_k}}{P_{i_k}}$, а $S' = (b_1, \dots, b_\eta)$ – деякий набор пікселів. Зафіксуємо числа $q_1, q_2 \geq 0$ та вектор $(\varepsilon_1, \dots, \varepsilon_\eta)$, $\varepsilon_i \geq 0$. Уведемо метрику p_{i_j} між a_{i_j} та b_{i_j}.

Отже, зважаючи на все вище зазначене, та по аналогії з пунктом 3.2.1 даного розділу, можемо ввести наступні визначення інформативності відносно деякого зображення s та класу розпізнавання H_j.

Визначення 3.27. Т – опорна множина $\binom{e_{i_1}}{P_{i_1}}, \dots, \binom{e_{i_k}}{P_{i_k}}$, називається інформативною до об‘єкта s (зображення), якщо в системі нерівностей $p_1(a_1, b_1) \le \varepsilon_1, \dots, p_\eta(a_\eta, b_\eta) \le \varepsilon_\eta$ виконано не менше q_1 та не більше q_2 нерівностей.

Визначення 3.28. Т – опорна множина $\binom{e_{i_1}}{P_{i_1}}, \dots, \binom{e_{i_k}}{P_{i_k}}$ називається інформативною по відношенню до деякого $j-$ тового класу зображень за початковою інформацією $I(l)$, якщо в фіксованій системі нерівностей $p_1(a_1^j 0, b_1) \le \varepsilon_1, \dots, p_\eta(a_\eta^i 0, b_\eta) \le \varepsilon_\eta$ виконано не менше q_1 та не виконано не більше q_2 нерівностей для певних зображень тільки $j-$ тового класу. Відносно зображень, що не входять у $I(l)$, ніяких обмежень не накладається.

На наступному етапі введемо позначення $\Omega^{2T} = \{w_1, \dots, w_\xi\}$ де w_i – ($i-$това) 2Т – опорна множина, ξ – загальна кількість 2Т – опорних множин, H_j – множина зображень (клас), що належать класу j. Той факт, що 2Т – опорна множина w інформативна до класу H_j, будемо позначати наступним чином: $w * H_j$, аналогічно $w * s$ – інформативність 2Т – опорної множини w до зображення s. Запис $w * H_j$ означає, що існує, принаймні, одне таке зображення $s \in H_j$, що $w * s$.

Визначення 3.29. Систему 2Т – опорних множин $\Omega^{2T} = \{w_1, \dots, w_\xi\}$ назвемо інформативною, якщо виконуються такі умови:

1) $\forall i \exists j (w_i * H_j), i = 1, \dots, \xi, j = 1, \dots, l$;

2) $\forall s' \exists i (w_i * s')$;

3) $\forall s \in H_j \; \exists i (w_i * s)$.

Визначення 3.30. Систему 2Т – опорних множин $\Omega^{2T} = \{w_1, \dots, w_\xi\}$ назвемо несуперечливою, якщо:

1) $\forall i \exists j \left(w_i * H_j\right), i = 1, \dots, \xi, j = 1, \dots, l;$

2) $\forall s \in H_j \; \exists i (w_i * s);$

3) $\overline{\exists}(i_1, i_2), i_1 \neq i_2, \left(w_{i_1} * s \text{ та } w_{i_2} * s\right) \rightarrow \left(w_{i_1} * H_{j_1} \text{ та } w_{i_2} * H_{j_2}\right), j_1 \neq j_2.$

Визначення 3.31. Інформативну несуперечливу систему 2Т – опорних множин назвемо простою, якщо виконується умова $\forall s \exists! \, i(w_i * s)$.

Далі будемо використовувати 2Т – опорні множини в якості ознак, що характеризують зображення, тобто якщо 2Т – опорна множина інформативна по відношенню до відповідного зображення, то за ознакою приписується значення 1, в іншому випадку – 0. Для побудови алгоритмів розпізнавання або класифікації зображень використовуються розпізнаючи дерева моделей (алгебраїчних систем), у вершинах яких знаходяться 2Т – опорні множини.

Як підсумок, можна зробити наступні висновки:

1) Зовнішню координатну сітку для довільного зображення з фіксованою системою числення K, розміром $N \times M$ можна вибрати $N! \times M!$ способом.

2) Внутрішню координатну сітку для довільного зображення з фіксованою системою числення K, розміром $N \times M$ можна вибрати $(N \times M)!$ способом.

3) Т – опорна множина – це фіксований набір символів $P_{i_1}, \dots, P_{i_k}$ з фіксованими їх значеннями: $\begin{pmatrix} e_{i_1} & & e_{i_k} \\ P_{i_1}, & \dots, & P_{i_k} \end{pmatrix} (e_{i_k} \in \{0,1, \dots, k-1\}, \quad j = 1, \dots, k,$ $k \in \{1,2, \dots, R\}$, а $R -$ кількість позицій, необхідних для представлення координатної сітки), характеризує цілком визначену частину зображення з кількістю пікселів, що дорівнюють $\eta = K^{R-k}$.

4) Т – опорна множина $\begin{pmatrix} e_{i_1} & & e_{i_k} \\ P_{i_1}, & \dots, & P_{i_k} \end{pmatrix}$ називається інформативною до об‘єкта s (зображення), якщо в системі нерівностей $p_1(a_1, b_1) \leq \varepsilon_1, \dots, p_\eta\left(a_\eta, b_\eta\right) \leq \varepsilon_\eta$ виконано не менше q_1 та не більше q_2 нерівностей, називається інформативною по відношенню до деякого $j -$ тового класу зображень за початковою інформацією $I(l)$, якщо в фіксованій системі

нерівностей $p_1\left(a_1^j 0, b_1\right) \leq \varepsilon_1, \ldots, p_\eta\left(a_\eta^i 0, b_\eta\right) \leq \varepsilon_\eta$ виконано не менше q_1 та не виконано не більше q_2 нерівностей для певних зображень тільки $j-$тового класу.

5) Система 2Т – опорних множин $\Omega^{2T} = \{w_1, \ldots, w_\xi\}$ буде інформативною, якщо виконуються умови повноти, інформативності щодо класу та інформативності щодо зображення.

6) Система 2Т – опорних множин $\Omega^{2T} = \{w_1, \ldots, w_\xi\}$ буде несуперечливою, якщо виконуються умови неперетинання, інформативності відносно класу та інформативності відносно зображення.

7) Інформативна, несуперечлива система 2Т – опорних множин буде простою, якщо виконується умова одиничності $(\forall s \exists! \, i(w_i * s))$.

8) Важливим моментом є те, що 2Т – опорні множини можна використовувати в якості ознак, що характеризують деяке зображення, тобто якщо 2Т – опорна множина інформативна по відношенню до відповідного зображення, то за ознакою приписується значення – (True/1), в іншому випадку – (False/0), причому для побудови алгоритмів розпізнавання або класифікації зображень доречно використовувати дерева моделей класифікації (алгебраїчних систем), у вершинах яких знаходяться 2Т – опорні множини.

3.8. Задача мінімізації вихідного опису об‘єктів при побудові алгоритмів класифікації

Позначимо через $R\big(I(l)\big) = \big\{\big(I(s_1), f_R(s_1)\big), \dots, \big(I(s_m), f_R(s_m)\big)\big\}$ – початкову інформацію про зображення з множини H, де $f_R(s_m) \in \{1,2, \dots, l\}$, $(i = 1,2, \dots, m)$, m – кількість зображень в $I(l)$, $f_R(s)$ – деяка скінчено-значна функція, що задає розбиття множини H на класи (функція розпізнавання). Відношення $f_R(s_i) = j$, $(j = 1, \dots, l)$ означає, що $s_i \in H_j$.

Визначимо на початковій інформації $R\big(I(l)\big)$ наступні функціонали:

$$W(P_i) = \frac{1}{M}\sum_{j=G_i} \max_{0\le k\le l} b_j^k \tag{3.1}$$

Зауважимо, що M – множина об’єктів $I(l)$, G_i – множина значень P_i – тової ознаки, b_j^k – кількість значень j - P_i – тової ознаки в класі H_k,$(k = 1, \dots, l)$, $l -$ кількість класів в H.

$$W\big(P_i \backslash P_{i_1} = \eta_1, \dots, P_{i_t} = \eta_t\big) = \frac{1}{h}\sum_{j=G_i} \max_{0\le k\le l} b_j^k \tag{3.2}$$

Зауважимо, що $i \neq i_1, \dots, i_t$. Функціонал (3.2) обчислюється аналогічно функціоналу (3.1) з урахуванням того, що беруться тільки ті об’єкти, для яких $P_{i_1} = \eta_1, \dots, P_{i_t} = \eta_t,$ h – кількість таких об’єктів.

$$W\big(P_{i_1}, \dots, P_{i_t}\big) = \frac{1}{M}\sum_{\Delta_j=G_{\Delta_j}} \max_{0\le k\le l} b_{\Delta_j}^k \tag{3.3}$$

Зауважимо, що $\Delta_j = (x_{i_1}, \dots, x_{i_t})$, $(x_{i_k} \in G, k = 1, \dots, t)$ – фіксований набір значень ознак $P_{i_1}, \dots, P_{i_t}$, G_{Δ_j} – множина значень ознак, що відрізняються між собою значеннями хоча би однієї ознаки, $b_{\Delta_j}^k$ – кількість Δ_j класі H_k, $(k = 1, \dots, l)$.

$$W\big(P_{i_1}, \dots, P_{i_t} | P_{i_1} = \eta_1, \dots, P_{i_r} = \eta_r\big) = \frac{1}{h}\sum_{\Delta_j=G_{\Delta_j}} \max_{0\le k\le l} b_{\Delta_j}^k \tag{3.4}$$

Відмітимо, що функціонали (3.3) і (3.4) обчислюються аналогічно функціоналам (3.1) та (3.2) з урахуванням того, що беруться тільки ті зображення, для яких $P_{i_1} = \eta_1, \dots, P_{i_r} = \eta_r,$ h – кількість таких об’єктів.

Ці функціонали призначені для обчислення деякої міри (коефіцієнта), що характеризує роздільні можливості окремих ознак (відносно$f_R(s_i)$ за початковою інформацією $R\big(I(l)\big)$.

Відмітимо, що використовуючи функціонали (3.1) та (3.4), можна ввести похідні від них:

$$W(P_i) = W(P_1, P_2, \dots, P_{i-1}, P_{i+1}, \dots, P_n) \tag{3.5}$$

$$W\left(P_{i_1}, \dots, P_{i_j}\right) = W(P_1, P_2, \dots, P_n \backslash P_{i_1}, \dots, P_{i_j}) \tag{3.6}$$

$$W\big(P_i | P_{j_1} = \eta_1, \dots, P_{j_v} = \eta_v\big) =$$

$$= W(P_1, P_2, \dots, P_{i-1}, P_{i+1}, \dots, P_n | P_{j_1} = \eta_1, \dots, P_{j_v} = \eta_v) \tag{3.7}$$

$$W\left(P_{i_1}, \dots, P_{i_j} | P_{j_1} = \eta_1, \dots, P_{j_v} = \eta_v\right) =$$

$$= W(P_1, \dots, P_n \backslash P_{i_1}, \dots, P_{i_j} | P_{j_1} = \eta_1, \dots, P_{j_v} = \eta_v) \tag{3.8}$$

Зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) Питання мінімізації опису вихідного зображення тісно пов‘язано з функціональною оцінкою важливості дискретних ознак (які характеризують дане зображення), що дає можливість максимально компактно та якісно описати об‘єкт класифікації (зображення).

2) Відмітимо, що якісні функціонали (3.1) – (3.4) призначено для обчислення деякої роздільної можливості окремих ознак відносно$f_R(s_i)$ за початковою інформацією $R\big(I(l)\big)$.

3) Звернемо увагу, що запропонований набір функціоналів оцінки якості дискретних ознак дає змогу побудувати мінімальне за ознаковим описом зображення довільної структури.

Висновки до розділу 3

1. У розділі зафіксовано, що задача пошуку набору всіх тупикових тестів не є тривіальною та не дає можливість простого розв‘язання шляхом прямого перебору навіть для відносно невеликих масивів НВ, а пряме обчислення оцінки $\Gamma_j(S)$, як правило, ускладнене або взагалі неможливе, хоча і запропоновані підходи, що значно спрощують цей процес за рахунок особливостей вибору набору опорних множин, причому кожний фіксований алгоритм класифікації в моделі алгоритмів обчислення оцінок буде однозначно визначатися вибором системи опорних множин, функцією близькості, визначенням ваги об’єктів за навчаючою інформацією та правилом класифікації.

2. Запропонована концепція Т – опорних множини, яка полягає у відборі (фіксації) певного набору ознак разом зі своїми значеннями на основі інформації деякої початкової НВ з можливістю наступної оцінки даних опорних множин за допомогою відповідних функціоналів. Виділено основні способи задання систем Т – опорних множин, причому найпростіший і природний випливає безпосередньо з самого визначення даного поняття і полягає в фіксації деякого набору ознак та їх значень на основі програмного генератора PRG, а процес знаходження простих систем Т – опорних множин за допомогою масиву деякої початкової інформації та дерев розпізнавання (схем ЛДК) мало залежить від об‘єму, структури та виду початкової інформації $I(l)$.

3. На основі концепції Т – опорних множин запропоновано нове формальне визначення алгоритму розпізнавання (класифікації), що дало змогу ввести нові означення інформативності Т – опорних множин щодо довільного об‘єкту S та класу H_j, дослідити основні підмоделі алгоритму розпізнавання (на основі Т – опорних множин). Зафіксовано, що найбільший інтерес викликають саме прості системи Т – опорних множин, алгоритми

розпізнавання, що їх визначають, не дають помилок розпізнавання на даних початкової навчальної інформації $I(l)$ та відмов класифікації на всій множині допустимих об‘єктів.

4. Показано, що деяке правило розпізнавання (класифікації) приводить довільний допустимий об‘єкт S (фактично послідовно задіює ознаки $P_{i_1}, \dots, P_{i_\xi}$ у вигляді вершин в конструкції ЛДК) за його інформаційним описом $I(S)$ у кінцеву вершину дерева класифікації – тобто задає фіксований шлях у структурі логічного дерева, а множина $D = \{D_1, \dots, D_\xi\}$ (шляхів структури ЛДК) утворює базис задачі розпізнавання, якщо виконуються умови визначеності та повноти.

5. Уведено дерева моделей для задачі класифікації, які представляють собою зв‘язаний граф без циклів, у некінцевих вершинах якого знаходяться фіксовані моделі з $\{I(_j^{\eta_k}), P_{\eta_k}\}$, а ребра нумеруються значеннями предикатів (узагальнених ознак) з цих моделей, причому в кінцевих вершинах знаходяться значення функції, що задає розбиття $I(l)$ на класи або символ Δ невизначеності, а на фіксованому шляху ДМ одна й та сама модель може зустрічатися тільки один раз. Показано, що деяка множина моделей $\{I(_j^{\eta_k}), P_{\eta_k}\}$, що представляють алгебраїчні системи розпізнавання (у вигляді структур дерев класифікації) при фіксованому η_k та P_{η_k} перетворюється на конкретну модель класифікації $\{I(_j^{\eta_k}), P_{\eta_k}\}$ (деревоподібну логічну структуру), причому правилом розпізнавання (класифікації) ДМ являється деякий оператор, що відносить довільний допустимий об‘єкт S, (за його початковим інформаційним описом) у фіксовану кінцеву вершину дерева моделей.

6. Запропоновано використання концепції Т – опорних множин для опису дискретних об‘єктів, на основі чого представлено нові означення інформативності по відношенню до класу та об‘єкту класифікації, причому важливим моментом є те, що 2Т – опорні множини можна використовувати в якості ознак, що характеризують деякий об‘єкт, а, відповідно, для побудови алгоритмів розпізнавання або класифікації дискретних об‘єктів доречно

використовувати дерева моделей класифікації (алгебраїчних систем), у вершинах яких знаходяться 2Т – опорні множини.

7. Запропоновано якісні функціонали для обчислення деякої роздільної можливості (інформативності) окремих дискретних ознак відносно ФР за початковою інформацією $I(l)$, причому даний набір функціоналів оцінки якості дискретних ознак дозволяє побудувати мінімальне за ознаковим описом представлення дискретного об‘єкту довільної структури. Підкреслено, що питання мінімізації опису дискретних об‘єктів тісно пов‘язано з функціональної оцінкою важливості дискретних ознак (які характеризують даний об‘єкт), що забезпечує можливість максимально компактно та якісно описати об‘єкт класифікації.

РОЗДІЛ 4
ЗАГАЛЬНІ МЕТОДИ ПРЕДСТАВЛЕННЯ ДИСКРЕТНИХ СТРУКТУР У ВИГЛЯДІ ЛОГІЧНИХ ДЕРЕВ

4.1. Взаємозв‘язок логічних дерев та логічних функцій

Для всебічного дослідження концепції логічних дерев на першому етапі необхідно розглянути принципове питання представлення функції k – значної логіки в граф – схемній (графічній) формі. Як відомо, функції k – значної логіки, по аналогії із двозначною, можна представити в табличній, аналітичній та графічній формах. У даному дослідженні головну увагу буде приділено графічній формі представлення функцій k – значної логіки. Основна ідея якої полягає в тому, що довільну k – значну логічну функцію можна представити у вигляді деякого зв’язного графа, який не містить циклів (у теорії графів такий вигляд графа називається деревом). Отже, назвемо вказане представлення логічної функції – логічним деревом (або просто деревом).

4.1.1. Графічна форма представлення функцій багатозначної логіки

Наприклад на рис. 4.1 приведено логічне дерево або граф функції трьох-значної логіки, яка залежить від трьох змінних, аргументів (зокрема коло - x_1, трикутник - x_2, квадрат - x_3).

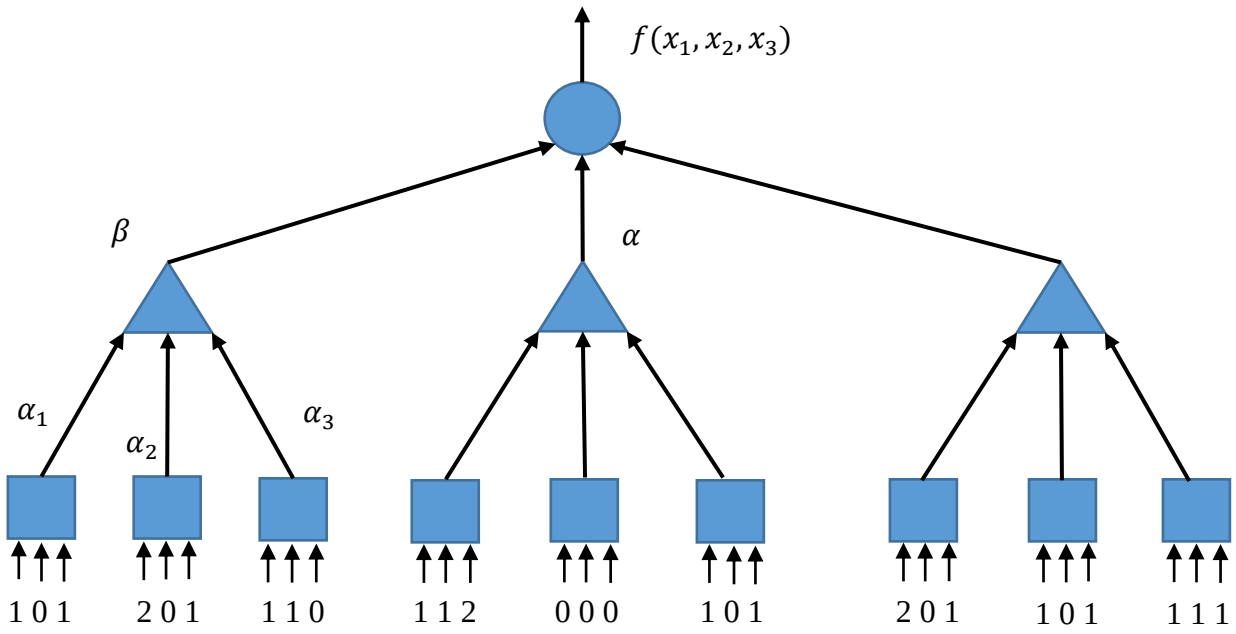


Рис. 4.1. Логічне дерево функції трьох-значної логіки $f(x_1, x_2, x_3)$.

Звернемо увагу – вершини графа відповідають змінним, від яких залежить результуюча функція. В кожну вершину входить три ребра, які відповідають різним значенням змінних (0,1,2). Усього ребер нижнього ярусу графа – 3^n, тобто ця кількість дорівнює кількості різних наборів аргументів функції від $n-$ змінних. Якщо рухатися від верхньої вершини графа вниз по ребрам до найнижчого ярусу ребер графа, то кожний такий шлях буде відповідати визначеному набору змінних. Зокрема, для прикладу, нехай $a\ b\ c$ на рис. 4.1 відповідає набору $\alpha = (1,0,2)$. Кожному кінцевому ребру такого шляху ставиться у відповідність значення функцій на наборі, який відповідає фіксованому шляху, до якого належить дане ребро, наприклад, на наборі $\alpha = (1,0,2),\ f(x_1, x_2, x_3) = 2.$

Наступним кроком кожному кінцевому ребру ставимо у відповідність деяку функцію, яку визначимо наступним чином: нехай деяка вершина графа відповідає змінній $x_i = 0$, у неї входять три ребра, які відповідають різним значенням змінної; нехай далі ребру, яке відповідає значенню змінної $x_i = 0$, поставлено у відповідність функцію α_1, ребру, яке відповідає $x_i = 1$, поставимо α_2, а ребру, відповідному $x_i = 2$, поставимо α_3, тоді на виході з вершини x_i отримаємо функцію:

$$\beta = \alpha_1 * \varphi_0(x_i) \vee \alpha_2 * \varphi_1(x_i) \vee \alpha_3 * \varphi_2(x_i). \qquad (4.1)$$

Якщо вершина x_i є самою верхньою, то на виході її отримаємо саму функцію $f(x_1, x_2, x_3)$, тобто $\beta = f(x_1, x_2, x_3)$. Отже, за допомогою логічного дерева (рис. 4.2) отримаємо деякі дужкові представлення логічної функції $f(x_1, x_2, x_3)$.

Аналогічним чином будується логічне дерево (граф-схема) довільної $k\ -$значної функції від будь-якої кількості змінних. Дужкова форма довільної функції будується за тими самими правилами, що і у випадку трьох-значної логіки. На рис. 4.2 приведено граф функції $k-$значної логіки від n змінних. Де $f_{i_r} -$ значення функцій на $i_r -$ наборах, а $x_{j_p} \in \{x_1, x_2, \dots, x_n\}$. Мітки

$\alpha, \dots, \gamma, w$ – це вихідні функції, які відповідають вершинам графа, які знаходяться за описаним вище правилом. Можливість вказаного представлення функцій за допомогою графа обумовлена теоремою про розклад довільної $k-$значної функції за фіксованою змінною, аналогом якої в булевій алгебрі є теорема про розклад Шеннона. Доведемо цю теорему.

Теорема 4.1. Довільну логічну функцію $f(x_1, x_2, \dots, x_n)$ $k-$ значної логіки від n змінних можна представити у вигляді:

$$f(x_1, x_2, \dots, x_n) = \varphi_0(x_i) f(x_1, x_2, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n) \vee$$
$$\vee \varphi_1(x_i) f(x_1, x_2, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n) \vee \dots \vee$$
$$\vee \varphi_{k-1}(x_i) f(x_1, x_2, \dots, x_{i-1}, k-1, x_{i+1}, \dots, x_n). \qquad (4.2)$$

Зауважимо, що $f \in \{0,1, \dots, k-1\}, x_i \in \{0,1, \dots, k-1\}, (i = 1,2, \dots, n)$.

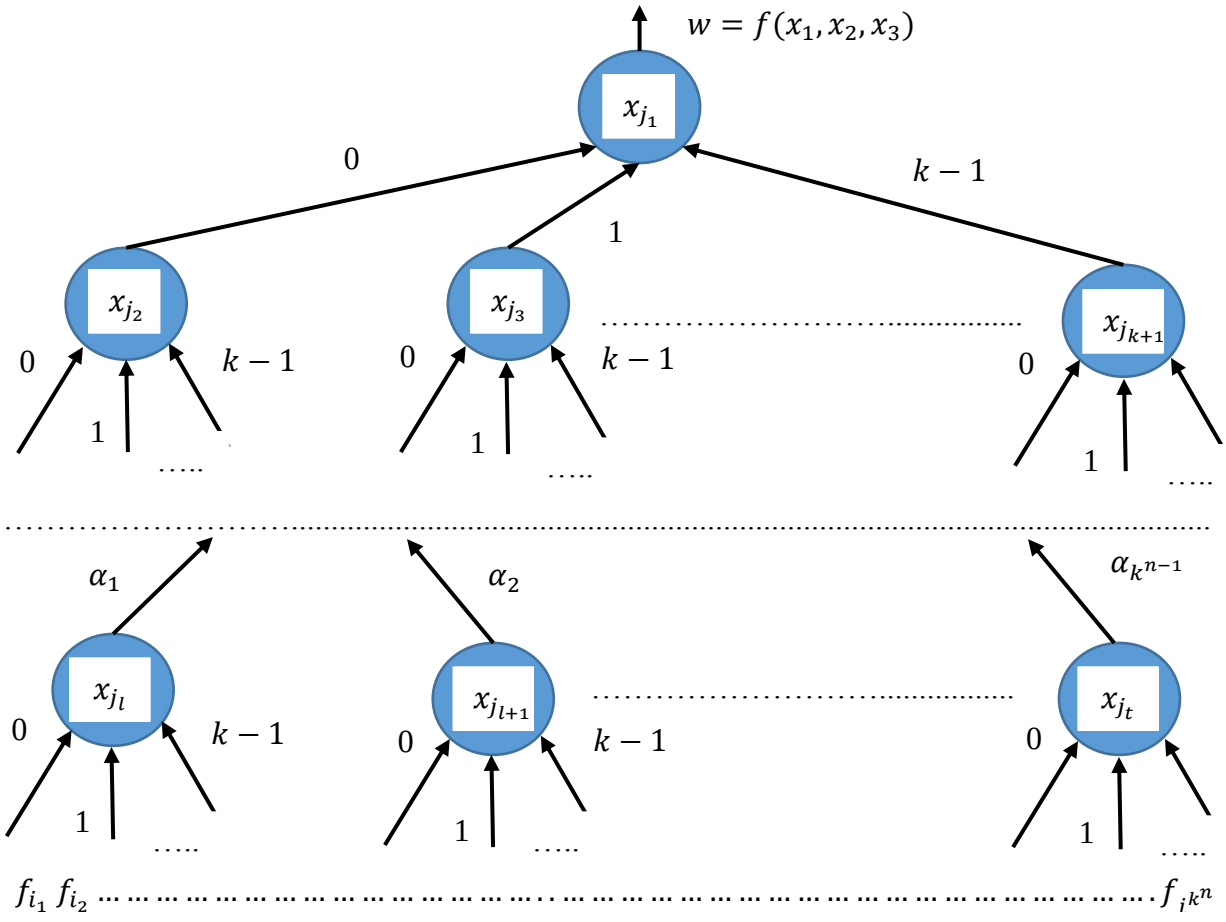


Рис. 4.2. Логічне дерево (граф) функції $k-$значної логіки від n змінних.

Доведення. Для мінімізації запису приймемо наступні позначення:

$f(x_1, x_2, \dots, x_{i-1}, l, x_{i+1}, \dots, x_n) = f_{\varphi_l}(x_i)$, тут $l \in \{0,1,2, \dots, k-1\}$.

Тоді умову теореми можна скоротити у такий спосіб:

$$f(x_1, x_2, \dots, x_n) = \bigvee_{l=0}^{k-1} \varphi_l(x_i) * f_{\varphi_l}(x_i) \qquad (4.3)$$

Для доведення теореми покажемо, що значення лівої та правої частини виразу (4.3) співпадають на всіх наборах. Нехай $x_i = p$. Тоді в лівій частині виразу (4.3) отримаємо наступне:

$$f(x_1, x_2, \dots, x_{i-1}, p, x_{i+1}, \dots, x_n) = f_{\varphi_p}(x_i).$$

У правій частині виразу (4.3) маємо отримати:

$$\varphi_p(p) * f_{\varphi_p}(x_i).$$

Усі інші диз‘юнктивні члени правої частини дорівнюють нулю, оскільки вони містять множник $\varphi_l(p) = 0, (l \neq p); p, l \in \{0,1, \dots, k-1\}$. Але за визначенням характеристичної функції $\varphi_p(p) = k - 1$, тобто отримаємо:

$$\varphi_p(p) * f_{\varphi_p}(x_i) = (k-1) * f_{\varphi_p}(x_i) = f_{\varphi_p}(x_i).$$

Отже, показано, що якщо $x_i = p$, то права та ліва частини виразу (4.3) співпадають незалежно від того, які значення приймають інші змінні. Але через те, що значення $x_i = p \in \{0,1, \dots, k-1\}$ вибираються довільно, показано що ліва та права частини виразу (4.3) співпадають на всіх наборах, що і доводить теорему.

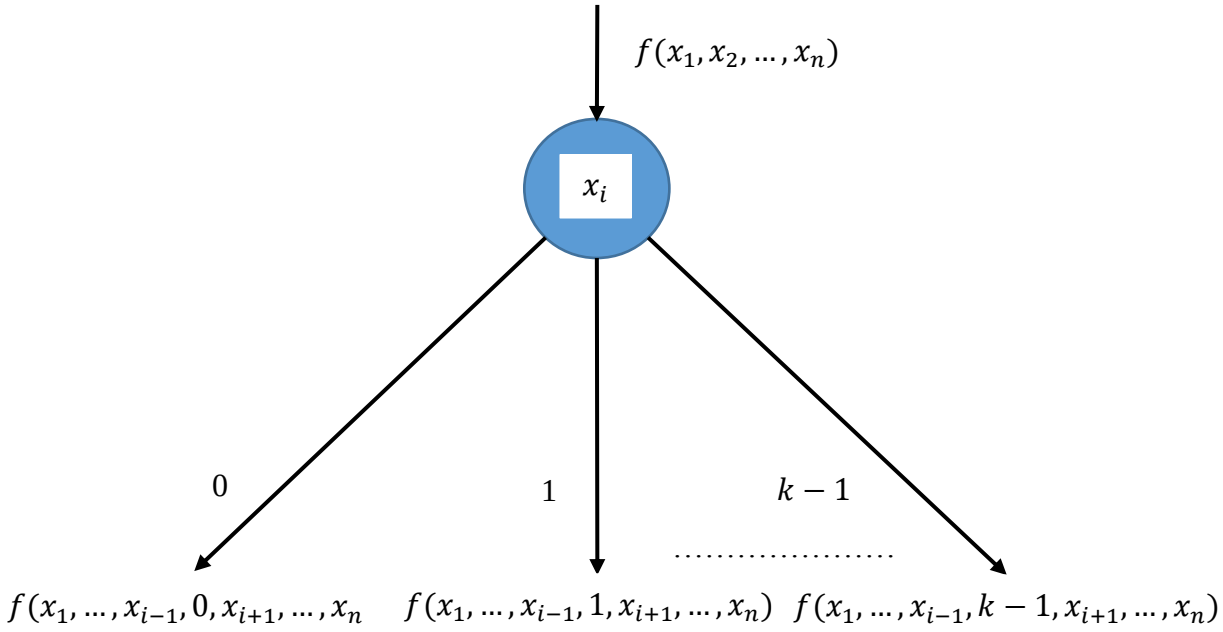


Рис. 4.3. Графічна інтерпретація представлення формули (4.2).

Якщо $f_{\varphi_0}(x_i) = f_{\varphi_1}(x_i) = \cdots = f_{\varphi_{k-1}}(x_i)$, то змінна x_i називається фіктивною. В цьому випадку всі ребра, які входять у вершину x_i, відмічені

однаковими або тотожно рівними функціями. Будемо вважати, що всі ребра відмічені в цьому випадку ідентично. Графічна інтерпретація представлення (4.2) має вигляд, показаний на рис. 4.3.

Відмітимо, що якщо застосувати теорему 4.1 послідовно до функцій $f(x_1, x_2, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n)$, $f(x_1, x_2, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n)$ тощо, починаючи розклад $f(x_1, x_2, \dots, x_n)$ за змінною x_1 та закінчуючи x_n, отримаємо:

$$f(x_1, x_2, \dots, x_n) = \bigvee_{i=0}^{k-1} \varphi_i(x_1) * \bigvee_{j=0}^{k-1} \varphi_j(x_2) * \bigvee_{p=0}^{k-1} \varphi_p(x_3) * \dots$$

$$\dots * \bigvee_{l=0}^{k-1} \varphi_l(x_{n-1}) * f(i, j, \dots, l, x_n).$$

Граф (логічне дерево), який інтерпретує дане представлення, показаний на рис. 4.2.

Для ілюстрації вищезазначеного наведемо приклад. Нехай деяка логічна функція $f(x_1, x_2)$ задається наступною таблицею:

Таблиця 4.1. Таблична форма $f(x_1, x_2)$ при $k = 4$.

f	0	0	1	2	3	0	1	0	1	3	0	1	1	2	0	3
x_1	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
x_2	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3

Структура дерева буде мати наступний вигляд (зокрема, коло - x_1, квадрат - x_2):

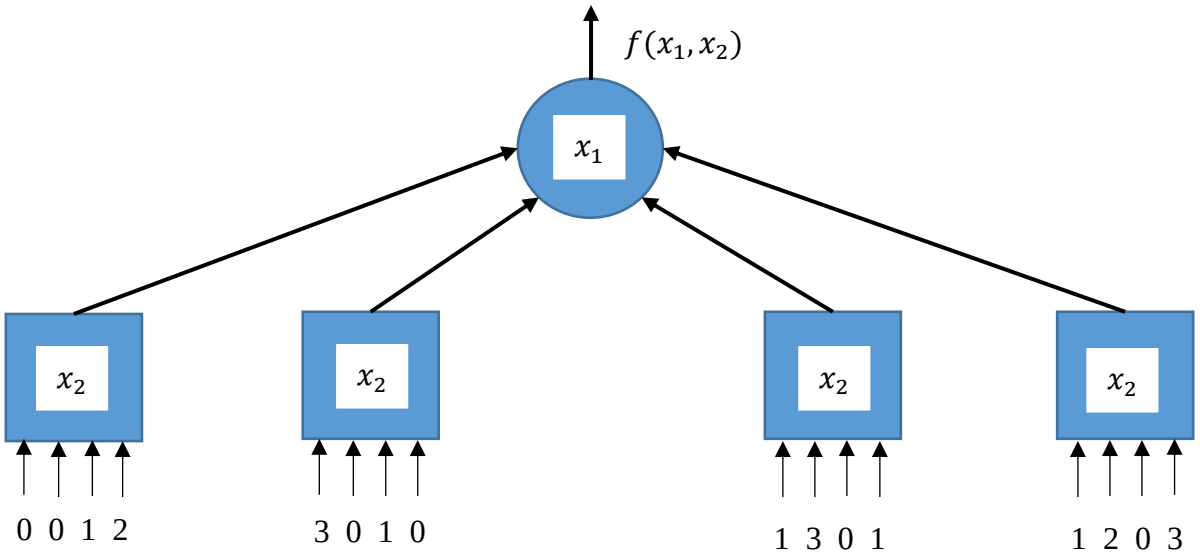


Рис. 4.4. Логічне дерево функції $f(x_1, x_2)$ при $k = 4$.

Функції – мітки на другом ярусі $-\alpha, \beta, \gamma, \xi$. Відповідно до (4.1) для кожної мітки будемо мати:

$$\alpha = 1 * \varphi_2(x_2) \vee 2 * \varphi_3(x_2);$$
$$\beta = 3 * \varphi_0(x_2) \vee 1 * \varphi_2(x_2);$$
$$\gamma = 1 * \varphi_0(x_2) \vee 3 * \varphi_1(x_2) \vee 1 * \varphi_3(x_2);$$
$$\xi = 1 * \varphi_0(x_2) \vee 2 * \varphi_1(x_2) \vee 3 * \varphi_3(x_2).$$

Зауваження. При визначенні міток у структурі логічного дерева деякі з них очевидно можуть дорівнювати нулю.

4.1.2. Задача мінімізації структури логічного дерева

Дерево є деякою графічною багатоярусною (під ярусом будемо розуміти горизонтальний ряд вершин графу зі змінною одного індексу) структурою. Причому, на першому ярусі маємо одну вершину, на другому - k вершин, на $n-$ му ярусі k^{n-1}. У кожній вершині довільного ярусу маємо змінну одного індексу. Загальна кількість ярусів дорівнює кількості змінних даної функції. Розташування змінних по ярусам можуть бути різними, тобто не обов‘язково, щоб на першому ярусі була змінна x_1, на другому – x_2 і так далі. Перестановкою змінних по ярусах отримаємо нове дерево. Кількість всіх можливих для функції від n аргументів дорівнює кількості перестановок з символів (атрибутів), тобто $n!$

На останньому ярусі в загальному випадку може бути k^k різних функцій (як відомо, k^k – кількість всіх різних функцій $k-$значної логіки одного аргументу). На $i-$ товому ярусі можуть зустрічатися довільні суперпозиції з k^k функцій.

Дерево, в якому на останньому ярусі k^n вершин, назвемо повним. Уведемо декілька визначень, які дозволять будувати неповні дерева.

Визначення 4.1. Логічне дерево (граф), що представляє $k-$ значну функцію $f(x_1, \dots, x_n)$ – це зв’язаний граф без циклів, у некінцевих вершинах якого знаходяться змінні $x_1, \dots, x_n$, а ребра нумеруються значеннями цих

змінних. У кінцевих вершинах дерева знаходяться значення функції $f(x_1, \dots, x_n)$. Причому на фіксованому шляху дерева одна і та сама змінна зустрічається тільки один раз.

Нехай a – вершина логічного дерева. Через $q(a)$ позначатимемо символ, який стоїть у вершині a. Вершина a називається наступною за вершиною b, якщо деяка стрілка, що виходить з вершини b входить у вершину a. Позначимо цей факт через $a = b$. Якщо в вершину a входить j– това стрілка вершини b, то це позначатимемо $a = (b)_j$.

Послідовність вершин $a_1, \dots, a_r$ називається шляхом в дереві, якщо $a_{j+1} = (a_j)$, $(j = 1,2, \dots, r-1)$, $1 \le r \le n$. Кожна стрілка, що входить в деяку вершину, нумерується значенням змінної, що знаходиться в вершині, із якої виходить ця стрілка. Крім цієї стандартної нумерації стрілок, що слідує із визначення дерева, введемо в розгляд нумерацію стрілок мітками.

Визначення 4.2. Мітка стрілки, що входить у вершину дерева, характеризує функцію піддерева, визначеного цією міткою. Якщо всі вихідні стрілки деякої вершини відмічено однаковою міткою α, то і вихідну мітку даної вершини відмічено тією ж самою міткою α. Це відповідає використанню співвідношення :

$$A * \varphi_0(x) \vee A * \varphi_1(x) \vee \dots \vee A * \varphi_{k-1}(x) = A \qquad (4.4)$$

Зауважимо, що тут A – довільна функція.

Умова (4.4) фактично дозволяє забезпечити мінімізацію функції за допомогою логічного дерева за такою схемою:

1) Якщо мітка деякої стрілки, що входить деяку вершину дерева являється константою, то замість усього піддерева, визначеного цією вершиною, ставимо одну вершину із цією константою, атрибутом (процедура скорочення блоку конструкції логічного дерева).

2) Якщо мітки, що виходять із деякої вершини, характеризують одну і ту саму функцію, то замість усього піддерева, що визначається цією вершиною, ставимо піддерево, що характеризує дану функцію.

Приклад 1. Нехай структура логічного дерева функції $f(x_1, x_2, x_3)$ має вигляд, зображений на рис. 4.5 (зауважимо, що тут коло – x_1, квадрат – x_2, трикутник – x_3).

Стандартна нумерація стрілок послідовна – зліва направо – і далі не відображена. Зокрема, відображено мітки, характеристичні функції відповідних піддерев (базового логічного дерева).

Мінімізуємо дане логічне дерево вказаним вище способом. Після проведення всіх етапів мінімізоване дерево буде мати вигляд, показаний на рис. 4.6 (зауважимо, що тут так само, коло - x_1, квадрат - x_2, трикутник - x_3).

Функції, що реалізовуються мітками $\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5,\alpha_6$ виглядають таким чином:

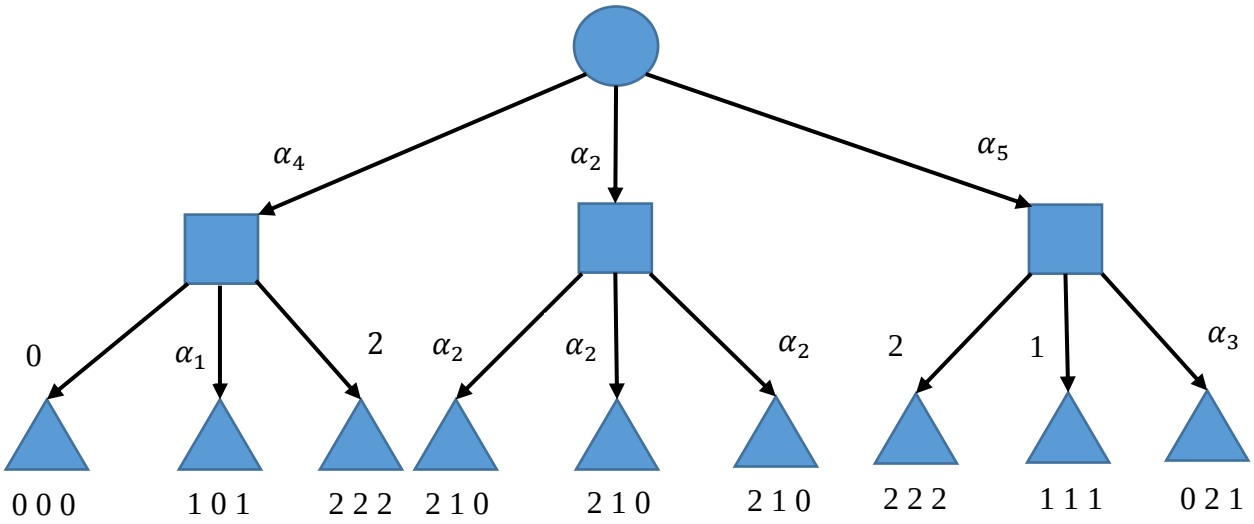


Рис. 4.5. Логічне дерево $f(x_1, x_2, x_3)$ прикладу 1.

$$\alpha_1 = 1 * \varphi_0(x_3);\ \alpha_2 = 2 * \varphi_0(x_3) \vee 1 * \varphi_1(x_3);\ \alpha_3 = 2 * \varphi_1(x_3) \vee 1 * \varphi_1(x_3);$$

$$\alpha_4 = \alpha_1 * \varphi_1(x_2) \vee 2 * \varphi_2(x_2) = 1 * \varphi_0(x_3) * \varphi_1(x_2) \vee 2 * \varphi_2(x_2);$$

$$\alpha_5 = 2 * \varphi_0(x_2) \vee 1 * \varphi_1(x_2) \vee \alpha_3 * \varphi_2(x_2) = 2 * \varphi_0(x_2) \vee 1 * \varphi_1(x_2) \vee$$

$$\vee \left(2 * \varphi_1(x_3) \vee 1 * \varphi_2(x_3)\right) * \varphi_2(x_2);$$

$$\alpha_6 = \alpha_4 * \varphi_0(x_1) \vee \alpha_2 * \varphi_1(x_1) \vee \alpha_5 * \varphi_2(x_1) = (1 * \varphi_0(x_3) \vee \varphi_1(x_2) \vee$$

$$\vee 2 * \varphi_2(x_2)) \vee (2 * \varphi_0(x_3) \vee 1 * \varphi_1(x_3)) * \varphi_1(x_1) \vee$$

$$\vee \left(2 * \varphi_1(x_3) \vee 1 * \varphi_2(x_3)\right) * \varphi(x) = f(x_1, x_2, x_3).$$

У прикладі 1 представлено мінімізацію повного логічного дерева, котре реалізує деяку функцію. Також показано, як на основі мінімізованого логічного дерева, отримати аналітичний вигляд даної функції.

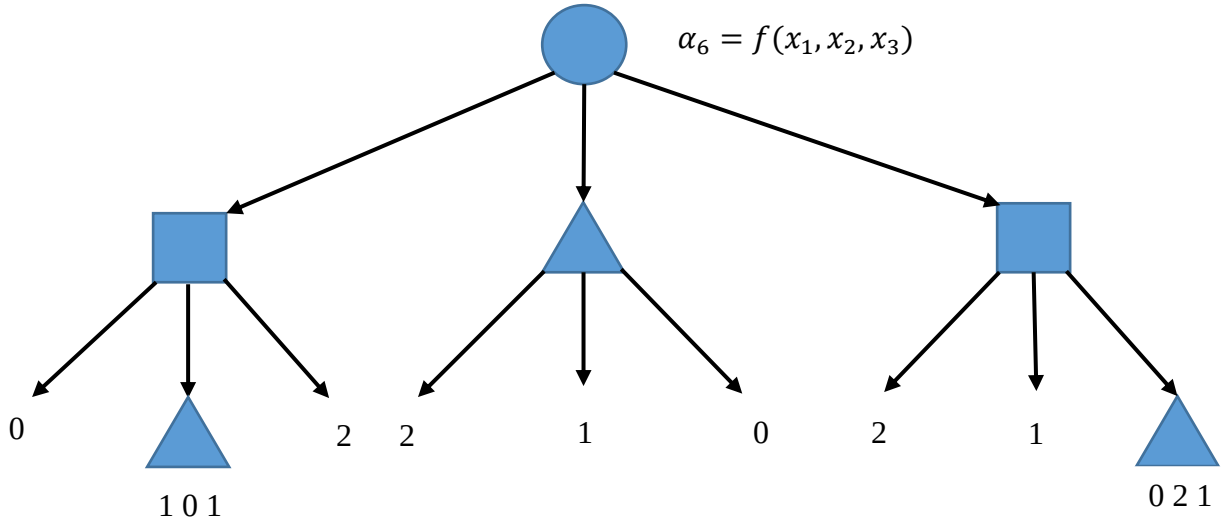


Рис. 4.6. Мінімізоване логічне дерево $f(x_1, x_2, x_3)$ прикладу 1.

Приклад 2. Розглянемо логічне дерево з рис. 4.7 (зауважимо, що тут коло - x_1, квадрат - x_2, трикутник - x_3). Необхідно аналітично визначити функцію, яка представляє дане логічне дерево.

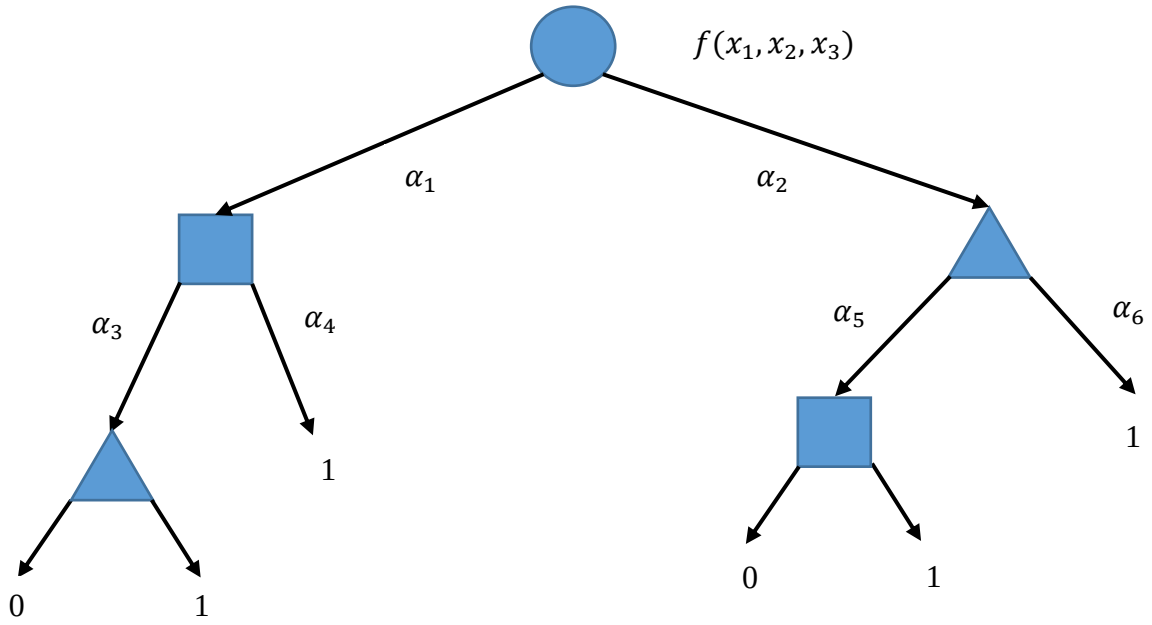


Рис. 4.7. Логічне дерево $f(x_1, x_2, x_3)$ прикладу 2.

$\alpha_1 = x_1 * \overline{x_2} \vee x_3$ при $(x_1 = 0)$;

$\alpha_2 = x_2 * \overline{x_3} \vee \overline{x_2} * x_3 \vee x_2 * \overline{x_3} \vee x_3$ при $(x_1 = 1)$;

$\alpha_3 = x_3$ при $(x_2 = 0)$; $\alpha_4 = 1$; $\alpha_5 = x_2$ при $(x_3 = 1)$; $\alpha_6 = 1$;

Тобто дереву на рис. 4.7 відповідає наступне дужкове представлення функції $f(x_1, x_2, x_3)$:

$$f(x_1, x_2, x_3) = \overline{x_1} * \overline{\alpha_1} \vee x_1 * \alpha_1 = \overline{x_1} * (\overline{x_2} * x_3 \vee x_2) \vee x_1 * (\overline{x_3} * x_2 \vee x_3).$$

Побудуємо ще одне логічне дерево, яке характеризує початкову функцію (рис. 4.8) (зауважимо, що тут коло - x_1, квадрат - x_2).

За цим логічним деревом можна записати наступну функцію, яка еквівалентна початковій функції – $f(x_1, x_2, x_3) = x_2 * \overline{x_3} \vee x_3$.

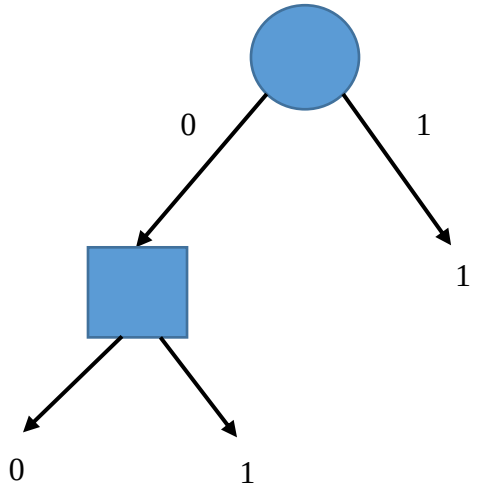


Рис. 4.8. Дерево, яке характеризує початкову функцію прикладу 2.

На останньому етапі введемо деякі визначення, які будуть необхідні в подальшому для даного дослідження.

Визначення 4.3. Якщо всі ребра, які входять у вершину логічного дерева (графа) x_i, відмічені ідентично, то вершина x_i називається подібною.

Визначення 4.4. Логічне дерево (піддерево), верхня вершина якого є подібною, називається особливим деревом (піддеревом).

Усі інші логічні дерева та піддерева (графи) в даному дослідженні будемо називати неособливими.

Визначення 4.5. Логічне дерево (граф функція), у вершинах кожного горизонтального ярусу якого записані однакові змінні (атрибути), називається регулярним. У протилежному випадку логічне дерево (граф функція) називається нерегулярним.

Отже, резюмуючи все вищезазначене щодо зв‘язку логічних дерев та багатозначних логічних функцій, приходимо до наступних пунктів:

1) Довільну функцію $k-$ значної логіки від n змінних, задану в аналітичному вигляді можна представити у формі логічного дерева.

2) Довільному логічному дереву можна поставити у відповідність цілком визначену функцію $f(x_1, \dots, x_n)$ $k-$ значної логіки від n змінних в аналітичному вигляді.

3) За допомогою логічного дерева можна мінімізувати $k-$ значну функцію $f(x_1, \dots, x_n)$ від n змінних в аналітичному вигляді.

4) Якщо $k-$ значна функція $f(x_1, \dots, x_n)$ від n змінних задана таблично та визначена на всіх наборах, тоді:

- можна побудувати відповідне повне логічне дерево, мінімізувати його і записати аналітичний вигляд даної функції;
- за табличним представленням логічної функції знайти її аналітичне представлення у вигляді ДНФ або КНФ та побудувати логічне дерево;
- якщо всі ребра, які входять у вершину фіксованого логічного дерева (графа), відмічені ідентично (ідентичні мітки), то цю вершину будемо далі називати подібною;
- ЛДК, у вершинах кожного горизонтального ярусу якого розташовані однакові змінні, будемо називати регулярним; регулярні логічні дерева буде розглянуто в даному дослідженні далі в розрізі питання можливої мінімізації їх структури;
- ЛДК, початкова вершина (вершина нульового ярусу) якого є подібною, в подальшому в даному дослідженні будемо називати особливим деревом.

Отже, якщо деяка $k-$ значна функція $f(x_1, \dots, x_n)$ від n змінних визначена не на всіх наборах, то виникає задача довизначення даної функції. Ця задача в даній роботі буде розв’язуватися із точки зору розпізнавання образів. Результати цього розділу будуть використовуватися в подальшому для розв’язування інших задач ЛДК.

4.2. Представлення неповністю визначених багатозначних логічних функцій заданих таблично у вигляді логічних дерев

На даному етапі дослідження розглянемо $k-$ значні функції від n змінних, визначені на бінарних наборах. Як відомо, функції $k-$ значної логіки, по аналогії із двозначною логікою, можна представляти в табличній, аналітичній та графічній формах. Далі розглядаються всі три форми представлення функцій. Це пояснюється тим, що в теорії розпізнавання образів, як правило, невідома функція задана таблично (мається на увазі представлення НВ), причому тільки на деякій підмножині її можливих значень, одержаних найчастіше експериментальним шляхом. Потрібно відновити її значення на всіх наборах. Очевидно, що, не накладаючи фіксованих обмежень на вигляд та властивості цієї функції, дана задача не має єдиного розв'язку. Основна вимога, що накладається на функцію (функцію розпізнавання), яку ми шукаємо – це правильне її значення на об'єктах (наборах) тестової вибірки, тобто весь експериментальний матеріал $I(l)$ (масив об'єктів) розбивається на НВ, з якої будується функція (ФР) та ТВ, за якою вона перевіряється. Зауважимо, що хоча при даних умовах задача також не має єдиного розв'язку (якщо НВ та ТВ не об'єднують всі набори функцій), але пошук таких функцій при визначених умовах часто викликає труднощі. Якщо ввести ще додатково ряд обмежень на шукану ФР, (наприклад складність, інформаційну ємність тощо), то розв'язок даної задачі (пов'язаний із побудовою ФР) буде сильно ускладненим.

Отже, нехай задана деяка $k-$ значна функція на фіксованих наборах:

$$\Big(\big(x_1, f_R(x_1)\big), \big(x_2, f_R(x_2)\big), \dots, \big(x_m, f_R(x_m)\big)\Big). \tag{4.5}$$

Де $x_i = \big(x_1^i, x_2^i, \dots, x_N^i\big), x_j^i \in \{0,1\}, j = 1,2, \dots, N;$

$f_R(x_i) = \big(x_1^i, x_2^i, \dots, x_N^i\big)$, $N-$ кількість змінних (ознак об'єкта), $m-$ загальна кількість наборів $f_R(x_i) = \in \{0,1, \dots, k-1\}$.

Зауважимо, що часто набори вигляду (4.5) і називають навчальною вибіркою, а f_R буде функцією розпізнавання.

Для отримання тестової вибірки розіб‘ємо вибірку вигляду (4.5) на дві частини (НВ та ТВ).

$$\begin{cases} \Big(\big(x_1, f_R(x_1)\big), \big(x_2, f_R(x_2)\big), \dots, \big(x_m, f_R(x_m)\big)\Big) - \text{НВ}; \\ \Big(\big(x_{m+1}, f_R(x_{m+1})\big), \big(x_{m+2}, f_R(x_{m+2})\big), \dots, \big(x_{m+\delta}, f_R(x_{m+\delta})\big)\Big) - \text{ТВ}. \end{cases}$$

Розглянемо питання щодо побудови k – значної функції, заданої таблично на деякій підмножині її двійкових наборів (об’єктів). У літературі відомі різні способи побудови (знаходження) таких функцій [298-301]. У дисертаційній роботі досліджено знаходження таких функцій (ФР) за допомогою графічних методів, зокрема орієнтованих дерев або логічних дерев.

Нехай деяка функція $f(x_1, x_2, x_3)$ задана таблично (табл. 4.2).

Таблиця 4.2. Таблична форма $f(x_1, x_2, x_3)$.

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
f	0	1	2	0	3	1	0	1

Відповідно до табличного представлення, на рис. 4.9 зображено відповідне логічне дерево (тут коло - x_1, квадрат - x_2, трикутник - x_3).

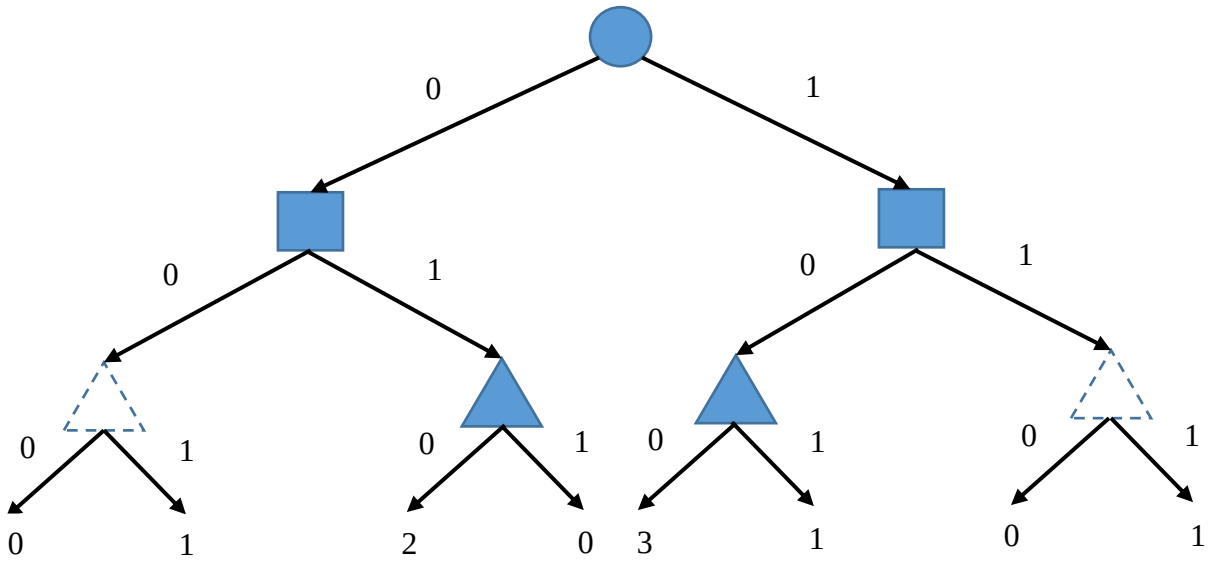


Рис. 4.9. Логічне дерево функції з – (Табл. 4.2).

Відмітимо, що вершини даного дерева, крім кінцевих, відповідають змінним, від яких залежить початкова функція. В кожну вершину, крім

початкової, входить одне і виходять два ребра, що відповідають значенням відповідної змінної. В кінцевих вершинах дерева знаходяться значення функцій f_R на наборі x_i (листи ЛДК). Якщо рухатися від верхньої вершини схеми дерева вниз по ребрах до найнижчого ярусу дерева класифікації, то кожний такий шлях буде відповідати визначеному набору змінних. Так, наприклад, {100} на рис. 4.9 відповідає набору ($x_1 = 1,\ x_2 = 0,\ x_3 = 0$), а кінцева вершина дерева визначає значення функції на цьому наборі, тобто $f(1,0,0) = 3$.

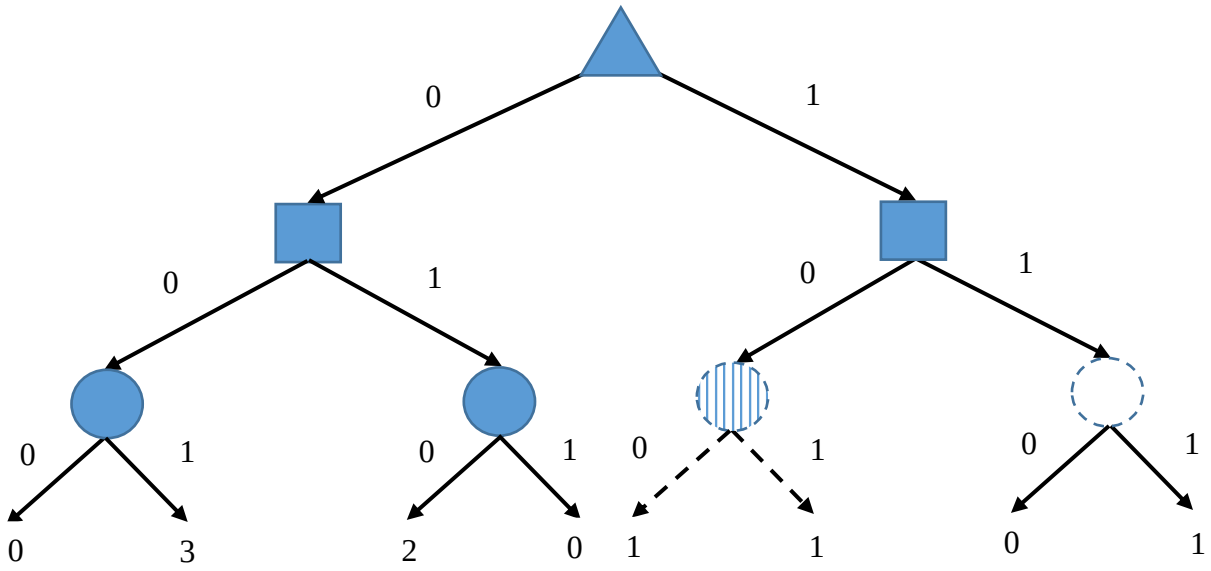


Рис. 4.10. Логічне дерево функції зі змінними в оберненому порядку.

На рис. 4.9 задано повне логічне дерево, тобто дерево, в якому функція визначена на бінарних наборах. У цьому разі логічне дерево будується дуже просто: спочатку будується повне дерево, а потім проставляються значення функції на відповідному наборі. Проте, ми матимемо справу з функціями, заданими на деякій підмножині будь-яких значень. Даний алгоритм придатний і для побудови логічних дерев для цього випадку, якщо в кінцевому підсумку ще мінімізувати його. Зауважимо, що в логічному дереві (рис. 4.9) на третьому ярусі є два блоки (зліва направо – перший і четвертий), які підлягають оптимізації для спрощення загальної складності логічного дерева. В даному прикладі відповідні блоки достатньо просто замінити на змінну x_3, не змінюючи логіки самого дерева. Тоді мінімізація логічного дерева полягає в

тому, що потрібно викинути всі незадіяні вершини (або навіть цілі блоки) в структурі дерева. Пояснимо це на прикладі (рис. 4.9 – рис. 4.10).

Як видно з прикладу, дане дерево задає функцію, визначену на всіх наборах, тобто $f(0,0,1) = 1, f(1,0,0) = 3$. Проте, якщо порядок змінних у дереві змінити в інший напрямок (рис. 4.10), то отримаємо іншу функцію, тобто $f(0,0,1) = 3, f(1,0,0) = 1$. При такій структурі дерева, як і для дерева на рис. 4.9 доступна мінімальна оптимізація, яка полягає в заміні на останньому ярусі (рівні) третього блоку на символ 1 (константу), а четвертого блоку – на змінну x_1.

Із усього вищезазначеного можна зробити такі висновки:

1) Одержання тієї чи іншої логічної функції залежить від порядку проходження змінних у логічному дереві, що безпосередньо впливає на кінцеву структуру логічного дерева.

2) За підмножиною будь-яких значень набору змінних можна побудувати логічне дерево, яке представляє функцію, визначену на всіх наборах. Ця обставина дозволяє застосовувати логічні дерева в розпізнаванні образів. Важливою особливістю являється задіяння принципу прецедентності або часткової прецедентності.

3) Логічні дерева, що застосовуються для задач розпізнавання образів ще не гарантують ефективний та успішний розв‘язок. Тільки всебічне вивчення з математичної точки зору дозволяє з’ясувати можливості в області їх застосування.

4) Відмітимо, що важливим питанням для ефективного застосуванні ЛДК у задачах розпізнавання образів залишається питання мінімальної (оптимальної) структури відносно початкової НВ. Отже, такі ЛДК у практичних задачах також дозволяють проведення процедури хоча б найпростішої мінімізації – за рахунок відкидання однотипних блоків (блоків, що повторюються) структури логічного дерева.

4.3. Мінімізація регулярного логічного дерева методом перестановки ярусів у його структурі

На даному етапі дослідження розглянемо принципове питання ефективної мінімізації регулярного логічного дерева за допомогою методу перестановки ярусів у його структурі. У попередніх пунктах даного дослідження було показано, що логічне дерево також можна досить ефективно використовувати для мінімізації функції багатозначної логіки, яке воно представляє [300]. Далі розглянемо для цього приклад.

Приклад 3. Дано деяку логічну функцію, яка записана в такому вигляді ($k = 3$):

$$f(x_1, x_2) = 1 * \varphi_0(x_1) \vee 1 * \varphi_1(x_2) \vee 1 * \varphi_2(x_2) \vee \varphi_2(x_1) * \varphi_2(x_2).$$

Використовуючи базові співвідношення:

$$P * \varphi_0(x) \vee P * \varphi_1(x) \vee \ldots \vee P * \varphi_{k-1}(x) = P;$$

$$P * 1 * \varphi_1(x) \vee P * 2 * \varphi_2(x) \vee \ldots \vee P * (k-1) * \varphi_{k-1}(x) = P * x;$$

знаходимо відповідну ДДНФ:

$$1 * \varphi_0(x_1) * \varphi_0(x_2) \vee 1 * \varphi_0(x_1) * \varphi_1(x_2) \vee 1 * \varphi_0(x_1) * \varphi_2(x_2) \vee$$
$$\vee 1 * \varphi_1(x_1) * \varphi_1(x_2) \vee 1 * \varphi_2(x_1) * \varphi_1(x_2) \vee 1 * \varphi_1(x_1) * \varphi_2(x_2) \vee$$
$$\vee \varphi_2(x_1) * \varphi_2(x_2).$$

Якщо провести всі можливі склейки, то на виході отримаємо скорочену ДНФ вигляду – $1 * \varphi_0(x_1) \vee x_1 x_2 \vee 1 * x_2$. На наступному етапі складанням імплікативної таблиці або іншим шляхом знаходимо мінімальну та єдину (тупікову) ДНФ, яка має наступний вигляд – $1 * \varphi_0(x_1) \vee x_1 x_2$.

Описаний метод, аналогічний відомому в булевій алгебрі методу Квайна, зв‘язаний із досить громіздкими обчисленнями при великій кількості змінних та фактично при $n > 3, k > 3$ не може мати практичного застосування.

Відмітимо лише, що інші методи (інтерпретації карт Карно та методи Мак-Класки для багатозначного випадку) також мають вказані вище недоліки.

Розглянемо застосування методу логічного дерева для даного прикладу. Так, загальна структура логічного дерева для заданої функції має вид, зображений на рис. 4.11.

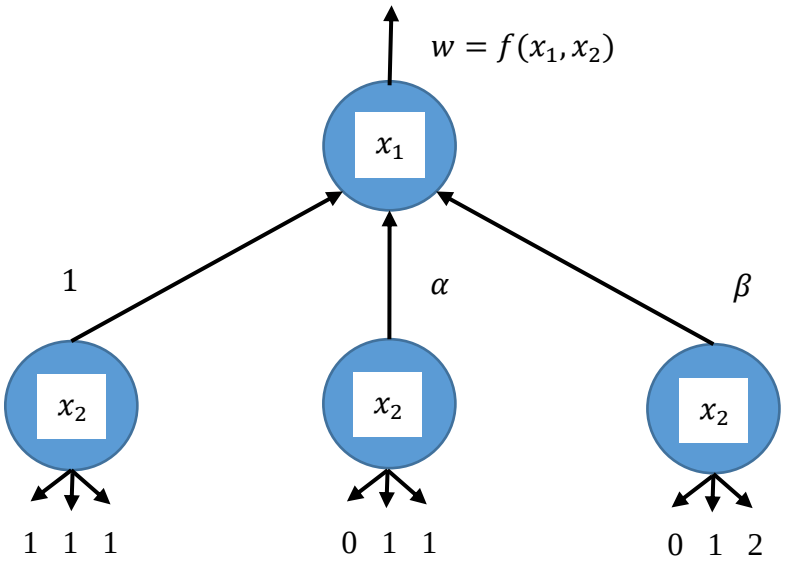


Рис. 4.11. Логічне дерево $f(x_1, x_2)$ з прикладу 3.

Після заміни міток α, β та w через змінні x_1, x_2 маємо:

$$\alpha = 1 * \varphi_1(x_2) \vee 1 * \varphi_2(x_2) = 1 * \big(\varphi_1(x_2) \vee \varphi_2(x_2)\big) = 1 * x_2;$$

$$\beta = x_2; w = f(x_1 x_2) = 1 * \varphi_0(x_1) \vee \alpha * \varphi_1(x_1) \vee \beta * \varphi_2(x_1) =$$

$$= 1 * \varphi_0(x_1) \vee \varphi_1(x_1) * 1 * x_2 \vee \varphi_2(x_1) * x_2 =$$

$$= 1 * \varphi_0(x_1) \vee x_2 * \big(1 * \varphi_1(x_1) \vee \varphi_2(x_1)\big) = 1 * \varphi_0(x_1) \vee x_1 x_2.$$

Тобто отримали мінімальну форму початкової функції $f(x_1 x_2)$ значно швидшим шляхом ніж при використанні попереднього методу. Однак слід зауважити, що в даному прикладі не використовувався метод пошуку оптимального регулярного дерева, тобто дерево з природним розташуванням змінних x_1, x_2 вже дало мінімальну форму.

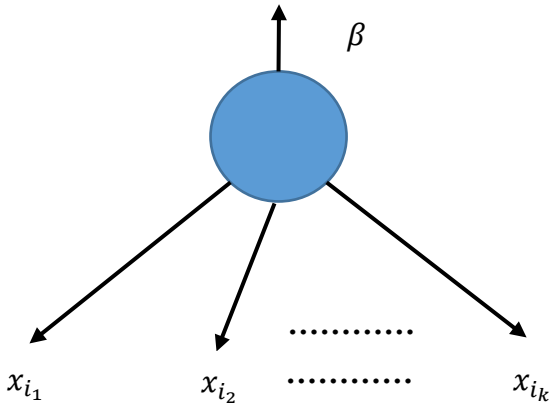


Рис. 4.12. Елементарне логічне дерево.

4.3.1. Метод мінімізації структури логічного дерева шляхом перестановки структурних блоків

При використанні концепції логічного дерева для цілей мінімізації важливим моментом другого етапу є розстановка функцій – міток (або просто міток) на структурі логічного дерева. Відмітимо при цьому, що будемо враховувати наступні зауваження, які стосуються елементарного дерева, наступного вигляду (рис. 4.12).

Зокрема, $x_{i_j} \in \{0,1,2,\dots,k-1\}, j = 1,2,\dots,k$, також $x_{i_1},\dots,x_{i_k}$ – деяка вхідна дія (вхідна функція), а $\beta -$ вихідна функція (мітка).

1) Якщо на вході $x_{i_1} = x_{i_2} = \dots = x_{i_k} = \alpha$, то вихідна функція – мітка має вигляд $\alpha * \varphi_0(x) \vee \alpha * \varphi_1(x) \vee \dots \vee \alpha * \varphi_{k-1}(x) = \alpha * (k-1) = \alpha.$

2) При вході $x_{i_1} = 0;\ x_{i_2} = 1;\dots;\ x_{i_k} = k-1$ маємо наступне:

$\beta = 0 * \varphi_0(x) \vee 1 * \varphi_1(x) \vee \dots \vee (k-1) * \varphi_{k-1}(x) = x.$

3) При вході $x_{i_1} = k-1;\ x_{i_2} = k-2;\ \dots\ ; x_{i_k} = 0$ маємо:

$\beta = (k-1) * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee .. \vee 0 * \varphi_{k-1}(x) = \overline{x} = k-1-x.$

Відмітимо, що доведення цього факту випливає зі збігу правої та лівої частин при довільних значеннях x із множини $\{0,1,2,\dots,k-1\}$.

4) Нехай вхідна дія полягає в наступному $- 0,1,\dots,i-1,i,i,\dots,i$ в цьому випадку будемо мати наступне:

$\beta = 0 * \varphi_0(x) \vee 1 * \varphi_1(x) \vee \dots \vee i * \varphi_i(x) \vee i * \varphi_{i+1}(x) \vee \dots \vee i * \varphi_{k-1}(x).$

Помножимо перші $(i+1)$ членів на i. Очевидно, що при цьому значення кон‘юнкцій не поміняється, оскільки $i -$ більше любої з цих кон‘юнкцій. Далі, починаючи з $(i+2)$ члена, кожну кон‘юнкцію множимо на індекс характеристичної функції, яка входить у дану кон‘юнкцію. Після виконання вказаних перетворень будемо мати наступне:

$$0 * i * \varphi_0(x) \vee 1 * i * \varphi_1(x) \vee \dots \vee i * i * \varphi_i(x) \vee i * (i+1) * \varphi_{i+1}(x) \vee \dots$$

$$\dots \vee i * (k-1) * \varphi_{k-1}(x) = (0 * \varphi_0(x) \vee 1 * \varphi_1(x) \vee \dots \vee i * \varphi_i(x) \vee$$

$$\vee (i+1) * \varphi_{i+1}(x) \vee \dots \vee (k-1) * \varphi_{k-1}(x)) * i = i * x$$

(тут використовується пункт 2).

5) Нехай вхідна дія полягає в наступному – $k-1, k-2, \dots, k-i-1, \dots, k-i-1$ в цьому випадку будемо мати:

$$\beta = (k-1) * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee \dots \vee (k-i) * \varphi_{i-1}(x) \vee$$
$$\vee (k-i-1) * \varphi_i(x) \vee \dots \vee (k-i-1) * \varphi_{k-1}(x).$$

На наступному етапі представимо вихідну функцію за формулою (4.1):

$$(k-1) * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee \dots \vee (k-i) * \varphi_{i-1}(x) \vee$$
$$\vee (k-i-1) * \varphi_i(x) \vee \dots \vee (k-i-1) * \varphi_{k-1}(x),$$

значення $k-1, \dots, k-i-1$ представимо у вигляді диз‘юнкції:

$$k-1 \vee k-i-1 = k-1; k-i-1 \vee k-i-1 = k-i-1,$$

а значення $k-i-1$, починаючи з $(i+2)-$ го члена у вигляді:

$$k-i-1 \vee k-i-2 = k-i-1, \dots, k-i-1 \vee 1 = k-i-1,$$
$$k-i-1 \vee 0 = k-i-1.$$

Далі, підставляючи ці значення у вираз вихідної функції, отримаємо:

$$(k-1 \vee k-i-1) * \varphi_0(x) \vee (k-2 \vee k-i-1) * \varphi_1(x) \vee \dots$$
$$\dots \vee (k-i-1 \vee k-i-1) * \varphi_i(x) \vee (k-i-1 \vee k-i-2) * \varphi_{i+1}(x) \vee \dots$$
$$\dots \vee (k-i-1 \vee 1) * \varphi_{k-2}(x) \vee (k-i-1 \vee 0) * \varphi_{k-1}(x) =$$
$$= ((k-1 * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee \dots \vee (k-i-1) * \varphi_i(x) \vee \dots$$
$$\dots \vee 1 * \varphi_{k-2}(x) \vee 0 * \varphi_{k-1}(x)) \vee (k-i-1) * (\varphi_0(x) \vee \varphi_1(x) \vee \dots$$
$$\dots \vee * \varphi_i(x) \vee \dots \vee \varphi_{k-2}(x) \vee \varphi_{k-1}(x)) = \overline{x} \vee k-i-1.$$

6) Нехай вхідна дія полягає в наступному – $i, i, \dots, i+1, i+2, \dots, k-2, k-1$. У цьому випадку $\beta = x \vee i$.

Доведення. Вихідна функція за формулою (4.1) представимо в наступному вигляді:

$$i * \varphi_0(x) \vee i * \varphi_1(x) \vee \dots \vee i * \varphi_i(x) \vee (i+1) * \varphi_{i+1}(x) \vee \dots$$
$$\dots \vee (k-1) * \varphi_{k-1}(x).$$

Значення i, яке входить в перші $i+1$ кон‘юнкції, будуть представлені у вигляді диз‘юнкцій зі значенням індексів, які відповідають характеристичним функціям. Значення членів не будуть мінятися, оскільки значення індексів менше ніж i. Починаючи з $(i+2)-$ го члена, кожний множник

$i+1, i+2, \dots, k-1$ представимо у вигляді диз‘юнкції з i. На значення кон‘юнкцій це перетворення також не впливає, оскільки i менше цих множників. Вихідну функцію в перетвореному вигляді представимо наступним чином: $x \vee i$.

7) Нехай вхідними функціями є сигнали $i, i, \dots, i, k-1, k-2, \dots, i$ або $i, i, \dots, i, k-1, k-2, \dots, i+1$.

У цьому разі будемо мати наступне:

$$\beta = i \vee (k-1) * \varphi_{i+1}(x) \vee \dots \vee i * \varphi_{k-1}(x) \quad \text{або}$$

$$\beta = i \vee (k-1) * \varphi_{i+1}(x) \vee \dots \vee (i+1) * \varphi_{k-1}(x).$$

Доведення. Вихідна функція конструюється наступним чином:

$$i * \varphi_0(x) \vee i * \varphi_1(x) \vee \dots \vee i * \varphi_i(x) \vee (k-1) * \varphi_{i+1}(x) \vee \dots \vee i * \varphi_{k-1}(x) =$$

$$i * \varphi_0(x) \vee i * \varphi_1(x) \vee \dots \vee i * \varphi_i(x) \vee \big((k-1) \vee i\big) * \varphi_{i+1}(x) \vee \dots$$

$$\dots \vee (i \vee i) * \varphi_{k-1}(x) = i * (\varphi_0(x) \vee \varphi_1(x) \vee \dots \vee \varphi_i(x) \vee \varphi_{i+1}(x) \vee \dots$$

$$\dots \vee \varphi_{k-1}(x)) \vee (k-1) * \varphi_{i+1}(x) \vee \dots \vee i * \varphi_{k-1}(x) =$$

$$= i \vee (k-1) * \varphi_{i+1}(x) \vee \dots \vee i * \varphi_{k-1}(x).$$

Для вхідної функції $i, i, \dots, i, k-1, k-2, \dots, i+1$ доведення буде аналогічним.

8) Нехай вхідними функціями є дія $k-1, k-2, \dots, i+1, i, i+1, \dots, k-2, k-1$. Вихідна функція представляється у вигляді $\beta = x \vee \overline{x}$.

Доведення. Вихідна функція конструюється наступним чином:

$$(k-1) * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee \dots \vee (i+1) * \varphi_{i-1}(x) \vee i * \varphi_0(x) \vee$$

$$\vee (i+1) * \varphi_{i+1}(x) \vee \dots \vee (k-1) * \varphi_{k-1}(x) = \big((k-1) \vee 0\big) * \varphi_0(x) \vee$$

$$\vee \big((k-2) \vee 1\big) * \varphi_1(x) \vee \dots \vee \big((i+1) \vee (i-1)\big) * \varphi_{i-1}(x) \vee (i \vee i) * \varphi_i(x) \vee$$

$$\vee \big((i+1) \vee (k-i-2)\big) * \varphi_{i+1}(x) \vee \dots \vee \big((k-1) \vee 0\big) * \varphi_{k-1}(x).$$

Беручи до уваги дистрибутивний закон, матимемо наступне:

$$\big(0 * \varphi_0(x) \vee 1 * \varphi_1(x) \vee \dots \vee i * \varphi_i(x) \vee \dots \vee (k-1) * \varphi_{k-1}(x)\big) \vee$$

$$\vee \big((k-1) * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee \dots \vee i * \varphi_i(x) \vee \dots \vee 0 * \varphi_{k-1}(x)\big) =$$

$$= x \vee \overline{x}.$$

9) Нехай вхідними функціями є дія $0,1, \dots, i-1, i, i-1, \dots, 1,0$.

Вихідна функція представляється у вигляді $x * \overline{x}$.

Доведення. Вихідна функція конструюється наступним чином:

$$0 * \varphi_0(x) \vee 1 * \varphi_1(x) \vee \ldots \vee (i-1) * \varphi_{i-1}(x) \vee i * \varphi_i(x) \vee$$
$$\vee (i-1) * \varphi_{i+1}(x) \vee \ldots \vee 0 * \varphi_{k-1}(x).$$

На наступному етапі всі члени цього виразу до $(i+1)-$ го включно помножимо відповідно на значення $k-1, k-2, \ldots, i$. При цьому значення даних кон‘юнкцій не змінюється. Всі члени, починаючи з $(i+1)-$ го місця, помножимо на індекс характеристичної функції, яка входить у дану кон‘юнкцію. Після таких перетворень отримаємо наступний вираз:

$$0 * (k-1) * \varphi_0(x) \vee 1 * (k-2) * \varphi_1(x) \vee \ldots \vee (i-1) * (i+1) * \varphi_{i-1}(x) \vee$$
$$i * i * \varphi_i(x) \vee (i-1) * (i+1) * \varphi_{i+1}(x) \vee \ldots \vee (k-2) * 1 * \varphi_{k-2}(x) \vee$$
$$\vee (k-1) * 0 * \varphi_{k-1}(x).$$

Відмітимо, що отриманий вираз можна представити наступним чином:

$$\big(0 * \varphi_0(x) \vee 1 * \varphi_1(x) \vee \ldots \vee i * \varphi_i(x) \vee \ldots \vee (k-1) * \varphi_{k-1}(x)\big) \wedge$$
$$\wedge ((k-1) * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee \ldots \vee i * \varphi_i(x) \vee \ldots$$
$$\ldots \vee 0 * \varphi_{k-1}(x)) = x * \overline{x}.$$

10) Якщо вхідними функціями є наступна дія $i, \ldots, i, i, i-1, \ldots, 1, 0$ – де i до $i-$ місця, тоді вихідна функція буде представлена в наступному вигляді: $\overline{x} \wedge i$.

Доведення. Вихідна функція за визначенням може бути представлена у вигляді:

$$i * \varphi_0(x) \vee i * \varphi_1(x) \vee \ldots \vee i * \varphi_{i-1}(x) \vee (i-1) * \varphi_i(x) \vee (i-2) * \varphi_{i+1}(x) \vee \ldots$$
$$\ldots \vee 0 * \varphi_{k-1}(x) = i * (k-1) * \varphi_0(x) \vee i * (k-2) * \varphi_1(x) \vee \ldots$$
$$\ldots \vee (i+1) * i * \varphi_{i-1}(x) \vee (i-1) * i * \varphi_i(x) \vee (i-2) * i * \varphi_{i+1}(x) \vee \ldots$$
$$\ldots \vee 0 * i * \varphi_{k-1}(x) = ((k-1) * \varphi_0(x) \vee (k-2) * \varphi_1(x) \vee \ldots$$
$$\ldots \vee (i+1) * \varphi_{i-1}(x) \vee \ldots \vee 0 * \varphi_{k-1}(x) \wedge i = \overline{x} \wedge i.$$

11.(а)) Нехай вхідна дія буде в наступному вигляді:

$$\underbrace{i, i, \ldots, i}_{l}, i+j, i+j, \ldots, i+j \text{ де } i+j \leq k-1.$$

У цьому разі будемо мати наступне:

$$\beta = i \vee (i+j) * \varphi_l(x) \vee (i+j) * \varphi_{l+1}(x) \vee \ldots \vee (i+j) * \varphi_{k-1}(x),$$

при $i + j < k - 1$, а якщо $i + j = k - 1$, то відповідно в наступному вигляді: $i \vee \varphi_l(x) \vee \ldots \vee \varphi_{k-1}(x)$.

Доведення. Представимо дану вихідну функцію за допомогою початкової формули (4.1):

$$i * \varphi_0(x) \vee i * \varphi_1(x) \vee \ldots \vee i * \varphi_{l-1}(x) \vee (i+j) * \varphi_l(x) \vee (i+j) * \varphi_{l+1}(x) \vee$$
$$\vee \ldots \vee (i+j) * \varphi_{k-1}(x) = i * \varphi_0(x) \vee \ldots \vee i * \varphi_{l-1}(x) \vee \big(i \vee (i+j)\big) * \varphi_l(x) \vee$$
$$\vee \big(i \vee (i+j)\big) * \varphi_{l+1}(x) \vee \ldots \vee \big(i \vee (i+j)\big) * \varphi_{k-1}(x) = (i * \varphi_0(x) \vee \ldots$$
$$\ldots \vee i * \varphi_{l-1}(x) \vee i * \varphi_l(x) \vee i * \varphi_{l+1}(x) \vee \ldots \vee i * \varphi_{k-1}(x)) \vee$$
$$\vee (i+j) * \varphi_l(x) \vee (i+j) * \varphi_{l+1}(x) \vee \ldots \vee (i+j) * \varphi_{k-1}(x) =$$
$$= i * \big(\varphi_l(x) \vee \ldots \vee \varphi_{l-1}(x) \vee \ldots \vee \varphi_{k-1}(x)\big) \vee (i+j) * \varphi_l(x) \vee \ldots$$
$$\ldots \vee (i+j) * \varphi_{k-1}(x) = i \vee (i+j) * \varphi_l(x) \vee \ldots \vee (i+j) * \varphi_{k-1}(x).$$

11.(b)) Нехай вхідна дія буде наступною:

$$\underbrace{i, i, \ldots, i}_{l}, \underbrace{i+j, i+j, \ldots, i+j}_{m+1}, i, i, \ldots, i.$$

Вихідна функція буде представлена в такому вигляді:

$$\beta = i \vee (i+j) * \varphi_l(x) \vee \ldots \vee (i+j) * \varphi_{l+m}(x).$$

11.(c)) Нехай вхідна дія буде наступною:

$$\underbrace{i+j, i+j, \ldots, i+j}_{m+1}, i, i, \ldots, i.$$

Вихідна функція буде представлена в такому вигляді:

$$\beta = (i+j) * \varphi_0(x) \vee \ldots \vee (i+j) * \varphi_m(x) \vee i.$$

Відмітимо, що доведення зауважень (11.(b)) та (11.(c)) є аналогічним доведенню (11.(a)).

Визначення 4.6. Нехай задано довільну функцію двох змінних $f(x_1, x_2)$ багатозначної логіки. Трикутником будемо називати логічне дерево, яке можна побудувати для даної функції (рис. 4.13).

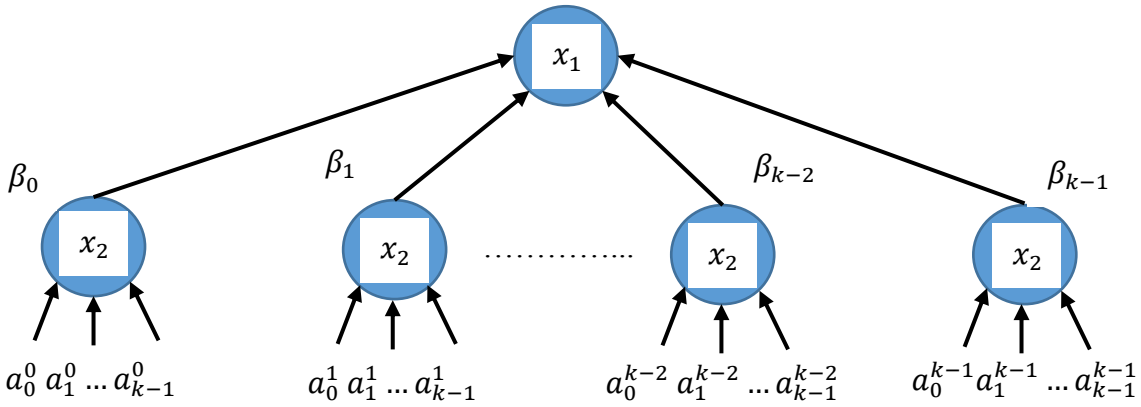


Рис. 4.13. Трикутник (логічне дерево) функції $f(x_1, x_2)$.

Теорема 4.2.

$$x_1 * (x_2 * (\alpha_0^0, \alpha_1^0, \dots, \alpha_{k-1}^0); x_2 * (\alpha_0^1, \alpha_1^1, \dots, \alpha_{k-1}^1); \dots$$
$$\dots; x_2 * (\alpha_0^{k-1}, \alpha_1^{k-1}, \dots, \alpha_{k-1}^{k-1})) = x_2 * (x_1 * (\alpha_0^0, \alpha_0^1, \dots, \alpha_0^{k-1});$$
$$; x_1 * (\alpha_1^0, \alpha_1^1, \dots, \alpha_1^{k-1}); \dots; x_1 * (\alpha_{k-1}^0, \alpha_{k-1}^1, \dots, \alpha_{k-1}^{k-1})).$$

Теорема визначає, що при перестановці ярусів логічного дерева, відповідних змінних x_1 та x_2, вхідні значення $\alpha_0^0, \dots, \alpha_{k-1}^0, \alpha_0^1, \alpha_1^1, \dots, \alpha_{k-1}^1, \alpha_{k-1}^1, \dots, \alpha_{k-1}^{k-1}$ міняються місцями та представляються у вигляді $\alpha_0^0, \dots, \alpha_0^{k-1}, \alpha_1^0, \dots, \alpha_{k-1}^0, \dots, \alpha_{k-1}^{k-1}$ (рис. 4.14).

Доведення. Застосувавши теорему 4.2, функція $f(x_1, x_2)$ може бути представлена в наступному вигляді:

$$f(x_1, x_2) = \left(\alpha_0^0 * \varphi_0(x_2) \vee \alpha_1^0 * \varphi_1(x_2) \vee \dots \vee \alpha_{k-1}^0 * \varphi_{k-1}(x_2)\right) * \varphi_0(x_1) \vee$$
$$\vee \left(\alpha_0^1 * \varphi_0(x_2) \vee \alpha_1^1 * \varphi_1(x_2) \vee \dots \vee \alpha_{k-1}^1 * \varphi_{k-1}(x_2)\right) * \varphi_1(x_1) \vee \dots$$
$$\dots \vee \left(\alpha_0^{k-1} * \varphi_0(x_2) \vee \alpha_1^{k-1} * \varphi_1(x_2) \vee \dots \vee \alpha_{k-1}^{k-1} * \varphi_{k-1}(x_2)\right) * \varphi_{k-1}(x_1).$$

На наступному етапі, провівши перетворення даного виразу за дистрибутивним законом, отримаємо:

$$\left(\alpha_0^0 * \varphi_0(x_1) \vee \alpha_0^1 * \varphi_1(x_1) \vee \dots \vee \alpha_0^{k-1} * \varphi_{k-1}(x_1)\right) * \varphi_0(x_2) \vee$$
$$\vee \left(\alpha_1^0 * \varphi_0(x_1) \vee \alpha_1^1 * \varphi_1(x_1) \vee \dots \vee \alpha_1^{k-1} * \varphi_{k-1}(x_1)\right) * \varphi_1(x_2) \vee \dots$$
$$\dots \vee \left(\alpha_{k-1}^0 * \varphi_0(x_1) \vee \alpha_{k-1}^1 * \varphi_1(x_1) \vee \dots \vee \alpha_{k-1}^{k-1} * \varphi_{k-1}(x_1)\right) * \varphi_{k-1}(x_2).$$

Відмітимо, що отриманий вираз представляє новий запис логічної функції $f(x_1, x_2)$, яка відповідає перестановці місцями ярусів змінних x_1 та x_2 в логічному дереві. Отриманий порядок вхідних функцій співпадає з тим порядком, зазначеним у теоремі 4.2. Саме цим фактом і доводиться справедливість даної теореми.

Припустимо, що задано багатозначну функцію від n змінних. Побудуємо дерево даної функції. Розглянемо трикутник, який утворюється першим та другим каскадами (даний трикутник співпадає з рис. 4.13, але вже $w = f(x_1, \dots, x_2)$) та трикутник, який отримується з нього перестановкою x_1 та x_2 (цей трикутник співпадає з рис. 4.14) але $w = f(x_1, x_2, \dots, x_n)$).

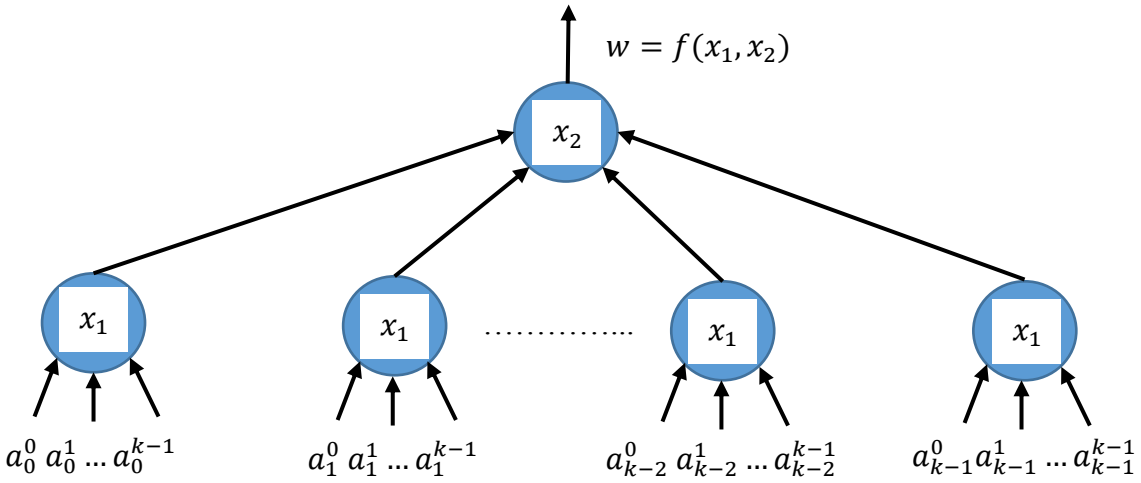


Рис. 4.14. Трикутник (логічне дерево) функції $f(x_1, x_2)$ при перестановці ярусів відповідних змінних x_1 та x_2.

Оскільки мінімізація отримується за рахунок зменшення кількості різних функцій, то доцільно розглянути випадок, коли $\alpha_0^0 = \alpha_0^1 = \dots = \alpha_0^{k-1}$ або $\alpha_1^0 = \alpha_1^1 = \dots = \alpha_1^{k-1}, \dots, \alpha_{k-1}^0 = \dots = \alpha_{k-1}^{k-1}$, тому що при цій перестановці змінних можна отримати зменшення кількості різних функцій.

Для визначеності припустимо, що:

$\alpha_0^0 = \alpha_1^0 = \dots = \alpha_0^{k-1}, \alpha_0^0 \neq \alpha_0^1 \neq \dots \neq \alpha_{k-1}^0, \dots, \alpha_0^{k-1} \neq \alpha_1^{k-1} \neq \dots \neq \alpha_{k-1}^{k-1}$.

Зауважимо, що для всіх інших випадків загальна схема мінімізації буде аналогічною. У цьому разі $\alpha_0^1 = \beta_0^1$. Відмітимо, що при дослідженні можуть виникнути декілька варіантів:

1) Функції $\beta_0, \beta_1, \dots, \beta_{k-1}$ (рис. 4.13) у нижніх ярусах дерева не зустрічались, а отже, перестановкою змінних x_1 та x_2 замість k функцій $\beta_0, \beta_1, \dots, \beta_{k-1}$ отримаємо тільки $k-1$ функцій, серед яких можуть бути і такі, які вже зустрічалися на нижніх ярусах. Відповідно перестановка x_1 та x_2 приводить до зменшення кількості різних функцій.

2) Деякі m логічних функцій з $\beta_0, \beta_1, \dots, \beta_{k-1}; (1 \le m \le k-1)$ в нижніх ярусах дерева зустрічались, тоді знову можливі два варіанти:

2.(a)) Якщо при перестановці змінних x_1 та x_2 отримаємо кількість функцій $p, (p > m)$, які вже зустрічались на нижніх ярусах, то тим самим досягнуть зменшення кількості різних функцій.

2.(b)) Якщо при перестановці змінних x_1 та x_2 отримаємо, що функції $\beta_0, \beta_1, \dots, \beta_{k-1}$ (рис. 4.14) в нижніх ярусах не зустрічались, то перестановка не зменшує кількості різних функцій. Тоді змінні x_1 та x_2 переставляти не вигідно.

3) Якщо всі функції $\beta_0, \beta_1, \dots, \beta_{k-1}$ структури логічного дерева (рис. 4.13) в нижніх ярусах зустрічались, тоді перестановка змінних x_1 та x_2 не може зменшити кількість різних функцій (а, отже, і структурну складність логічного дерева, що мінімізується), відповідно вона буде невигідною щодо процедури мінімізації.

Використовуючи вище сформульовану в даному розділі дослідження властивість трикутника, можна визначити алгоритм мінімізації логічного дерева (графа) довільної функції.

Нехай задано довільну логічну функцію n змінних $f(x_1, x_2, \dots, x_n)$. Побудуємо для неї логічне дерево. Розглянемо трикутник, який утворюється першим та другим каскадами, використовуючи властивості трикутника, якщо потрібно, переставляємо змінні x_1 та x_2. На наступному етапі перевіряємо трикутники, які утворені другим та третім каскадами. Застосуємо аналогічні судження до кожного трикутника, спускаючись вниз за ярусами, доки не

перевіримо всі трикутники. Отже, отримаємо наступний алгоритм мінімізації за логічним деревом.

4.3.2. Загальна схема мінімізації функції на основі логічного дерева

1) На першому етапі необхідно за табличною (або іншою) формою задання довільної логічної функції n змінних $f(x_1, x_2, \dots, x_n)$ побудувати відповідне логічне дерево.

2) Далі – на отриманій структурі логічного дерева провести розстановку відповідних міток (атрибутів) дерева та проаналізувати можливий ефект (в плані оптимізації структури дерева) від перестановки в його структурі ярусів (цим питанням буде приділено увагу далі в межах даного дослідження).

3) На наступному етапі мінімізації, використовуючи теорему 4.1 (про розклад функції n змінних $f(x_1, x_2, \dots, x_n)$ за фіксованою змінною), необхідно представити логічну функції через відповідні мітки (атрибути).

4) Усі мітки логічного дерева відповідної функції n змінних $f(x_1, x_2, \dots, x_n)$ необхідно представити через змінні $x_1, x_2, \dots, x_n$, тобто отримаємо деяке дужкове представлення логічної функції.

В отриманому представленні логічної функції n змінних $f(x_1, x_2, \dots, x_n)$, використовуючи різні співвідношення, проводиться процес подальшого спрощення (наприклад, 2 – й дистрибутивний закон або інші) та дужкові перетворення.

З вищезазначеного можна відмітити такі переваги запропонованого простого метода мінімізації логічної функції відносно існуючих методів:

1) простота роботи за вказаною схемою на відповідній графічній структурі (конструкції логічного дерева);

2) значна наочність запропонованого методу порівняно з іншими методами є прямим або опосередкованим узагальненням двійкових (бінарних) методик для багатозначного випадку;

3) можливість нескладної програмної реалізації, що значно розширює сферу дій для великих n та k у випадку багатозначної логіки;

4) природність запропонованого графічного представлення функцій багатозначної логіки (на основі концепції логічних дерев).

На даному етапі дослідження розглянемо приклад:

Приклад 4. Спростити функцію $f(x_1, x_2)$, задану таблично $(k = 5)$.

Таблиця 4.3. Таблична форма логічної функції $f(x_1, x_2)$ прикладу 4.

f	0	0	0	4	4	1	1	0	3	4	2	2	0	2	2	3	3	0	1	1	4	4	0	0	0
x_1	0	0	0	0	0	1	1	1	1	1	2	2	2	2	2	3	3	3	3	3	4	4	4	4	4
x_2	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4

Побудуємо логічне дерево для даної функції (рис. 4.15).

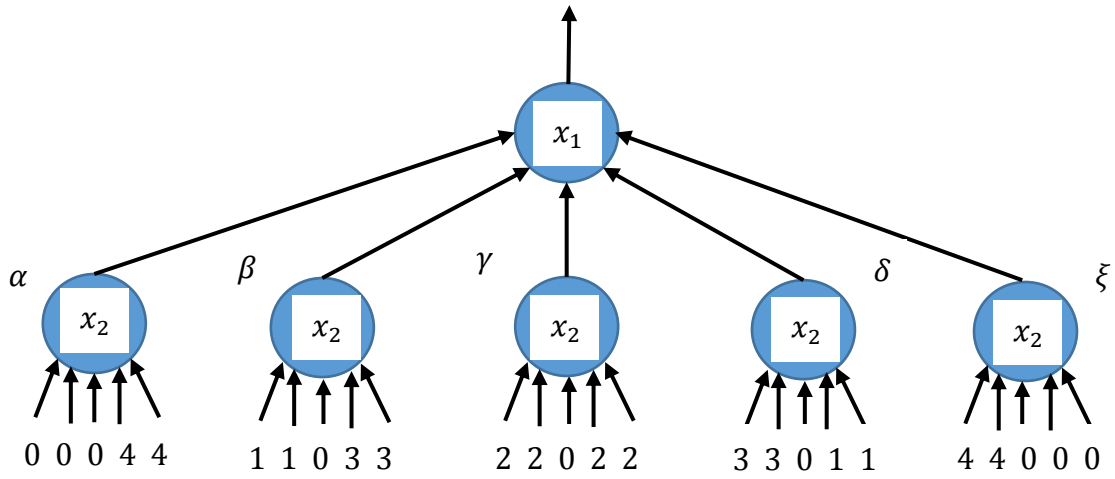


Рис. 4.15. Логічне дерево функції $f(x_1, x_2)$ прикладу 4.

$$\alpha = \varphi_3(x_2) \vee \varphi_4(x_2);$$

$$\varepsilon = \varphi_0(x_2) \vee \varphi_1(x_2);$$

$$\beta = 1 * \varphi_0(x_2) \vee 1 * \varphi_1(x_2) \vee 3 * \varphi_3(x_2) \vee 3 * \varphi_4(x_2);$$

$$\gamma = 2 * \varphi_0(x_2) \vee 2 * \varphi_1(x_2) \vee 2 * \varphi_3(x_2) \vee 2 * \varphi_4(x_2);$$

$$\delta = 3 * \varphi_0(x_2) \vee 3 * \varphi_1(x_2) \vee 1 * \varphi_3(x_2) \vee 1 * \varphi_4(x_2);$$

$$w = \alpha * \varphi_0(x_1) \vee \beta * \varphi_1(x_1) \vee \gamma * \varphi_2(x_1) \vee \delta * \varphi_3(x_1) \vee \xi * \varphi_4(x_1);$$

$$w = \varphi_3(x_2) * \varphi_0(x_1) \vee \varphi_4(x_2) * \varphi_0(x_1) \vee 1 * \varphi_0(x_2) * \varphi_1(x_1) \vee$$

$$\vee 1 * \varphi_1(x_2) * \varphi_1(x_1) \vee 3 * \varphi_3(x_2) * \varphi_1(x_1) \vee 3 * \varphi_4(x_2) * \varphi_1(x_1) \vee$$

$$\vee 2 * \varphi_0(x_2) * \varphi_2(x_1) \vee 2 * \varphi_1(x_2) * \varphi_2(x_1) \vee 2 * \varphi_4(x_2) * \varphi_2(x_1) \vee$$

$$\vee 3 * \varphi_0(x_2) * \varphi_3(x_1) \vee 3 * \varphi_1(x_3) * \varphi_3(x_1) \vee 1 * \varphi_3(x_2) * \varphi_3(x_1) \vee$$

$$\vee 1 * \varphi_4(x_2) * \varphi_2(x_1) \vee \varphi_0(x_2) * \varphi_4(x_1) \vee \varphi_1(x_2) * \varphi_4(x_1).$$

Відмітимо, що дане представлення початкової логічної функції двох змінних $f(x_1, x_2)$ не є її мінімальною ДНФ, оскільки можна провести ще додатково операцію поглинання.

Тому на наступному етапі побудуємо та розглянемо структуру логічного дерева після проведення операції перестановки змінних x_1 та x_2 (рис. 4.16).

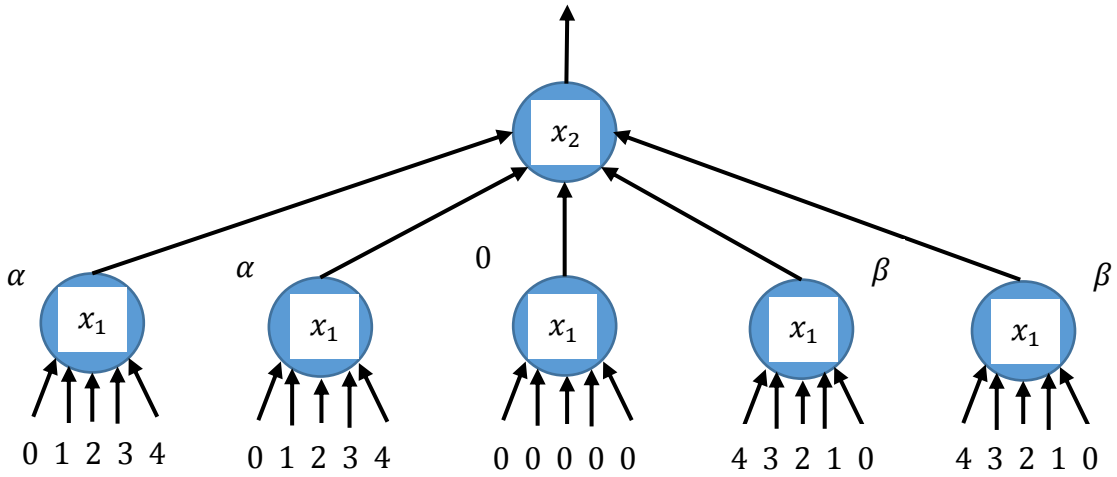


Рис. 4.16. Логічне дерево функції $f(x_1, x_2)$ прикладу 4 після перестановки змінних x_1 та x_2.

$$\alpha = x_1;\ \beta = \overline{x_1}; w = x_1 * \varphi_0(x_2) \vee x_1 * \varphi_1(x_2) \vee \overline{x_1} * \varphi_3(x_2) \vee \overline{x_1} * \varphi_4(x_2).$$

Далі проведемо процедури мінімізації даної логічної функції, для цього представимо логічну функцію в ДДНФ та проведемо відповідні перетворення.

$$f(x_1, x_2) = 4 * \varphi_0(x_1) * \varphi_3(x_2) \vee 4 * \varphi_0(x_1) * \varphi_4(x_2) \vee 1 * \varphi_1(x_1) * \varphi_0(x_2) \vee$$
$$\vee 1 * \varphi_1(x_1) * \varphi_1(x_2) \vee 3 * \varphi_1(x_1) * \varphi_3(x_2) \vee 3 * \varphi_1(x_1) * \varphi_2(x_2) \vee$$
$$\vee 2 * \varphi_2(x_1) * \varphi_0(x_2) \vee 2 * \varphi_2(x_1) * \varphi_1(x_2) \vee 2 * \varphi_2(x_1) * \varphi_3(x_2) \vee$$
$$\vee 2 * \varphi_2(x_1) * \varphi_4(x_2) \vee 3 * \varphi_3(x_1) * \varphi_0(x_2) \vee 3 * \varphi_3(x_1) * \varphi_1(x_2) \vee$$
$$\vee 1 * \varphi_3(x_1) * \varphi_3(x_2) \vee 1 * \varphi_3(x_1) * \varphi_4(x_2) \vee 4 * \varphi_4(x_1) * \varphi_0(x_2) \vee$$
$$\vee 4 * \varphi_4(x_1) * \varphi_1(x_2) = (1 * \varphi_1(x_1) \vee 2 * \varphi_2(x_1) \vee 3 * \varphi_3(x_1) \vee$$
$$\vee 4 * \varphi_4(x_1)) * \varphi_0(x_2) \vee (1 * \varphi_1(x_1) \vee 2 * \varphi_2(x_1) \vee 3 * \varphi_3(x_1) \vee$$
$$\vee 4 * \varphi_4(x_1)) * \varphi_1(x_2) \vee (4 * \varphi_0(x_1) \vee 3 * \varphi_1(x_1) \vee 2 * \varphi_2(x_1) \vee 1 * \varphi_3(x_1)) *$$
$$* \varphi_3(x_2) \vee \big(4 * \varphi_0(x_1) \vee 3 * \varphi_1(x_1) \vee 2 * \varphi_2(x_1) \vee 1 * \varphi_3(x_1)\big) * \varphi_4(x_2) =$$
$$= x_1 * \varphi_0(x_2) \vee x_1 * \varphi_1(x_2) \vee \overline{x_1} * \varphi_3(x_2) \vee \overline{x_1} * \varphi_4(x_2).$$

Отже, на виході отримаємо наступний вираз:

$$f(x_1, x_2) = x_1 * \varphi_0(x_2) \vee x_1 * \varphi_1(x_2) \vee \overline{x_1} * \varphi_3(x_2) \vee \overline{x_1} * \varphi_4(x_2).$$

Зважаючи на все вищезазначене, зафіксуємо наступне:

1) Відмітимо, що актуальним залишається питання ефективної мінімізації регулярного логічного дерева (моделі ЛДК) за допомогою методу перестановки ярусів у його структурі, оскільки логічне дерево також можна досить ефективно використовувати для мінімізації функції багатозначної логіки, яке воно представляє.

2) Запропонований метод мінімізації на основі концепції логічних дерев можна досить ефективно використовувати для мінімізації функції багатозначної логіки, а, отже, забезпечувати мінімальну форму дерева класифікації, яке вона представляє. Підкреслимо також, що даний метод дозволяє просту та ефективну програмну реалізацію – що значно розширює сферу дій для великих n та k у випадку багатозначної логіки.

3) Відмітимо, що важливим моментом при такому методі мінімізації логічних функції є пошук серед $n!$ логічних дерев (логічних структур) найкращого (оптимального в деякому сенсі) дерева, тобто логічного дерева, яке дає мінімальну або близьку до неї форму.

4) Зауважимо, що, як буде показано в даному дослідженні далі, загальна схема пошуку оптимального логічного дерева, яке дає мінімальну або близьку до неї форму (логічну структуру), значно простіше ніж застосовувати інші актуальні методи мінімізації функцій багатозначної логіки.

4.4. Схема оцінки ефекту перестановки ярусів найскладнішого логічного дерева для бінарного випадку

Для того, щоб оцінити вплив процедури перестановки ярусів логічного дерева, розглянемо питання, як зміниться найскладніше (регулярне) логічне дерево після перестановки ярусів (для спрощення розглянемо випадок двозначної логіки, тобто $k = 2$). На даному етапі відмітимо лише, що загальну схему побудови самого складного логічного дерева буде розглянуто в даному дослідженні далі.

Так, структура найскладнішого логічного дерева має вигляд, представлений на рис. 4.17. Зауважимо, що число, яке стоїть біля кожного ярусу, є номером даного ярусу, а набор чисел з лівого боку – кількістю вершин, з правого – кількістю функцій.

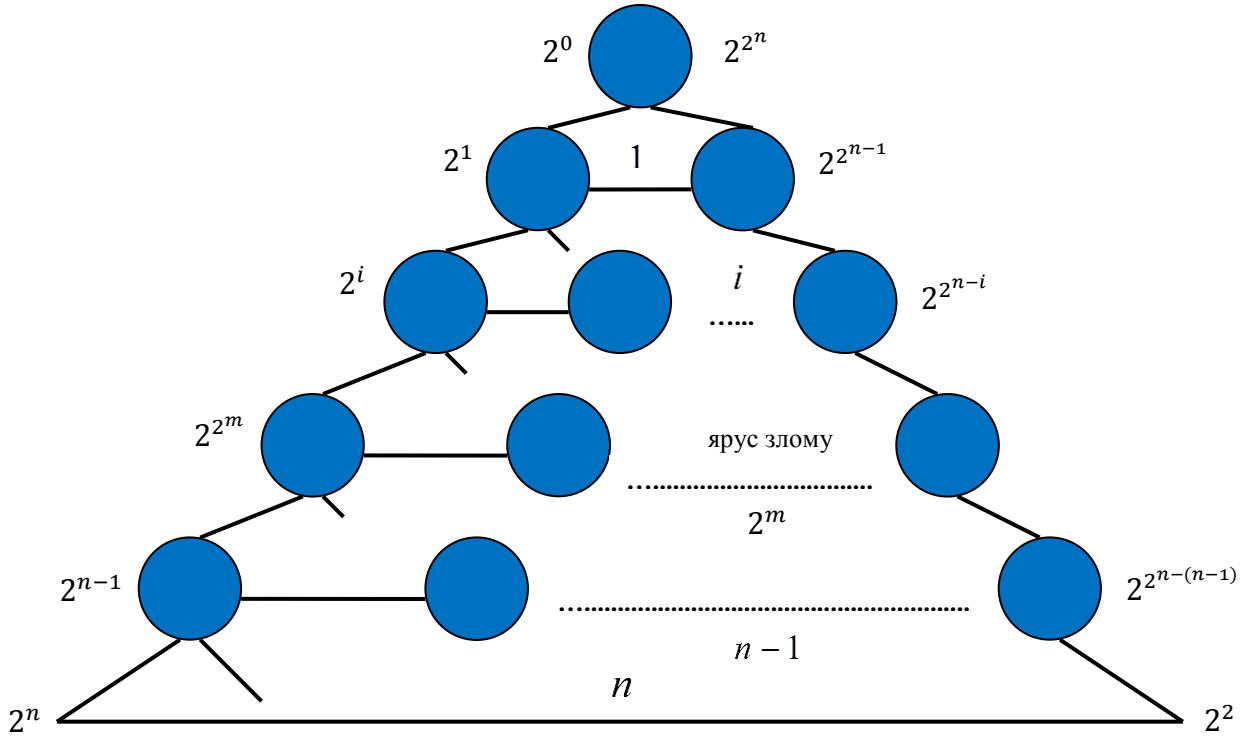


Рис. 4.17. Загальна структура найскладнішого логічного дерева.

Для спрощення представлення виберемо $n -$ кількість аргументів – так, щоб $n = 2^m$ $(m = 1,2, \dots)$. У цьому разі номер ярусу злому завжди дорівнюватиме 2^m.

4.4.1. Ярус злому структури логічного дерева

Визначення 4.7. Під ярусом злому будемо розуміти такий ярус деякого логічного дерева за номером i, де всі функції різні в усіх вершинах (вузлах) дерева, тобто - $2^i = 2^{2^{n-i}}$.

Дійсно, при $i = 2^m$ маємо $2^{2^m} = 2^{2^{2^m+m-2^m}} = 2^{2^m}$. Отже, на ярусі злому $i = 2^m$ є 2^{2^m} вершин і в кожній вершині стоять різні функції, тобто різних функцій також всього 2^{2^m}.

На ярусі з номером $i - 1 = 2^m - 1$ буде 2^{2^m+1} вершин та $2^{2^{m-1}}$ різних функцій, тобто менше ніж вершин, тому в деяких вершинах будуть стояти однакові функції.

На наступному етапі позначимо всі функції $(2^m - 1)$ ярусу через $f_1, f_2, \dots, f_{2^{2^{m-1}}}$. Покажемо, що ці функції можна розташувати на $(2^m - 1)$ ярусі способом, зображеним на рис. 4.18.

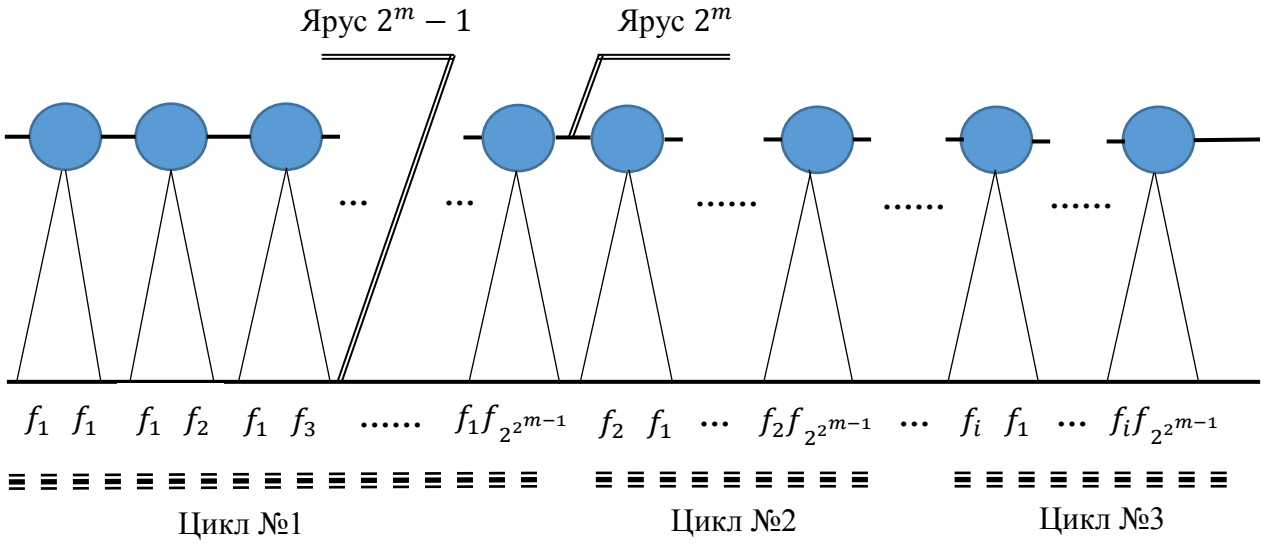


Рис. 4.18. Розбиття міток логічного дерева за циклами.

Відмітимо, що в кожному циклі маємо $2^{2^{m-1}} * 2 = 2^{2^{m-1}+1}$ функцій, а всього циклів буде відповідно $- 2^{2^{m-1}}$, звідси можна зробити висновок, що всього функцій буде $2^{2^{m-1}+1} * 2^{2^{m-1}} = 2^{2*2^{m-1}+1} = 2^{2^{m+1}}$, тобто фактично стільки, скільки є вершин на $(2^m - 1) -$ ому ярусі логічного дерева.

На наступному етапі розрахуємо загальну кількість різних функцій для логічного дерева, починаючи з ярусу злому. Зауважимо, що всі функції в ярусах логічного дерева, які розташовані вище, будуть різними, що визначається використанням оператора розкладу.

Отже, зважаючи на все вищезазначене, будемо мати наступну структуру логічного дерева – (рис. 4.19).

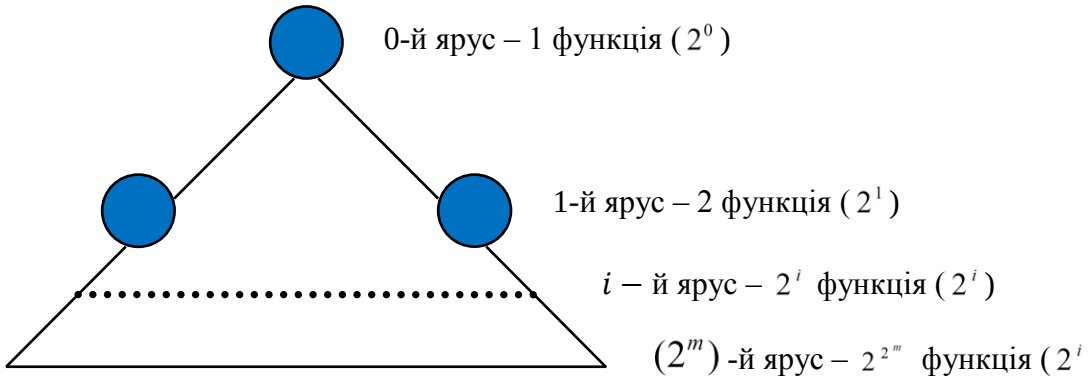


Рис. 4.19. Схема розрахунку вершин за ярусами.

Таким чином, всього функцій буде $2^0 + 2^1 + 2^2 + \cdots + 2^{2^m} = 2^{2^m} - 1$. Відмітимо, що це слідує з того простого факту, що:

$$1 + 2 + 2^2 + \cdots + 2^k = 2^{k+1} - 1.$$

Зокрема при цьому не потрібно врахувати функції, які розташовані до ярусу злому, оскільки якщо деяка функція – мітка ($\alpha = \varphi(x_{m+1}, x_{m+2}, \dots, x_n)$) зустрічалась в нижньому ярусі, то вона, очевидно, буде дорівнювати деякій функції мітці $d(x_m, x_{m+1}, \dots, x_n)$ на ярусі злому.

4.4.2. Вплив перестановки ярусів на структуру логічного дерева

Після цього розглянемо вплив перестановок ярусів на логічне дерево, починаючи з ярусу злому (2^m – го ярусу). Якщо переставити (2^m) – й ярус та $(2^m - 1)$ – й ярус, то отримаємо структуру, зображену на рис. 4.20. Відмітимо, що дана схема тільки для 1-го циклу, для всіх інших циклів вона аналогічна.

Зауважимо, що на ярусі $(2^m - 1)$ отримаємо таке саме розташування міток, яке було на (2^m) ярусі.

На наступному етапі поміняємо містами яруси $(2^m - 1)$ – й та $(2^m - 2)$ – й і так далі до $(2^m - 1)$ – го ярусу, після чого будемо мати структуру, подану на рис. 4.20.

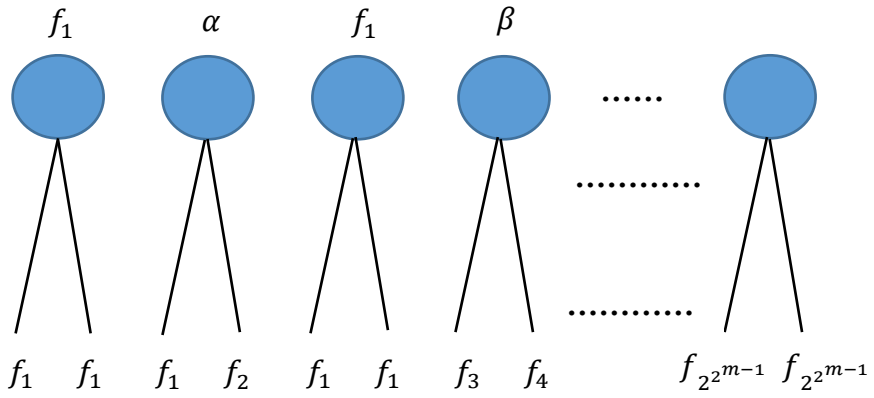


Рис. 4.20. Схема перестановки ярусів у логічному дереві.

Відмітимо, що тепер на $(2^m - 1)$ – му ярусі логічного дерева отримаємо конструкцію, яка буде аналогічною 1-му циклу на ярусі злому (тобто на (2^m) – му ярусі логічного дерева) – (рис. 4.21).

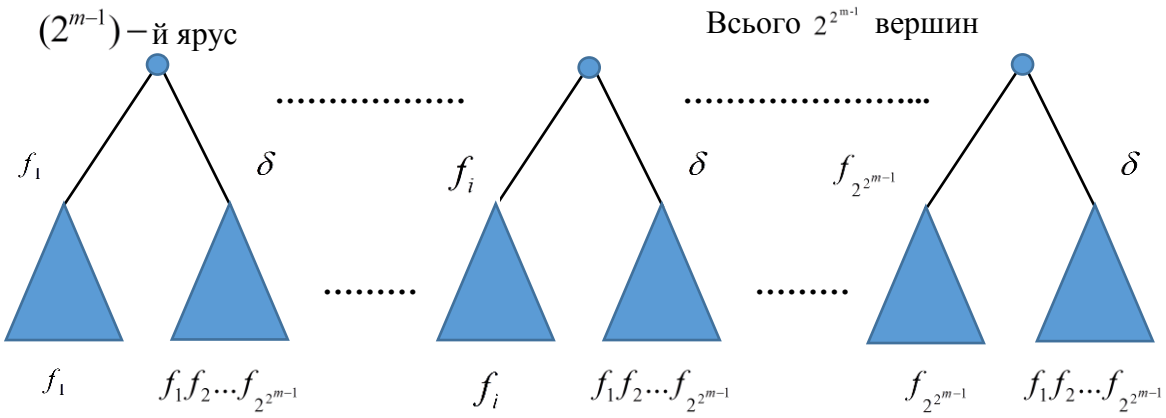


Рис. 4.21. Схема наступного кроку перестановки ярусів у логічному дереві.

Тобто процедурою перестановки ярусів в логічному дереві можна отримати наступну структуру логічного дерева, починаючи з $(2^{m-1}-1)$ ярусу дерева – (рис 4.22).

Відмітимо, що аналогічно вищевказаному кількість різних функцій у лівому трикутнику буде дорівнювати $2^0 + 2^1 + \cdots + 2^{2^{m-1}} = 2^{2^{m-1}+1} - 1$.

Остаточну структуру логічного дерева зображено на рис. 4.23.

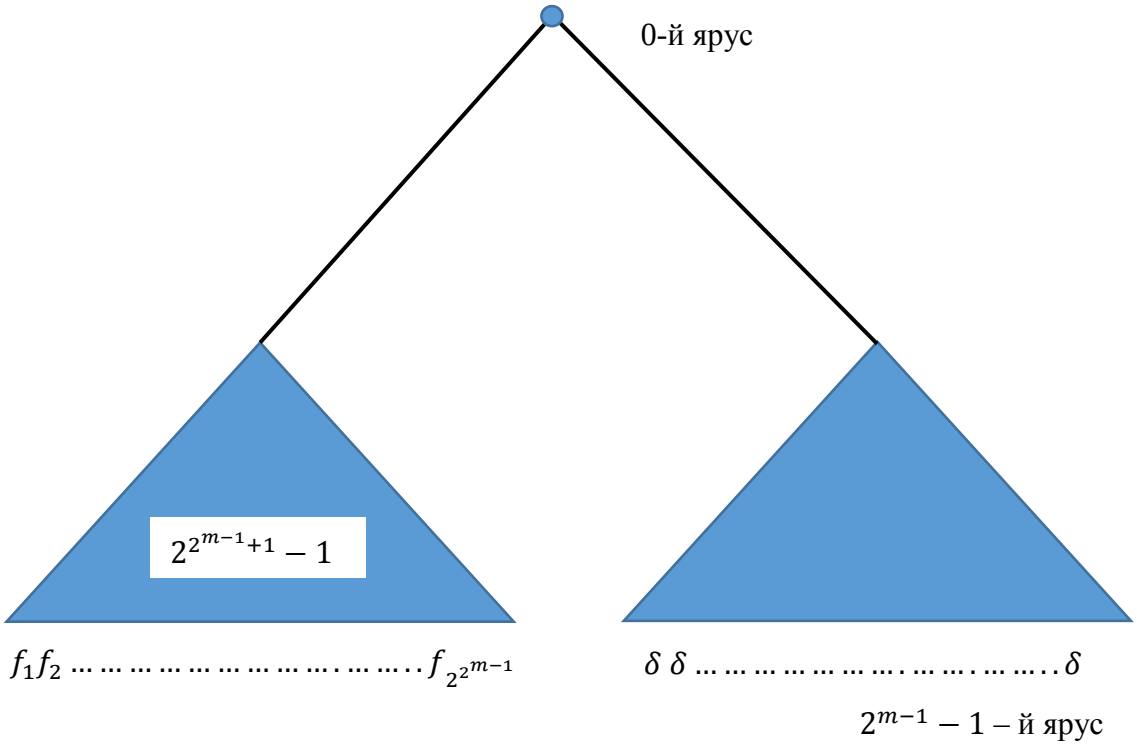


Рис. 4.22. Отримана структура логічного дерева після процедури перестановки ярусів.

Отже, загальна кількість всіх міток після проведення всіх перестановок буде дорівнювати:

$$2^{2^{m-1}}-1+2^{2^{m-1}+1}-1-1-2^{2^{m-1}}-1=3*2^{2^{m-1}}-1.$$

Так при $m=2$ до проведення перестановок ярусів маємо $2^{2^2+1}-1=31$ мітку, а після даної процедури: $3*2^{2^{2-1}}-1=11$.

Тобто після проведення процедури перестановок маємо виграш у складності фінальної структури логічного дерева майже втричі.

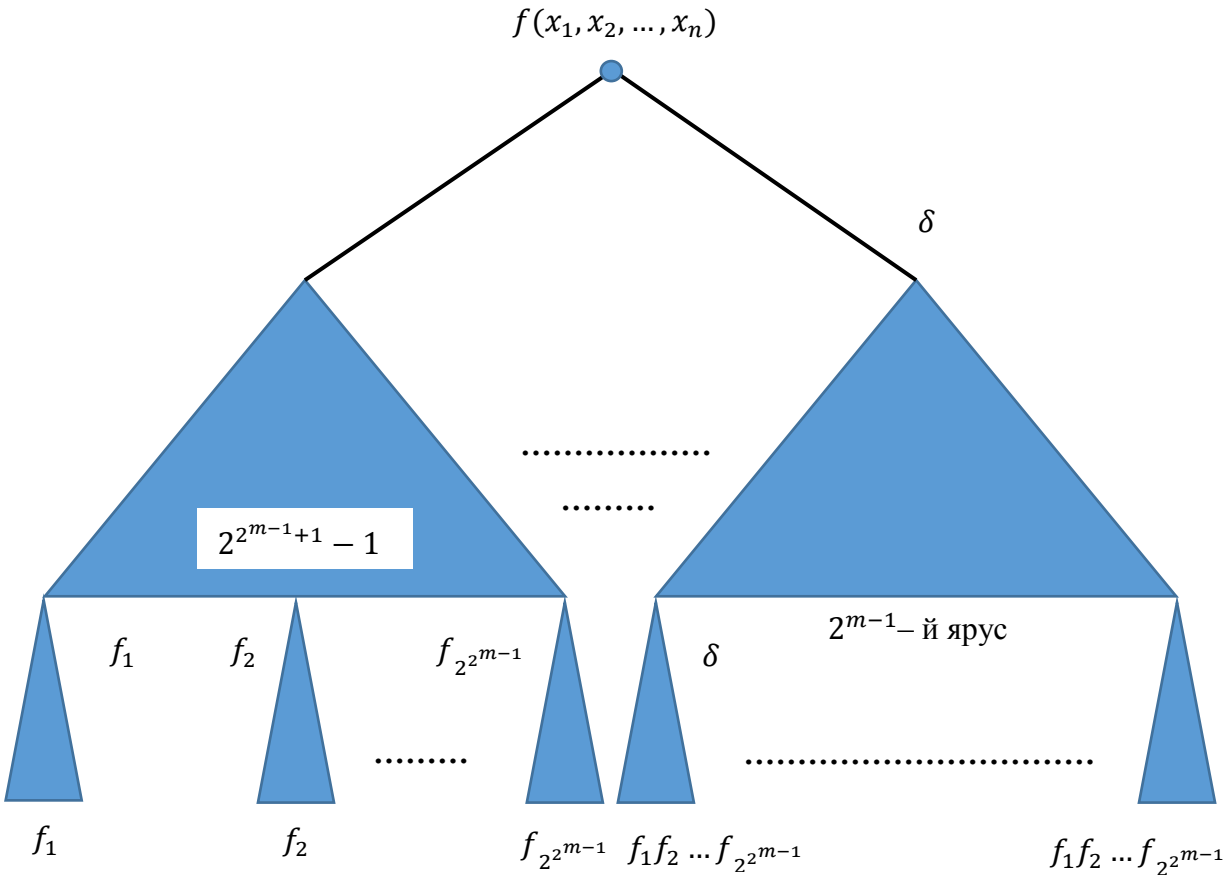


Рис. 4.23. Остаточна структура логічного дерева після проведення процедури перестановки ярусів.

На наступному етапі розрахуємо виграш у складності структури логічного дерева в загальному випадку:

$$\frac{2^{2^m+1}}{3*2^{2^{m-1}}} = \frac{1}{3} * 2^{2^{m-1}+1}. \quad (4.6)$$

При $m = 3$ маємо $\frac{1}{3} * 2^{2^{3-1}+1} \cong 10.$

При $m = 4$ маємо $\frac{1}{3} * 2^{2^{4-1}+1} \cong 160.$

Отже, можна зробити висновок, що зі зростанням числа m, та відповідно n, ефективність перестановки ярусів логічного дерева дуже швидко зростає.

Наприклад, при $m = 5$ маємо наступне: $\frac{1}{3} * 2^{2^{5-1}+1} = \frac{2^{17}}{3}$, при цьому число $n = 2^5 + 5 = 37$.

Отже, зважаючи на все вищезазначене, можна зробити висновки:

1) Важливим механізмом мінімізації (оптимізації структур) логічних дерев є процедура перестановки ярусів (блоків) у структурі дерева, яка дозволяє досягти помітного ефекту зменшення складності логічного дерева, причому слід відмітити, що зі зростанням числа m та, відповідно, n (структурної складності логічного дерева) ефективність перестановки ярусів дуже швидко зростає.

2) У розділі показано, що ефект перестановки ярусів логічного дерева є надзвичайно значним: дужкові форми логічних функцій, які знайдено за довільним деревом та деревом з перестановками, можуть відрізнятися в $\frac{1}{3} * 2^{2^{m-1}+1}$ разів. Отже, пошук ефективних перестановок (у структурі логічного дерева) є важливою задачею в даному підході. Питання такого пошуку в структурі логічного дерева ще буде підніматися в даному дослідженні далі.

3) Важливим напрямком дослідження даної проблематики в перспективі є вивчення питання, як зміниться найскладніше (регулярне за структурою) логічне дерево після перестановки ярусів для випадків ($k > 2$). Вирішення даного питання дозволить впровадити ефективніші схеми мінімізації структур ЛДК/АДК (для алгоритмічних дерев маються на увазі структури АДК типу II) без необхідності їх зведення до бінарного випадку ($k = 2$).

Висновки до розділу 4

1. У розділі підкреслено, що довільну функцію $k-$значної логіки від n змінних (задану в аналітичному вигляді) можна представити у формі логічного дерева, і навпаки – логічному дереву можна поставити у відповідність цілком визначену функцію $f(x_1, \dots, x_n)$ $k-$значної логіки від n змінних в аналітичному вигляді. Показано, що якщо $k-$значна функція $f(x_1, \dots, x_n)$ від n змінних задана таблично та визначена на всіх наборах, то можна побудувати відповідне повне логічне дерево, мінімізувати його і записати аналітичний вигляд даної функції, а за табличним представленням логічної функції знайти її аналітичне представлення у вигляді ДНФ або КНФ та побудувати логічне дерево.

2. Показано, що за підмножиною будь-яких значень набору змінних можна побудувати повне логічне дерево, яке представляє функцію розпізнавання, визначену на всіх наборах, що забезпечує можливість застосовувати логічні дерева в розпізнаванні образів, причому одержання тієї чи іншої логічної функції залежить від порядку проходження змінних у логічному дереві, що безпосередньо впливає на його кінцеву структуру, а принциповою задачею залишається питання мінімальної (оптимальної) структури ЛДК відносно початкової НВ.

3. Запропоновано простий метод мінімізації дискретних структур на основі концепції логічних дерев (процедури перестановки ярусів у його структурі), причому даний підхід характеризується простотою роботи за вказаною схемою на відповідній графічній структурі (конструкції логічного дерева), значною наочністю порівняно з іншими методами (є прямим або опосередкованим узагальненням двійкових (бінарних) методик для багатозначного випадку), можливістю нескладної програмної реалізації, що значно розширює сферу дій для великих n та k у випадку багатозначної логіки.

4. Розроблено метод оцінки впливу процедури перестановки ярусів структури регулярного логічного дерева на його складність для бінарного випадку ($k = 2$), причому ефект перестановки ярусів є значним – так дужкові

форми логічних функцій, які знайдені за довільним деревом та деревом з перестановками, можуть відрізнятися в $\frac{1}{3} * 2^{2^{m-1}+1}$ разів. Запропоновано важливий механізм мінімізації логічних дерев – процедуру перестановки ярусів (блоків) у структурі дерева, яка дає змогу досягти помітного ефекту зменшення складності логічного дерева, причому зі зростанням числа m та, відповідно, n (структурної складності логічного дерева) ефективність перестановки ярусів дуже швидко зростає.

РОЗДІЛ 5
МЕТОДИ ТА СХЕМИ ПОБУДОВИ ЛОГІЧНИХ ДЕРЕВ НА ОСНОВІ ПРОЦЕДУРИ ПЕРЕСТАНОВКИ ЯРУСІВ

5.1. Загальна схема побудови логічного дерева максимальної складності

На початку даного розділу введемо необхідні в подальшому для даного дослідження структур логічних дерев визначення.

Визначення 5.1. Найскладнішим логічним деревом (за конструкцією) будемо називати таке дерево, яке містить у своїй структурі максимальну кількість різних міток (вершин, функцій, атрибутів).

Визначення 5.2. Під ярусом логічного дерева в даному дослідженні будемо розуміти горизонтальний ряд вершин даного графу (логічного дерева) зі змінною одного індексу (для випадку регулярного логічного дерева). В літературі також зустрічаються терміни каскад, рівень, полоса (stripe).

5.1.1. Схема максимального логічного дерева

Далі для довільної функції $f(x_1, x_2, \dots, x_n)$ побудуємо логічне дерево максимальної структурної складності (рис. 5.1). Відмітимо, що функції $\alpha_1, \alpha_2, \dots, \alpha_k$ будуємо різними, відповідно функції $\beta_1, \beta_2, \dots, \beta_n$ будуємо також різними. Цей процес продовжуємо до тих пір, доки це можливо.

На наступному етапі розглянемо побудоване логічне дерево. Зауважимо, що яруси в даній структурі позначено числами $0,1,2, \dots, n-1$. Причому, номер 0 буде відноситися до першого ярусу, а $n-1$ номер – до ярусу n. Відмітимо, що на l −му ярусі знаходиться k^l вершин ($l = 0,1,2, \dots, n-1$).

Зрозуміло, що всі функції, які відповідають вершинам l −го ярусу, залежать від змінних $x_{l+1}, x_{l+2}, \dots, x_n$, причому загальна кількість цих аргументів дорівнює $n-l$. Кількість всіх функцій k–значної логіки, які залежать від $n-l$ аргументів, буде дорівнювати $k^{k^{n-l}}$. Причому принципово важливою буде залежність між k^l та $k^{k^{n-l}}$.

Відмітимо, що якщо $n \le k+1$, то $k^{k^{n-l}} \ge k^k$ та $k^l \le k^k$, $(l = 0,1, \dots, n-1)$. Отже, в цьому випадку $k^l \le k^{k^{n-l}}$ при всіх

$l = 0,1,\dots,n-1$. Тому можна зробити висновок, що при $n \le k+1$ в усіх вершинах $l-$ го ярусу логічного дерева можна розмістити різні функції. Відмітимо, що якщо $k^{k^{n-l}} < k^l$, то в усіх вершинах $l-$ го ярусу вже неможливо розмістити різні функції. Зрозуміло, що такі яруси знайдуться тоді, коли $n > k+1$. Дійсно, якщо $n > k+1$, то $k^{k^{n-(n-1)}} = k^k < k^{n-1}$. Отже, в цьому разі на $n-1$ ярусі логічного дерева вже неможливо в різних вершинах розмістити різні функції.

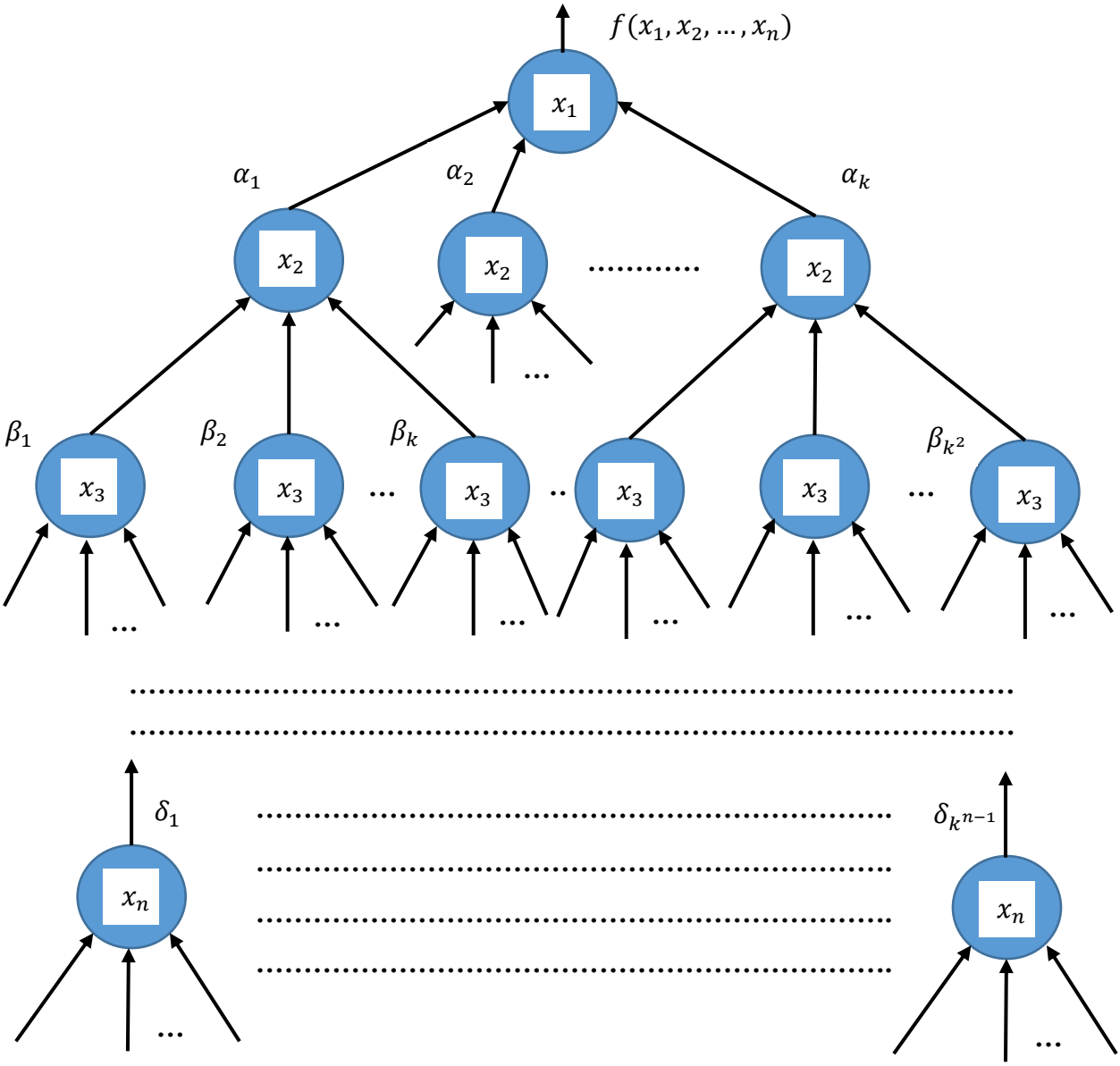


Рис. 5.1. Структура логічного дерева функції $f(x_1, x_2, \dots, x_n)$ максимальної складності.

Відмітимо, що якщо $n > k+1$, то $k^0 = 1 < k^{k^{n-0}} = k^{k^n}$, та відповідно $k^{n-1} > k^{k^{n-(n-1)}} = k^k$.

Отже, при $n > k+1$ завжди знайдеться таке число l_0, яке буде задовольняти наступні умови:

$$k^{l_0} \le k^{k^{n-l_0}};$$
$$k^{l_0} > k^{k^{n-(l_0+1)}}. \qquad (5.1)$$

Число l_0, яке задовольняє співвідношення (5.1), є єдиним. Ярус із номером l_0, який задовольняє співвідношення (5.1), будемо називати ярусом злому логічного дерева.

На наступному етапі дослідження знайдемо формулу, за допомогою якої можна буде визначити номер ярусу злому логічного дерева через k та n. Далі будемо вважати, що $k \ge 2$ та $n > k+1$.

Спочатку співвідношення (5.1) зведемо до еквівалентних йому:

$$n - z \le k^z;$$
$$n - (z-1) > k^{z-1}. \qquad (5.2)$$

Зокрема $z = n - l_0$; $(z = 1,2,\dots,n)$.

На наступному етапі покажемо, що для довільних натуральних чисел k та n знайдуться такі невід'ємні числа m, s та w, що $n = s * k^m + w$, де відповідно $1 \le s \le k-1;\ 0 \le w \le k^m$.

Розглянемо числа $k^j, (j = 0,1,2,\dots.)$. Оскільки $k^0 \le n$ та $k^N > n$ при достатньо великому числі n, то знайдеться таке m, що $k^m \le n < k^{m+1}$.

Нехай $n \ne s * k^m + w$, де $0 \le w < k^m$. З $k^m \le n < k^{m+1}$ слідує, що $1 \le s \le k-1$.

Далі покажемо, що представлення $n = s * k^m + w$, причому $(1 \le s \le k-1, 0 \le w \le k^m)$ при даному $k > 2$ є єдиним.

Нехай $s * k^m + w = s' * k^{m'} + w'$, де $1 \le s,\ s' \le k-1,\ 0 \le w,\ w' < k^m$. Покладемо, наприклад, що $m > m'$. Отже, будемо мати, що $m = m' + r$, де $r > 0$. Тоді одержуємо наступну ситуацію:

$s^{'} * k^{m^{'}} + w^{'} < s^{'} * k^{m^{'}} + k^{m^{'}} = (s^{'} + 1) * k^{m^{'}} \leq k^{m^{'}+1} < k^{m} < s * k^{m} + w.$

Отже, матимемо наступне: $s^{'} * k^{m^{'}} + w^{'} < s * k^{m} + w.$

Однак слід відмітити, що це протирічить першому припущенню, тобто $s^{'} * k^{m^{'}} + w^{'} = s * k^{m} + w$. Отже, будемо мати, що $m = m^{'}$.

Звідси слідує, що $w = w^{'}$ та $s = s^{'}$. На основі останнього робимо висновок, що існування єдиного представлення $n = s * k^{m} + w$, де відповідно $1 \leq s \leq k - 1$ та $0 \leq w < k^{m}$ доведено.

Оскільки умова (5.1) фактично була зведена до умови (5.2), то для знаходження числа l_0, необхідно знати таке z, яке задовільнило б умову (5.2).

1) На наступному етапі дослідження розглянемо спочатку випадок, коли $n = k^{m} + w$, тобто $s = 1$.

У цьому разі можливі такі три варіанти:

a) $0 < m < w$;

b) $m = w$;

c) $w < m < k^{m}$.

Спочатку розглянемо варіант (a), коли $0 < m < w < k^{m}$. Оцінимо наступні різниці:

$$n - (m + 1) = k^{m} + w - (m + 1) = k^{m} + (w - m) - 1 <$$
$$< k^{m} \leq 2 * k^{m} \leq k^{m+1};$$
$$n - (m + 1 - 1) = n - m = k^{m} + w - m = k^{m} + (w - m) > k^{m}.$$

Виходячи з цих двох різниць та умови (5.2), отримаємо що:

$$z_1 = m + 1. \quad (5.3)$$

Далі розглянемо варіант (b), коли $m = w$. Оцінимо наступні різниці:

$$n - m = k^{m} + w - m = k^{m};$$
$$n - (m - 1) = n - m + 1 = k^{m} + w - m + 1 > k^{m} > k^{m-1}.$$

Виходячи з цих двох різниць та враховуючи (5.2), отримаємо що:

$$z_2 = m. \quad (5.4)$$

Розглянемо варіант (c), коли $w < m < k^{m}$. Оцінимо наступні різниці:

$$n - m = k^{m} + w - m < k^{m};$$

$$n-(m-1)=k^m+w-(m-1)=k^{m-1}+w+(k-1)*k^{m-1}-$$
$$-(m-1)=k^{m-1}-(m-1)+(k-1)*k^{m-1}+w.$$

Оскільки $k^{m-1}-(m-1)\geq 0$ та $(k-1)\geq 2$, то будемо мати наступне:

$$n-(m-1)\geq (k-1)*k^{m-1}+w>k^{m-1}.$$

Провівши порівняння цих двох різниць та враховуючи (5.2), отримаємо наступне:

$$z_3=m \tag{5.5}$$

2) На другому етапі розглянемо випадок, коли $s\geq 2$. Тоді нехай $n=s*k^m+w,(s\geq 2,\ k\geq 2,\ w<k^m)$.

Оцінимо наступні різниці:

$$n-(m+1)=s*k^m+w-(m+1)=s*k^m+(w-m-1)<$$
$$<s*k^m+k^m=(s+1)*k^m\leq k^{m+1};$$

$$n-(m+1-1)=s*k^m+w-m=k^m+(w-m)+(s-1)*k^m.$$

Оскільки $k^m+(w-m)>0,(k^m-m>0,w\geq 0)$, то будемо мати наступне відношення:

$$n-(m+1-1)>(s-1)*k^m\geq k^m=k^{(m+1)-1}.$$

Далі, провівши процедуру порівняння цих двох різниць та враховуючи умову (5.2), можна зробити висновок, що:

$$z=m+1. \tag{5.6}$$

Але, оскільки $n=s*k^m+w$, тоді можна представити що $m\leq log_k n<m+1$, тобто будемо мати наступне:

$$log_k n=m+\xi,(0\leq\xi<1). \tag{5.7}$$

З виразів (5.3), (5.4), (5.5), (5.6) отримаємо наступне:

$$z=m+\beta,(0\leq\beta\leq 1). \tag{5.8}$$

З виразів (5.7), (5.8) також слідує наступне:

$$m=log_k n-\xi.$$

Тоді $z=log_k n+j,(-1<j\leq 1)$. Але $l_0=n-z$. Підставивши значення z у представлення l_0, будемо мати наступне:

$$l_0=n-(log_k n+j). \tag{5.9}$$

Таким способом, вище було представлено l_0 через n та k.

Оскільки l_0 – ціле число, то $l_0 = n - (log_k n + j)$, або можна записати, що $(n-1) - log_k n < l_0 < (n+1) - log_k n$.

На наступному етапі дослідження доведемо наступну лему.

5.1.2. Структурна складність логічного дерева

Лема 5.1. Нехай на l −му ярусі в деякого логічного дереві D в усіх вершинах стоять різні функції (мітки, атрибути). Тоді в усіх вершинах $0,1,2,\dots,l-1$ ярусів даного логічного дерева стоять різні функції.

Доведення. Спочатку покажемо, що на кожному j −му ярусі логічного дерева ($j = 0,1,\dots,l$) в різних вершинах стоять різні функції (мітки). Для цього достатньо показати, що якщо в l −му ярусі в усіх вершинах стоять різні функції, то в $l-1$ −му ярусі в усіх вершинах стоять також різні функції. Відмітимо, що останнє випливає з того очевидного факту, що:

$$f(x_1, x_2, \dots, x_n) = \overline{x_1} * f(0, x_2, \dots, x_n) \vee x_1 * f(1, x_2, \dots, x_n);$$

$$\varphi(x_1, x_2, \dots, x_n) = \overline{x_1} * \varphi(0, x_2, \dots, x_n) \vee x_1 * \varphi(1, x_2, \dots, x_n).$$

Крім того, обов‘язково має місце хоча б одне з наступних співвідношень:

a) $f(0, x_2, \dots, x_n) \neq \varphi(0, x_2, \dots, x_n)$;

b) $f(1, x_2, \dots, x_n) \neq \varphi(1, x_2, \dots, x_n)$.

Тоді відповідно будемо мати, що:

$$f(x_1, x_2, \dots, x_n) \neq \varphi(x_1, x_2, \dots, x_n).$$

На наступному етапі покажемо, що функції, які стоять у вершинах різних ярусів i та j ($i, j \in \{0,1,2,\dots,l\}$) будуть різні.

Тому припустимо, що $i > j$, тоді функція i −го ярусу не залежить від змінної x_{j+1}. Це означає, що лему доведено.

Далі нехай $D_{n,k}$ – довільне логічне дерево з n ярусами, яке відповідає деякій функції $k-$ значної логіки. Через $\left|D_{n,k}\right|$ позначимо кількість всіх різних міток (потужність), які стоять в вершинах логічного дерева $D_{n,k}$. Дерево $D_{n,k}^0$ називається найскладнішим серед логічних дерев $D_{n,k}$, якщо виконується умова $\left|D_{n,k}^0\right| > \left|D_{n,k}\right|$ для довільного дерева $D_{n,k}$.

Нехай n та k – деякі числа, де $n > k + 1$ та $n \geq 2$. Припустимо, що l_0 – це ярус злому логічного дерева, який визначається співвідношенням (5.1). Далі розташуємо в l_0 – ярусі в різних вершинах різні функції, які залежать від змінних $x_{l_0+1}, \ldots, x_n$. У результаті отримаємо деяке фінальне логічне дерево $D^*_{n,k}$. Доведемо наступну теорему.

5.1.3. Випадок найскладнішого логічного дерева

Теорема 5.1. Дерево структури $D^*_{n,k}$ є найскладнішим логічним деревом.

Доведення. Для доведення даної теореми спочатку розглянемо випадок, коли виконується рівність $k^{l_0} = k^{k^{n-l_0}}$. Через A позначимо множину всіх функцій логічного дерева $D^*_{n,k}$, які стоять у $0,1,2, \ldots, l_0 - 1$ ярусах даної структури, а через B – множину всіх функцій, які стоять у $l_0, l_0 + 1, \ldots, n - 1$ ярусах. У даному логічному дереві очевидно, що буде виконуватися рівність:

$$A \cap B = \emptyset. \tag{5.10}$$

На наступному етапі введемо наступні позначення: $|A|$ – кількість функцій множини A (потужність), $|B|$ – кількість функцій множини B, $|A \cup B| -$ кількість функцій множин A та B разом. На основі (5.10) можна записати наступне:

$$|A \cup B| = |A| + |B|. \tag{5.11}$$

Розглянемо довільне логічне дерево $D'_{n,k}$. Аналогічно введемо позначення: A' – множина функцій, які стоять в $1,2, \ldots, l_0 - 1$ ярусах дерева, та B' – множина функцій, які стоять в $l_0, l_0 + 1, \ldots, n - 1$ ярусах. Для дерева $D'_{n,k}$ можна записати наступне:

$$|A' \cup B'| \leq |A'| + |B'|. \tag{5.12}$$

Але очевидно, що:

$$|A'| \leq |\mathrm{A}| \quad \text{та} \quad |B'| \leq |B|. \tag{5.13}$$

Тоді, виходячи з (5.12) та (5.13), представимо наступне:

$$|A' \cup B'| \leq |A'| + |B'| \leq |A| + |B|. \tag{5.14}$$

З виразів (5.11) та (5.14) слідує наступне:

$$\left|A^{'} \cup B^{'}\right| \leq |A \cup B|.$$

Отже, вище було показано, що логічне дерево $D^{*}_{n,k}$ є найскладнішим деревом. Тоді можна зробити висновок, що теорему доведено для першого випадку.

На наступному етапі дослідження розглянемо другий випадок, коли виконується наступне:

1) $k^{l_0} < k^{k^{n-l_0}}$;

2) $k^{l_0+1} > k^{k^{n-(l_0+1)}}$.

У цьому разі можливі знову два варіанти:

a) у $l_0 + 1$ ярусі розташовані всі можливі різні функції, тоді доведення проводиться аналогічно доведенню першого випадку;

b) у $l_0 + 1$ ярусі розташована тільки певна частина всіх функцій (міток), а інші знаходяться на наступних ярусах.

Тоді розглянемо варіант (b). У цьому випадку можна змінювати логічне дерево так, щоб у $l_0 + 1$ ярусі стояли всі можливі різні функції.

Відомо, що в l_0 ярусі всі функції в вершинах різні. В $l_0 + 1$ ярусі маємо тільки частину різних функцій, причому кількість вершин тут більша ніж кількість функцій, тобто в деяких вершинах знаходяться однакові функції (є дублювання міток).

Нехай $\{\varphi_i, \dots, \varphi_r\}$ – набір функцій (міток), яких не вистачає і які розташовані в $l_0 + 2, \dots, n$ ярусах. В одну з вершин, яка містить однакові функції, помістимо функцію φ_i з $\{\varphi_i, \dots, \varphi_r\}$.

Тоді виникає питання: чи будуть міститися в l_0 ярусі вершини з однаковими функціями (тобто необхідно перевірити факт дублювання міток). Але для цього необхідно, щоб вершини $l_0 + 1$ ярусі, які є вхідними для деяких вершин l ярусу, містили відповідно різні функції.

Однак цього факту бути не може. Якщо в l_0 ярусі всі функції різні в вершинах логічного дерева, то очевидно, що вхідні вершини $l_0 + 1$ ярусу для кожної з вершин l_0 ярусу є відповідно різними. Та якщо в одну з вершин, які є

вхідними для деякої вершини l_0 ярусу, помістимо деяку функцію, відмінну від всіх у $l_0 + 1$ ярусі, то, очевидно, в l_0 ярусі всі функції в вершинах логічного дерева залишаться різними.

Отже, вище було показано, що підстановка функції φ_i замість іншої в ярус $l_0 + 1$ не змінила загальної складності логічного дерева. В $l_0 - 1$ ярусі кількість різних функцій збільшилась на одиницю. Відмітимо, що так само можна поступити з наступною функцією φ_{i+1} з початкового набору міток (функцій) $\{\varphi_i, \varphi_{i+1}, \dots, \varphi_r\}$.

Продовжимо цей процес до тих пір, доки не розмістимо всі $\{\varphi_i, \dots, \varphi_r\}$ функції в $l_0 + 1$ ярусі. Шляхом такої перестановки в структурі логічного дерева отримаємо всі можливі різні функції в $l_0 + 1$ ярусі, тобто прийдемо до варіанту (а) другого випадку.

Можна бачити, що подальше доведення теореми проводиться аналогічно доведенню першого випадку.

Наслідок. Розрахуємо тепер величину $\left|D^*_{n,k}\right|$. Для спрощення запису величину $\left|D^*_{n,k}\right|$ позначимо через $S_{n,k}$, тобто $S_{n,k}$ – загальна кількість різних функцій (міток) логічного дерева $D^*_{n,k}$.

1) На першому етапі розглянемо випадок $k^{l_0} = k^{k^{n-l_0}}$.

У цьому випадку $n = k^m$. Отже, $n - m = n - log_k n$.

Величина $S_{n,k}$ буде представлятися формулою:

$$S_{n,k} = 1 + k + k^2 + \cdots + k^{l_0} = \frac{k^{l_0+1} - 1}{k - 1}.$$

З представлення $l_0 = n - log_k n$ отримаємо, що:

$$S_{n,k} = \frac{k^{n-log_k n} - 1}{k - 1} = \frac{k^{n+1} * k^{-log_k n} - 1}{k - 1} = \frac{\frac{k^{n+1}}{n} - 1}{k - 1} =$$

$$= \frac{k^{n+1} - n}{n * (k - 1)} = \frac{k^{n+1}}{n * (k - 1)} - \frac{1}{k - 1}.$$

Звідси отримаємо наступне:

$$S_{n,k} \approx \frac{k^{n+1}}{n*(k-1)}. \qquad (5.15)$$

2) На наступному етапі розглянемо випадок, коли $k^{l_0} \neq k^{k^{n-l_0}}$.

У цьому разі будемо мати наступне:

$$1 + k + \cdots + k^{l_0} \leq S_{n,k} \leq 1 + k + \cdots + k^{l_0} + 1.$$

Звідси $\frac{k^{l_0+1}-1}{k-1} \leq S_{n,k} \leq \frac{k^{l_0+2}-1}{k-1}$. Отже, $S_{n,k} = \frac{k^{l_0+1+r}-1}{k-1}$, де відповідно $0 \leq r \leq 1$. Але відмітимо, що з іншого боку $l_0 = n - log_k n + j$, де $-1 \leq j < 1$. Отже, можна зробити висновок, що $S_{n,k} = \frac{k^{1+n-log_k n+\rho}-1}{k-1}$, де $-1 \leq \rho < 2$. Звідси слідує наступний результат:

$$S_{n,k} = \frac{\frac{k^{1+n+\rho}}{n}-1}{k-1} = \frac{k^{1+n+\rho}}{n*(k-1)} - \frac{1}{k-1}, \text{ де } -1 \leq \rho < 2.$$

Остаточно будемо мати, що:

$$S_{n,k} \approx \frac{k^{1+n+\rho}}{n*(k-1)}. \quad (5.16)$$

Зауважимо, що число $S_{n,k}$ – ціле.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) Найскладнішим логічним деревом буде таке дерево, яке містить у своїй структурі максимальну кількість різних міток (атрибутів, вершин, функцій).

2) Для довільного регулярного логічного дерева, якщо в l –му ярусі в усіх вершинах стоять різні функції (мітки), тоді в усіх вершинах $0,1,2, \ldots, l-1$ ярусів даного логічного дерева стоять різні функції.

3) Відмітимо, що загальна кількість різних міток (функцій) найскладнішого логічного дерева $D^*_{n,k}$ залежно від розташування ярусу злому логічного дерева буде дорівнювати величині $\frac{k^{n+1}}{n*(k-1)}$ або $\frac{k^{1+n+\rho}}{n*(k-1)}$.

5.2. Критерій оптимальності регулярного логічного дерева на основі поняття подібності

Нехай є деяке логічне дерево (довільної структури), яке представляє функцію $f(x_1, x_2, \dots, x_n)$ для $k-$ значної логіки від n змінних (наприклад логічне дерево з рис. 3.2). Далі визначимо кількість характеристичних функцій $L_{\text{дуж}}(f)$ у дужковій формі функції $f(x_1, x_2, \dots, x_n)$, яка отримана за допомогою даного логічного дерева.

Відмітимо, що переходи логічного дерева (ребра графа) відповідають характеристичним функціям у дужковій формі, але через те, що існують подібні вершини (перший підрозділ розділу 4) та тотожність $\varphi_i(x) * 0 = 0$, то кількість міток (атрибутів) у дереві буде менше ніж кількість ребер (переходів у структурі логічного дерева). Тоді будемо мати наступне:

$$L_{\text{дуж}}(f) = \frac{k^{n+1}-k}{k-1} - S_n.$$

Зауважимо, що $\frac{k^{n+1}-k}{k-1}-$ загальна кількість ребер логічного дерева функції $k-$ значної логіки від n змінних, величину S_n назвемо подібністю логічного дерева відповідної функції $f(x_1, x_2, \dots, x_n)$, де $S_n > 0$. Очевидно, що оптимальним є таке логічне дерево, для якого буде виконуватися $S_n = \max(S_{n_i})$.

Зауважимо, що величина подібності S_n може слугувати критерієм оптимальності логічного дерева (графа).

5.2.1. Методи знаходження подібності структур логічних дерев

На наступному етапі дослідження розглянемо два можливі підходи знаходження подібності логічного дерева S_n.

Метод (А) знаходження подібності S_n для логічного дерева.

Нехай маємо логічне дерево функції $f(x_1, x_2, \dots, x_n)$. При цьому можливі два випадки:

а) Верхня вершина (вершина першого ярусу) логічного дерева є подібною. В цьому разі логічне дерево можна представити

піддеревом – лівою гілкою початкового дерева. Нехай верхня вершина логічного дерева відповідає змінній x_i. Тоді має місце наступна формула:

$$L_{\text{дуж}}(f) = L_{\text{дуж}}\left(f_{\varphi_0}(x_i)\right) = \frac{k^n - k}{k-1} - S_{n-1}^{\varphi_0^i}. \quad (5.17.(a))$$

Зауважимо, що $S_{n-1}^{\varphi_0^i}$ – подібність піддерева, яке представляє ліву гілку (піддерево) початкового логічного дерева.

Тобто в цьому випадку будемо мати наступне:

$$S_n = k^n + S_{n-1}^{\varphi_0^i}. \quad (5.17.(b))$$

b) Верхня вершина логічного дерева (вершина першого ярусу) не є подібною, тоді будемо мати наступне:

$$L_{\text{дуж}}(f) = L_{\text{дуж}}\left(f_{\varphi_0}(x_i)\right) + L_{\text{дуж}}\left(f_{\varphi_1}(x_i)\right) + \cdots$$

$$\ldots + L_{\text{дуж}}\left(f_{\varphi_{k-1}}(x_i)\right) + k. \quad (5.17.(c))$$

Зауважимо, що формула (5.17.(c)) має місце в тому випадку, якщо на деякому ребрі логічного дерева є функція $\equiv 0$, тоді з (5.17.(a)) та (5.20), отримаємо, що $L_{\text{дуж}}(f \equiv 0) = -1$.

Відмітимо, що кількість міток кожного з піддерев початкового логічного дерева визначається наступним чином:

$$L_{\text{дуж}}\left(f_{\varphi_0}(x_i)\right) = \frac{k^n - k}{k-1} - S_{n-1}^{\varphi_0^i},$$

$$L_{\text{дуж}}\left(f_{\varphi_1}(x_i)\right) = \frac{k^n - k}{k-1} - S_{n-1}^{\varphi_1^i},$$

… … … … … … … … … … … … … … ….

$$L_{\text{дуж}}\left(f_{\varphi_{k-1}}(x_i)\right) = \frac{k^n - k}{k-1} - S_{n-1}^{\varphi_{k-1}^i}.$$

Зауважимо, що $S_{n-1}^{\varphi_0^i}, S_{n-1}^{\varphi_1^i}, \ldots, S_{n-1}^{\varphi_{k-1}^i}$ – подібність відповідних піддерев (підграфів) початкового логічного дерева.

Остаточно будемо мати наступне:

$$L_{\text{дуж}}(f) = \frac{k^n - k}{k-1} - S_{n-1}^{\varphi_0^i} + \frac{k^n - k}{k-1} - S_{n-1}^{\varphi_1^i} + \cdots + \frac{k^n - k}{k-1} - S_{n-1}^{\varphi_{k-1}^i} + k =$$

$$= \frac{k^n - k}{k-1} * k + k - S_{n-1}^{f_i} = \frac{k^{n+1} - k^2 + k^2 - k}{k-1} - S_{n-1}^{f_i} = \frac{k^{n+1} - k}{k-1} - S_{n-1}^{f_i}.$$

Отже, на виході отримаємо:

$$S_{n-1}^{f_i} = \sum_{j=1}^{k-1} S_{n-1}^{\varphi_j^i}. \quad (5.18)$$

З (5.18) отримаємо $S_n = S_{n-1}^{f_i}$.

Далі, об‘єднавши результати пунктів (a) та (b), можна представити наступне:

$$S_n = \begin{cases} k^n + S_{n-1}^{\varphi_0^i}, \text{ якщо верхня вершина дерева є подібною;} \\ S_{n-1}^{f_0^i}, \text{ якщо верхня вершина дерева не є подібною.} \end{cases} \quad (5.19)$$

На наступному етапі, розглядаючи піддерево, яке складається з однієї вершини нижнього ярусу дерева, приходимо до величини S_0 – подібності кінцевого ребра логічного дерева, причому:

$$S_0 = \begin{cases} 1, \text{ якщо кінцеве ребро відповідає нулю;} \\ 0, \text{ якщо кінцеве ребро не відповідає нулю.} \end{cases} \quad (5.20)$$

5.2.2. Повна подібність логічної структури

Метод (B) знаходження подібності S_n для логічного дерева.

Визначення 5.3. Сукупність комірок діаграми Венна, відповідає наборам змінних:

$$x_1^{\sigma_1}, x_2^{\sigma_2}, \dots, x_{i-1}^{\sigma_{i-1}}, 0, x_{i+1}^{\sigma_{i+1}}, \dots, x_n^{\sigma_n};$$
$$x_1^{\sigma_1}, x_2^{\sigma_2}, \dots, x_{i-1}^{\sigma_{i-1}}, 1, x_{i+1}^{\sigma_{i+1}}, \dots, x_n^{\sigma_n};$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$
$$x_1^{\sigma_1}, x_2^{\sigma_2}, \dots, x_{i-1}^{\sigma_{i-1}}, k-1, x_{i+1}^{\sigma_{i+1}}, \dots, x_n^{\sigma_n}.$$

Будемо називати її повною системою x_i – порівнюваних комірок.

Запис $x_l^{\sigma_l}$ означає, що змінна x_l на даному наборі приймає відповідне значення σ_l, $(\sigma_l \in \{0,1,\dots,k-1\})$.

Так, якщо в усіх комірках x_i – порівнюваної системи комірок записано одне і те саме число, тобто одне і те саме значення функції, то будемо говорити, що вони є x_i – подібними. Кількість x_i – подібних систем комірок

у таблиці Венна логічної функції $f(x_1, x_2, \dots, x_n)$ назвемо $x_i -$ подібністю функції. Позначимо цю величину через S_f^i.

Визначення 5.4. Повною подібністю деякої логічної функції будемо називати наступну величину:

$$S_f = k * \sum_{i=1}^{n} S_f^i. \tag{5.21}$$

Далі введемо для розгляду наступну величину:

$$R_i = k * \sum_i S_{f_{r_i}}^i. \tag{5.22}$$

Зауважимо, що індекс f_{r_i} означає, що береться вихідна функція $i -$ої вершини (тобто вершини, в якій записано x_i). Верхній індекс i означає, що для функції f_{r_i} є подібність тільки за x_i.

Відмітимо, що тут $f_{r_i} -$ вихідна функція $i -$ої подібної вершини. Сумування ведеться по всіх подібних вершинах, які не входять у ліві гілки особливих піддерев.

Уведемо ще одну величину:

$$R_j = k * \sum_j S_{f_{r_j}}^j. \tag{5.23}$$

Аналогічно $f_{r_i} -$ вихідна функція $j -$ої вершини, яка не є подібною і в яку записано j. Сумування ведеться по всім вершинам, які не є подібними.

Подібність на основі кінцевих ребер логічного дерева позначимо через Q. Іншими словами, $Q -$ це кількість нульових ребер у логічному дереві, які враховуються при підрахунку подібності за вказаним вище методом.

Лема 5.2. Повну подібність логічної функції $f(x_1, x_2, \dots, x_n)$ можна визначити з наступного співвідношення:

$$S_f = k * \sum_{l=1}^{n} \sum_{f_{r_l}} S_{f_{r_l}}^l. \tag{5.24}$$

Зауважимо, тут $f_{r_l} -$ вихідна функція $l -$ої вершини логічного дерева. Сумування ведеться в усіх вершинах логічного дерева (графа).

Доведення. Нехай у верхній вершині логічного дерева знаходиться деяка змінна x_{i_1}. Вихідна функція верхньої вершини логічного дерева співпадає з

самою функцією $f(x_1, x_2, \dots, x_n)$, тому $S_f^{i_1}$ функції та логічного дерева співпадають.

Нехай у лівій гілці другого ярусу логічного дерева знаходиться змінна x_{i_2} (у вершині відповідного піддерева). Тоді жоден із наборів, які відповідають вихідним функціям вершин x_{i_2}, розташованих в інших піддеревах початкового дерева, не може бути $x_{i_2}-$ порівнюваним із наборами лівого піддерева, оскільки вони відрізняються змінною x_{i_1}.

Аналогічно набори, які відповідають вихідним функціям різних вершин дерева, в яких записано x_{i_2}, відрізняються хоча б тією змінною, яка знаходиться у вершині, де зустрічаються дані вихідні функції. Тобто деяка вершина логічного дерева є вершиною, де сходяться вихідні функції двох інших вершин логічного дерева, якщо при руху вверх по ребрах від двох останніх вершин цих функцій першою їх спільною вершиною (через яку вони пройдуть) буде вищевказана вершина.

Отже, було показано, що набори, які відповідають вихідним функціям різних вершин даного логічного дерева не можуть бути $x_{i_2}-$ порівнюваними, а тому не можуть бути і $x_{i_2}-$ подібними. Отже, можна записати наступне:

$$S_f^{i_2} = \sum_{f_{r_{i_2}}} S_{f_{r_{i_2}}}^{i_2}.$$

Відмітимо, що тут сумування ведеться за всіма вихідними функціями вершин логічного дерева, в яких записано i_2.

Отже, на наступному етапі, міркуючи аналогічним чином для вершин логічного дерева, які відповідають довільній x_l, приходимо до затвердження леми:

$$S_f = k * \sum_{l=1}^{n} S_f^l = k * \sum_l \sum_{f_{r_l}} S_{f_{r_l}}^l.$$

Звідси робимо висновок, що лему доведено.

Теорема 5.2. Для довільного логічного дерева, яке представляє деяку фіксовану логічну функції $f(x_1, x_2, \dots, x_n)$ $k-$значної логіки маємо наступне:

$$S_n = S_f - R_i - R_j + Q. \qquad (5.25)$$

Доведення. Відмітимо спочатку, що вираз $R_i + R_j$ дорівнює сумі $x_i -$ подібності за всіма вихідними функціями всіх вершин логічного дерева, крім тих його вершин, які беруть участь у підрахунку подібності, тобто крім вершин, які входять у ліві гілки особливих піддерев (це слідує з визначення величин R_i та R_j).

Але подібність S_n дорівнює подібності за рахунок вершин плюс величина Q. Позначимо подібність за рахунок подібних вершин через S_n^1. Тоді будемо мати наступне:

$$S_n = S_n^1 + Q. \tag{5.26}$$

Але з іншого боку відомо:

$$S_n^1 + R_i + R_j = S_f. \tag{5.27}$$

Далі з (5.27) отримаємо наступне:

$$S_n^1 = S_f - R_i - R_j. \tag{5.28}$$

На останньому етапі, підставляючи (5.28) у (5.26), будемо мати наступне:

$$S_n = S_f - R_i - R_j + Q.$$

Отже, можна зробити висновок, що теорему доведено.

Таким чином, вище було представлено два методи визначення подібності функції S_n. При підрахунку подібності за допомогою логічного дерева раціонально використовувати перший спосіб, значення теореми 5.2 буде роз‘яснено в даному дослідженні пізніше.

Приклад 1. Для пояснення вищевикладених методик розглянемо приклад визначення подібності функції, яка задана у вигляді таблиці Венна (табл. 5.1).

Відмітимо, що при підрахунку подібності враховуються тільки ті подібні вершини логічного дерева, які не входять у праві гілки особливих піддерев та такі ж нулі. Тут використовувався перший спосіб (Метод (А) для логічного дерева) визначення подібності.

Зауважимо, що на рисунку нулі, які враховувались при підрахунку подібності, позначено нижніми рисками (підкреслюванням).

Визначимо кількість характеристичних функцій у дужковій формі логічної функції:

$$L_{\text{дуж}}(f) = \frac{k^{n+1} - k}{k-1} - S_n = \frac{3^5 - 3}{2} - 89 = 120 - 89 = 31.$$

Таблиця 5.1. Таблиця Венна функції $f(x_1, x_2, x_3, x_4)$ трьох-значної логіки прикладу 1.

	x_4^0	x_4^1	x_4^2	x_4^0	x_4^1	x_4^2	x_4^0	x_4^1	x_4^2	
x_1^0	0	0	1	2	2	2	0	0	0	x_2^0
	0	0	1	2	2	2	0	0	0	x_2^1
	0	0	1	2	2	2	0	0	0	x_2^2
x_1^1	1	1	1	0	0	1	2	2	2	x_2^0
	1	0	2	1	0	2	1	0	2	x_2^1
	0	0	2	0	0	0	1	1	0	x_2^2
x_1^2	2	2	2	0	0	0	2	1	0	x_2^0
	1	0	1	1	0	1	1	0	1	x_2^1
	0	2	1	0	2	1	0	2	1	x_2^2
	x_3^0			x_3^1			x_3^2			

Логічне дерево функції $f(x_1, x_2, x_3, x_4)$ показано на рис. 5.2. Відмітимо, що всі подібні вершини логічного дерева позначено верхньою рискою. Визначивши подібність даного логічного дерева, отримаємо $S_n = 89$.

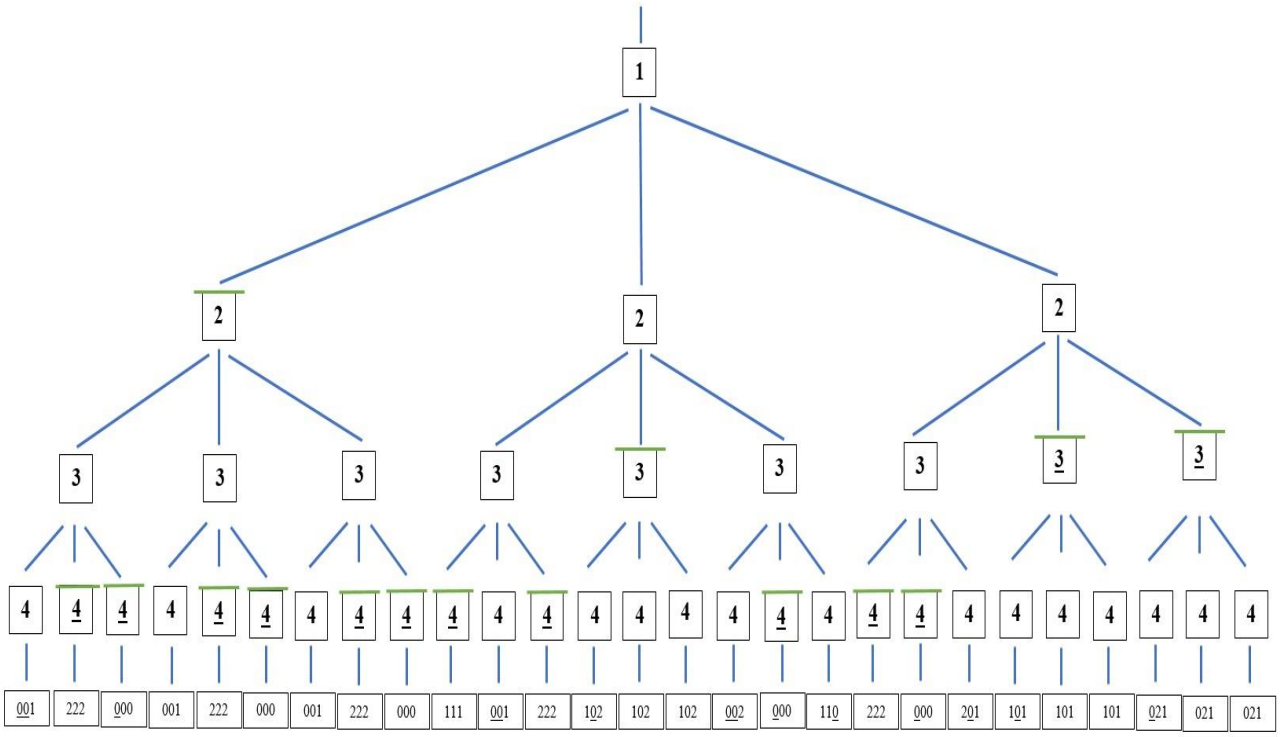


Рис. 5.2. Логічне дерево функції $f(x_1, x_2, x_3, x_4)$ прикладу 1.

Зважаючи на все вищезазначене, можна зафіксувати наступне:

1) Відмітимо, що кількість характеристичних функцій у дужковій формі $L_{\text{дуж}}(f)$ функції $f(x_1, x_2, \dots, x_n)$ можна визначити за допомогою формули $L_{\text{дуж}}(f) = \frac{k^{n+1}-k}{k-1} - S_n$.

2) Повною подібністю деякої логічної функції будемо називати величину $S_f = k * \sum_{i=1}^{n} S_f^i$.

3) Повну подібність логічної функції $f(x_1, x_2, \dots, x_n)$ можна визначити за допомогою співвідношення $S_f = k * \sum_{l=1}^{n} \sum_{f_{r_l}} S_{f_{r_l}}^l$.

4) Для довільного логічного дерева, яке представляє деяку фіксовану логічну функцію $f(x_1, x_2, \dots, x_n)$ $k-$ значної логіки, маємо, що $S_n = S_f - R_i - R_j + Q$.

5.3. Схеми вибору оптимального розташування змінних в структурі логічного дерева

Як було показано в даному дослідженні вище, складність дужкової форми логічної функції $f(x_1, x_2, \dots, x_n)$ $k-$значної логіки залежить від розташування змінних у вершинах відповідного логічного дерева. На даному етапі дослідження розглянемо методи, які здебільшого дають оптимальне логічне дерево (або близьке до оптимального відносно його структури).

Визначення 5.5. Логічне дерево (піддерево), у якого подібними є всі вершини нижнього ярусу (і тільки вони), називається простим логічним деревом (піддеревом).

Теорема 5.3. Якщо логічна функція $f(x_1, x_2, \dots, x_n)$ має тільки одну фіктивну змінну x_1, для довільного неособливого логічного дерева можна побудувати особливе дерево з тією ж подібністю, у верхній вершині якого розташовано змінну x_1.

Доведення. Нехай маємо неособливе логічне дерево (граф), яке представляє деяку логічну функцію $f(x_1, x_2, \dots, x_n)$. Далі побудуємо особливе логічне дерево цієї функції. У вершину верхнього ярусу дерева записуємо змінну x_1. Ліве піддерево цієї вершини є даним неособливим логічним деревом, в якому замість піддерев із вершиною x_1 стоять піддерева, які представляють їх ліві гілки.

Назвемо їх піддеревами типу A. Кожному піддереву, яке враховується при підрахунку подібності логічних дерева типу A з даного неособливого дерева в побудованому особливому дереві відповідає тотожне піддерево. Тоді можна зробити висновок, що подібність за рахунок піддерев типу A в неособливому та особливому піддеревах одинакова.

Позначимо її через S_A. У побудованому особливому логічному дереві всі вершини, крім верхньої та вершин із піддерев типу A, знаходятся на один ряд нижче відповідних вершин даного неособливого дерева. Піддерево, яке складається з таких вершин, назвемо піддеревом типу B. Подібність піддерева

типу B у неособливому піддереві позначимо через S_B. Тоді подібність піддерева типу B в особливому дереві буде дорівнювати $\frac{S_B}{k}$. У результаті будемо мати наступне:

$$S_H = S_A + S_B + S_{x_1};$$

$$S_0 = S_A + \frac{S_B}{k} + k^n.$$

Зокрема, $S_H -$ подібність неособливого дерева, $S_0 -$ подібність особливого дерева, $S_{x_1} -$ подібність за рахунок вершин із міткою (змінною) x_1.

На наступному етапі візьмемо в даному неособливому логічному дереві (графі) піддерево, яке складається з піддерев типу B та вершин з x_1. Вершини з x_1 є кінцевими, тобто їх гілки в піддерево, що розглядається, не входять. Аналогічно назвемо це піддерево піддеревом типу C, і для нього будемо мати наступну ситуацію:

$$S_C = S_B + S_{x_1}.$$

Відмітимо, що якщо піддерево типу C є простим, то для всякої некінцевої вершини всі його гілки дають однакову подібність, яка буде дорівнювати k^{l-1}, (де $l -$ номер ярусу вершини), а все піддерево має подібність, яка дорівнює:

$$S_{C\text{ просте}} = k^n.$$

Відмітимо, що якщо піддерево C не є простим, то кожна його некінцева подібна вершина збільшує подібність на k^l та зменшує на $k^{l-1} * (k-1)$ (на подібність усіх правих гілок цієї вершини) порівняно з простим піддеревом. Отже, подібність збільшується на S_B та зменшується на $\frac{S_B*(k-1)}{k}$ в порівнянні з простим піддеревом. Тобто одержуємо наступну ситуацію:

$$S_c = S_{c\text{ просте}} + S_B - \frac{S_B*(k-1)}{k} = k^n + \frac{S_B}{k}. \qquad (5.29)$$

Далі з (5.29) будемо мати, що:

$$S_H = S_A + S_B + S_{x_1} = S_A + S_C = S_A + k^n + \frac{S_B}{k}.$$

Зауважимо, що це буде за умови, якщо $S_H = S_0$, що і треба було довести.

Наслідок. Відмітимо, що оптимальне логічне дерево функції з фіктивною змінною можна побудувати, записавши в його верхню вершину номер фіктивної змінної, тобто оптимальне дерево функції з фіктивною змінною можна завжди зробити особливим логічним деревом.

На наступному етапі дослідження перейдемо безпосередньо до побудови оптимального логічного дерева. Розглянемо вираз (5.25).

$$S_n = S_f - R_i - R_j + Q.$$

Відмітимо, що величина S_f є постійною для даної функції. Для того, щоб отримати оптимальне логічне дерево, треба зменшити величини R_i та R_j, та збільшити величину Q.

У більшості випадків $R_j > R_i$ та Q, тому домогтися максимальної подібності можна, зменшуючи величину R_j.

5.3.1. Схеми розташування міток структур логічних дерев

Враховуючи це, можливі наступні алгоритми вибору змінних у вершинах логічного дерева.

Алгоритм (А). У верхню вершину логічного дерева записуємо змінну x_i з мінімальною величиною S_f^i. Якщо таких вершин декілька, то записується будь-яка з них. У вершині піддерев функцій $f_{\varphi_0}(x_i), f_{\varphi_1}(x_i), \dots, f_{\varphi_{k-1}}(x_i)$ записуються відповідно номера змінних із мінімальними величинами $S_{f_{\varphi_l}}^j(x_i)$, де $l \in \{0,1,2, \dots, k-1\}$, $j \in \{1,2, \dots, n\}$.

Відмітимо, що якщо мінімальне значення цих величин однакове для декількох змінних, то записується будь-яка з них. Аналогічним чином знаходяться номера змінних для всіх інших вершин даного логічного дерева. В результаті застосування даного алгоритму отримується нерегулярне логічне дерево (граф), де всі його особливі піддерева складаються з подібних вершин.

Алгоритм (В). Якщо функція $f(x_1, x_2, \dots, x_n)$ має фіктивну змінну, то фіктивна змінна розміщується у верхній вершині відповідного логічного дерева. Якщо функція фіктивних змінних не має, то у верхню вершину

логічного дерева розміщуємо змінну з мінімальною величиною S_f^i. У вершини піддерев функцій $f_{\varphi_0}(x_i), f_{\varphi_1}(x_i), \dots, f_{\varphi_{k-1}}(x_i)$ записуються номера змінних, фіктивних для даних функцій, а якщо таких немає – змінна x_l з мінімальною величиною $S_{f_{\varphi_l}}^j(x_i)$, де відповідно $l \in \{0,1,2,\dots,n\}$, $j \in \{0,1,2,\dots,k-1\}$.

Відмітимо, що якщо змінних, які задовольняють вказаним вище умовам декілька, то у відповідну вершину логічного дерева записується довільна з них. Аналогічним чином знаходяться змінні для всіх інших вершин. Зауважимо, що змінна x_l є фіктивною для функції f_{r_i}, якщо $S_{f_{r_i}}^l = k^{n-t-1}$, де $t-$ кількість характеристичних функцій у функції r_i. Використовуючи це зауваження, легко можна визначити, яку змінну помістити у вершину дерева. Відмітимо, що, враховуючи наслідок із теореми 5.3, алгоритм (B) буде давати кращій результат ніж представлений алгоритм (A).

5.3.2. Приклади побудови оптимальних логічних дерев

На наступному етапі розглянемо приклади побудови оптимальних логічних дерев, використовуючи запропоновані вище алгоритми.

Приклад 2. Нехай маємо логічну функцію $f(x_1, x_2, x_3, x_4)$ трьох-значної логіки чотирьох змінних, яка задана таблицею Венна (табл. 5.2).

Таблиця 5.2. Таблиця Венна функції $f(x_1, x_2, x_3, x_4)$ трьох-значної логіки прикладу 2.

	x_3^0			x_3^1			x_3^2			
x_1^0	1	2	0	0	2	1	1	0	0	x_2^0
	1	0	0	1	0	0	1	0	0	x_2^1
	1	1	2	1	0	1	1	1	2	x_2^2
x_1^1	1	2	0	0	2	1	1	0	0	x_2^0
	2	1	2	2	1	2	2	1	2	x_2^1
	1	1	2	1	0	1	1	2	2	x_2^2
x_1^2	1	2	0	0	2	1	1	0	0	x_2^0
	0	2	1	0	2	1	0	2	1	x_2^1
	1	1	2	1	0	1	2	1	2	x_2^2
	x_4^0	x_4^1	x_4^2	x_4^0	x_4^1	x_4^2	x_4^0	x_4^1	x_4^2	

Підрахуємо x_i- подібність:

$$S_f^1 = 16;\ S_f^2 = 3;\ S_f^3 = 11;\ S_f^4 = 0.$$

У верхній вершині логічного дерева помістимо x_4, оскільки $S_f^4 < S_f^i$, причому $(i = 1,2,3)$. На наступному етапі побудуємо таблицю подібності (табл. 5.3).

Таблиця 5.3. Таблиця подібності для $f(x_1, x_2, x_3, x_4)$.

$\boldsymbol{i\backslash r}$	$\boldsymbol{x_4^0}$	$\boldsymbol{x_4^1}$	$\boldsymbol{x_4^2}$	$\boldsymbol{x_1^0x_2^0}$	$\boldsymbol{x_4^0x_2^1}$	$\boldsymbol{x_4^0x_2^2}$	$\boldsymbol{x_4^1x_2^0}$	$\boldsymbol{x_4^1x_2^1}$	$\boldsymbol{x_4^1x_2^2}$	$\boldsymbol{x_4^2x_2^0}$	$\boldsymbol{x_4^2x_2^1}$	$\boldsymbol{x_4^2x_2^2}$
1	5	5	6	3	0	2	3	0	2	3	0	3
2	2	0	1	-	-	-	-	-	-	-	-	-
3	5	3	3	0	3	2	0	3	0	0	3	0

Підрахуємо подібність логічного дерева S_n:

$$S_n = 3^2 * 7 + 3 * 4 + 8 = 83;$$

$$L_{\text{дуж}}(f) = \frac{3^5 - 3}{2} - 83 = 37.$$

Отже, можна зробити висновок, що дужкова форма функції, яка отримана за допомогою оптимального логічного дерева, буде містити 37 характеристичних функцій.

На наступному етапі побудуємо її:

$$f(x_1, x_2, x_3, x_4) = \varphi_0(x_4) * \gamma_1 \vee \varphi_1(x_4) * \gamma_2 \vee \varphi_2(x_4) * \gamma_3 =$$
$$= \varphi_0(x_4) * (\varphi_0(x_2) * \alpha_1 \vee \varphi_1(x_2) * \alpha_2 \vee \varphi_2(x_2) * \beta_1) \vee$$
$$\vee \varphi_1(x_4) * (\varphi_0(x_2) * \alpha_2 \vee \varphi_1(x_2) * \alpha_5 \vee \varphi_2(x_2) * \beta_2) \vee$$
$$\vee \varphi_2(x_4) * (\varphi_0(x_2) * \alpha_7 \vee \varphi_1(x_2) * \alpha_8 \vee \varphi_2(x_2) * \alpha_9) =$$
$$= \varphi_0(x_n) * (\varphi_0(x_2) * (1 * \varphi_0(x_3) \vee 1 * \varphi_2(x_3)) \vee 1 * \varphi_1(x_3) \wedge$$
$$\wedge \left(1 * \varphi_0(x_3) \vee \varphi_1(x_3)\right) \vee \varphi_2(x_2) * (1 * \varphi_0(x_1) \vee 1 * \varphi_1(x_1) \vee$$
$$\vee \varphi_2(x_1) * (1 * \varphi_0(x_3) \vee 1 * \varphi_1(x_3) \vee \varphi_2(x_3)))) \vee$$
$$\vee \varphi_1(x_4) * (\varphi_0(x_2) * (\varphi_0(x_3) \vee \varphi_1(x_3)) \vee$$
$$\vee \varphi_1(x_2) * \left(1 * \varphi_1(x_1) \vee \varphi_2(x_1)\right) \vee \varphi_2(x_2) * (1 * \varphi_0(x_3) \vee$$
$$\vee \varphi_2(x_3) * (1 * \varphi_0(x_1) \vee \varphi_1(x_1) \vee 1 * \varphi_2(x_1)))) \vee$$
$$\vee \varphi_2(x_4) * (1 * \varphi_0(x_2) * \varphi_1(x_3) \vee \varphi_1(x_2) * (\varphi_1(x_1) \vee 1 * \varphi_2(x_1)) \vee$$
$$\vee \varphi_2(x_2) * (\varphi_0(x_3) \vee 1 * \varphi_1(x_3) \vee \varphi_2(x_3))).$$

Так для того, щоб продемонструвати ефективність побудови оптимального логічного дерева, можна побудувати оптимальне дерево з природним розташуванням змінних за ярусами логічного дерева, а, підрахувавши подібність S_n для даного регулярного логічного дерева, отримаємо наступне:

$$S_n = 3^2 * 3 + 8 = 35;$$

$$L_{\text{дуж}}(f) = \frac{3^5-3}{2} - 35 = 85.$$

Так, дужкова форма логічної функції, яка отримана за допомогою даного регулярного логічного дерева, буде містити 85 характеристичних функцій.

Мінімальна ДНФ для даної функції буде мати наступний вигляд:

$$f(x_1, x_2, x_3, x_4) = 1 * \varphi_1(x_1) * \varphi_1(x_2) \vee 1 * \varphi_2(x_2) * \varphi_0(x_3) \vee$$
$$\vee 1 * \varphi_0(x_2) * \varphi_0(x_3) * \varphi_0(x_4) \vee 1 * \varphi_0(x_1) * \varphi_1(x_2) * \varphi_0(x_4) \vee$$
$$\vee 1 * \varphi_1(x_1) * \varphi_0(x_3) * \varphi_1(x_4) \vee 1 * \varphi_0(x_2) * \varphi_1(x_3) * \varphi_2(x_4) \vee$$
$$\vee 1 * \varphi_0(x_2) * \varphi_1(x_3) * \varphi_2(x_4) \vee 1 * \varphi_0(x_2) * \varphi_2(x_3) * \varphi_0(x_4) \vee$$
$$\vee 1 * \varphi_0(x_1) * \varphi_2(x_2) * \varphi_0(x_3) \vee 1 * \varphi_1(x_1) * \varphi_2(x_2) * \varphi_0(x_3) \vee$$
$$\vee 1 * \varphi_2(x_2) * \varphi_1(x_3) * \varphi_2(x_4) \vee 1 * \varphi_2(x_2) * \varphi_2(x_3) * \varphi_1(x_4) \vee$$
$$\vee 1 * \varphi_2(x_1) * \varphi_1(x_2) * \varphi_2(x_4) \vee \varphi_0(x_2) * \varphi_2(x_3) * \varphi_1(x_4) \vee$$
$$\vee \varphi_1(x_1) * \varphi_1(x_2) * \varphi_0(x_4) \vee \varphi_1(x_1) * \varphi_1(x_2) * \varphi_2(x_4) \vee$$
$$\vee \varphi_2(x_2) * \varphi_0(x_3) * \varphi_2(x_4) \vee \varphi_2(x_2) * \varphi_2(x_3) * \varphi_2(x_4) \vee$$
$$\vee \varphi_2(x_1) * \varphi_2(x_2) * \varphi_1(x_4) \vee \varphi_0(x_1) * \varphi_0(x_2) * \varphi_0(x_3) * \varphi_0(x_4) \vee$$
$$\vee \varphi_1(x_1) * \varphi_0(x_2) \wedge \varphi_1(x_3) * \varphi_1(x_4) \vee \varphi_1(x_1) * \varphi_2(x_2) * \varphi_2(x_3) * \varphi_1(x_4) \vee$$
$$\vee \varphi_2(x_1) * \varphi_0(x_2) * \varphi_1(x_3) * \varphi_1(x_4) \vee \varphi_2(x_1) * \varphi_2(x_2) * \varphi_1(x_3) * \varphi_0(x_4) \vee$$
$$\vee \varphi_2(x_1) * \varphi_2(x_2) * \varphi_2(x_2) * \varphi_2(x_3) * \varphi_0(x_4).$$

Відмітимо, що мінімальна ДНФ містить 76 характеристичних логічних функцій, тобто в два рази більше ніж дужкова форма, яка отримана за допомогою оптимального логічного дерева, але зауважимо, що на 9 характеристичних функцій менше ніж дужкова форма, яка отримана за допомогою регулярного логічного дерева, в якому змінні розміщені за ярусами дерева в природньому порядку.

Це показує, що найпростіше представлення функцій $k-$ значної логіки отримано саме за допомогою оптимального (мінімальної форми) логічного дерева. Слід зауважити, що побудувати оптимальне логічне дерево набагато простіше процедури знаходження мінімальної ДНФ, особливо при великій кількості змінних.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступне:

1) Під простим логічним деревом будемо розуміти структуру, в якій подібними є всі вершини нижнього ярусу і тільки вони.

2) Якщо деяка логічна функція $f(x_1, x_2, \dots, x_n)$ має тільки одну фіктивну змінну x_1, для довільного неособливого логічного дерева можна побудувати особливе логічне дерево з тією ж величиною подібності, у верхній вершині якого розташовано змінну x_1.

3) Оптимальне логічне дерево функції з фіктивною змінною можна побудувати, записавши в його верхню вершину номер фіктивної змінної, тобто оптимальне дерево функції з фіктивною змінною можна завжди зробити особливим логічним деревом, а, отже, забезпечити можливість його ефективної мінімізації.

5.4. Питання мінімізації функцій за допомогою регулярних логічних дерев

Слід відмітити, що, проводячи процедуру мінімізації логічних дерев та функцій $k-$ значної логіки, які вони представляють, за допомогою алгоритмів, які було приведено в попередньому підрозділі, враховувалося тільки одне співвідношення склеювання, а саме:

$$\mathrm{C}A\varphi_0(x) \vee \mathrm{C}A\varphi_1(x) \vee \ldots \vee \mathrm{C}A\varphi_{k-1}(x) = CA. \qquad (5.30)$$

Слід зауважити, що в багатьох випадках можна домогтися значно більшої ефективності мінімізації, якщо використовувати співвідношення наступного вигляду:

$$C_1A\varphi_0(x) \vee C_1A\varphi_1(x) \vee \ldots \vee C_1A\varphi_l(x) \vee C_2B\varphi_{l+1}(x) \vee C_3D\varphi_{l+2}(x) \vee \ldots$$
$$\ldots \vee C_m\varphi_{k-1}(x)N = C_1A \vee C_2B\varphi_{l+1}(x) \vee C_3D\varphi_{l+2}(x) \vee \ldots$$
$$\ldots \vee C_mN\varphi_{k-1}(x). \qquad (5.31)$$

Зокрема, $\mathrm{C}_j \in \{1,2,\ldots,k-1\}-$ константи, причому маємо, що $C_2,\ldots,C_m \geq C_1$, $l = 0,1,\ldots.,k-1$, $A \cap B = B, A \cap D = D,\ldots\ldots, A \cap N = N$.

Також, тут $A,B,\ldots,N-$ деякі елементарні добутки.

Використання співвідношень вигляду (5.31) дає можливість здебільшого досягти додаткових спрощень дужкових виразів логічних функцій, що безпосередньо впливає на структуру логічного дерева. На практиці ці співвідношення реалізуються наступним чином.

На першому етапі нехай маємо піддерева функції n змінних, які складаються в свою чергу з однієї вершини нижнього ярусу логічного дерева та ребер, які в нього входять. Нехай ці ребра відповідають значенням функції, причому жодне з яких не дорівнює нулю.

Тоді найменше значення функції буде входити в мітку вихідного ребра даної вершини без множення на відповідну характеристичну функцію у вигляді диз‘юнктивного члена. Якщо таких найменших значень є декілька, то жодне з них не множиться на відповідну характеристичну функцію.

Приклад 3. Розглянемо реалізацію даного правила на прикладі піддерева деякої функції чотирьох-значної логіки, яке складається з однієї вершини нижнього ряду.

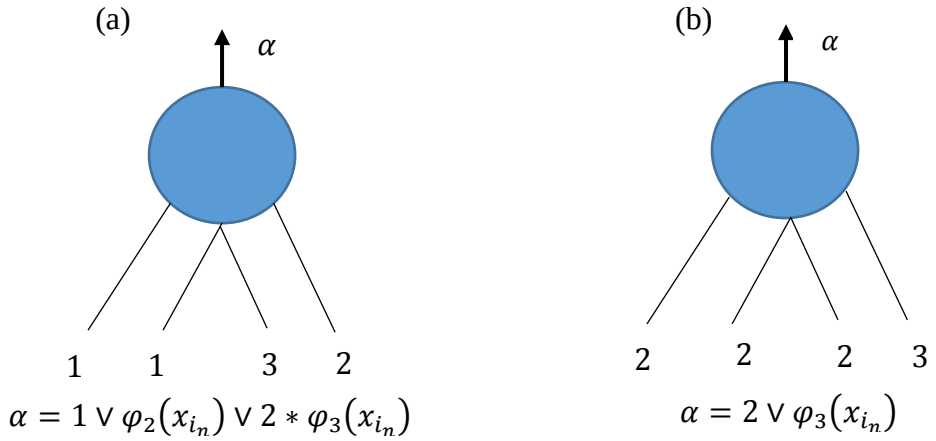


Рис. 5.3. (a,b) Піддерево функції чотирьох-значної логіки прикладу 3.

З рис. 5.3 можна бачити, що використання співвідношень вигляду (5.31) розширює можливість мінімізації логічного дерева.

Слід зауважити, що мітки, які відповідають вихідним функціям вершин нижнього ярусу, будуть функціями однієї змінної, тобто на першому етапі проходить мінімізація одномісних функцій.

Якщо під $A, B, \dots, N$ у (5.31) розуміти довільні функції, то співвідношення (5.31) буде мати місце в тому випадку, якщо функції $B, D, \dots, N$ містять в собі функцію A.

Під виразом «функція B містить у собі функцію A» будемо розуміти, що значення функції B на всіх наборах не менше значень функції A на відповідних наборах (тобто функція B буде поглинати функцію A).

На наступному етапі дослідження розглянемо декілька прикладів використання співвідношень вигляду (5.31).

Приклад 4.

$\alpha_1 = \boxed{1} \vee 2 * \varphi_0(x_{i_1}) \vee \varphi_1(x_{i_1});$

$\alpha_2 = \boxed{1} \vee \varphi_0(x_{i_1}) \vee 2 * \varphi_1(x_{i_1});$

$\alpha_3 = \boxed{1} \vee \varphi_1(x_{i_1});$

$$\alpha_4 = \boxed{1 * \varphi_2(x_{i_1}) \vee 1 * \varphi_3(x_{i_1})};$$

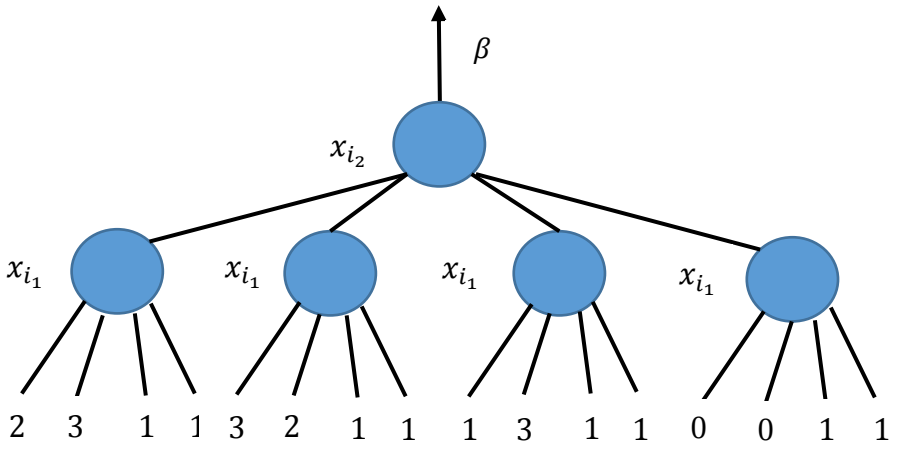


Рис. 5.4. Дерево функції прикладу 4.

Оскільки вирази в мітках $\alpha_1, \alpha_2, \alpha_3$, які взято в рамку, містять у собі вирази, які також взяті і в рамку в α_4 (тобто вони містять у собі α_4), тоді α_4 не будемо множити на характеристичну функцію $\varphi_3(x_{i_2})$ а, отже, отримаємо у виразі для β ще на одну характеристичну функцію менше.

$$\beta = \varphi_0(x_{i_2}) * \alpha_1 \vee \varphi_1(x_{i_2}) * \alpha_2 \vee \varphi_2(x_{i_2}) * \alpha_3 \vee \varphi_3(x_{i_2}) * \alpha_4 =$$

$$= \varphi_0(x_{i_2}) * \left(1 \vee 2 * \varphi_0(x_{i_1}) \vee \varphi_1(x_{i_1})\right) \vee \varphi_1(x_{i_2}) * (1 \vee \varphi_0(x_{i_1}) \vee$$

$$\vee 2 * \varphi_1(x_{i_1})) \vee \varphi_2(x_{i_2}) * (1 \vee \varphi_1(x_{i_1})) \vee 1.$$

Відмітимо, що без застосування співвідношення вигляду (5.31) будемо мати наступне:

$$L_{\text{дуж}}(f) = \frac{4^3 - 4}{3} - 2 = 18.$$

Тобто маємо ситуацію, що в дужковому представленні для функції β отримали 18 характеристичних функцій, в той час як в даному прикладі всього 10 характеристичних функцій.

Приклад 5. Нехай функцію $f(x_1, x_2, x_3)$ трьох-значної логіки, яка залежить від трьох змінних, задано логічним деревом (рис. 5.5) (відмітимо, що ярус з колом - x_1, ярус квадратами - x_2, ярус з трикутниками - x_3). Тоді:

$$\alpha_1 = 1 * \varphi_1(x_3) \vee 1 * \varphi_2(x_3);$$

$$\alpha_2 = 1 * \varphi_1(x_3) \vee \varphi_2(x_3);$$

$$\beta_1 = 1 * \varphi_1(x_3) \vee 1 * \varphi_2(x_3) \vee \varphi_1(x_2) * \big(1 \vee \varphi_1(x_3) \vee \varphi_2(x_3)\big) \vee$$

$$\vee \varphi_2(x_2) * \big(1 \vee \varphi_1(x_3) \vee \varphi_2(x_3)\big);$$

$$\beta_2 = \varphi_1(x_2) * \big(1 \vee \varphi_1(x_3) \vee \varphi_2(x_3)\big) \vee \varphi_2(x_2) * \big(1 \vee \varphi_1(x_3) \vee \varphi_2(x_3)\big);$$

$$\gamma = (1 \vee \varphi_1(x_3) \vee \varphi_2(x_3)) * (\varphi_1(x_1) \vee \varphi_1(x_2) \vee \varphi_1(x_1)) \vee$$

$$\vee \varphi_0(x_1) * \big(1 * \varphi_1(x_3) \vee 1 * \varphi_2(x_3)\big) = f(x_1, x_2, x_3).$$

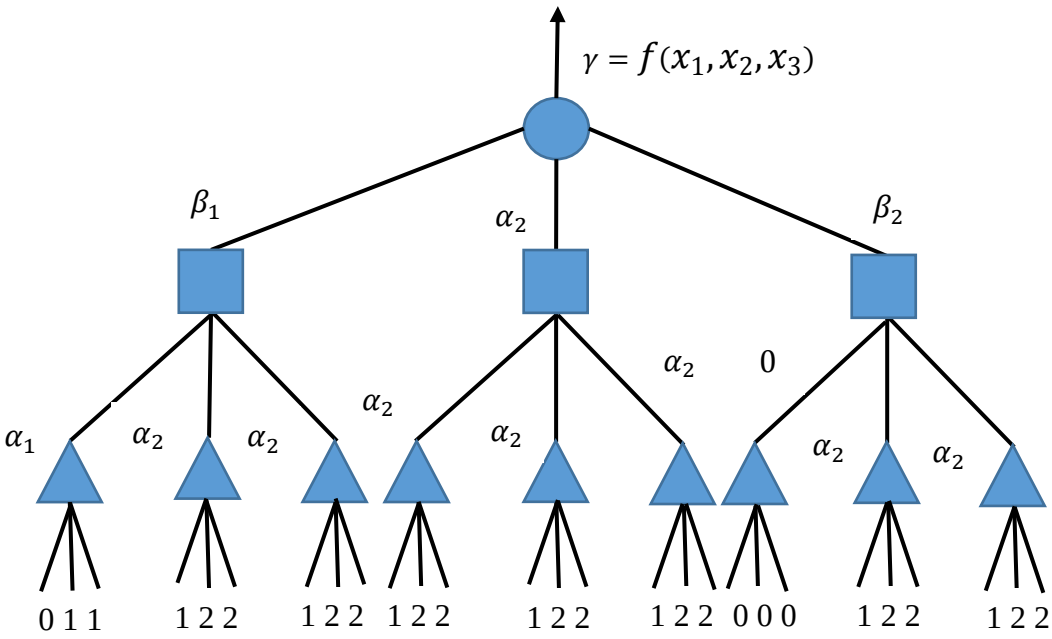


Рис. 5.5. Дерево функції трьох-значної логіки $f(x_1, x_2, x_3)$ прикладу 5.

Відмітимо, що дана дужкова форма містить 8 характеристичних функцій, причому без застосування співвідношення (5.31) маємо наступну ситуацію:

$$L_{\text{дуж}}(f) = \frac{3^4 - 3}{2} - 3^2 - 4 = 26.$$

Тобто дужкова форма містить фактично 26 характеристичних функцій.

Зрозуміло, що приведене вище логічне дерево відповідної функції $f(x_1, x_2, x_3)$ – рис. 5.5 – неможливо вважати оптимальним, але така задача і не ставилася, в свою чергу цю форму можна спростити з використанням співвідношення (5.31).

Враховуючи умову, яка була наведена після прикладу 1, застосуємо співвідношення (5.31) для подальшого спрощення логічної функції. Тоді будемо мати наступну ситуацію:

$$\alpha_1 = 1 * \varphi_0(x_3) \vee 1 * \varphi_2(x_3);\ \ \alpha_2 = 1 * \varphi_0(x_1) \vee \varphi_1(x_1);$$

$$\alpha_3 = 1 \vee \varphi_2(x_3);\ \alpha_4 = \varphi_0(x_3) \vee \varphi_1(x_3);$$

$$\alpha_5 = 1 * \varphi_1(x_1) \vee \varphi_2(x_1);\ \ \alpha_6 = 1 * \varphi_1(x_1);$$

$$\alpha_7 = 1 * \varphi_1(x_3);\ \ \alpha_8 = \varphi_1(x_1) \vee 1 * \varphi_2(x_1);$$

$$\alpha_9 = 1 * \varphi_0(x_3) \vee \varphi_2(x_3);$$

$$\beta_1 = 1 \vee \varphi_2(x_3) * \varphi_2(x_1);\ \ \beta_2 = 1 \vee \varphi_0(x_3) \vee \varphi_2(x_3) * \big(1 \vee \varphi_1(x_1)\big);$$

$$\gamma_1 = \varphi_0(x_2) * \alpha_1 \vee \varphi_1(x_2) * \alpha_2 \vee \varphi_2(x_2) * \beta_1;$$

$$\gamma_2 = \varphi_0(x_2) * \alpha_4 \vee \varphi_1(x_2) * \alpha_5 \vee \varphi_2(x_2) * \beta_2;$$

$$\gamma_3 = \varphi_0(x_2) * \alpha_7 \vee \varphi_1(x_2) * \alpha_8 \vee \varphi_2(x_2) * \alpha_9;$$

$$w = \varphi_0(x_4) * \gamma_1 \vee \varphi_1(x_4) * \gamma_2 \vee \varphi_2(x_4) * \gamma_3 = \varphi_0(x_4) * (\varphi_0(x_2) * (1 *$$

$$* \varphi_0(x_3) \vee 1 * \varphi_2(x_3)) \vee \varphi_1(x_2) * (1 * \varphi_0(x_1) \vee \varphi_1(x_1)) \vee \varphi_2(x_2) *$$

$$* (1 \vee \varphi_2(x_3) * \varphi_2(x_1))) \vee \varphi_1(x_4) * (\varphi_0(x_2) * (\varphi_0(x_3) \vee \varphi_1(x_3)) \vee$$

$$\vee \varphi_1(x_2) * \big(1 * \varphi_1(x_1) \vee \varphi_2(x_2)\big) \vee \varphi_2(x_2) * (1 * \varphi_0(x_3) \vee \varphi_2(x_3) *$$

$$* (1 \vee \varphi_1(x_1)))) \vee \varphi_2(x_4) * (\varphi_0(x_2) * (1 * \varphi_1(x_3)) \vee \varphi_1(x_2) *$$

$$* \big(\varphi_1(x_1) \vee 1 * \varphi_2(x_1)\big) \vee \varphi_2(x_2) * (1 \vee \varphi_0(x_3) \vee \varphi_2(x_3))).$$

Дана дужкова форма логічної функції $f(x_1, x_2, x_3, x_4)$ містить 30 характеристичних функцій, тобто на 7 характеристичних функцій менше ніж дужкова форма, яку отримали без застосування співвідношення (5.31), та в два з половиною рази менше мінімальної ДНФ.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) Використання співвідношень вигляду (5.31) дає можливість у більшості випадків досягти додаткових спрощень дужкових виразів логічних функцій, що безпосередньо впливає на складність логічного дерева.

2) Наведені вище в цьому підрозділі приклади показують, що для отримання простих дужкових виразів функцій $k-$ значної логіки необхідно

побудувати мінімальне логічної дерево, а після цього – мінімізувати функцію, застосовуючи співвідношення вигляду (5.30) та (5.31).

3) Можна привести й інші співвідношення, специфічні для багатозначної логіки, які дозволяють проводити подальше спрощення дужкової форми довільної функції $f(x_1, x_2, \dots, x_n)$, яка отримана за допомогою концепції логічного дерева.

Висновки до розділу 5

1. У розділі досліджено питання побудови найскладнішого логічного дерева (на основі критерію структурної складності), яке містить у своїй конструкції максимальну кількість різних міток (атрибутів, вершин, функцій). Показано, що для довільного регулярного дерева, якщо на фіксованому ярусі в усіх вершинах стоять різні функції (мітки), тоді в усіх вершинах ярусів, які розташовані вище в структурі даного логічного дерева, також стоять різні функції (мітки). Дано загальну числову оцінку (структурної складності) кількості різних міток (функцій, атрибутів) найскладнішого логічного дерева залежно від розташування ярусу злому в його структурі.

2. Розроблено прості методи знаходження величини подібності для конструкції логічних дерев у задачах мінімізації їх структур, та на основі цього досліджено питання критерію оптимальності регулярного логічного дерева. Розв‘язок даного питання має принципову важливість з точки зору як пошуку ефективних перестановок у процедурі мінімізації структур регулярних логічних дерев, так і оцінки загального ефекту результату такої оптимізації логічних дерев.

3. Запропоновано ефективні схеми оптимального розташування змінних у структурі логічного дерева, які здебільшого дають оптимальне логічне дерево (або близьке до оптимального відносно його структури). Показано, що оптимальне логічне дерево (деякої функції з фіктивною змінною) можна побудувати шляхом запису в його верхню вершину номера фіктивної змінної – тобто оптимальне дерево функції з фіктивною змінною можна завжди зробити особливим логічним деревом, що в перспективі дозволяє провести процедуру ефективної мінімізації його структури.

4. Запропоновано схему мінімізації логічної функції, яка базується на побудові мінімального логічного дерева і забезпечує мінімізацію функції із застосуванням відповідних співвідношень, дає змогу проводити подальше спрощення дужкової форми довільної функції, яка отримана за допомогою концепції логічного дерева.

РОЗДІЛ 6
ЗАГАЛЬНІ МЕТОДИ ПОБУДОВИ МОДЕЛЕЙ ЛОГІЧНИХ ТА АЛГОРИТМІЧНИХ ДЕРЕВ КЛАСИФІКАЦІЇ

6.1. Загальна концепція методів побудови логічних дерев класифікації

На даному етапі дослідження розглянемо загальну ідею методів побудови ЛДК, які базуються на ідеї апроксимації початкової НВ набором ранжованих елементарних ознак. Нехай на початку задано деяку НВ у вигляді начальних пар (з початку першого розділу даного дослідження) та деяку систему (обмежений набір) елементарних ознак (з ознакового простору поточної задачі) – $\varphi_1(x), \varphi_2(x), \dots, \varphi_n(x)$.

На наступному етапі введемо набір наступних множин:

$$G_{r_1,\dots,r_n} = \{x \in G / \varphi_1(x) = r_1, \dots, \varphi_n(x) = r_n\}. \qquad (6.1)$$

Причому система множин $G_{r_1,\dots,r_n}$ представляє собою повне розбиття початкової множини G, яке реалізується виділеним набором елементарних ознак $\varphi_1, \varphi_2, \dots, \varphi_n$, а відповідно $(r_1, r_2, \dots, r_n) \in \{0,1\}$.

Тут слід зауважити, що можлива ситуація, коли деякі з множин $G_{r_1,\dots,r_n}$ можуть будуть пустими.

На наступному етапі через величину $S_{r_1,\dots,r_n}$ виділимо загальну кількість входжень у початкову НВ тих навчальних пар $(x_i, f_R(x_i))$, причому $(1 \le i \le m)$, які задовольняють основну умову належності:

$$x_i \in G_{r_1,\dots,r_n}. \qquad (6.2.(a))$$

Аналогічно через величину $S^j_{r_1,\dots,r_n}$, $(j = 0,1,\dots,k-1)$ визначимо кількість входжень у початкову НВ тих пар $(x_i, f_R(x_i))$ $(i = 1,2,\dots,m)$, які задовольняють умови належності та визначеності:

$$x_i \in G_{r_1,\dots,r_n} \text{ та } f_R(x_i) = j. \qquad (6.2.(b))$$

6.1.1. Вибір критерію розгалуження для структури ЛДК

Далі на даному етапі дослідження введемо наступні необхідні в подальшому базові величини:

$$\delta_{r_1,\dots,r_n} = \frac{S_{r_1,\dots,r_n}}{m}; \quad (6.3.(a))$$

$$\psi^j_{r_1,\dots,r_n} = \frac{S^j_{r_1,\dots,r_n}}{S_{r_1,\dots,r_n}}; \quad (6.3.(b))$$

$$\rho_{r_1,\dots,r_n} = \max_j \psi^j_{r_1,\dots,r_n}. \quad (6.3.(c))$$

Слід зауважити, що якщо не виконується умова належності $x_i \notin G_{r_1,\dots,r_n}$ для всіх $i = 1, \dots, m$, тоді величини $\delta_{r_1,\dots,r_n} = 0$ та $\psi^j_{r_1,\dots,r_n} = 0$, причому $(j = 0,1, \dots, k-1)$.

Відмітимо, що величина $\delta_{r_1,\dots,r_n}$ характеризує собою частоту входжень членів послідовності сигналів (дискретних об‘єктів) $x_1, x_2, \dots, x_m$ у множину $G_{r_1,\dots,r_n}$.

Аналогічно величина $\psi^j_{r_1,\dots,r_n}$ характеризує частоту приналежності деякого дискретного об‘єкту x образу (класу) $H_0, H_1, \dots, H_{k-1}$ при умові, що $x \in G_{r_1,\dots,r_n}$. Тут слід зауважити, що умова належності $x \in G_{r_1,\dots,r_n}$ фактично еквівалентна наступній умові набору елементарних ознак:

$$\varphi_1(x) = r_1, \varphi_2(x) = r_2, \dots, \varphi_n(x) = r_n.$$

Аналогічно величина $\delta_{r_1,\dots,r_n}$ характеризує інформаційну ефективність розпізнавання приналежності дискретного об‘єкта x до одного з класів $H_0, H_1, \dots, H_{k-1}$ звичайно при виконанні базової умови:

$$x \in G_{r_1,\dots,r_n}.$$

Нехай маємо ситуацію, що $x \in G_{r_1,\dots,r_n}$, тоді виникає принципове питання, до якого класу з $H_0, H_1, \dots, H_{k-1}$ слід віднести даний дискретний об‘єкт x. Зрозуміло, що природньо дискретний об‘єкт x слід віднести до того класу H_l, для якого виконується наступне співвідношення:

$$\rho_{r_1,\dots,r_n} = \psi^l_{r_1,\dots,r_n}, \{0 \le l \le k-1\}. \quad (6.4)$$

На цьому етапі можна зробити висновок, що дане співвідношення представляє собою деяке правило класифікації. Тоді з цього можна побачити, що чим більше величина $\rho_{r_1,\dots,r_n}$, тим відповідно вища загальна ефективність розпізнавання.

Зауваження. Як було вказано вище, в якості єдиної початкової інформації, яка характеризує класи $H_0, H_1, \dots, H_{k-1}$, є набір початкових навчальних пар $(x_i, f_R(x_i))$ НВ. Тому доцільно під довільним класом H_l даної задачі розуміти сукупність усіх навчальних пар $(x_i, f_R(x_i))$ НВ, які задовольняють просте співвідношення $f_R(x_i) = l$.

6.1.2. Оцінка якості апроксимації НВ набором елементарних ознак

Тоді можна зафіксувати, що середня ефективність класифікації на образи $H_0, H_1, \dots, H_{k-1}$, які задані початковими даними НВ за допомогою фіксованого набору елементарних ознак $\varphi_1, \varphi_2, \dots, \varphi_n$, може бути оцінена наступною величиною:

$$O_S(\varphi_1, \varphi_2, \dots, \varphi_n) = \sum_{r_1,\dots,r_n} \delta_{r_1,\dots,r_n} * \rho_{r_1,\dots,r_n}. \quad (6.5)$$

Тоді величину $O_S(\varphi_1, \dots, \varphi_n)$ можна вважати оцінкою апроксимації початкової НВ за допомогою набору елементарних ознак $\varphi_1, \varphi_2, \dots, \varphi_n$. Так, з формули (6.3) випливають наступні співвідношення для представлених вище величин $\delta_{r_1,\dots,r_n}$, $\psi^j_{r_1,\dots,r_n}$ та $\rho_{r_1,\dots,r_n}$:

$$\delta_{r_1,\dots,r_n} \geq 0, \sum_{0 \leq r_1,\dots,r_n \leq 1} \delta_{r_1,\dots,r_n} = 1;$$

$$\psi^j_{r_1,\dots,r_n} \geq 0, \sum_{j=0}^{k-1} \psi^j_{r_1,\dots,r_n} = 1;$$

$$\frac{1}{k} \leq \rho_{r_1,\dots,r_n} \leq 1, (r_1, \dots, r_n \in \{0,1\}). \quad (6.6)$$

Можна бачити, що з формул (6.6) безпосередньо випливають такі властивості величини $O_S(\varphi_1, \dots, \varphi_n)$:

1) $\frac{1}{k} \leq O_S(\varphi_1, \varphi_2, \dots, \varphi_n) \leq 1$;

2) $O_S(\varphi_1, \varphi_2, \dots, \varphi_n) = 1 \Leftrightarrow \rho_{r_1,\dots,r_n} = 1$; (6.7)

Тут слід зауважити, що ці властивості актуальні для всіх $r_1, \dots, r_n$, які задовольняють наступному співвідношенню:

$$\exists i (1 \leq i \leq m,\ x_i \in G_{r_1,\dots,r_n}).$$

Так з другої частини (6.7) випливає, що деякий набір елементарних ознак $\varphi_1, \dots, \varphi_n$ тоді і тільки тоді реалізує повне розпізнавання на класи

$H_0, H_1, \dots, H_{k-1}$ (які задані початковою НВ), коли буде в певному сенсі тестом (тобто при умові, що $O_S(\varphi_1, \varphi_2, \dots, \varphi_n) = 1$).

Відмітимо, що формула (6.5) дає можливість фактично просто та ефективно знаходити такі набори елементарних ознак (тести).

На наступному етапі дослідження відмітимо ще одну важливу особливість функціональної оцінки набору елементарних ознак відносно даних початкової НВ.

Так, нехай маємо деяке число $b, (\frac{1}{k} \leq b < 1)$, відповідно величина M_b представляє кількість всіх входжень об‘єктів x_i у послідовність $x_1, x_2, \dots, x_m$, для яких виконуються співвідношення:

$$\begin{cases} x_i \in G_{r_1,\dots,r_n} \\ \rho_{r_1,\dots,r_n} > b \end{cases}. \quad (6.8)$$

Відмітимо, що дане співвідношення фактично представляє ефективність розпізнавання для деяких дискретних об‘єктів x_i за допомогою фіксованого набору елементарних ознак $\varphi_1, \varphi_2, \dots, \varphi_n$, що перевищує число b.

Відповідно величина $\gamma_b = \frac{m_b}{m}$ є часткою тих входжень об‘єктів x_i у послідовність $x_1, x_2, \dots, x_m$, для яких ефективність розпізнавання за допомогою набору елементарних ознак $\varphi_1, \varphi_2, \dots, \varphi_n$ більша ніж число b.

Величина γ_b виражається наступним чином:

$$\gamma_b = \sum_{\rho_{r_1,\dots,r_n} > b} \delta_{r_1,\dots,r_n}. \quad (6.9)$$

Тут слід зауважити, що вираз $\sum_{\rho_{r_1,\dots,r_n} > b}$ означає сумування за всіма $r_1, \dots, r_n$, які задовольняють співвідношення $\rho_{r_1,\dots,r_n} > b$.

На наступному етапі припустимо, що величина $O_S(\varphi_1, \dots, \varphi_n) = c$ та відповідно $\frac{1}{k} \leq b < c$. Тоді з формули (6.5) та співвідношення $O_S(\varphi_1, \dots, \varphi_n) = c$ будемо мати наступне:

$$c = \sum_{\rho_{r_1,\dots,r_n} > b} \delta_{r_1,\dots,r_n} * \rho_{r_1,\dots,r_n} + \sum_{\rho_{r_1,\dots,r_n} \leq b} \delta_{r_1,\dots,r_n} * \rho_{r_1,\dots,r_n}. \quad (6.10)$$

Далі з формул (6.6) та (6.9) отримаємо рівняння:

$$\sum_{\rho_{r_1,\dots,r_n} \leq b} \delta_{r_1,\dots,r_n} = 1 - \gamma_b. \quad (6.11)$$

Відповідно з формул (6.6), (6.9), (6.10) та (6.11) будемо мати наступу нерівність:

$$c \leq \gamma_b + (1 - \gamma_b) * b. \quad (6.12)$$

З якої безпосередньо випливає, що:

$$\gamma_b \geq \frac{c-b}{1-b}. \quad (6.13)$$

На наступному етапі підставимо в нерівність (6.13) наступні значення:

$$c = 1 - \varepsilon^2 \text{ та } b = 1 - \varepsilon.$$

Відмітимо, що тут величина $\varepsilon -$ нескінчене мале число. Тоді будемо мати наступну ситуацію:

$$\gamma_b \geq 1 - \varepsilon. \quad (6.14)$$

Можна побачити, що з нерівності (6.14) випливає, що якщо величина $c \to 1$, то існує така відповідна величина b (зрозуміло, яка залежить від величини c), що виконується наступне:

$$\frac{1}{k} \leq b < c, \lim_{c \to 1} b = 1.$$

Зокрема, величина b у цьому разі не залежить від даних початкової НВ (а отже від її об‘єму, потужності – числа m).

Отже, зважаючи на все вищезазначене, можна зафіксувати наступне:

1) Головна ідея методу логічного дерева класифікації полягає в покроковій апроксимації масиву початкових даних НВ набором ранжованих за інформативністю елементарних ознак.

2) Критерієм розгалуження в методах ЛДК залишається ефективність (інформативність) фіксованої елементарної ознаки φ_i (набору ознак та їх сполучень) відносно деякої частини (підмножини) даних початкової НВ.

3) Важливим напрямком розвитку методів апроксимації даних НВ набором елементарних ознак є вибір ефективного критерію розгалуження (наприклад у випадках малоінформативних ознак [66]) у дереві класифікації (питання швидкої та простої оцінки ефективності наборів та сполучень елементарних ознак φ_i відносно даних початкових НВ).

6.2. Схема методу побудови моделі ЛДК на основі поетапної селекції елементарних ознак

На даному етапі дослідження наведемо просту та ефективну схему методу побудови моделі ЛДК, яка базується на ідеї поетапної селекції елементарних ознак для синтезу результуючої моделі за початковими даними НВ поточної задачі.

Зауважимо лише, що отримана структура ЛДК представляє собою набір ранжованих елементарних ознак (розташованих на відповідному ярусі логічного дерева) з ознакового простору задачі та зв‘язаних між собою множиною переходів – гілок даного логічного дерева.

Нехай на початку задачі задано деяку НВ стандартного вигляду:

$$\left(x_1, f_R(x_1)\right), \dots, \left(x_M, f_R(x_M)\right). \quad (6.15)$$

Зокрема $x_i \in G$ ($G -$ деяка множина початкових сигналів), а відповідно ФР $f_R(x_i) \in \{0,1,2, \dots, k-1\}, (i = 1,2, \dots, M)$, причому $M -$ загальна кількість навчальних пар (об‘єктів відомої класифікації) початкової НВ у задачі.

Відповідно $f_R(x_i) = l, (0 \le l \le k-1)$ означає, що об‘єкт $x_i \in H_l, H_l \subset G$. Причому $f_R -$ деяка скінчено-значна функція (ФР), яка задає початкове розбиття R множини G, котре складається з підмножин (образів, класів) $H_0, H_1, H_2, \dots H_{k-1}$ (заданої НВ).

Отже, можна зафіксувати, що початкова НВ – це сукупність (точніше послідовність) деяких наборів (об‘єктів відомої класифікації), причому кожний набір – це сукупність значень деяких ознак та значення деякої функції (ФР) на цьому наборі. Іншими словами можна підсумувати, що сукупність значень ознак – це деяке зображення (дискретний об‘єкт), а значення функції (ФР) відносить цей об‘єкт до відповідного образу [204].

Отже, зважаючи на зазначене вище, на цьому етапі дослідження буде стояти задача побудови конструкції L деякого ЛДК, структурні параметри p якого були б оптимальними $F(L(p, x_i), f_R(x_i)) \to opt$ щодо початкових даних НВ.

Відмітимо, що головна ідея методу поетапної селекції елементарних ознак полягає в тому, щоб максимізувати величину якості ознаки $W_M(f)$ [205]. Останнє означає, що в алгоритмах логічного дерева має бути знайдена для навчальної вибірки (6.15) така узагальнена ознака f, для якої величина $W_M(f)$ є по можливості найбільшою [206].

Відмітимо, що під важливістю елементарної ознаки (інформативністю) будемо розуміти величину, яка може бути розрахована, як варіант, наступними функціоналами:

$$W(\varphi) = \sum_{i=1}^{M} \frac{b_i}{M} * \rho_i, \text{де } \rho_i = \max_{1 \le m \le k} \frac{q_i^M}{b_i}; \qquad (6.16.(a))$$

$$W(\varphi) = \frac{1}{M} \sum_{j=G_i} \max_{0 \le k \le l} b_j^k. \qquad (6.16.(b))$$

Зрозуміло, що аналогічно можна оцінити важливість інших ознак. Величину $\frac{q_i^m}{b_i}$ можна інтерпретувати як імовірність того, що функція $f_R(x)$ прийме значення $O_m, (1 \le m \le k)$, при умові, що значення ознаки φ дорівнює $i, (1 \le i \le k)$. Величина ρ_i представляє собою максимальну з цих ймовірностей. Можна сказати, що величина ρ_i представляє собою ту інформацію, яку можна отримати про значення функції $f_R(x)$, знаючи, що на наборі z значення ознаки φ дорівнює i.

Тут величина $W(\varphi)$, яка визначається даною формулою, характеризує ту інформацію, яку можна отримати про функцію $f_R(x)$, якщо відомо значення ознаки φ на наборі z. Зрозуміло, що елементарна ознака, для якої ця інформація є найбільшою, вважається найбільш важливою ознакою стосовно $f_R(x)$. Відповідно G_i – множина значень φ_i – тової ознаки, b_j^k – кількість значень j - φ_i – тової ознаки в класі $H_k, (k = 1, \dots, l)$, l – кількість класів H.

Відмітимо, що за аналогією з деревами рішень функціонали (6.16) в даній схемі побудови ЛДК будуть виступати базовим критерієм розгалуження самої структури логічного дерева.

6.2.1. Задача оптимальної апроксимації НВ за допомогою деякої УО

Зауважимо, що вибірка вигляду (6.15) може мати імовірнісний характер, тобто пари $\left(x_i, f_R(x_i)\right), (i = 1,2, \dots, M)$ можуть у ній з‘являтися відповідно деяким імовірнісним розподілам $p(x/H_0), \dots, p(x/H_{k-1})$, але фінальна узагальнена ознака є детермінованою [208].

Отже, ставиться задача оптимальної апроксимації взагалі імовірнісної вибірки типу (6.15) за допомогою деякої детермінованої функції, яка в загальному випадку представлена узагальненою ознакою f.

Очевидно, що задача має сенс тоді, коли характер образів (класів) $H_0, H_1, H_2, \dots H_{k-1}$ достатньо близький до детермінованого. Останнє буде означати, що основну частину займають ті точки (дискретні об‘єкти) x, для яких величина $\max\limits_{0 \le i \le k-1} (p(x/H_0), \dots, p(x/H_i))$ близька до одиниці. Ця величина може істотно змінюватись тільки в точках (дискретних об‘єктах), які лежать на межі декількох класів $H_0, H_1, H_2, \dots H_{k-1}$.

Зауважимо, що на практиці алгоритми ЛДК в основному застосовуються для практичних задач, де образи (класи) $H_0, H_1, H_2, \dots H_{k-1}$ мають характер, близький до детермінованого випадку.

Відмітимо, що дані алгоритми побудови логічних дерев мають наступну особливість – кожний алгоритм представляє собою фіксований поетапний процес, який складається з певних кроків $(s_0, s_1, \dots, s_i)$. Тут слід відмітити, що кожний крок цього процесу s_j складається в свою чергу з двох етапів (режимів): навчання та перевірочного тесту.

Так, в режимі навчання на кроці s_i формується деяка узагальнена ознака f_i. Далі в режимі тесту для цієї узагальненої ознаки розраховується ефективність $W_M(f_i)$ відносно навчальної вибірки типу (6.15).

Відмітимо, що, якщо $W_M(f_i) \ge \delta$, то на цьому процес навчання завершується, якщо $W_M(f_i) < \delta$, тоді здійснюється перехід до кроку s_{i+1}. Зауважимо, що величина δ представляє собою наперед заданий параметр, який характеризує оцінку ефективності навчання, відносно поточної задачі (НВ).

Отже, критерій завершення етапу навчання (аналогічно методам дерев рішень) для даної схеми побудови ЛДК представляє собою наступну логічну умову:

$$\begin{cases} 1, if\ W_M(f_i) \geq \delta; \\ 0, if\ W_M(f_i) < \delta. \end{cases} \quad (6.17)$$

На наступному кроці слід відмітити особливості подачі НВ типу (6.15) на початковому етапі навчання. Так, у практичних задачах побудови ЛДК можливі два наступні випадки:

1) Випадок, коли НВ вигляду (6.15) фіксованого розміру, тобто вся вона подається на кожному кроці s_i процедури навчання.

2) Випадок, коли вибірка вигляду (6.15) залежить від поточного кроку s_i, тобто на кожному кроці процедури навчання s_i подається своя вибірка (або певна частина, фрагмент початкової НВ).

Відмітимо, що випадок (1) має місце тоді, коли початкова НВ типу (6.15) представляє собою наприклад дані деякого експерименту (комп‘ютерних замірів, масиви показників цифрових датчиків), які записані в постійну пам‘ять системи. Алгоритм навчання в цьому випадку представляє собою процедуру багатократної обробки вибірки типу (6.15).

6.2.2. Метод побудови ЛДК на основі ранжування наборів ознак

Зауважимо, що вибірка типу (6.15) може мати дуже великий об‘єм (масив надвеликого об‘єму), тоді єдиним виходом із ситуації є те, що алгоритми обробки початкової НВ мають бути такими, щоб при їх роботі вибірка (6.15) не заносилася в оперативну пам‘ять системи.

Якщо відсутній випадок (1), та не потрібно зберігати дані в постійній пам‘яті комп‘ютера, то маємо випадок (2). У цьому разі всі навчальні пари початкової вибірки (6.15), які обробляються на деякому кроці s_i не запам‘ятовуються, і тому на кроці s_{i+1} подається вже деяка інша серія (послідовність) навчаючих пар типу (6.15).

Для визначеності та спрощення представлення схеми побудови ЛДК далі будемо вважати, що має місце саме випадок (1), тобто на кожному кроці s_i процедури навчання подається одна і та сама початкова НВ типу (6.15).

Отже, в даній схемі методу побудови ЛДК спочатку вибирається деяка елементарна ознака φ_1^1. Від цієї елементарної ознаки вимагається, щоб величина $W_M(\varphi_1^1)$, яка характеризує фактично її інформативність відносно вибірки (6.15), була по можливості найбільшою.

Зауважимо же раз, що $W_M(\varphi_1^1)$ розраховується відповідно до методики (6.16). Наступні кроки методу побудови логічного дерева зручно інтерпретувати за допомогою схеми (рис. 6.1).

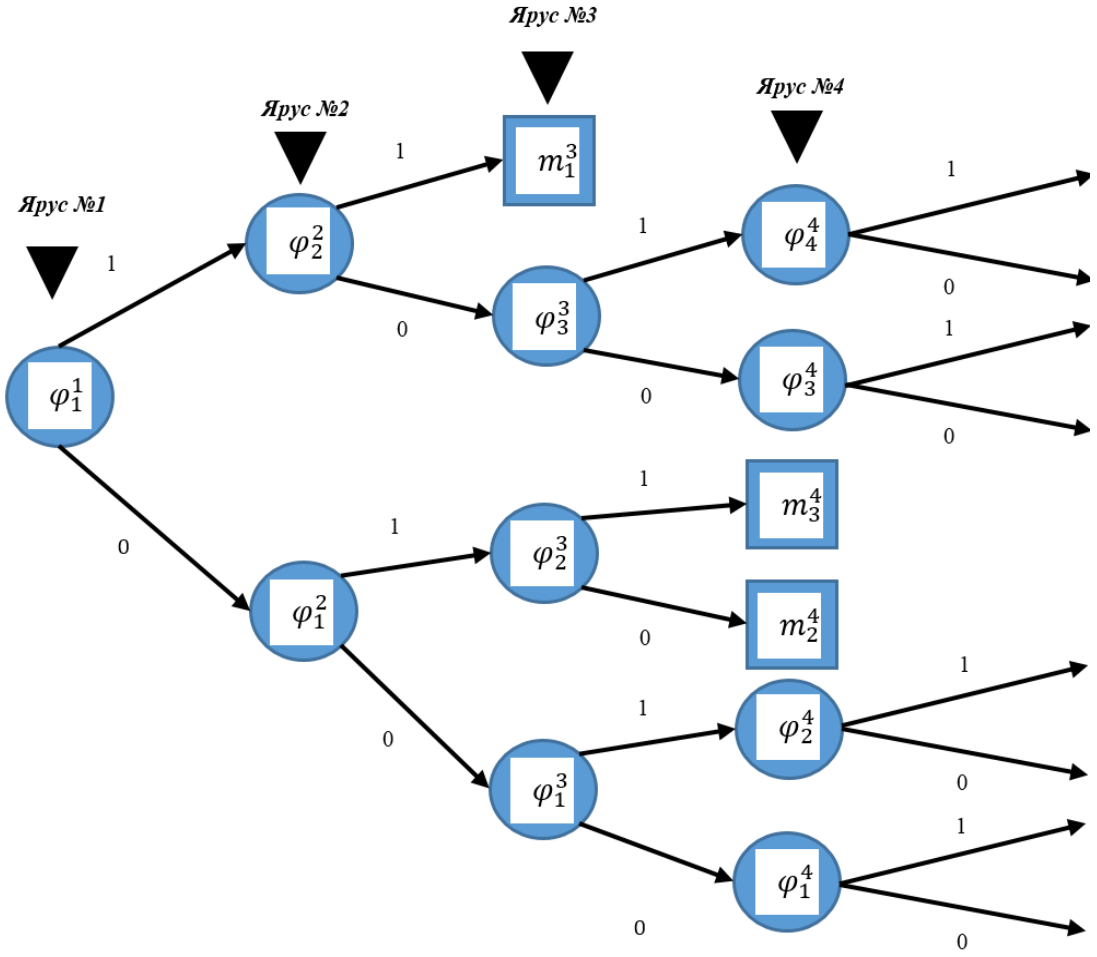


Рис. 6.1. Схема структури ЛДК на основі методу поетапної селекції елементарних ознак.

Відмітимо, що в кожній вершині ЛДК (рис. 6.1) стоїть або деяка ознака φ_i^j (мітка, атрибут), або число m_i^j, яке належить множині $\{0,1,\dots,k-1\}$. Так,

вершина, в якій стоїть m_i^j, називається кінцевою вершиною дерева ЛДК (таку вершину ще називають листом логічного дерева).

Від кожної вершини, в якій стоїть ознака φ_i^j, виходять дві направляючі (стрілки) які позначені 0 та 1. Направляючій, яка позначена 0, відповідає значення $\varphi_i^j = 0$, а позначеній 1 – значення $\varphi_i^j = 1$. Логічне дерево розбито умовно за ярусами (рівнями), причому в $j-$ вому ярусі ЛДК стоять відповідні елементарні ознаки $\varphi_1^j, \varphi_2^j, ...,$.

Відмітимо, що всі елементарні ознаки, які стоять в усіх ярусах логічного дерева, починаючи з першого та закінчуючи $n-$ товим, фактично представляють собою ті ознаки, які отримані після проведення n кроків (етапів) процесу побудови дерева класифікації (ЛДК). Причому елементарні ознаки, які стоять на $n-$ товому ярусі, представляють собою ті ознаки, які отримані відповідно на $n-$ товому кроці (етапі) процедури побудови логічного дерева.

Далі припустимо, що проведено лише три кроки побудови ЛДК та набір $\varphi_1^1, \varphi_1^2, \varphi_2^2, \varphi_1^3, \varphi_2^3, \varphi_3^3 -$ всі елементарні ознаки, які отримані в результаті цих трьох кроків. Логічне дерево, яке отримаємо за ці три кроки, буде мати вигляд, зображений на рис. 6.2.

Відмітимо, що кожній зв‘язаній парі $(x_i, f_R(x_i))$, $(1 \le i \le M)$ НВ вигляду (6.15) буде відповідати певний шлях (Т $-$ опорна множина) в побудованому логічному дереві (рис. 6.2).

Даний шлях у ЛДК реалізується наступним чином:

1) На першому етапі розраховується $\varphi_1^1(x_i) = r_i$.

2) Далі від вершини φ_1^1 спускаємося вниз за стрілкою, яка позначена через r_i. Нехай наприклад $\varphi_1^1(x_i) = r_i = 0$, тоді спускаємось у вершину ЛДК, в якій стоїть ознака φ_1^2. Далі розрахуємо $\varphi_1^2(x_i) = r$ та спускаємося за стрілкою, яка виходить із вершини φ_1^2 та позначена значенням r_2 і так далі.

Так, шлях, якій відповідає навчальній парі $(x_i, f_R(x_i))$ (він повністю визначається значенням x_i), позначимо через T_i.

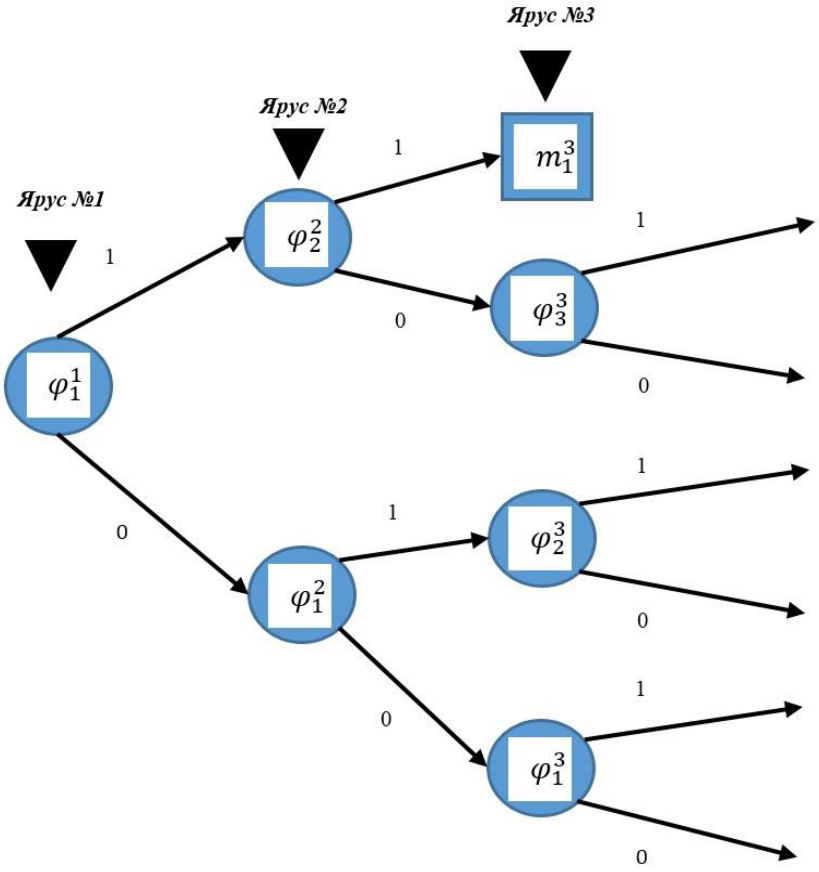


Рис. 6.2. Структура ЛДК після трьох кроків процедури селекції елементарних ознак.

Відмітимо, що в цій ситуації можливі два випадки:

1) Нехай шлях T_i закінчується деякою направляючою стрілкою. Наприклад, якщо маємо ситуацію, що $\varphi_1^1(x_i) = 0$, $\varphi_1^2(x_i) = 0$, $\varphi_1^3(x_i) = 0$, то шлях T_i закінчується стрілкою, яка виходить із вершини φ_1^3 та позначається символом 0 (рис. 6.2).

2) Нехай шлях T_i закінчується деякою вершиною, в якій стоїть відповідне значення – m_i^j. Наприклад, коли $\varphi_1^1(x_i) = 1$, $\varphi_2^2(x_i) = 1$, то шлях T_i закінчується вершиною, в якій стоїть значення m_1^3 (рис. 6.2).

Далі шляхи у випадку (1) будемо називати незакінченими, а шляхи у випадку (2) – закінченими.

Якщо шлях T_i, який відповідає навчальній парі $(x_i, f_R(x_i))$, є закінченим та в кінці стоїть значення $m_i^j, (m_i^j \in \{0,1,\dots,k-1\})$, то це означає, що $f_R(x_i) = m_i^j$.

Наприклад на рис. 6.2 для всіх навчальних пар $(x_i, f_R(x_i))$, які задовільнять умову $\varphi_1^1(x_i) = \varphi_2^2(x_i) = 1$, виконується ще така умова – $f_R(x_i) = m_1^3$. Можна зазначити, що для значення x_i, якому відповідає закінчений шлях T_i, реалізується повне розпізнавання за логічним деревом (рис. 6.2). Тобто можна зазначити, що навчальна пара $(x_i, f_R(x_i))$, належить відповідному шляху T_i даного ЛДК.

Далі при проведенні наступних етапів процедури побудови дерева класифікації розглядаються лише незакінчені шляхи в структурі даного дерева.

Кожний шлях на дереві класифікації, що будується, будемо позначати відповідним двійковим набором $r_1, r_2, r_3, \ldots (r_i \in \{0,1\})$. Наприклад, двійковий набір 010 на дереві класифікації (рис. 6.2) позначає шлях, який закінчується кінцевою стрілкою, з вершини φ_2^3 та позначеною символом 0 (листом, міткою). Очевидно, що сукупність $000, 001, 010, 011, 100, 101$ є множиною всіх незакінчених шляхів у структурі даного дерева класифікації (рис. 6.2).

Нехай величина M_{r_1,r_2,r_3} – загальна кількість усіх навчальних пар $(x_i, f_R(x_i))$ вибірки типу (6.15), які належать незакінченому шляху r_1, r_2, r_3 у структурі логічного дерева (рис. 6.2), та $M^j_{r_1,r_2,r_3}, (0 \le j \le k-1)$ – загальна кількість всіх пар, які належать даному шляху r_1, r_2, r_3 та, крім того, для них ще виконується співвідношення $f_R(x_i) = j$.

На наступному етапі для кожного незакінченого шляху r_1, r_2, r_3 дерева класифікації (рис. 6.2) розрахуємо наступні величини:

$$t^j_{r_1,r_2,r_3} = \frac{M^j_{r_1,r_2,r_3}}{M_{r_1,r_2,r_3}}. \tag{6.18}$$

Зауважимо, що $j = 0,1, \ldots, k-1$.

Далі знайдемо таку величину $l(r_1, r_2, r_3)$, що:

$$l(r_1, r_2, r_3) \in \{0,1, \ldots, k-1\};\ t^{l(r_1,r_2,r_3)}_{r_1,r_2,r_3} = \max_j t^j_{r_1,r_2,r_3}.$$

Тоді, підставивши в кінці кожного шляху r_1, r_2, r_3 на ЛДК структури (рис. 6.2) величину $l(r_1, r_2, r_3)$, отримаємо дерево класифікації, зображене на рис. 6.3.

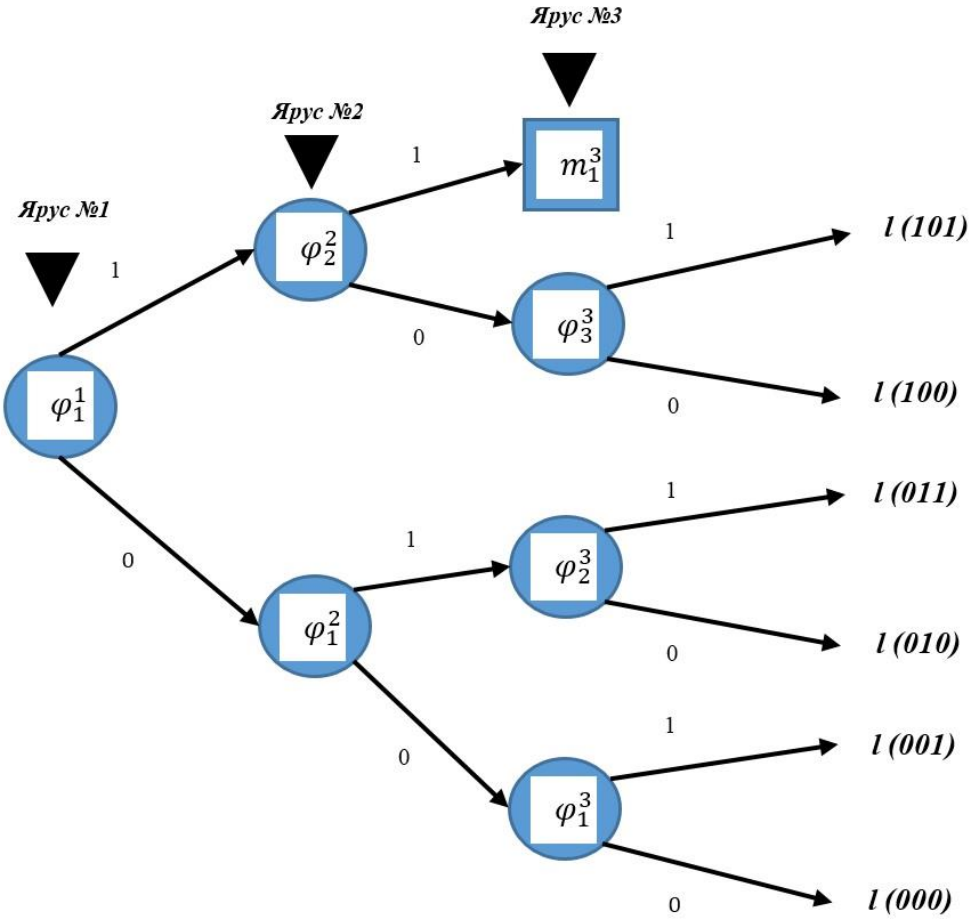


Рис. 6.3. Логічне дерево після трьох кроків процедури генерації з фіксованими шляхами.

Відмітимо, що дерево класифікації (рис. 6.3) фактично реалізує (представляє) деяку узагальнену ознаку $f_3(x)$, яка визначена на множині G та приймає значення з множини $\{0,1,\dots,k-1\}$. Ознака $f_3(x)$ розраховується наступним чином. Спочатку знаходимо для сигналів (дискретних об‘єктів) x весь шлях T_x, якій відповідає даному елементу (об‘єкту НВ). Наприклад, якщо $\varphi_1^1(x)=0, \varphi_1^2(x)=1, \varphi_2^3(x)=1$ то відповідно $T_x = 011$.

Відмітимо, що в якості значень $f_3(x)$ береться елемент із множини $\{0,1,\dots,k-1\}$, яким закінчується шлях T_x.

Наприклад, якщо $T_x = 011$, то $f_3(x) = l(011)$. Відмітимо, що якщо об‘єкту x відповідає закінчений шлях T_x, у кінці якого стоїть число m_x $(0 \le m_x \le l)$ – результуюче значення ФР, тоді фіксуємо відповідне значення узагальненої ознаки – $f_3(x) = m_x$.

Після побудови узагальненої ознаки $f_3(x)$ починається етап перевірочного тесту. Так, у режимі перевірочного тесту підраховується загальна кількість S всіх тих пар $(x_i, f_R(x_i))$ з вибірки типу (6.15), для яких виконується співвідношення $f_R(x_i) = f_3(x)$.

На наступному етапі необхідно перевірити умову $\frac{S}{M} \ge \delta$. Відмітимо, що δ – параметр, який характеризує оцінку ефективності навчання відносно поточної задачі (НВ).

Якщо цю умову виконано, то на цьому процес побудови дерева класифікації закінчується, а узагальнена ознака $f_3(x)$, яка представляється логічним деревом структури (рис. 6.3) є такою, що забезпечує апроксимацію вибірки вигляду (6.15).

Якщо виконується умова $\frac{S}{M} < \delta$, то процес побудови дерева класифікації продовжується.

Відмітимо, що при побудові дерева класифікації спочатку виділяються на ЛДК (рис. 6.3) всі ті значення $l(r_1, r_2, r_3)$, для яких виконується співвідношення $t_{r_1,r_2,r_3}^{l(r_1,r_2,r_3)} = 1$. Шляхи r_1, r_2, r_3, для яких виконується тільки що вказане співвідношення, можна вважати закінченими.

Наприклад нехай:

$$t_{010}^{l(010)} = t_{011}^{l(011)} = 1;\ t_{000}^{l(000)} < 1;$$
$$t_{001}^{l(001)} < 1;\ t_{100}^{l(100)} < 1;\ t_{101}^{l(101)} < 1.$$

Відмітимо, що в цьому разі буде отримано дерево класифікації (рис. 6.4), де $m_2^4 = l(010), m_3^4 = l(011)$.

Можна побачити, що всі шляхи $000, 100$ та 101 на дереві класифікації (рис. 6.4) є незакінченими. Для кожного з цих шляхів r_1, r_2, r_3 розглядаються множини H_{r_1,r_2,r_3}.

Зауважимо, що H_{r_1,r_2,r_3} – відповідні множини всіх тих пар $(x_i, f_R(x_i))$ навчальної вибірки типу (6.15), які належать шляху r_1, r_2, r_3. Фактично множини H_{r_1,r_2,r_3} можна вважати деякими вибірками (фрагментами початкової НВ). Відмітимо, що у випадку логічного дерева (рис. 6.4) будемо мати наступні вибірки (підмножини початкової НВ): $H_{000}, H_{001}, H_{100}, H_{101}$.

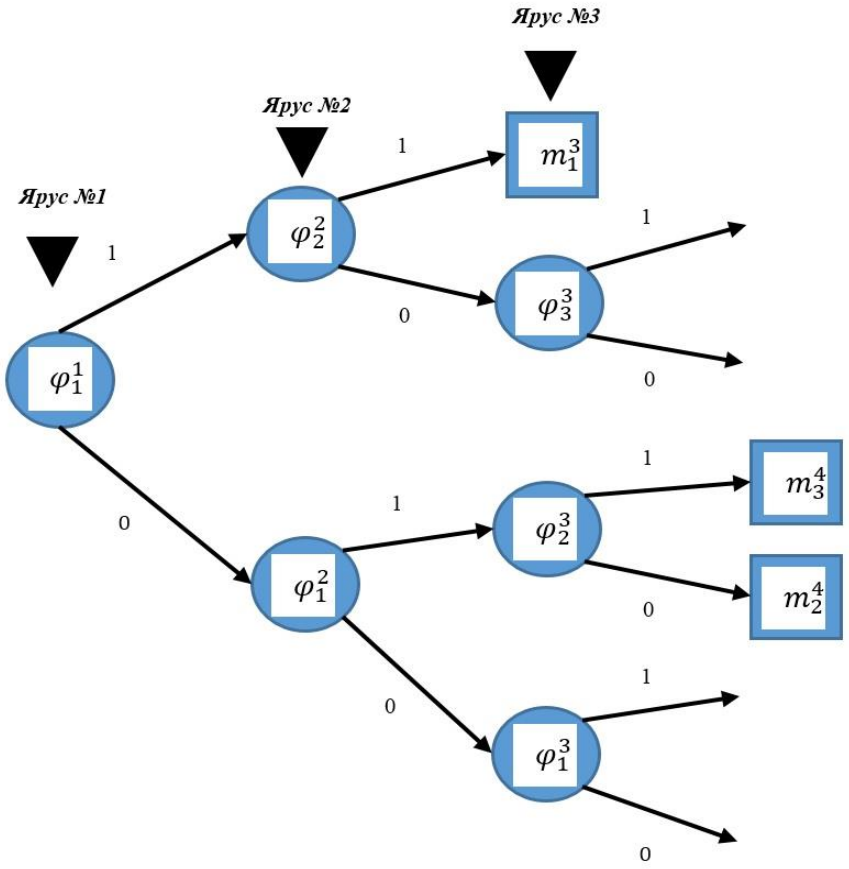


Рис. 6.4. Фрагмент фінальної структури ЛДК.

Відмітимо, що для кожної вибірки H_{r_1,r_2,r_3} вибирається така елементарна ознака φ_{r_1,r_2,r_3}, для якої величина $W_{H_{r_1,r_2,r_3}}(\varphi_{r_1,r_2,r_3})$ є по можливості найбільшою. Тут величина $W_{H_{r_1,r_2,r_3}}(\varphi_{r_1,r_2,r_3})$ представляє собою ефективність розпізнавання вибірки H_{r_1,r_2,r_3} за допомогою ознаки φ_{r_1,r_2,r_3}. Після вибору ознак φ_{r_1,r_2,r_3} буде отримано нове ЛДК структури, зображеної на рис. 6.1.

Зокрема, будемо мати наступну ситуацію:

$$\varphi_1^4 = \varphi_{000};\ \varphi_2^4 = \varphi_{001};\ \varphi_3^4 = \varphi_{100};\ \varphi_4^4 = \varphi_{101}.$$

Далі до дерева структури (рис. 6.1) застосовується той самий процес, що й до дерева (рис. 6.2).

Важливо підкреслити, що для реалізації кожної вибірки H_{r_1,r_2,r_3} не потрібно формувати окрему множину навчаючих пар. Всі ці вибірки реалізуються таким способом: послідовно подаються пари початкової вибірки (6.15) та до уваги беруться тільки ті навчальні пари, які належать шляху r_1, r_2, r_3. У результаті цього процесу і буде реалізовуватись вибірка H_{r_1,r_2,r_3}.

6.2.3. Етап експериментальної перевірки моделей ЛДК

Відмітимо, що на основі однієї початкової НВ можна побудувати набір ЛДК, використовуючи відповідні методи та алгоритми. Так, застосовуючи в якості критерію розгалуження запропоновані функціонали (6.16) можна побудувати як мінімум два ЛДК залежно від того, чи проводити оцінку якості елементарних ознак на кожному кроці генерації логічного дерева, чи зробити це один раз на початку побудови, і тим самим заощадити апаратні ресурси системи. Тому виникає актуальна задача порівняння отриманих моделей із метою вибору найкращої відносно поточної початкової НВ.

Так, важливим етапом порівняння побудованих деревоподібних моделей розпізнавання є етап визначення показників, які характеризують базові властивості отриманих моделей. Причому порівняння моделей проводиться на основі інтегрального показника якості – центрального критерію порівняння моделей ЛДК.

За аналогією з роботою [66] зафіксуємо базові характеристики синтезованих ЛДК наступному переліку:

V_{tr} – загальна кількість вершин побудованого ЛДК; відмітимо, що мінімальна кількість вершин ЛДК складає $(m + 1)$;

N_{tr} – загальна кількість елементарних ознак, які використовуються в структурі побудованої моделі ЛДК;

C_{tr} – загальна кількість переходів (зв‘язків) у структурі побудованої моделі ЛДК;

O_{tr} – загальна кількість результуючих значень ФР (листків дерева) в структурі побудованої моделі ЛДК;

En_{tr} – помилка моделі ЛДК на масиві даних НВ;

Et_{tr} – помилка моделі ЛДК на масиві даних ТВ;

Er_H – помилка на кожному з класів дискретних об‘єктів поточної задачі, причому $(H = 1, \dots, k)$.

На наступному етапі зафіксуємо основні параметри моделі ЛДК відносно його характеристик:

$C_{avg} = (C_{tr}/V_{tr})$ – середня кількість переходів дерева класифікації на одну вершину в структурі ЛДК

$N_{tr}^V = (N_{tr}/n)$ – частка елементарних ознак, які використовуються в структурі побудованого ЛДК;

$O_{tr}^V = (O_{tr}/V_{tr})$ – частка результуючих значень ФР у загальній структурі побудованого ЛДК;

$Q_{avg} = (M/O_{tr})$ – середня кількість наборів НВ на результуюче значення ФР (листків дерева) в структурі побудованого ЛДК.

Далі введемо показник узагальнення даних початкової НВ моделі ЛДК, який розраховується наступним чином:

$$I_{Main} = \frac{M * n}{V_{tr} + 2C_{tr}}.$$

Зрозуміло, що критично важливими параметрами отриманої моделі ЛДК, які необхідно мінімізувати, є помилки моделі En_{tr}, Et_{tr}, Er_H (відповідно на даних НВ, ТВ та для кожного з класів початкового розбиття множини G поточної задачі).

Зауважимо, що принциповим моментом довільної практичної задачі залишається питання зменшення загальної складності структури ЛДК (тут маються на увазі параметри: N_{tr} – кількість ознак у структурі ЛДК, V_{tr} –

кількість вершин моделі ЛДК та C_{tr} – загальна кількість переходів в структурі ЛДК), параметри витрат пам‘яті λ та процесорного часу τ.

Відмітимо, що доречно в структурі ЛДК збільшити параметри O_{tr} та O_{tr}^{V}, що дозволить зменшити час прийняття рішень за даною моделлю логічного дерева та заощадити процесорний час системи.

Слід відмітити, що також необхідно максимізувати параметр I_{Main} (показник узагальнення моделі ЛДК), що дає можливість домогтися оптимальної структури ЛДК та забезпечує фактично максимальний стиск даних початкової НВ (представлення масиву початкових даних мінімальним за структурною складністю логічним деревом класифікації), та параметр Q_{avg} (середня кількість навчальних наборів НВ на результуючі значення ФР – листи структури ЛДК).

Важливим показником якості побудованої моделі у вигляді ЛДК з урахуванням вище зазначених параметрів є інтегральний показник якості моделі в наступній формі:

$$Q_{Main} = \frac{O_{tr} * e^{-\frac{|En_{tr} * Et_{tr} - \delta^2|}{M * M_{ts}}}}{N_{tr} * V_{tr} * C_{tr}}.$$

Тут слід зауважити, що $M, M_{ts} -$ потужності навчальної та тестової вибірки відповідно, а даний інтегральний показник Q_{Main} якості моделі ЛДК має сенс лише при виконанні умови:

$$(En_{tr}/M) \leq \delta.$$

Зрозуміло, що в протилежному випадку він буде дорівнювати нулю. Збільшення цього показника характеризує зростання якості моделі ЛДК і навпаки – зменшення вказує на погіршення якості класифікації.

Так, на основі методу дерева класифікації та принципу модульності в Ужгородському національному університеті було розроблено програмний комплекс «Оріон ІІІ» для генерації автономних систем розпізнавання. Алгоритмічна бібліотека системи нараховує

15 алгоритмів розпізнавання, серед яких запропонована вище алгоритмічна реалізація побудови ЛДК [94].

Базовою задачею комплексу було конструювання автономної системи розпізнавання на основі геологічних даних (задача про розділення нафтоносних пластів). Для розпізнавання об‘єктів використовувалися 12 основних елементарних ознак та 10 додаткових.

Відмітимо, що в НВ представлено інформацію про об‘єкти двох класів. Так, на етапі екзамену побудована система класифікації має забезпечити ефективне розпізнавання об‘єктів невідомої класифікації відносно цих двох класів. Перед початком роботи навчальну вибірку було автоматично перевірено на коректність (пошук та видалення однакових об‘єктів різної належності – помилки першого роду), хоча в системі і реалізовано схему донавчання та виправлення помилок в дереві класифікації (алгоритм ДВП), а оскільки генерація проходила в автоматичному режимі, то даний алгоритм не використовувався.

Зауважимо, що навчаюча вибірка складалася з 1250 об‘єктів (з них нафтоносні – 756 об‘єктів), причому ефективність сконструйованої системи розпізнавання оцінювалася на тестовій виборці об‘єму 240 об‘єктів. Дані навчаючих та тестових вибірок отримано на основі геологічної розвідки на території Закарпатської області в період з 2001 року по 2011 рік.

Відмітимо, що фрагмент основних результатів приведених вище експериментів представлено в табл. 6.1. Причому побудовані моделі ЛДК забезпечили необхідний рівень точності, який задано умовою задачі, швидкодію та витрати робочої пам‘яті системи.

Зауважимо, що запропоновані оцінки якості моделі ЛДК фіксують найважливіші характеристики логічних дерев та можуть бути застосовані в якості критерію оптимальності в процедурі побудови ЛДК та фінальному відборі з множини моделей ЛДК.

Відмітимо, що дана схема побудови ЛДК порівнювалась із методом алгоритмічного дерева класифікації (АДК) та показала прийнятний результат.

Головна ідея АДК полягає в апроксимації набором алгоритмів початкової НВ. Отримана структура АДК характеризується високою універсальністю та відносно компактною структурою самої моделі, однак вимагає великих апаратних витрат для зберігання узагальнених ознак та початкової оцінки якості алгоритмів класифікації за НВ. Порівняно з нею ЛДК має високу швидкодію правил класифікації, незначні апаратні витрати для зберігання та роботи самої структури дерева та високу якість класифікації.

Таблиця 6.1. Порівняльна таблиця методів синтезу ЛДК.

Метод синтезу структури логічного дерева	*Початкові дані*				*Інтегральний показник якості моделі* Q_{Main}	*Загальна кількість помилок моделі на ТВ* Et_{tr}
	Кількість класів в НВ	*Кількість ознак об'єктів НВ*	*Загальна потужність НВ*	*Загальна потужність ТВ*		
Запропонований метод селекції елементарних ознак (метод ЛДК) [95].	2	22	1250	240	0,00231412	0
Обмежений метод побудови моделі ЛДК регульованої точності [118].	2	22	1250	240	0,00172347	3
Метод дерева класифікації на основі автономних алгоритмі класифікації та розпізнавання (метод АДК типу І) [92].	2	22	1250	240	0,00213489	1
Метод дерева класифікації на основі автономних алгоритмі класифікації та розпізнавання (метод АДК типу ІІ) [163].	2	22	1250	240	0,00183608	2

Отриманий результат базується на тому, що запропонований простий метод побудови ЛДК на основі селекції елементарних ознак із постійною оцінкою їх важливості на кожному кроці генерації дерева класифікації дає компактні та ефективні моделі класифікації. Причому на кожному кроці розгалуження враховується вплив того чи іншого значення ознаки на результуюче значення ФР у структурі дерева. Відмітимо, що запропонований функціонал можна використовувати не тільки для оцінки інформативності окремих елементарних ознак, але й для розрахунку важливості наборів ознак та їх сполучень, що в перспективі дозволяє досягти оптимальної структури синтезованого ЛДК за початковими даними НВ.

У даній частині роботи запропоновано набір загальних показників, який дозволяє ефективно представити загальні характеристики моделі ЛДК, можливе його використання для відбору оптимального ЛДК із множини побудованих (наприклад у випадку алгоритмів побудови випадкових ЛДК із першого розділу даного дослідження).

Відмітимо, що запропонований метод побудови ЛДК на основі селекції елементарних ознак був реалізований в основній бібліотеці алгоритмів класифікації універсальної програмної системі “ОРІОН ІІІ” для розв‘язку різноманітних практичних задач класифікації (розпізнавання) масивів дискретних об‘єктів.

Зауважимо, що проведені практичні випробовування підтвердили працездатність запропонованих моделей ЛДК та програмного забезпечення, що дає можливість рекомендувати використання даного підходу та його програмної реалізації для широкого спектру прикладних задач класифікації та розпізнавання дискретних об‘єктів.

Відмітимо, що перспективи подальших досліджень будуть спрямовані в бік розвитку методів алгоритмічних дерев класифікації оптимізації програмних реалізацій запропонованого методу побудови ЛДК, а також його практичної апробації на множині реальних задач класифікації та розпізнавання.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступне:

1) Апроксимація початкової НВ набором відсортованих елементарних ознак (оцінених за інформативністю) дає відносно компактні та ефективні структури моделей ЛДК.

2) Важливим питанням у задачах побудови структур ЛДК залишається проблема вибору якісного критерію розгалуження, а, отже, ефективних методів оцінки інформативності елементарних ознак, їх наборів та різноманітних сполучень.

3) Слід відмітити принциповий момент, що за аналогією з підходом апроксимації НВ набором елементарних ознак, можна запропонувати метод

апроксимації початкової навчальної інформації набором автономних різнотипних алгоритмів розпізнавання і класифікації та отримати деревоподібну модель АДК, що і буде зроблено далі.

4) Важливим моментом даного методу є можливість уведення критерію зупинки розгалуження, що дозволяє в прикладній задачі керувати складністю (точністю) моделі ЛДК, що будується. Далі в межах даного дослідження буде запропоновано обмежений метод побудови ЛДК на основі селекції елементарних ознак.

5) Принциповою особливістю даного підходу побудови дерева класифікації є те, що у випадку надвеликих початкових НВ отримана структура ЛДК буде характеризуватися великою складністю конструкції, що негативно впливає на можливість аналізу (виділення правил класифікації) та роботи з такою структурою. Одним із варіантів виходу з цієї ситуації є модифікований (обмежений) метод побудови ЛДК (на основі селекції елементарних ознак), який проводить добудову структури дерева класифікації лише за шляхами, де відбувається найбільша кількість помилок класифікації.

6) Важливим питанням теорії дерев класифікації (саме для випадку ЛДК) залишається питання використання в якості вершин ЛДК наборів елементарних ознак та їх фіксованих сполучень, тобто випадок малоінформативних ознак [66]. Причому принципове вирішення даного питання дозволить суттєво вдосконалити моделі (концепцію) ЛДК, підняти параметри універсальності щодо прикладних застосувань та надати нове життя методам дерев рішень.

6.3. Загальна концепція методів побудови алгоритмічних дерев класифікації

На наступному етапі дослідження за аналогією з методами апроксимації НВ набором оцінених елементарних ознак представимо головну ідею методів алгоритмічних дерев класифікації, яка в свою чергу полягає в апроксимації масиву початкової НВ набором автономних різнотипних алгоритмів класифікації.

Нехай задано початкову НВ загального вигляду (6.15) як послідовність навчальних пар відомої класифікації (потужності m) та деяку систему (набір) незалежних та автономних алгоритмів розпізнавання (класифікації) для початкової НВ $\alpha_1(x), \alpha_2(x), \dots, \alpha_n(x)$. Далі необхідно ввести наступні множини, які представляють розбиття даних НВ відповідними алгоритмами класифікації a_i:

$$G_{a_1,\dots,a_i} = \{x \in G / \alpha_i(x) = 1\}, (i = 1, \dots, n). \quad (6.19)$$

Зауважимо, що для спрощення пояснень кожний з автономних алгоритмів класифікації $\alpha_i(x)$ генерує на виході значення лише в межах бінарної множини $\{0,1\}$, зокрема $\alpha_i(x) = 1$ у разі вдалої класифікації об‘єкта x та $\alpha_i(x) = 0$ – у протилежному випадку.

Відмітимо, що система множин $G_{a_1,\dots,a_i}$ буде фактично представляти собою повне поетапне розбиття множини G (зі зростанням величини i – задіяних алгоритмів класифікації), яке реалізується незалежними алгоритмами $\alpha_1, \alpha_2, \dots, \alpha_n$. Зауважимо, що залежно від початкового обрання набору алгоритмів класифікації $\alpha_1, \alpha_2, \dots, \alpha_n$ деякі з множин $G_{a_1,\dots,a_i}$ можуть бути порожніми (випадок непридатності одного конкретного або декількох алгоритмів для апроксимації поточної НВ).

На наступному етапі через величину $S_{a_1,\dots,a_n}$ позначимо кількість входжень у початкову НВ тих навчальних пар $(x_s, f_R(x_s))$, $(1 \le s \le m)$, які задовольняють базову умову належності $x_s \in G_{a_1,\dots,a_i}$.

Відповідно через величину $S^{j}_{a_1,\dots,a_i}$, $(j = 0,1,\dots,k-1)$ позначимо кількість входжень у НВ тих пар $(x_s, f_R(x_s))$ $(s = 1,2,\dots,m)$, які задовольняють умови $x_i \in G_{a_1,\dots,a_n}$ та $f_R(x_s) = j$.

6.3.1. Вибір критерію розгалуження для структури АДК

Отже, зважаючи на зазначене вище, сказане та за аналогією з методами селекції наборів елементарних ознак, можна ввести наступні величини, які доцільно розглядати, як певний критерій розгалуження в структурі АДК:

$$\delta_{a_1,\dots,a_i} = \frac{s_{a_1,\dots,a_i}}{m}, \psi^{j}_{a_1,\dots,a_i} = \frac{s^{j}_{a_1,\dots,a_i}}{s_{a_1,\dots,a_i}}, \rho_{a_1,\dots,a_i} = \max_j \psi^{j}_{a_1,\dots,a_i}. \qquad (6.20)$$

Зауважимо, що якщо об‘єкт $x_s \notin G_{a_1,\dots,a_i}$ для всіх $s = 1,\dots,m$, тоді зрозуміло, що $\delta_{a_1,\dots,a_i} = 0$ та $\psi^{j}_{a_1,\dots,a_i} = 0$ при $j = 0,1,\dots,k-1$.

Зокрема величина $\delta_{a_1,\dots,a_i}$ характеризує частоту входжень членів послідовності $x_1, x_2, \dots, x_m$ (дискретних об‘єктів) у множину $G_{a_1,\dots,a_i}$, а відповідно величина $\psi^{j}_{a_1,\dots,a_i}$ характеризує частоту приналежності деякого об‘єкта x образу (класу) H_j, за умови, що $x \in G_{a_1,\dots,a_i}$. Слід зауважити, що наведена умова еквівалентна умові, що в послідовності алгоритмів $a_1,\dots,a_i$ знайдеться такий алгоритм a_y, що $a_y(x) = 1$. Тоді величина $\delta_{a_1,\dots,a_i}$ характеризує інформаційну ефективність розпізнавання приналежності деякого об‘єкту x до одного з класів $H_0, H_1, \dots, H_{k-1}$ звичайно за умови, що $x \in G_{a_1,\dots,a_i}$.

На наступному етапі знову виникає принципове питання щодо належності об‘єкта x класам $H_0, H_1, \dots, H_{k-1}$ (питання формування правила класифікації). Зрозуміло, що слід віднести об‘єкт x до того класу H_j, для якого виконується просте співвідношення:

$$\rho_{a_1,\dots,a_i} = \psi^{j}_{a_1,\dots,a_i}. \qquad (6.21)$$

Зауважимо, що тут $\{0 \le j \le k-1\}$, а співвідношення (6.21) представляє собою деяке правило класифікації, причому зрозуміло, що чим більше значення величини $\rho_{a_1,\dots,a_i}$, тим вища ефективність правила.

Оскільки єдиною інформацією, яка представляє розбиття образів $H_0, H_1, \dots, H_{k-1}$, є початкова НВ, то під класом H_j розуміється сукупність усіх навчальних пар $(x_s, f_R(x_s))$ НВ, які задовольняють співвідношенню $f_R(x_i) = j$, тобто умові належності.

6.3.2. Оцінка якості апроксимації НВ набором незалежних алгоритмів класифікації

Аналогічно середня ефективність розпізнавання набору образів $H_0, H_1, \dots, H_{k-1}$, які задані даними НВ за допомогою алгоритмів розпізнавання $\alpha_1, \alpha_2, \dots, \alpha_n$ оцінюється наступною величиною:

$$F_S(\alpha_1, \alpha_2, \dots, \alpha_n) = \sum_{a_1,\dots,a_i} \delta_{a_1,\dots,a_i} * \rho_{a_1,\dots,a_i}. \quad (6.22)$$

Отже в даному випадку величину $F_S(\alpha_1, \dots, \alpha_n)$ можна вважати оцінкою апроксимації початкової НВ за допомогою набору незалежних алгоритмів класифікації $\alpha_1, \alpha_2, \dots, \alpha_n$.

Зважаючи на саму ідею алгоритмічного дерева класифікації, яка була представлена вище, величину $F_S(\alpha_1, \dots, \alpha_n)$ можна отримати ще з таких міркувань: функцією розпізнавання F назвемо деяке відображення, яке кожному набору $a_1, \dots, a_i$ ставить у відповідність деякий елемент множини $\{0,1, \dots, k-1\}$ (тобто відповідний номер класу).

Таким чином, ФР F представляє собою функцію вигляду $F(a_1, \dots, a_i)$, де $a_1, \dots, a_i$ приймають значення з множини $\{0,1\}$. Відповідно ФР $F(a_1, \dots, a_i)$ об‘єкт $x, (x \in G)$ однозначно відноситься до того класу (образу) $H_j, (0 \le j \le k-1)$, для якого виконується наступне співвідношення:

$$F(a_1, \dots, a_i) = l, (0 \le l \le k-1). \quad (6.23)$$

Нехай задано початкову НВ вигляду (6.15), тоді будемо вважати, що ФР $F(a_1, \dots, a_i)$ правильно класифікує набір $\left(x_j, f_R(x_j)\right), (1 \le j \le m)$ масиву даних (6.15), якщо $F\left(a_1(x_j), \dots, a_i(x_j)\right) = f_R(x_j)$, в протилежному випадку ФР $F(a_1, \dots, a_i)$ неправильно класифікує навчальну пару $\left(x_j, f_R(x_j)\right)$.

Далі, нехай величина $m_F -$ кількість всіх входжень навчальних пар $\left(x_j, f_R(x_j)\right)$ у початкову НВ, які правильно класифікуються ФР $F(a_1, \dots, a_i)$. Тоді введемо наступну величину:

$$\tau_F = \frac{m_F}{m}. \tag{6.24}$$

Зауважимо, що дана величина τ_F може вважатися загальною ефективністю ФР $F(a_1, \dots, a_i)$ для початкової НВ відносно деякого набору алгоритмів класифікації $\alpha_1, \dots, \alpha_n$.

На наступному етапі дослідження виразимо величину τ_F через запропоновані раніше величини $\delta_{a_1,\dots,a_i}$ та $\psi^j_{a_1,\dots,a_i}$, $(i = 1,2, \dots, n; j = 0, \dots, k-1)$. Для цього підрахуємо кількість тих навчальних пар $\left(x_s, f_R(x_s)\right)$, які правильно класифікуються ФР $F(a_1, \dots, a_i)$ та для яких виконується відношення належності $x_s \in G_{a_1,\dots,a_i}$. Нехай $F(a_1, \dots, a_i) = l$, тоді кількість всіх навчальних пар $\left(x_s, f_R(x_s)\right)$, які правильно класифікуються (при виконанні базової умови належності $x_s \in G_{a_1,\dots,a_i}, f_R(x_s) = l$) дорівнює $S^l_{a_1,\dots,a_i}$ (відповідно до попередніх виразів (6.20)). Тоді стає очевидно, що введена вище величина m_F розраховується за наступною формулою:

$$m_F = \sum_{0 \le a_1,\dots,a_i \le 1} S^{F(a_1,\dots,a_i)}_{a_1,\dots,a_i}. \tag{6.24}$$

Тут слід зауважити, що в формулі (6.24) можна брати до уваги тільки ті навчальні набори, для яких виконується співвідношення $S_{a_1,\dots,a_i} \neq 0$ (величина $S_{a_1,\dots,a_i}$ аналогічна величинам із (6.20)). Отже, беручи до уваги щойно вказане зауваження щодо (6.24), можна представити величину m_F в наступному вигляді:

$$m_F = \sum_{0 \le a_1,\dots,a_i \le 1} S_{a_1,\dots,a_i} * \psi^{F(a_1,\dots,a_i)}_{a_1,\dots,a_i}. \tag{6.25}$$

Зауважимо, що величина $\psi^{F(a_1,\dots,a_i)}_{a_1,\dots,a_i}$ була попередньо визначена в виразах (6.20).

На наступному етапі з формул (6.25) та (6.24) отримаємо наступне:

$$\tau_F = \frac{m_F}{m} = \sum_{0 \le a_1,\dots,a_i \le 1} \delta_{a_1,\dots,a_i} * \psi^{F(a_1,\dots,a_i)}_{a_1,\dots,a_i}. \tag{6.26}$$

Оскільки $\psi_{a_1,\dots,a_i}^{F(a_1,\dots,a_i)} \le \rho_{a_1,\dots,a_i}$, то будемо мати наступну ситуацію:

$$\tau_F \le F_S(\alpha_1, \alpha_2, \dots, \alpha_n). \quad (6.27)$$

Далі, функцію розпізнавання (6.21) позначимо через F_0, причому функція F_0 задаватиметься наступним співвідношенням:

$$F_0(a_1, \dots, a_n) = l, \text{ якщо } \psi_{a_1,\dots,a_i}^{l} = \rho_{a_1,\dots,a_i} = \max_{0 \le j \le k-1} \psi_{a_1,\dots,a_i}^{j}. \quad (6.28)$$

Тоді з формул (6.26) та (6.28) безпосередньо будемо мати наступне:

$$\tau_{F_0} = \frac{m_{F_0}}{m} = \sum_{0 \le a_1,\dots,a_n \le 1} \delta_{a_1,\dots,a_i} * \psi_{a_1,\dots,a_i}^{F_0(a_1,\dots,a_i)} =$$

$$= \sum_{0 \le a_1,\dots,a_n \le 1} \delta_{a_1,\dots,a_i} * \rho_{a_1,\dots,a_i} = F(\alpha_1, \alpha_2, \dots, \alpha_n). \quad (6.29)$$

Отже, з формул (6.27) та (6.29) для всіх ФР $F(a_1, \dots, a_n)$ випливає наступне:

$$\tau_F \le \tau_{F_0} \quad (6.30)$$

Так, у даному розділі дослідження в підрозділі щодо концепції ЛДК було зафіксовано, що для довільного ЛДК система ознак (вершин, атрибутів) $\varphi_1, \varphi_2, \dots, \varphi_n$, яка фігурує в його структурі задає відображення: кожному об‘єкту $w, (w \in G)$ ставить у відповідність двійковий набір $r_1, r_2, \dots, r_n$, де $r_1 = \varphi_1(w), \dots, r_n = \varphi_n(w)$. Очевидно, що це відображення приймає тільки скінчене число значень, не більше ніж 2^n. З постанови задачі розпізнавання відомо, що довільна функція $f(w)$, яка задана на G та приймає скінчене число значень, буде називатися ознакою об‘єкта w. Таким чином, довільна система ознак $\varphi_1, \varphi_2, \dots, \varphi_n$ задає деяку узагальнену ознаку (а фактично – алгоритм класифікації) на множині G. Тобто набір ознак $\varphi_1, \varphi_2, \dots, \varphi_n$ буде набором елементарних ознак – ознак у вигляді предикатів, які мають детерміновану природу – приймають значення з множини $\{0,1\}$. Головна вимога до набору ознак $\varphi_1, \varphi_2, \dots, \varphi_n$ – це їх простота (звичайно для структур ЛДК).

Як вже наголошувалося в роботі [161], довільна ознака, яка отримана за певною схемою з фіксованого набору елементарних ознак $\varphi_1, \varphi_2, \dots, \varphi_n$, називається узагальненою ознакою. Тобто під терміном узагальнена ознака розуміється довільна детермінована ознака, на яку не накладається вимога

простоти. З іншого боку, слід зауважити, що довільна вершина в структурі АДК (по аналогії з елементарною ознакою в структурі ЛДК) у вигляді алгоритму класифікації a_i буде в деякому сенсі частковим випадком узагальненої ознаки (саме в сенсі визначення узагальненої ознаки роботи [208]).

Отже, оцінка якості (ефективності) набору алгоритмів $W(a_1, a_2, \dots, a_n)$ є ефективністю розпізнавання деякої побудованої структури АДК, яке задається набором алгоритмів $a_1, a_2, \dots, a_n$, а величину W можна визначити для довільного алгоритму розпізнавання в структурі дерева класифікації.

Отже зважаючи на все вище сказане можна зафіксувати наступне:

1) Головна ідея методів алгоритмічного дерева класифікації полягає в покроковій апроксимації масиву початкових даних НВ набором відібраних та оцінених алгоритмів класифікації.

2) Зауважимо, що ФР вигляду (6.22) структури АДК є найефективнішим з усіх правил класифікації, які покривають дані початкової НВ на основі набору алгоритмів класифікації $\alpha_1, \alpha_2, \dots, \alpha_n$.

3) Принципово важливим питанням концепції методів АДК залишається задача оцінки якості (ефективності, інформативності) набору алгоритмів класифікації (вершин структури дерева класифікації), а, отже, відбору критерію розгалуження в конструкції АДК.

4) Величина $F_S(\alpha_1, \alpha_2, \dots, \alpha_n)$ представляє собою найбільшу ефективність розпізнавання даних початкової НВ за допомогою фіксованого набору незалежних алгоритмів класифікації $\alpha_1, \alpha_2, \dots, \alpha_n$.

5) Величину $F_S(\alpha_1, \alpha_2, \dots, \alpha_n)$ для методів АДК можна інтерпретувати як частку всіх правильно класифікованих членів (навчальних пар) початкової НВ за допомогою ФР (6.22), причому жодна інша ФР (при заданому деякому наборі алгоритмів $\alpha_1, \alpha_2, \dots, \alpha_n$) не може збільшити цієї частки.

6.4. Схема методу побудови моделі АДК на основі апроксимації НВ набором автономних алгоритмів класифікації

На наступному етапі дослідження запропонуємо загальну схему побудови алгоритмічного дерева класифікації на основі апроксимації фіксованим набором автономних, різнотипних алгоритмів класифікації масиву даних початкової НВ. Відмітимо, що в даному пункті буде запропоновано схеми алгоритмічних дерев різних типів (у роботі введено АДК двох типів, причому сама концепція апроксимації набором незалежних алгоритмів масиву НВ не обмежується лише даними типами дерев класифікації).

Нехай задано деяку НВ загального типу (6.15) у вигляді послідовності навчальних пар $\left(x_i, f_R(x_i)\right)$, потужністю – M, розмірністю ознакового простору – n, та фіксований набір різнотипних алгоритмів класифікації $(\alpha_1, \alpha_2, \dots, \alpha_m)$. Відмітимо, що робота побудованих моделей дерев класифікації перевіряється на масиві даних ТВ потужністю – T (класова належність яких також відома).

Зауважимо, що дані початкової НВ задають деяке розбиття R на класи $(H_1, H_2, \dots, H_k)$, а відповідні алгоритми α_i можуть бути не зв‘язані єдиною концепцією розпізнавання, а реалізовувати різноманітні методи та алгоритми класифікації (для прикладу це можуть бути звичайні геометричні алгоритми, принцип роботи яких полягає в апроксимації навчальної вибірки відповідними геометричними об‘єктами [91, 229], алгоритми обчислення оцінок [55], потенціальних функції тощо).

6.4.1. Структура моделі АДК першого типу

Відмітимо, що результатом роботи кожного із зафіксованих (відібраних із бібліотеки алгоритмів деякої інформаційної системи) автономних алгоритмів класифікації та розпізнавання a_i, на відповідному кроці генерації АДК, є одна або декілька узагальнених ознак – f_j (певних правил класифікації НВ), які і описують (апроксимують) визначену частину початкової навчальної вибірки. Так, для випадку відомих геометричних алгоритмів розпізнавання

[91] – відповідними результуючими узагальненими ознаками будуть геометричні об‘єкти, які покривають НВ в ознаковому просторі задачі розмірності n.

Зрозуміло, що в реальних прикладних задачах можливі випадки, коли відповідний алгоритм класифікації a_i не може побудувати узагальнену ознаку f_j – в зв‘язку зі складним розташуванням класів H_k в ознаковому просторі задачі або певними концептуальними та реалізаційними обмеженнями самого алгоритму класифікації. Тоді, по аналогії з ЛДК можливий випадок, коли побудовані алгоритмом класифікації a_i (побудовані узагальнені ознаки f_j) неповністю апроксимують початкову НВ, або така ситуація передбачена самою схемою алгоритму генерації АДК (як приклад, наявність початкового обмеження в схемі алгоритму дерева класифікації – про генерацію не більше однієї узагальнені ознаки f_j на кожному етапі побудови моделі АДК).

Зауважимо, що об‘єкти початкової НВ які не підпадають під побудовану схему апроксимації вибірки послідовністю узагальнених ознак f_j (на останньому етапі процедури синтезу АДК) відносяться до відмов (помилок) класифікації першого типу – En_{tr} і аналогічно для даних ТВ неправильно класифіковані дискретні об‘єкти – також відносять до помилок першого типу – Et_{tr}.

Отже, можна зробити припущення, що структура АДК (типу І) буде мати загальну конструкцію (рис. 6.5), де кожний ярус такого дерева класифікації визначає етап побудови АДК у вигляді апроксимації поточним алгоритмом класифікації a_i певної частини НВ, та завдяки такому підходу дозволяє регулювати фінальну складність (точність) отриманої моделі дерева класифікації.

Відмітимо, що на кожному кроці генерації моделі АДК (рис. 6.5) подається свій алгоритм α_i класифікації та відповідна НВ (або підмножина початкової НВ), причому початкова НВ у повному складі подається лише на першому кроці, далі з наступними етапами побудови дерева класифікації

потужність масиву даних НВ буде падати за рахунок набору побудованих УО f_j, які будуть відрізати (описувати) певну частину даних початкової НВ. Також важливо зауважити, що залежно від структури схеми побудови АДК та особливостей поточного алгоритму α_i на кожному кроці можливо генерувати більше однієї УО f_j.

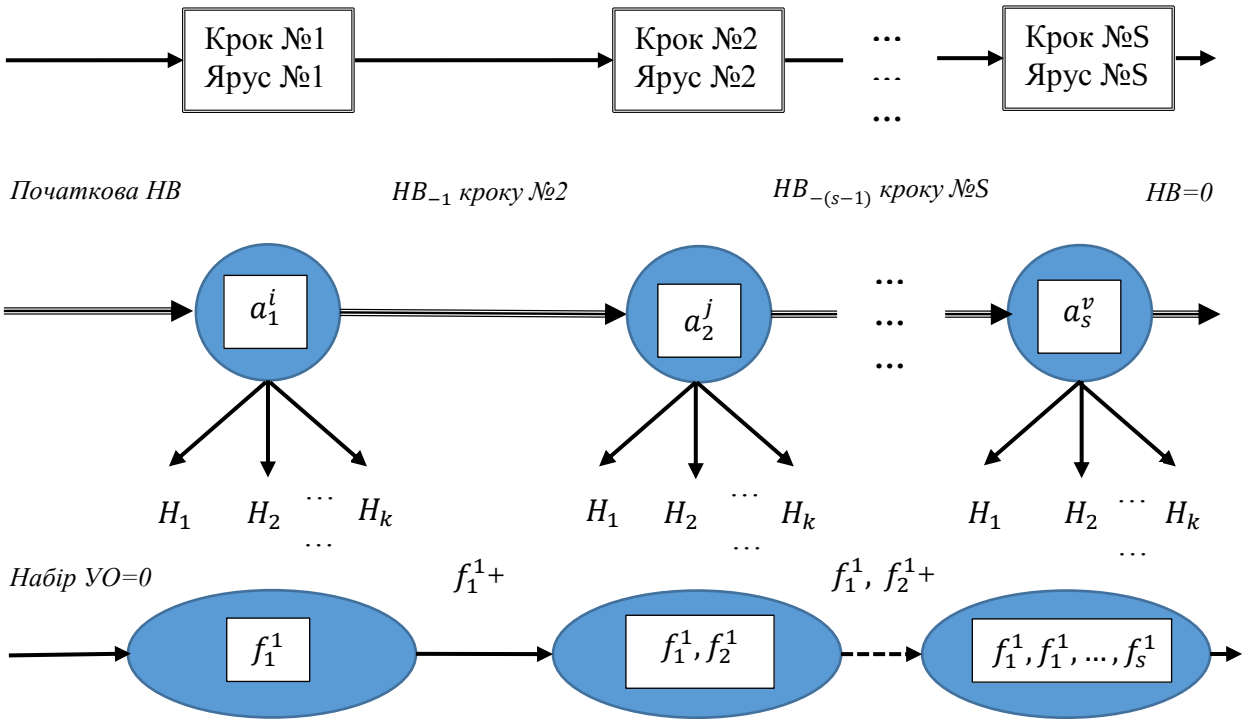


Рис. 6.5. Загальна схему структури АДК типу І.

На наступному етапі дослідження для методу АДК введемо два базові критерії побудови моделі дерева класифікації – критерій зупинки процедури розгалуження K_{stop} (він фактично регулює складність та точність отриманої моделі АДК) та критерій відбору розгалуження $W(a)$ (вибору алгоритму класифікації на поточному кроці) для дерева класифікації, що будується.

Отже, виходячи з вище зазначеного, доцільно ввести критерій зупинки K_{stop} процесу розгалуження типу $(boolean)$ процедури побудови АДК, який полягає в перевірці потужності $P_{pt}(HB)$ навчальної вибірки наступного вигляду:

$$K_{stop} = \begin{cases} 0, if\ P_{pt}(HB) = 0 \\ 1, if\ P_{pt}(HB) > 0 \end{cases}. \quad (6.31)$$

Відмітимо, про процедура побудови дерева класифікації продовжується

до тих пір, поки $K_{stop} = 1$, а протилежна ситуація – коли $K_{stop} = 0$ – сигналізує про завершення етапу синтезу моделі АДК та необхідність переходу до кроку тестової перевірки за даними ТВ та оцінки якості отриманої моделі дерева класифікації.

Відмітимо, що в методах АДК стає принципове питання вибору критерію розгалуження (оцінки та вибору алгоритму класифікації a_i для поточного кроку) в структурі моделі дерева класифікації, що будується. Зрозуміло, що за аналогією з методом апроксимації НВ набором ранжованих елементарних ознак в якості критерію розгалуження можна запропонувати початкову оцінку ефективності набору алгоритмів $(\alpha_1, \alpha_2, \dots, \alpha_i)$ у наступному вигляді:

$$W(a_i) = \frac{\frac{1}{P_{pt}(HB)} * \sum_{j=1}^{k} (T_{\text{УЗ}} + S_{\text{УЗ}} + \frac{E_{\text{УЗ}}}{S_{\text{УЗ}}})}{k}. \quad (6.32)$$

Відмітимо, що в запропонованому функціоналі (6.32) введені величини мають наступну інтерпретацію:

1) k – загальна кількість класів поточної задачі, які задані розбиттям R даних початкової НВ;

2) $T_{\text{УЗ}}$ характеризує загальний час (апаратний час), який витрачається на побудову поточної УО f_j;

3) $E_{\text{УЗ}}$ – інформаційна ємність (структурна складність) побудованої УО f_j на поточному кроці генерації моделі АДК;

4) $S_{\text{УЗ}}$ – загальна кількість дискретних об‘єктів x_i початкової НВ, які узагальнює (описує) дана УО f_j;

5) $P_{pt}(HB)$ – потужність (об‘єм) початкової НВ (або її фіксованої частини для поточного кроку схеми побудови АДК).

Зауважимо, що в формулі (6.32) сумування ведеться по всіх класах, які задані масивом даних початкової НВ (хоча можуть бути і обмеження щодо сумування, обумовлені структурою (параметрами) самого алгоритму побудови дерева класифікації).

Важливим моментом у схемі побудови моделі АДК (рис 6.5) є те, що на

кожному кроці алгоритму дерева фактично будується своя фіксована (одна або декілька – залежно від структури самого алгоритму АДК) УО f_j, причому їх загальна кількість збільшується з кожним кроком алгоритму дерева класифікації, а саме: АДК із набором алгоритмів класифікації $(\alpha_1, \alpha_2, \dots, \alpha_m)$ породжує (реплікує) деревоподібну конструкцію – дерево узагальнених ознак (ДУО) з відповідними набором УО $(f_1, f_2, \dots, f_z)$.

Отже, зважаючи на все зазначене вище, можна запропонувати одну з можливих алгоритмічних схем побудови АДК (типу I).

6.4.2. Метод побудови АДК першого типу

Eman 1. На першому етапі побудови АДК фіксується в бібліотеці алгоритмів інформаційної системи (відбирається в інтерактивному режимі, випадковим чином або після процедури відповідної оцінки якості (ефективності) за даними початкової НВ) набір автономних алгоритмів класифікації та розпізнавання $(\alpha_1, \alpha_2, \dots, \alpha_m)$. Відмітимо, що відбираються як самі алгоритми класифікації, так і їх кількість (величина m) залежно від умов та аспектів прикладної задачі.

Eman 2. На цьому етапі побудови АДК відібраний набір алгоритмів класифікації $(\alpha_1, \alpha_2, \dots, \alpha_m)$ оцінюється та ранжується на основі функціоналу (6.32) за даними НВ у наборі відповідно до їх ефективності. Слід відмити, що за аналогією з ЛДК можливі два варіанти –залежно від алгоритмічної схеми побудови дерева класифікації:

a) Варіант, коли оцінка ефективності та ранжування набору алгоритмів класифікації $(\alpha_1, \alpha_2, \dots, \alpha_m)$ проводиться лише один раз на даному етапі, а далі на кожному кроці побудови АДК фіксується для апроксимації даних наступний алгоритм α_i початкової послідовності. Такий підхід значно економить апаратні ресурси інформаційної системи, але негативно впливає на складність отриманої моделі дерева класифікації.

b) Варіант, коли оцінка ефективності та ранжування набору алгоритмів класифікації $(\alpha_1, \alpha_2, \dots, \alpha_m)$ проводиться на кожному кроці

побудови АДК за відповідними даними підмножин (частин) початкової НВ із метою оцінки та виявлення найбільш якісного (ефективного) алгоритму класифікації для даної частини НВ (кроку генерації АДК). Такий підхід дає можливість за меншу кількість кроків завершити апроксимацію НВ та отримати більш економну конструкцію АДК порівняно з варіантом (а), однак потребує значно більших апаратних ресурсів інформаційної системи для другого етапу схеми побудови АДК та вимагає значної уваги і введення набору обмежень щодо початкового відбору послідовності алгоритмів класифікації $(\alpha_1, \alpha_2, \dots, \alpha_m)$.

Етап 3. На третьому етапі схеми побудови АДК фіксується, як початкова вершина АДК – алгоритм класифікації α_i найбільшої ефективності з відсортованого набору $(\alpha_1, \alpha_2, \dots, \alpha_m)$, та на вхід подається початкова НВ у вигляді послідовності навчальних пар. Алгоритм α_i забезпечує генерацію однієї або декількох УО f_i першого ярусу (кількість УО, які генеруються на кожному кроці, задається параметрами самого алгоритму побудови АДК), які апроксимують (класифікують) певну частину НВ.

Етап 4. На цьому етапі обирається в якості вершини другого ярусу наступний за ефективністю алгоритм класифікації α_i ранжованої послідовності $(\alpha_1, \alpha_2, \dots, \alpha_m)$ та повторюється процедура побудови УО третього етапу з тією різницею, що на вхід подається вже обмежена НВ без навчальних пар, які апроксимуються УО вершиною першого ярусу, і так далі. Отже, далі процедура побудови АДК буде зводитися по повторення даного етапу для наступних за ефективністю алгоритмів α_i послідовності $(\alpha_1, \alpha_2, \dots, \alpha_m)$, постійного відсікання частин НВ та перевірки критерію зупинки розгалуження (пуста НВ), якій фактично сигналізує про завершення процедури побудови моделі АДК та отримання на виході як дерева алгоритмів α_i класифікації, так і дерева узагальнених ознак f_i.

Відмітимо, що можливі й інші реалізаційні варіанти схеми побудови АДК першого типу, які відрізняються від запропонованої схеми варіаціями щодо кількості УО, які будуються на кожному кроці, критеріями та послідовністю

етапів оцінки якості алгоритмів класифікації, можливістю використання обмеженої кількості алгоритмів (навіть одного), можливістю забезпечувати апроксимацію кожного з класів НВ набором своїх відібраних алгоритмів, можливістю варіації критерію зупинки розгалуження в АДК.

Важливо підкреслити, що принциповою особливістю такої схеми побудови АДК є можливість регулювати точність моделі дерева класифікації, що будується, впродовж основної процедури конструювання АДК, причому є принципова можливість побудови моделі АДК з наперед заданою точністю відносно масиву даних початкової НВ. Така можливість досягається шляхом обмеження кількості кроків процедури генерації АДК, системою обмежень щодо інформаційної ємності, кількості та параметрів узагальнення (області НВ, що апроксимується) набору узагальнених ознак, які будуються на відповідних етапах конструювання результуючого дерева класифікації.

6.4.3. Структура моделі АДК другого типу

Зрозуміло, що алгоритмічне дерево першого типу не є єдиною можливою конструкцією (структурою) алгоритмів класифікації, які можна організувати у вигляді деревоподібної моделі розпізнавання. Далі запропонуємо ще одну схему організації набору алгоритмів класифікації та розпізнавання $(\alpha_1, \alpha_2, \dots, \alpha_m)$ у вигляді моделі АДК, яку назвемо алгоритмічним деревом класифікації другого типу.

У даному дослідженні раніше вже висловлювалася думка, що зафіксована сукупність автономних алгоритмів класифікації та розпізнавання $(\alpha_1, \alpha_2, \dots, \alpha_m)$ може виступити в якості набору первинних ознак (атрибутів) для довільного дискретного об‘єкту x_i деякої початкової НВ загального вигляду (6.15). Причому щодо фіксованого дискретного об‘єкта x_i початкової НВ, буде важливою для даної схеми АДК інформація про вигляд УО, яку будує поточний алгоритм класифікації, та інформація щодо загальної можливості розпізнавання даного дискретного об‘єкту (наявність відмови, неправильної класифікації, неможливість УО описати даний об‘єкт тощо).

Отже, нехай кожній навчальній парі $\left(x_i, f_R(x_i)\right)$ НВ відповідає своя навчальна пара наступного вигляду:

$$\left(x_i(\varphi(\alpha_1), \varphi(\alpha_2), \dots, \varphi(\alpha_m)), f_R(x_i)\right), \text{ де } \varphi\left(\alpha_j\right) \in \{0,1\}. \quad (6.33)$$

Причому $\varphi\left(\alpha_j\right) = 1$, якщо даний дискретний об‘єкт апроксимується деякою УО f_l, яка побудована алгоритмом класифікації α_j набору $(\alpha_1, \alpha_2, \dots, \alpha_m)$ на відповідному етапі генерації АДК. Аналогічно $\varphi\left(\alpha_j\right) = 0$, якщо для даного дискретного об‘єкта алгоритмом α_j не було побудовано відповідну УО (яка би забезпечувала його апроксимацію, класифікацію), також до цієї ситуації належать відмови та помилки класифікації (помилки першого та другого роду).

Під алгоритмічним деревом другого типу будемо розуміти деяку деревоподібну конструкцію, загальний вигляд представлено на рис 6.6, у вершинах якої розташовано відповідні мітки (алгоритми класифікації та розпізнавання α_j, а також набори УО, які вони генерують на певному кроці процедури побудови АДК). Відмітимо, що логічне дерево такої конструкції належить класу регулярних логічних дерев повної складності (дане логічне дерево буде еквіваленте логічній функції від чотирьох аргументів, аргументи якої приймають значення з множини $\{0,1\}$ – комплекс цих питань піднімався в попередніх розділах даного дослідження).

Зауважимо, що структура АДК (тип II) – рис. 6.6 – також складається з відповідних ярусів (на кожному з яких розташовано лише один відібраний алгоритм α_i з побудованою УО), нумерація яких співпадає з послідовністю ранжованого за показником ефективності набору алгоритмів класифікації $(\alpha_1, \alpha_2, \dots, \alpha_m)$ – тобто на першому рівні знаходиться алгоритм із найбільшим коефіцієнтом ефективності відносно НВ, на другому – з меншим і так далі. Причому така схема структури АДК на виході не фіксує клас належності об‘єкту класифікації, а остаточне рішення щодо віднесення до того чи іншого класу (на етапі тестування та роботи моделі АДК) приймається шляхом голосування після проходження об‘єктом усіх вузлів (вершин) даної логічної

структури. Важливим моментом у цій схемі АДК другого типу є те, що, хоча в прикладі (рис. 6.6) показано регулярне дерево класифікації, в якому на кожному ярусі розташовано однакові відібрані алгоритми класифікації a_i – воно генерує нерегулярне дерево УО, в якому на фіксованому ярусі логічної структури може розміщуватись різна кількість УО, які також можуть відрізнятися параметричною складністю та потужністю, тобто залежати від результатів попереднього кроку апроксимації даних НВ.

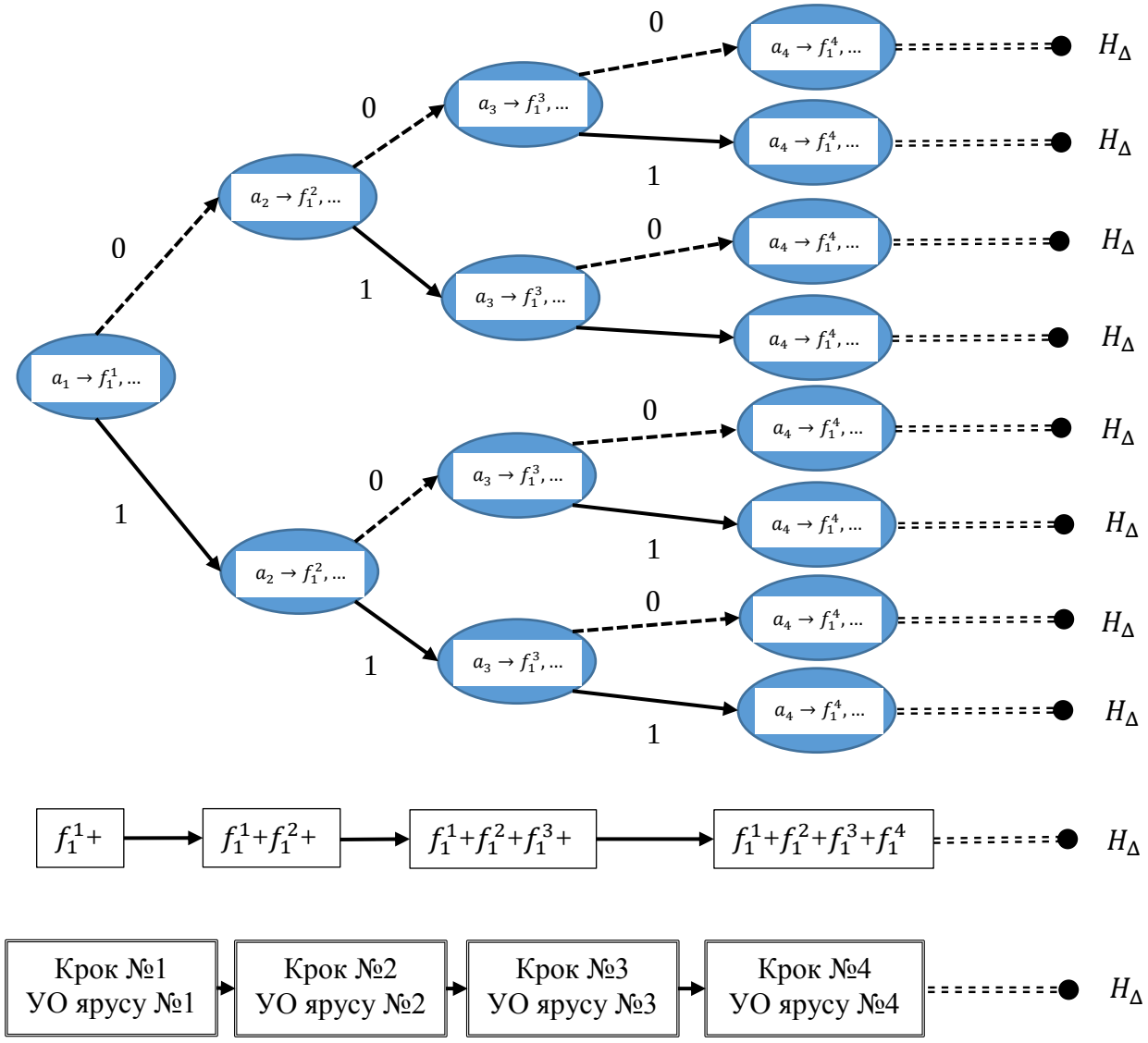


Рис. 6.6. Загальна схема структури АДК типу II.

Так, на кожній із вершин (вузлів) даної конструкції АДК, якщо відповідний алгоритм та його УО дозволяє забезпечити класифікацію (розпізнавання), то перед переходом до наступного ярусу (етапу) змінюється лічильник належності відповідному класу НВ, в іншому випадку перехід до

наступної вершини проходить без змін глобальних лічильників.

Отже, віднесення об‘єкту класифікації до того чи іншого класу НВ після проходження шляхом у структурі АДК здійснюється на основі перевірки та порівняння лічильників належності класам, причому їх нульове значення для всіх класів НВ означає відмову класифікації (помилку першого роду).

Підкреслимо, що як і в ситуації АДК (тип І) кількість УО, які генеруються на кроці схеми дерева класифікації, не обмежена і залежить від початкових параметрів ініціалізації алгоритму побудови АДК (тип ІІ), тому саме такий гнучкий підхід до вибору кількості УО при побудови дерева класифікації і дозволяє регулювати, як і в попередньому прикладі, фінальну ефективність (точність) моделі АДК або будувати дерево класифікації з наперед заданою точністю відносно початкових даних НВ.

Отже, зважаючи на все вищезазначене, та за аналогією до запропонованої схеми побудови АДК першого типу, можна запропонувати одну з можливих алгоритмічних схем генерації АДК другого типу.

6.4.4. Метод побудови АДК другого типу

Етап 1. На першому етапі побудови АДК відбирається на основі відповідного критерію ефективності (або в інтерактивному режимі, випадковим чином) набір автономних алгоритмів класифікації та розпізнавання $(\alpha_1, \alpha_2, \dots, \alpha_m)$ ранжованих відповідним чином. Зокрема відбираються як самі алгоритми класифікації, так і їх загальна кількість у наборі (величина m) залежно від умов та аспектів прикладної задачі. Саме цей етап має принципову важливість, оскільки безпосередньо впливає на фінальну складність отриманої моделі АДК.

Етап 2. На цьому етапі будується повне регулярне логічне дерево, де на кожному з ярусів даної структури розташовується відповідний алгоритм класифікації ранжованої послідовності $(\alpha_1, \alpha_2, \dots, \alpha_m)$. Відмітимо, що в даному логічному дереві кожна вершина має по два переходи на наступний ярус (два нащадки), які позначаються значенням із бінарної множини $\{0,1\}$. Оскільки маємо регулярне логічне дерево, то на кожному з ярусів даної

структури розташовано мітки (змінні) одного типу (порядку), що стосується лише самих алгоритмів класифікації a_i, а не УО f_j, які вони генерують.

Так, на другому етапі генерації АДК (тип II) послідовно на вхід алгоритмам класифікації a_i подається масив даних НВ (відповідно до структури побудованого дерева класифікації (рис. 6.6)) з метою отримання на виході набору відповідних УО f_j, причому їх загальна кількість у конструкції дерева та кількість для кожного алгоритму класифікації (кроку в схемі дерева, ярусу логічного дерева) залежить від початкових параметрів ініціалізації алгоритму побудови АДК (задаються в інтерактивному режимі або автоматично) та особливостей прикладної задачі, для якої будується модель АДК.

Після побудови набору всіх УО f_j для даної прикладної задачі, вони розташовуються у відповідних вершинах отриманого дерева класифікації з метою завершення процедури його побудови. Принциповим моментом даного етапу є те, що набір побудованих УО має перекривати весь масив даних НВ для забезпечення стовідсоткового розпізнавання початкових даних. Зокрема можуть бути певні відхилення, якщо будується модель АДК із наперед заданою точністю та складністю (це обмеження умов задачі може бути реалізовано за рахунок зміни кількості та потужності УО f_j, які будуються на другому етапі). Зауважимо, що дана умова може бути також реалізована за рахунок обмеження кроків (кількості ярусів конструкції) в процедурі побудови моделі АДК, додатковими обмеженнями кількості алгоритмів класифікації, які використовуються в структурі дерева класифікації.

Етап 3. На третьому етапі схеми побудови АДК після побудови основної конструкції дерева класифікації можна переходити безпосередньо до режиму тестування отриманої моделі АДК. Причому для кожного тестового об‘єкта, який подається на вхід дерева класифікації, обчислюються відповідні значення $\varphi(\alpha_j)$ (за допомогою набору побудованих раніше УО – для кожної вершини відповідного ярусу дерева), які забезпечують (визначають) відповідний

маршрут у структурі побудованого АДК другого типу. Так, УО кожної з вершин АДК – у випадку можливої апроксимації об‘єкта невідомої класифікації – інкрементують відповідний лічильник класу належності та залишають його без змін у разі відмови (неможливості) класифікації. На виході структури АДК об‘єкт невідомої класифікації відноситься до того класу, лічильник належності якого буде максимальним, у випадку їх нульової рівності маємо справу з відмовою класифікації.

Зауваження. Зі схеми побудови АДК другого типу, яка була представлена вище, можна бачити, що кількість УО (параметрична складність, потужність), які генеруються одним і тим самим відібраним алгоритмом класифікації α_j на деякому ярусі дерева класифікації для кожного із шляхів структури АДК може бути різною, причому, слідуючи в цьому напрямку, прийдемо до того, що конструкція моделі АДК не обов‘язково належати класу регулярних структур (логічних дерев), тобто в кожному ярусі конструкції АДК, що будується, разом із різною кількістю та типом (загальними параметрами) УО допускається наявність різних алгоритмів класифікації та розпізнавання α_j.

6.4.5. Етап експериментальної перевірки моделей АДК

Відмітимо, що запропоновані схеми побудови АДК дають змогу регулювати складність моделі дерева класифікації, що будується, будувати моделі з наперед заданою точністю, а сама структура дерева класифікації складається з різнотипних автономних алгоритмів класифікації як будівельних модулів (компонентів). Причому задача відбору моделі дерева класифікації серед набору побудованих АДК для конкретної задачі визначається набором параметрів, які мають визначальну важливість для поточної прикладної задачі (набору даних НВ).

Зрозуміло, що для порівняння та відбору конкретної моделі АДК з фіксованого набору, необхідно виділити найбільш важливі їх параметри (розмірність ознакового простору, кількість вершин, переходів, алгоритмів, тощо) та визначити їх похибку щодо масиву вхідних даних.

Принципово на даному етапі дослідження розглянути критерії якості отриманих інформаційних моделей, які залежать від похибки моделі, потужності початкового масиву даних НВ, об‘єму ТВ (кількості навчальних пар та розмірності ознакового простору задачі), кількості параметрів моделі тощо.

Зрозуміло, що критично важливими параметрами побудованої моделі АДК, які необхідно мінімізувати, є помилки моделі відповідно на масивах даних НВ, ТВ та для кожного з класів, які задані початковою умовою поточної прикладної задачі.

Зауважимо, що принциповим моментом залишається питання зменшення складності структури АДК (йдеться про кількість ознак, алгоритмів у структурі АДК, загальну кількість вершин моделі АДК та загальну кількість переходів у структурі АДК), параметрів загальних витрат пам‘яті та процесорного часу інформаційної системи. Так, важливим показником якості побудованої моделі у вигляді дерева класифікації з урахуванням параметрів структури моделі ЛДК є загальний інтегральний показник якості в наступній формі:

$$Q_{Main} = \frac{Fr_{All}}{V_{All} \cdot \sum_i p_i} \cdot e^{-\frac{Er_{All}}{M_{All}}}. \quad (6.34)$$

Відмітимо, що в формулі (6.34) набір параметрів p_i представляє собою найбільш важливі характеристики побудованого дерева класифікації, що оцінюється:

1) Er_{All} – загальна кількість помилок моделі АДК на масивах даних початкових тестової та навчальної вибірки;

2) M_{All} – загальна потужність (об‘єм) масивів даних навчальної та тестової вибірки;

3) Fr_{All} – кількість вершин отриманої моделі АДК із результуючими значеннями f_R (ФР, тобто листів дерева класифікації);

4) V_{All} – представляє загальну кількість усіх типів вершин у структурі моделі АДК;

5) O_{Uz} – загальна кількість узагальнених ознак, що використовуються в моделі дерева класифікації;

6) P_{All} – загальна кількість переходів між вершинами в структурі побудованої моделі дерева класифікації;

7) N_{Alg} – загальна кількість різних автономних алгоритмів класифікації a_i, що використовуються в моделі дерева класифікації.

Відмітимо, що даний інтегральний показник якості моделі АДК буде приймати значення від нуля до одиниці. Чим меншим він буде, тим гіршою буде якість побудованого дерева класифікації, а чим більшим буде показник, тим кращою буде отримана модель.

Далі розглянемо приклад побудови моделі АДК із такими початковими параметрами:

1) Фіксований набір різнотипних алгоритмів класифікації та розпізнавання $(\alpha_1, \alpha_2, \dots, \alpha_5)$, $(m = 5)$.

2) У масиві початкової НВ задано інформацію про розбиття R на класи, що не перетинаються – $(H_1, H_2, \dots, H_4)$, $(k = 4)$.

3) Початкова НВ вигляду (6.15) має потужність у 2000 навчальних пар (об‘єктів відомої класифікації), $(M = 2000)$.

4) Кожний із дискретних об‘єктів НВ x_i характеризуються набором ознак, атрибутів – $(x_i^1, x_i^2, \dots, x_i^{20})$, $(n = 20)$.

5) Для перевірки отриманої моделі АДК задано ТВ потужністю 500 елементів, $(T = 500)$.

Представлена в даному прикладі початкова НВ містить дані компонентного хімічного аналізу вмісту дизельного (вуглеводного) пального (задача оцінки якості пального) в спрощеному варіанті (кількість класів НВ задачі зменшена до чотирьох, розмірність ознакового простору з 38 до 20, а кількість алгоритмів класифікації також обмежена на початковому етапі шляхом відбору лише геометричних класифікаторів) заради демонстрації самої концепції алгоритмічного дерева.

Зауважимо, що в даному випадку тестова вибірка є фактично відокремленою частиною НВ з об‘єктами відомої класифікації, тобто оцінювати ефективність побудованої системи можна також на основі даних ТВ, крім процедури апроксимації даних набором узагальнених ознак (геометричних алгоритмів) та розрахунку інтегрального показника якості побудованої моделі АДК.

На першому етапі процедури побудови дерева класифікації оцінимо ефективність кожного з відібраних алгоритмів класифікації, на основі яких і буде побудовано загальну схему класифікації (модель АДК) стосовно даних початкової навчальної вибірки (за кількістю узагальнених ознак, що генеруються поточним алгоритмом та відмовам класифікації).

Таблиця 6.2. Оцінка ефективність фіксованих алгоритмів класифікації дискретних об‘єктів відносно початкового начальної вибірки.

(Номер класу/Тип алгоритму)	***Алгоритм*** a_1	***Алгоритм*** a_2	***Алгоритм*** a_3	***Алгоритм*** a_4	***Алгоритм*** a_5
Клас H_1	0/32	0/12	0/11	0/9	18/10
Клас H_2	0/16	12/17	1/16	14/6	12/8
Клас H_3	0/8	0/10	0/17	0/12	0/10
Клас H_4	0/11	15/3	9/16	16/6	14/11

У табл. 6.2 представлено ефективність кожного з відібраних для задач алгоритмів класифікації відносно класів початкової навчальної вибірки, причому перше число відповідає за кількість об‘єктів, яким відмовлено у класифікації відповідним алгоритмом (помилкам, відмовам класифікації), а друге – за кількість узагальнених ознак (для даного типу алгоритмів – геометричних об‘єктів), якими апроксимований відповідний клас початкової вибірки. Залежно від початкового вибору алгоритму як вершини дерева класифікації (моделі АДК), процедура побудови результуючої схеми класифікації може завершитися з різною кількістю кроків. Одну з можливих побудованих схем АДК представлено на рис. 6.7.

Так, з табл. 6.2 можна бачити, що ефективність усіх алгоритмів, за виключенням алгоритму a_5 (геометричного алгоритму гіперплощин) відносно

класу H_1 становить сто відсотків, тому для його апроксимації можна застосувати довільний алгоритм (зрозуміло, за виключенням a_5). На всіх наступних етапах побудови схеми розпізнавання (ярусах структури АДК) доцільно знову зафіксувати алгоритм a_1 (геометричний алгоритм гіперсфер), який виявився найбільш ефективним та економним відносно всіх інших класів даних початкової вибірки, зокрема його особливістю є велика універсальність в плані можливості побудови узагальненої ознаки, навіть в тих випадках, коли інші геометричні алгоритми цього зробити не можуть і дають великий відсоток відмов (помилок) класифікації для об‘єктів начальної вибірки (випадок складного, заплутаного розташування класів в ознаковому просторі задачі).

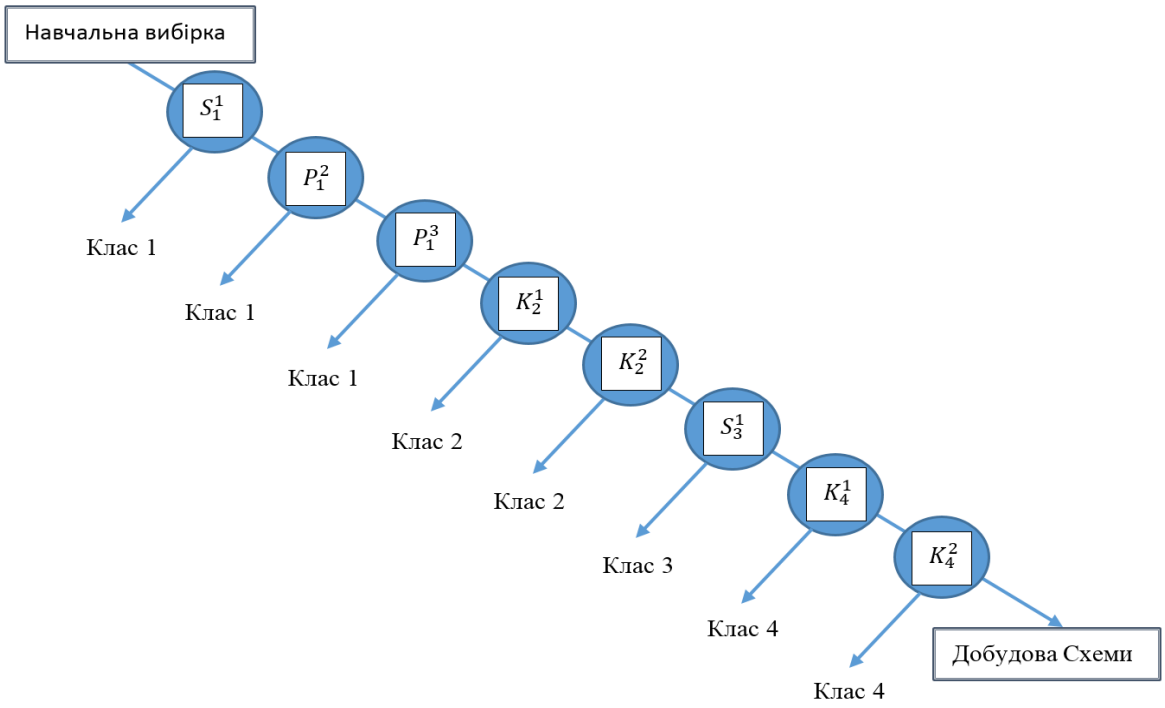


Рис. 6.7. Приклад сконструйованої моделі АДК.

Важливим моментом, також є той факт, що кожна з узагальнених ознак, згенерованих алгоритмом класифікації a_1, представляє собою набір координат центру гіперсфери (в ознаковому просторі задачі) і її радіус та потребує мінімального об‘єму пам‘яті інформаційної системи для свого зберігання, відповідно простих механізмів роботи з набором таких УО [161].

Відмітимо, що модель дерева класифікації (рис. 6.7) побудовано алгоритмами, ефективність яких оцінювалася щодо кількості узагальнених ознак (якими апроксимується початкова навчальна вибірка), причому для повної апроксимації масиву НВ даної задачі було достатньо лише трьох алгоритмів класифікації. Так, для апроксимації класу H_1 (з першого по третій крок) було застосовано два алгоритми – спочатку алгоритмом a_5 (геометричний алгоритм гіперплощин) побудовано узагальнену ознаку S_1^1, яка лише частково його описувала. На другому етапі застосовано алгоритм a_4 (геометричний алгоритм гіперпаралелепіпедів) – одержано ознаки P_1^2 та P_1^3, які остаточно і завершили розпізнавання (апроксимацію) даного класу H_1.

На наступних етапах побудови моделі АДК знову застосовано алгоритм класифікації a_1 (алгоритм гіперсфер) (ознаки $K_1^2, K_2^2, K_4^1, K_4^2$) та алгоритм a_5 (узагальнена ознака S_3^1).

Зокрема слід звернути увагу, що для побудови даної схеми (моделі) класифікації (рис. 6.7) застосовано три різні алгоритми розпізнавання (з п‘ятьох початково відібраних), які безпосередньо не впливають на роботу один одного – тобто на їх місці могли знаходитися зовсім різні за принципом та ідеологією алгоритми класифікації, з яких можна сконструювати схему розпізнавання (модель АДК) довільної складності та ефективності. Важливим є лише ефективність кожного з них на фіксованій вибірці та інформаційна ємність узагальнених ознак, генерованих ними.

Підкреслимо, що метод алгоритмічного дерева оперує лише вже готовими (побудованими) узагальненими ознаками, і його може зовсім не цікавити, яким алгоритмом чи способом (правилом, методом) вони отримані. Причому в схемі АДК (рис. 6.7) показано покрокову схему генерації не одиничних, а цілих наборів УО K_i^j, S_i^j, P_i^j, де $j-$ номер УО для відповідного класу апроксимації (номер етапу генерації АДК відносно класу), а $i-$ номер кроку процедури побудови дерева (поточного класу, який апроксимується).

Зрозуміло, що дану схему розпізнавання (модель АДК) сконструйовано

на основі методу логічного дерева (ЛДК), та може бути представлено як певну алгоритмічну схему (оператор), яку побудовано деяким алгоритмом мінімізації або максимізації відповідного функціонала, на основі якого оцінюється важливість ознаки, групи ознак або ефективність автономного алгоритму класифікації, однозначно пов‘язаного з помилками класифікації (перетинається з методами логічних дерев класифікації).

Зауважимо, що метод алгоритмічного дерева на основі вхідних даних (даних навчаючої вибірки) та асортименту (набору) алгоритмів формування узагальнених ознак, які зберігаються в його бібліотеці, конструює (генерує) оптимальну за витратами пам‘яті (складності) та ефективності розпізнавання (систему) певну схему (дерево класифікації). Під схемою в даному випадку будемо розуміти набір числових параметрів для набору УО K_i^j, S_i^j, P_i^j, які найкращим чином апроксимують масив початкових даних. Так, в нашому випадку, аргументи сконструйованої схеми розпізнавання – ознаки класів (гіперсфери, гіперпаралелепіпеди та інші) або міжкласові ознаки (гіперплощини). Параметри вказаних ознак та загальна структура АДК (схеми класифікації) зберігаються в пам‘яті комп‘ютера (інформаційної системи).

Кожна із сконструйованих схем за методом алгоритмічного дерева буде являти собою загальну систему розпізнавання (модель АДК), яку можна застосовувати для практичної роботи (обробки великих масивів експериментальних даних у вигляді дискретних наборів). Зауважимо що отримана схема буде в певній мірі новим алгоритмом розпізнавання (зрозуміло, синтезованим із відомих алгоритмів та методів). Крім того, для роботи даних отриманої моделі АДК немає необхідності зберігати в пам‘яті комп‘ютера об‘єкти вибірки, за якими вона була сконструйована, тобто великі інформаційні масиви даних, останнє, в свою чергу, веде до того, що процес побудови системи розпізнавання на основі методів дерев класифікації (ЛДК/АДК) в значній мірі схожий із процесом стиснення (йдеться про методи стиснення інформації з втратами) або кодуванням інформації (опис та кодування складних структур даних).

На наступному етапі дослідження проведемо оцінку та порівняння побудованих моделей дерев класифікації (ЛДК та АДК) для представленої вище прикладної задачі класифікації. Так, фрагмент основних результатів приведених вище експериментів (порівняльних тестів методів побудови моделей ЛДК/АДК на масиві даних даної прикладної задачі) представлено в табл. 6.3, причому синтезовані моделі різнотипних дерев класифікації забезпечили необхідний рівень точності, заданий умовою задачі, швидкодію та витрати робочої пам‘яті інформаційної системи, але показували різну структурну складність побудованих дерев класифікації (моделей) та набору узагальнених ознак (у випадку моделі алгоритмічного дерева класифікації).

Зауважимо, що запропонована в даному дослідженні оцінка якості моделі дерева класифікації (АДК) відображає базові параметри (характеристики) дерев класифікації та може бути застосована в якості критерію оптимальності в процедурі оцінки довільної деревоподібної схеми розпізнавання, наприклад у випадку застосування методів побудови та відбору випадкових ЛДК з роботи [112] (з урахуванням їх відповідних структурних параметрів).

Таблиця 6.3 Порівняльна таблиця моделей / методів дерев класифікації.

№	Метод синтезу структури логічного дерева	Інтегральний показник якості моделі Q_{Main}	Загальна кількість помилок моделі на НВ та ТВ Er_{All}
1	Метод повного ЛДК на основі селекції елементарних ознак	0,004789	2
2	Модель ЛДК з одноразовою оцінкою важливості ознак	0,002263	3
3	Обмежений метод побудови ЛДК	0,003181	4
4	Метод алгоритмічного дерева (типу I)	0,005234	0
5	Метод алгоритмічного дерева (типу II)	0,002941	0

Так, запропонований в даному дослідженні метод алгоритмічного дерева класифікації (методи АДК першого та другого типу) порівнювалися з методом повного ЛДК та обмеженого методу селекції елементарних ознак та показав у цілому прийнятний результат.

Практична цінність отриманих результатів полягає в тому, що запропонований метод побудови моделей АДК (першого та другого типу) дає можливість будувати економні та ефективні моделі класифікації заданої точності (даний метод був реалізований у бібліотеці алгоритмів системі "ОРІОН ІІІ" для розв'язку різноманітних прикладних задач класифікації), які характеризуються великим ступенем універсальності відносно широкого кола прикладних задач. Відмітимо, що практичні застосування підтвердили працездатність побудованих моделей дерев класифікації та розробленого програмного забезпечення. В якості перспективи, майбутні дослідження можуть бути спрямовані в бік подальшого розвитку методів АДК (введення нових типів, схем дерев класифікації), оптимізації програмних реалізацій запропонованого методу АДК, а також його практичної апробації на множині реальних задач класифікації та розпізнавання.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти.

1) Використання концепції АДК при побудові моделей дерев класифікації дозволяє досягти хороших результатів щодо розширення кола прикладних задач застосування (вимога універсальності), можливості побудови моделей класифікації, точність яких можна регулювати в процесі побудови (або будувати системи з наперед заданою точністю), можливості раціонального використання вже накопиченого потенціалу методів та алгоритмів розпізнавання та класифікації.

2) Концепція методів АДК дозволяє будувати моделі дерев класифікації різних типів, причому всі вони базуються на простій ідеї апроксимації навчальної вибірки набором автономних алгоритмів класифікації та представлення отриманої моделі у вигляді деякої

деревоподібної схеми.

3) Важливим моментом є те, що отримана модель АДК із різних алгоритмів та методів класифікації в свою чергу буде представляти собою новий алгоритм розпізнавання, тобто концепція АДК – це метод синтезу нових алгоритмів класифікації заданої точності відносно НВ на основі набору вже відомих.

4) Важливою особливістю даного підходу побудови дерева класифікації є те, що, навіть у випадку надвеликих початкових НВ, отримана структура АДК буде характеризуватися відносно невеликою складністю конструкції (компактністю), що дозволяє ефективно аналізувати (виділяти правила класифікації в конструкції, схемі АДК) та процедурно робити з такою структурою даних.

5) Зауважимо, що для даного метода побудови дерева класифікації є можливість уведення критерію зупинки розгалуження (не обов‘язково пов‘язаного з перевіркою потужності початкової НВ), що дозволяє в прикладній задачі керувати складністю (точністю) моделі АДК, що будується, або задати точність моделі АДК наперед.

6) Відмітимо, що апроксимація даних початкової НВ набором відібраних та оцінених незалежних алгоритмів класифікації та розпізнавання (оцінених відносно даних НВ) дає відносно компактні та ефективні структури моделей АДК (моделей ДУО, які вони фактично породжують).

6.5. Загальна схема побудови моделей дерев класифікації на основі обмежених методів ЛДК та АДК

На даному етапі дослідження розглянемо обмежені методи структур ЛДК та АДК, які дозволяють подолати певні негативні концептуальні моменти та обмеження, які притаманні даним схемам з точки зору ресурсних потреб та результуючої складності побудованих моделей дерев класифікації.

Так, спочатку звернемо увагу, що в загальній схемі методу побудови моделі ЛДК (на основі покрокової селекції елементарних ознак), яка була описана в цьому розділі дослідження раніше, є принциповий недолік, який пов‘язаний з тим, що зі зростанням кількості вершин (ярусів структури дерева класифікації) в конструкції ЛДК кількість елементарних ознак φ_i^j (де $i-$ номер елементарної ознаки в наборі, $j-$ номер ярусу розташування відповідної ознаки) в дереві значно збільшується. Звичайно таке ускладнення результуючої моделі ЛДК (конструкційної складності) негативно впливає на апаратні можливості системи класифікації (пам‘ять, процесорний час) та загальну можливість сприйняття і аналізу побудованої моделі без зовнішнього виділення правил класифікації в структурі дерева. Для того, щоб подолати ці принципово негативні моменти методу дерева класифікації запропонуємо наступну модифікацію методу ЛДК.

6.5.1. Обмежений метод побудови моделей ЛДК

На початковому етапі зафіксуємо деяке додатне число Z. Нехай маємо побудоване ЛДК (дерево класифікації після визначеної кількості кроків побудови моделі ЛДК) загальної структури (рис. 6.8), яке відображає деякий предикат (побудовану узагальнену ознаку) $p_1(x)$.

Звернемо увагу, що раніше при представленні методу побудови ЛДК (на основі селекції елементарних ознак) на етапі тесту обчислювали деяке число S, яке фігурує у співвідношенні $\frac{S}{M} \geq \delta$. Тепер, крім числа S, для кожного незакінченого шляху $r_1 r_2 r_3$ в структурі логічного дерева (рис. 6.8) розраховуємо ще число $S_{r_1 r_2 r_3}$, де $S_{r_1 r_2 r_3}-$ кількість усіх пар $(x_i, f_R(x_i))$ з НВ,

які фактично належать шляху $r_1r_2r_3$ і для яких виконується співвідношення $f_R(x_i) \neq l(r_1r_2r_3)$. Отже, $S_{r_1r_2r_3}$ – це число всіх тих помилок, які породжуються деяким предикатом $p_1(x)$ (узагальненою ознакою), яке представляється даним ЛДК загальної структури (рис. 6.8) на фіксованому шляху $r_1r_2r_3$ в конструкції даного дерева.

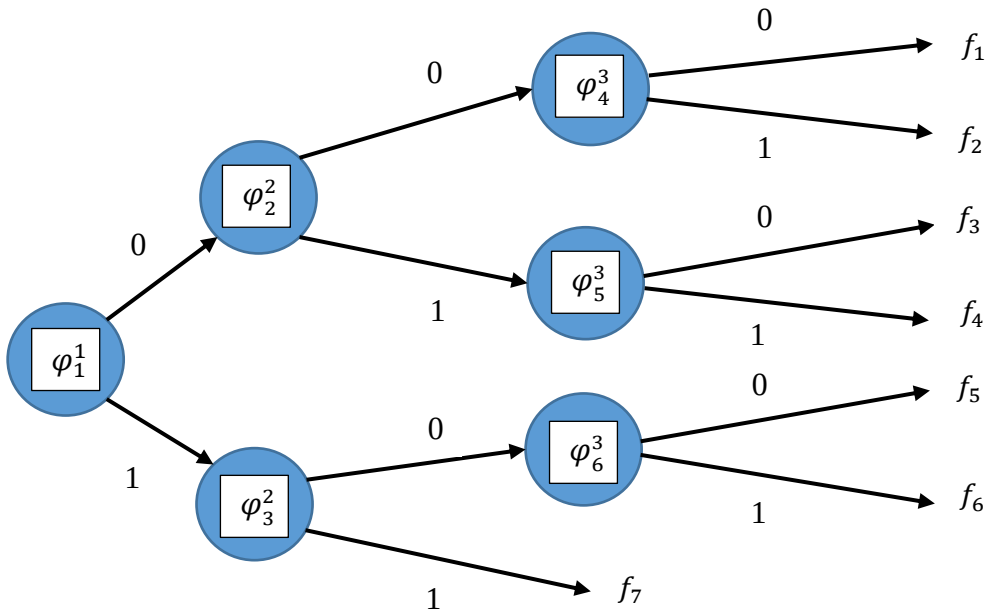


Рис. 6.8. Чотириярусне ЛДК, побудоване за даними початкової НВ.

На наступному етапі вибираємо число Z таких шляхів $(r_1r_2r_3)_1, .., (r_1r_2r_3)_Z$, для яких число $S_{r_1r_2r_3}$ буде найбільшим.

Приклад 1. Нехай $Z = 3$ та має місце наступне співвідношення:

$$S_{000} \geq S_{100} \geq S_{101} \geq S_{001} \geq S_{010} \geq S_{011}.$$

Тоді вибираються тільки шляхи $000, 100, 101$. Наступна добудова – відбір вершин (елементарних ознак) $\varphi_{r_1r_2r_3}$ – здійснюється тільки для цих шляхів.

Зафіксуємо, що дану модифіковану схему побудови дерева класифікації (ЛДК на основі селекції елементарних ознак) будемо називати обмеженим методом побудови ЛДК.

Відмітимо, що за даною схемою в процесі побудови дерева класифікації продовжуються тільки ті шляхи (загальної структури ЛДК), за якими виникає найбільша кількість помилок класифікації.

Зокрема при застосуванні тільки що вказаного процесу в кінці шляхів $r_1 r_2 r_3$, які не входять у відібрані Z шляхів, значення $l(r_1 r_2 r_3)$ зберігаються, причому процес модифікованого методу побудови ЛДК можна застосовувати в тому випадку, коли початкова НВ не є фіксованою – тобто коли на кожному кроці процедури побудови дерева класифікації подається своя вибірка (частина НВ).

Отже, запропонована вище схема побудови дерева класифікації забезпечує можливість фактично запровадити механізм регулювання точності моделі дерева, яка будується, враховуючи загальну кількість помилок класифікації на тому чи іншому шляху (етапі побудови) загальної структури логічного дерева.

Зрозуміло, що ця ідея може працювати і на рівні структури АДК, причому із урахуванням певних її особливостей. Отже, зважаючи на все вищезазначене, запропонуємо наступну модифікацію методу побудови структури АДК першого типу, який був представлений в даному дослідженні вище.

6.5.2. Обмежений метод побудови моделей АДК

Звернемо увагу, що АДК першого типу складається з ярусів, кожний з яких фактично відповідає певному кроку (етапу) побудови (апроксимації даних початкової НВ) дерева класифікації. Причому для кожного алгоритму класифікації на тому чи іншому кроці апроксимації можна розрахувати його ефективність щодо робочих даних – $(S/P_{pt}(HB^-))$. Ця величина має бути більшою (або рівною) ніж задане на початку обмеження δ (зрозуміло, що в деяких реалізаціях схеми АДК величина δ може бути використана в якості критерію зупинки процедури розгалуження в структурі дерева класифікації).

Зауважимо, що $S-$ загальна кількість помилок класифікації для фіксованого алгоритму на певному кроці (етапі генерації АДК), а $P_{pt}(HB^-)-$ потужність (об‘єм) підмножини початкової НВ, яка подається на вхід даного алгоритму на відповідному ярусі (рівні або етапі) дерева класифікації, що будується.

Тоді доцільно за аналогією з обмеженим методом ЛДК для кожного ярусу структури АДК (етапу побудови дерева класифікації) розраховувати величини $S_{a_1,\dots,a_i}$, які характеризують кількість усіх пар $(x_i, f_R(x_i))$ з масиву початкової НВ, які не можуть бути апроксимовані послідовністю фіксованих алгоритмів класифікації $a_1, \dots, a_i$. Отже, $S_{a_1,\dots,a_i}$ – це число всіх тих помилок класифікації, які здійснюються деякою послідовністю УО (побудованих на відповідних рівнях структури АДК), яке представляється АДК загальної структури (рис. 6.9) для фіксованого набору алгоритмів розпізнавання та класифікації $a_1, \dots, a_i$.

Тоді за даною схемою побудови структури АДК вибирається лише той шлях або певні шляхи в конструкції дерева (залежно від типів дерев класифікації, які можуть бути різними) для якого величина $S_{a_1,\dots,a_i}$ буде по можливості максимальною – тобто добудовується шлях у структурі АДК із найбільшою кількістю помилок класифікації. Причому наступна добудова та відбір вершин (алгоритмів класифікації – параметрів УО) $a_1, \dots, a_i$ здійснюється тільки для цих шляхів.

Зауважимо, що при такому підході в побудові обмежених методів ЛДК та АДК після побудови моделі дерева класифікації можлива його фактична добудова (донавчання), що дає пряму можливість впливу на точність побудованої моделі системи класифікації.

Відмітимо, що за обмеженою схемою АДК у процесі побудови дерева класифікації продовжуються (вибираються для добудови УО) тільки ті шляхи (загальної структури АДК), за якими виникає найбільша кількість помилок класифікації. Зрозуміло, що при застосуванні тільки що вказаного процесу побудови структури АДК у будь-який момент є можливість повернення до відкинутих шляхів для добудови структур (шляхів) дерева класифікації.

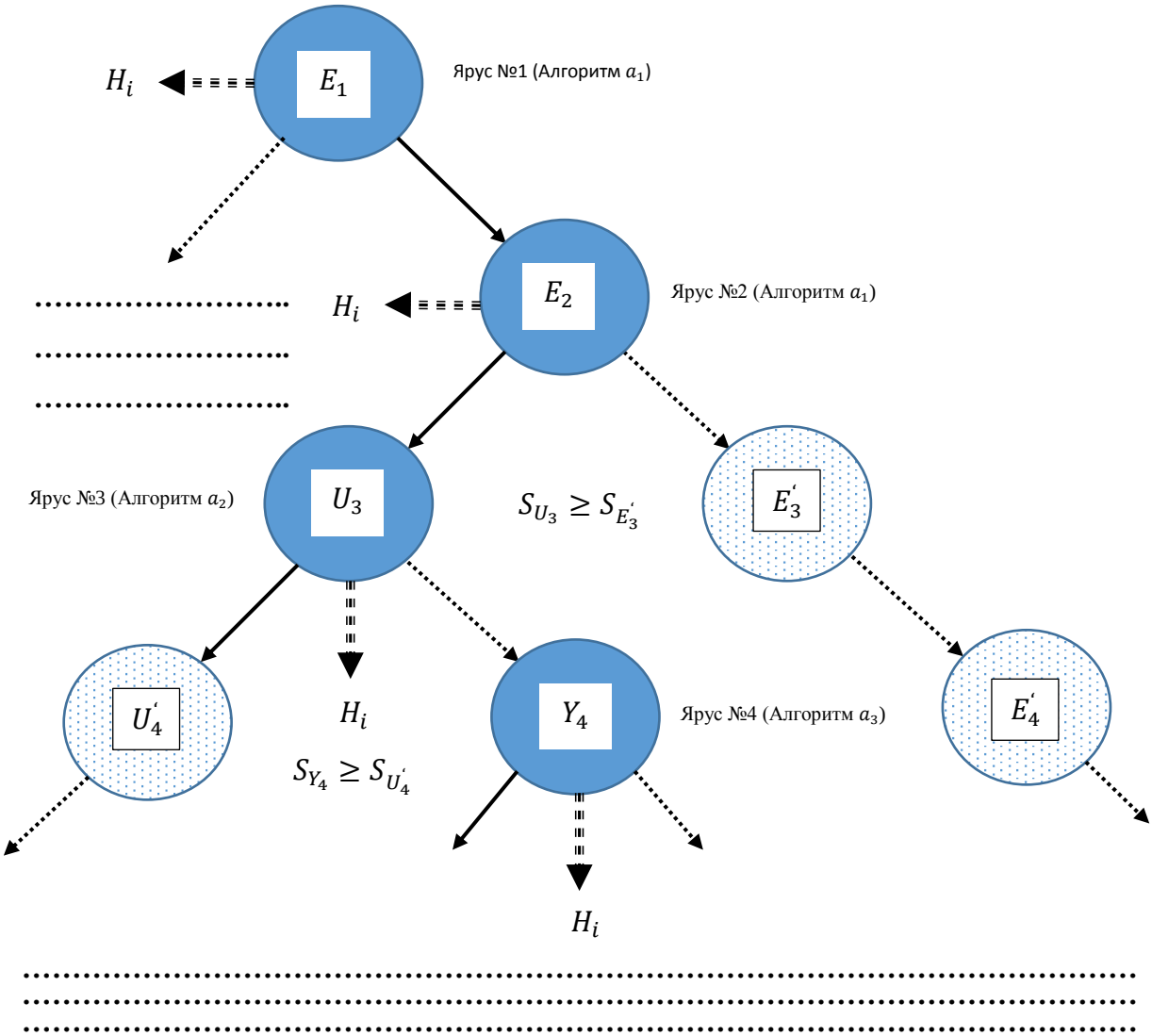


Рис. 6.9. Структура деякого дерева класифікації, побудована обмеженим методом АДК.

Підкреслимо, що запропонована вище схема побудови дерева класифікації (обмежений метод АДК) дозволяє регулювання точності (ефективність) моделі дерева, яка будується, враховуючи загальну кількість помилок (всіх типів) класифікації на тому чи іншому шляху (етапі побудови) загальної структури алгоритмічного дерева.

Отже, проведемо оцінку точності та порівняння побудованих моделей дерев класифікації (на основі обмежених методів ЛДК та АДК) для представленої в попередньому пункті прикладної задачі класифікації. Фрагмент основних порівнянь приведених вище експериментів (порівняльних тестів методів побудови обмежених моделей ЛДК/АДК на масиві даних даної

прикладної задачі) представлено в табл. 6.4. Відмітимо, що побудовані моделі дерев класифікації забезпечили необхідний рівень точності, заданий умовою задачі, можливість подальшої добудови структури (покращення якості моделі), швидкодію та витрати робочої пам‘яті інформаційної системи.

Отже, використання обмежених методів побудови дерев класифікації має свою доцільність в умовах обмеження апаратних ресурсів інформаційних систем, вимог щодо складності моделей дерев класифікації поточної прикладної задачі, можливості поетапної побудови моделі класифікації з покроковим підвищенням її фінальної якості.

Таблиця 6.4 Порівняльна таблиця обмежених методів ЛДК/АДК.

№	Метод синтезу структури логічного дерева	Інтегральний показник якості моделі Q_{Main}	Загальна кількість помилок моделі на НВ та ТВ Er_{All}
1	Метод повного ЛДК (на основі селекції елементарних ознак)	0,004789	2
2	Метод алгоритмічного дерева (типу I)	0,005234	0
3	Обмежений метод побудови ЛДК	0,003181	4
4	Обмежений метод побудови АДК	0,003043	5
5	Метод алгоритмічного дерева (типу II)	0,005148	0
6	Метод повного ЛДК (алг. Ю.А. Василенка)	0,050321	0

Зауважимо, що в табл. 6.4 оцінка якості побудованої моделі дерева класифікації проводилася на основі інтегрального показника (6.34) та відображає базові параметри (характеристики) фіксованого дерева класифікації, і може бути застосована додатково в якості критерію

оптимальності в процедурі оцінки довільної деревоподібної схеми (моделі) розпізнавання (класифікації).

Відмітимо, що практична цінність запропонованих обмежених методів побудови моделей дерев класифікації (ЛДК/АДК) дає можливість будувати економні та ефективні моделі класифікації з можливістю регулювати точність результуючого дерева класифікації (обмежені методи ЛДК/АДК були реалізовані в бібліотеці алгоритмів системи "ОРІОН ІІІ" для розв'язку різноманітних прикладних задач класифікації), причому вони характеризуються великим ступенем універсальності відносно широкого кола прикладних задач.

Важливим моментом є те, що практичні застосування підтвердили працездатність побудованих моделей дерев класифікації (на основі обмежених методів ЛДК/АДК) та розробленого програмного забезпечення.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) Важливою особливістю обмежених методів побудови моделей ЛДК/АДК є те, що вони спрямовані на добудову лише тих шляхів (ярусів) структури дерева класифікації, де є найбільша кількість помилок (всіх типів) класифікації. Такий підхід синтезу моделі розпізнавання дозволяє досить ефективно регулювати складність (точність) моделі дерева класифікації, що будується.

2) Оскільки обмежені методи побудови ЛДК/АДК мають ідейну основу у вигляді повних методів поетапної селекції елементарних ознак та апроксимації НВ набором алгоритмів класифікації та розпізнавання, тому доцільне їх застосування в разі обмежень щодо апаратних ресурсів інформаційної системи, обмежень точності та структурної складності моделі, що будується, обмежень структури, послідовності та глибини розпізнавання масиву даних НВ.

3) Важливим моментом є те, що з точки зору концепції дерев класифікації обмежені методи ЛДК/АДК вводяться за рахунок зміни

(модифікації) загального критерію розгалуження (вибору шляху для добудови структури) на етапі побудови дерева, що дає можливість зробити висновок про можливість подальшого вдосконалення (модифікації) базових методів шляхом введення більш ефективних (досконалих) критеріїв розгалуженого вибору вершин (структур ЛДК/АДК).

6.6. Питання збіжності процедури побудови дерева класифікації методів ЛДК/АДК

На даному етапі дисертаційного дослідження розглянемо принципове питання щодо методів дерев класифікації (моделей класифікації) – питання збіжності процедури побудови дерева класифікації (методів дерев класифікації). Отже, як і на початку розділу, припустимо, що маємо початкову НВ вигляду (6.15), яка представлена сукупністю навчальних пар відомої класифікації. Зауважимо, що тут НВ є детермінованою, тобто для неї буде виконуватися наступна умова:

$$If\ \left(x_k, f_R(x_k)\right)\ and\ \left(x_l, f_R(x_l)\right),\ x_k \neq x_l\ then\ f_R(x_k) \neq f_R(x_l).$$

6.6.1. Збіжність моделі дерева класифікації для випадку ЛДК

Нехай на кожному кроці в процесі побудови логічного дерева (деякої моделі ЛДК) буде вибиратися лише одна відібрана елементарна ознака з набору фіксованих ознак $(\varphi_1, \varphi_2, \dots, \varphi_n)$. Тоді на $n-$товому кроці процедури побудови дерева класифікації схема ЛДК буде представляти собою деякий предикат p_n (узагальнену ознаку, яка побудована з набору елементарних ознак) [95], який є найефективнішою апроксимацією початкової НВ загального вигляду (6.15) (звичайно, що це справедливо і для випадку структури АДК).

Зокрема p_n буде представляти деяку деревоподібну схему (дерево класифікації), яке складається з n вершин, тобто в структуру предикату p_n будуть входити всього n елементарних ознак (атрибутів дискретного об‘єкту НВ) з початкового набору.

Зауважимо, що послідовність предикатів $p_1, p_2, \dots, p_j$ (узагальнених ознак) збігається до початкової НВ вигляду (6.15), якщо, починаючи з деякого Q, буде виконуватись умова:

$$f_{Q+m} = f_R(x_i),\ (i = 1,2, \dots, m),\ (m \geq 0).$$

Деяку елементарну ознаку, яка буде вибиратися (фіксуватися) на $n-$товому кроці в схемі побудови моделі ЛДК, позначимо через φ_n.

Зрозуміло, що ознаці φ_n відповідає деякий фіксований шлях $r_1, r_2, \ldots$, який закінчується даним атрибутом (вершиною дерева класифікації – моделі ЛДК). Наприклад, на рис. 6.10 зображено ЛДК, в якому вершині φ_2 (ознаці) відповідає шлях $\{0\}$, а вершині φ_5 – шлях $\{0,1\}$.

Шлях, який відповідає елементарній ознаці φ_n вказаним чином, позначимо через T_n, а через D_n позначимо множину тих пар $\left(x_i, f_R(x_i)\right)$ початкової НВ загального вигляду (6.15), для яких об‘єкти w_i належать шляху T_n. Наприклад, для структури ЛДК (рис. 6.10), нехай $\varphi_n = \varphi_4$, тоді шлях T_n буде мати вигляд $\{1,0\}$.

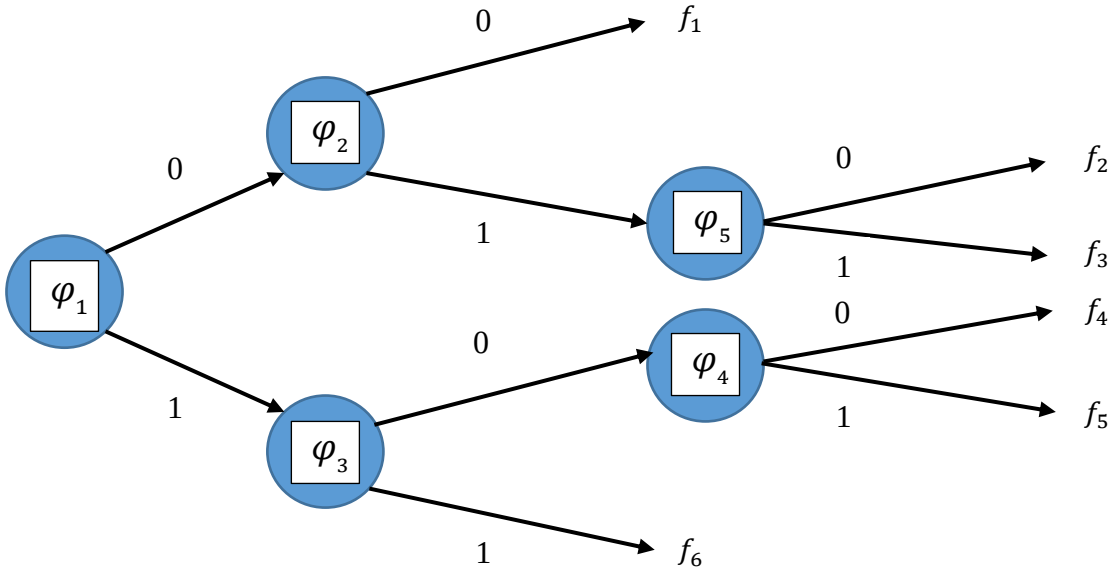


Рис. 6.10. Приклад структури ЛДК з елементарними ознаками в якості вершин.

У такому випадку деякий об‘єкт w_i належить шляху $\{1,0\}$, якщо виконуються умови: $\varphi_1(w_i) = 1$ та $\varphi_3(w_i) = 0$.

Далі будемо вважати, що елементарна ознака φ_n слабо розділяє множину D_n, якщо в D_n існують такі пари $\left(x_i, f_R(x_i)\right)$ та $\left(x_j, f_R(x_j)\right)$, що $\varphi_n(x_i) = 0$ та $\varphi_n(x_j) = 1$ (тобто $\varphi_n(x_i) \neq \varphi_n(x_j)$).

На наступному етапі дослідження введемо поняття кінцевої потужності схеми методу дерева класифікації.

Визначення 6.1. Кінцевою потужністю схеми методу дерева класифікації (моделей ЛДК/АДК) будемо називати кількість всіх кінцевих вершин (визначених листів) даної схеми. Наприклад, для ЛДК з рис. 6.10 кінцева потужність буде дорівнювати 6.

Очевидно, що кінцева потужність схеми методу дерева класифікації також дорівнює кількості всіх кінцевих шляхів у даній схемі. Зрозуміло, що індукцією за n легко довести, що кінцевою потужністю кожної з вищевказаних схем p_n (предикатів), дорівнює $n+1$. Дійсно, те, що кінцева потужність p_1, до складу якої входить тільки одна ознака або алгоритм (випадків ЛДК/АДК), дорівнює 2, є очевидним.

Нехай кінцева потужність схеми p_n дорівнює $n+1$. Підрахуємо кінцеву потужність p_{n+1}. Зрозуміло, що дана схема будується на основі схеми p_n, коли в деякій кінцевій вершині послідовно додається нова вершина (ознака, алгоритм) із номером $n+1$. Очевидно, що при додаванні цієї ознаки (алгоритму) в схему p_n зникає одна кінцева вершина та додаються дві нові кінцеві вершини. Отже, можна зробити висновок, що кількість усіх кінцевих вершин схеми p_n дорівнює $n+2$.

Припустимо, що на кожному $n-$ вому кроці процедури побудови дерева класифікації (моделі ЛДК) множина D_n слабо розділяється деякою ознакою φ_n. Далі розглянемо схему p_n. У цій схемі маємо відповідно до вищезазначеного, $n+1$ кінцевих шляхів. Завдяки тому, що D_n на кожному кроці слабо розділяється, кожний такий шлях містить хоча б одну пару початкової НВ загального вигляду (6.15). Крім того очевидно, що різні кінцеві шляхи в p_n не мають спільних пар із вибірки (6.15).

Отже, можна зробити висновок, що схема (предикат) p_n розділяє НВ (на основі базового критерію розгалуження введеного поточним методом дерева класифікації) на $n+1$ непустих частин (підмножин), що не перетинаються. Оскільки в початковій НВ всього знаходяться m навчальних пар, то схема

p_{m-1} (або предикат з меншим номером) повністю розділить початкову НВ, тобто p_{m-1} буде повністю розпізнавати вибірку.

Таким чином, якщо на кожному $n-$ вому кроці відібрана елементарна ознака φ_n слабо розділяє множину D_n, то в цьому випадку процес побудови ЛДК збігається відносно початкової НВ та закінчується не більше ніж за $m-1$ кроків, де $m-$ кількість усіх навчальних пар початкової НВ.

Зауважимо, що умова слабого розділення класів початкової НВ є доволі слабою – тому вона забезпечує невисоку збіжність процедури побудови дерева класифікації, отже, важливо розглянути питання збіжності процесу при більш сильній умові. Тому будемо припускати, що маємо справу з випадком, коли НВ містить інформацію про два класи (образи) H_0 та H_1, а сама НВ має детерміновану природу. Нехай n_j- кількість навчальних пар $\left(x_i, f_R(x_i)\right)$ у початковій НВ, які задовольняють співвідношення $f_R(x_i) = j, (j = 0,1)$, причому для спрощення та визначеності покладемо, що $n_0 \geq n_1$.

Зафіксувавши $f_R(x) \equiv 0$, буде отримано деяку узагальнену ознаку (схему) f_0, яка апроксимує (повністю або частково) початкову НВ. Очевидно, що в даному випадку (тобто в ситуації, коли ще не зроблено вибір жодної елементарної ознаки φ_n), узагальнена ознака (схема) f_0 є найкращою апроксимацією початкової НВ. Далі величину n_1 будемо називати безумовною кількістю помилок у початковій НВ.

Нехай на першому кроці побудови дерева класифікації відібрана (довільним шляхом) деяка елементарна ознака φ_1 – причому дана ознака розіб‘є початкову вибірку на дві частини (підмножини) H_0 та H_1, де H_j- множина всіх пар $\left(x_i, f_R(x_i)\right)$ початкової НВ, для яких виконується співвідношення $f_1(x_i) = j\ (j = 0,1)$.

Нехай n_m^j- множина всіх пар $\left(x_i, f_R(x_i)\right)$ з вибірки $H_j, (j = 0,1)$, для яких виконується співвідношення $f_R(x_i) = m\ (m = 0,1)$. Ознаку φ_1 можна вважати узагальненою ознакою f_1 (схемою), яка побудована на першому кроці процесу побудови ЛДК.

Уведемо величину $\rho = \max(n_0^0, n_1^0) + \max(n_0^1, n_1^1)$, яка представляє собою кількість правильних відповідей (класифікацій), які реалізуються узагальненою ознакою f_1, а відповідно величина n_0 є кількістю правильних відповідей (класифікацій), які реалізуються узагальненою ознакою f_0.

Під кількістю правильних відповідей розуміємо кількість тих навчальних пар $\left(x_i, f_R(x_i)\right)$ у початковій навчальній вибірці типу (6.15), для яких виконується співвідношення рівності $f_R(x_i) = f_1(x_i)$.

Оскільки $n_0^0 + n_0^1 = n_0$ та $n_1^0 + n_1^1 = n_1$, то будемо мати наступне:

$$\rho = \max(n_0^0, n_1^0) + \max(n_0^1, n_1^1) \geq n_0. \quad (6.35)$$

Отже, при виборі ознаки φ_1 кількість правильних відповідей як мінімум не зменшується. Кількість помилок, які дає узагальнений алгоритм f_1, буде дорівнювати:

$$m - \rho = n_1 - (\rho - n_0) \leq n_1. \quad (6.36)$$

Зауважимо, що (6.36) випливає з (6.35). Уведемо величину $\lambda_1 = \frac{n_1}{m-\rho}$ та назвемо її якістю елементарної ознаки φ_1 відносно початкової НВ, аналогічно визначається λ_n ознаки φ_n відносно початкової НВ ($n = 1,2,3, \dots$).

На наступному етапі дослідження зробимо припущення – якість λ_n елементарної ознаки φ_n відносно масиву початкової НВ не менше деякого числа y, де $y > 1$.

Проаналізуємо складність процедури побудови дерева класифікації при даній умові ($y > 1$), для цього оцінимо кількість кроків, за яку даний процес (процедура) реалізує повне розпізнавання масиву початкової навчальної вибірки.

Розглянемо для визначеності наступну схему побудови дерева класифікації (рис. 6.11).

Нехай $n_1 -$ безумовна кількість помилок початкової НВ. Елементарна ознака φ_1^1 розділяє НВ на дві вибірки: H_0 та H_1. Нехай h_0 та h_1 відповідно безумовна кількість помилок у вибірках H_0 та H_1. Ознака φ_1^2 розділить множину H_0 на дві множини H_{00} та H_{01}. Нехай h_{00} та $h_{01} -$ безумовна

кількість помилок у вибірках H_{00} та H_{01}. Аналогічно визначимо множини H_{10}, H_{11} та кількості h_{10} і h_{11} для елементарної ознаки φ_2^2.

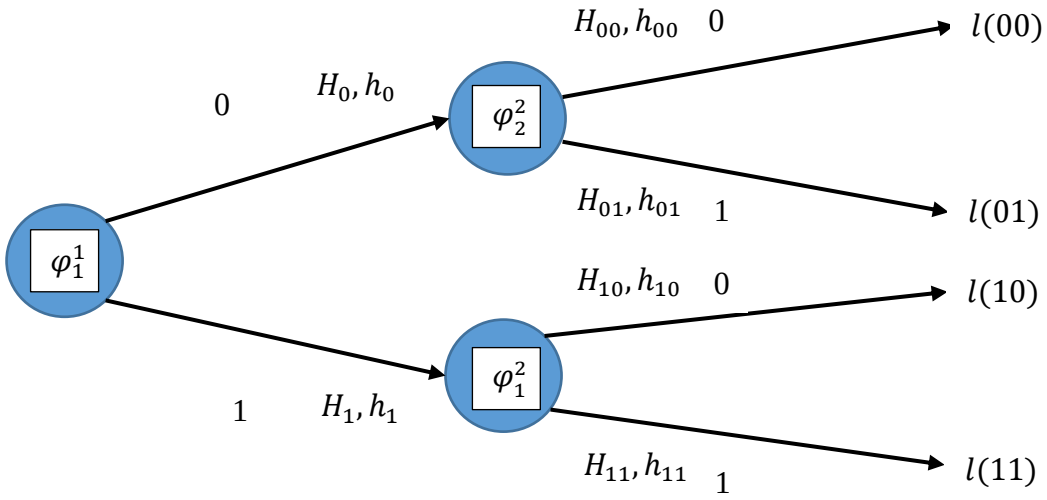


Рис. 6.11. Схема розбиття на підмножини в структурі дерева класифікації.

З початкової умови ($y > 1$) випливає:

$$\begin{cases} h_0 + h_1 \le \frac{1}{y} * n_1 \\ h_{00} + h_{01} \le \frac{1}{y} * h_0. \\ h_{10} + h_{11} \le \frac{1}{y} * h_1 \end{cases} \quad (6.37)$$

З (6.37) отримаємо наступне:

$$h_{00} + h_{01} + h_{10} + h_{11} \le \frac{1}{y^2} * n_1. \quad (6.38)$$

Зробимо такі припущення в даному відношенні: $h_0 \ge 1$, $h_1 \ge 1$, $h_{00} \ge 1$, $h_{01} \ge 1$, $h_{10} \ge 1$ та $h_{11} \ge 1$. Звідси будемо мати наступне:

$$2^1 \le \frac{1}{y} * n_1, 2^2 \le \frac{1}{y^2} * n_1. \quad (6.39)$$

Аналогічно для набору ознак $\varphi_1^i, \varphi_2^i, \dots$, які розташовані на $i-$ товому ярусі логічного дерева, будемо мати:

$$2^i \le \frac{1}{y^i} * n_1 \text{ або } (2y)^i \le n_1. \quad (6.40)$$

Звідси можна зробити висновок, що процес побудови дерева класифікації буде продовжуватися до тих пір, доки в структурі дерева не буде m ярусів (рівнів), де m має наступний вигляд:

$$m = R(\frac{\log_2 n_1}{1+\log_2 y}). \quad (6.41)$$

Під $R(x)$ розуміється заокруглення числа x до найблищого цілого числа, яке перевищує x. Наприклад $Q(1.2) = 2, Q(3.7) = 4, Q(4.1) = 5$.

Отже дерево класифікації, яке має m повних ярусів (тобто випадок, коли на $i-$товому ярусі стоять 2^{i-1} вершин), має $2^{m+1} - 1$ вершин – таким чином розпізнавання початкової НВ при умові $(y > 1)$ за допомогою повного ЛДК відбувається не більш ніж за $2^{m+1} - 1$ кроків, де m розраховується за допомогою виразу (6.41).

На наступному етапі дослідження розглянемо питання складності схеми (процедури побудови) дерева класифікації для випадку АДК, увівши спочатку необхідні в подальшому визначення.

Визначення 6.2. Потужністю деякої побудованої УО або набору УО (для фіксованого кроку схеми АДК) будемо називати кількість навчальних пар $\left(x_i, f_R(x_i)\right)$ початкової НВ вигляду (6.15), які апроксимує (правильно класифікує) дана узагальнена ознака (послідовність узагальнених ознак).

Важливим для схем АДК є те, що при покроковому розбитті НВ на дві вибірки H_0 та H_1 (і так далі) частина вибірки буде повністю покриватися поточним алгоритмом класифікації (узагальненою ознакою або їх набором) – тобто будемо мати випадок сильного розділення класів масиву НВ. Отже, можна зробити припущення, що складність кінцевої схеми АДК (загальна кількість кроків побудови дерева) буде в значній мірі залежати від процедури початкової оцінки та відбору набору незалежних алгоритмів класифікації a_i, їх початкових параметрів, параметрів наборів УО f_i, які вони генерують для кожного кроку схеми АДК.

Тоді для схеми АДК важливо розглянути загальну складність процедури побудови дерева класифікації за умови слабкої роздільності класів початкової НВ, при якій генерується не більше однієї УО потужністю в одиницю для кожної вершини дерева та умови сильної роздільності, коли обмежень на

кількість УО та їх потужність не накладається умовами задачі та практичною доцільністю, і можливо їх будувати.

6.6.2. Збіжність моделі дерева класифікації для випадку АДК

На першому етапі розглянемо випадок слабкого розділення класів з обмеженнями на набори УО, що будуються, схемою АДК.

Відмітимо, що процедура побудови алгоритмічного дерева має певні особливості з точки зору поетапної апроксимації початкової НВ послідовністю УО – нехай на кожному кроці побудови деякої моделі АДК буде вибиратися для роботи один фіксований алгоритм класифікації з набору відібраних алгоритмів $(a_1, a_2, \dots, a_n)$, причому дерево класифікації може бути побудовано одним алгоритмом a_i та послідовністю УО, які він генерує.

Отже, після проведення n кроків процедури побудови дерева класифікації структура АДК буде представляти собою деяку схему s_n (узагальнену ознаку другого порядку, яка побудована з набору синтезованих алгоритмами класифікації УО), яка є найбільш ефективною апроксимацією початкової НВ загального вигляду (6.15) набором незалежних алгоритмів класифікації та їх УО. Зокрема s_n буде представляти деяку деревоподібну схему (структуру ДУО), яка складається з n вершин, тобто в конструкцію схеми s_n будуть входити всього n алгоритмів класифікації (УО – при умові генерації для кожного кроку процедури побудови дерева не більше однієї узагальненої ознаки мінімальної потужності в одиницю) з початкового набору.

Отже, можна зробити висновок, що послідовність побудованих схем $s_1, s_2, \dots, s_j$ (узагальнених ознак другого порядку) збігається до початкової НВ вигляду (6.15) не більше ніж за M кроків (де $M -$ загальна потужність початкової НВ), навіть за умов генерації на кожному кроці лише однієї УО, потужність кожної з яких не більше одиниці.

Деякий алгоритм класифікації, який буде вибиратися (фіксуватися) на $n -$ товому кроці в процедурі побудови моделі АДК (для генерації відповідної УО), позначимо через a_n, причому зрозуміло, що даному алгоритму a_n

відповідає деяка схема s_n, яка складається з алгоритмів $a_1, a_2, \dots, a_{n-1}$ та закінчується даним атрибутом (вершиною дерева класифікації – моделі АДК). Наприклад, на рис. 6.12 зображено деяку модель АДК, в якій фіксованій схемі s_2 (вершині дерева класифікації, що будується) відповідає послідовність кроків (схем) $\{s_1\}$, а схемі s_M – послідовний шлях $\{s_1, s_2, \dots, s_{M-1}\}$.

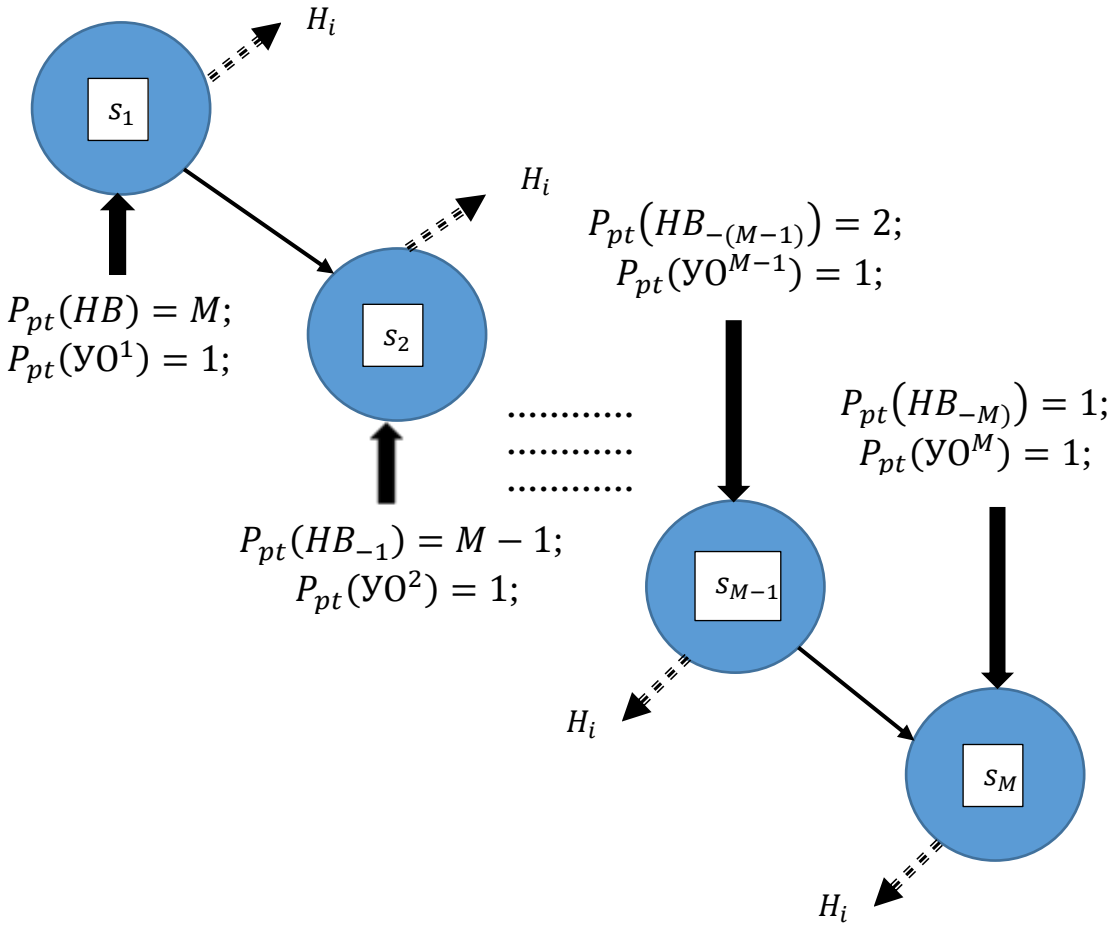


Рис. 6.12. Приклад структури АДК з УО в якості вершин.

Отже, для моделі АДК можна зробити наступний висновок: схема s_n (у структурі дерева класифікації) розділяє НВ на n непустих частин (підмножин), що не перетинаються, причому оскільки в початковій НВ всього знаходиться M навчальних пар, то схема s_M повністю розділить (апроксимує) початкову НВ (тобто s_M буде повністю розпізнавати вибірку за умови генерації на кожному кроці по одній УО потужністю один). Таким чином, якщо на кожному $n-$ вому кроці схеми побудови АДК згенерована УО (відібраним алгоритмом класифікації a_n) слабо розділяє множину початкової НВ, то в

цьому випадку процес побудови дерева класифікації збігається відносно початкової НВ та закінчується не більше ніж за M кроків, де $M -$ кількість усіх навчальних пар початкової НВ. На наступному етапі дослідження важливо розглянути випадок сильного розділення класів початкової НВ, коли жодних обмежень на алгоритми a_i щодо генерації УО не накладається (потужність побудованої УО обмежена лише практичною можливістю самого алгоритму класифікації a_i та структурними параметрами НВ). Нехай через $P(f_j)$ позначимо загальну потужність (апроксимаційну здатність) відповідної УО f_j, $(1 \leq j \leq s)$, де $s -$ кількість УО у схемі АДК, що будується. Далі на деякому кроці r $(1 \leq r \leq M)$ схеми АДК побудовано послідовність узагальнених ознак $f_1, \dots, f_r$ з відповідними їм величинами $P(f_i) = z_i$, де $(1 \leq z \leq M), (1 \leq i \leq r)$, $M -$ загальна потужність НВ, причому серед них є величини z^{max} та z^{min}, які є для них відповідно максимальними та мінімальними (відносно поточного кроку схеми АДК). Тоді в такому випадку схему (модель) АДК буде побудовано за t кроків, де величина t визначається співвідношенням (6.42).

$$t \leq 2 * \frac{P_{pt}(\text{НВ})}{z^{max}+z^{min}} = \frac{2M}{z^{max}+z^{min}}. \quad (6.42)$$

Зауважимо, що у разі, коли умовою прикладної задачі на схему АДК, що будується, накладаються обмеження щодо потужності синтезованих УО (неперевищення відповідної величини P) – схему дерева класифікації (модель АДК) буде побудовано за t кроків, де величина t визначається співвідношенням (6.43).

$$t \leq \frac{M}{P}. \quad (6.43)$$

Наприкінці нагадаємо, що при жорстких обмеженнях схеми АДК на одну генеровану УО (де за умовою $P(f_j) = 1, (1 \leq i \leq t)$), тобто у випадку слабкого розділення класів поточної задачі, схему дерева класифікації (модель АДК) буде побудовано за t кроків, де величина $t \leq M$.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) Для умови слабкого розділення класів у випадку ЛДК, якщо на кожному $n-$ вому кроці відібрана елементарна ознака φ_n слабо розділяє множину (підмножину) об‘єктів початкової НВ, то в цьому випадку процес побудови дерева класифікації збігається відносно початкової НВ та закінчується не більше ніж за $m-1$ кроків, де $m-$ кількість усіх навчальних пар початкової НВ.

2) Дерево класифікації (структури ЛДК) за умови сильного розділення класів множини об‘єктів початкової НВ, яке має m повних ярусів, рівнів (тобто випадок, коли на $i-$ товому ярусі стоять 2^{i-1} вершин), має $2^{m+1}-1$ вершин –отже розпізнавання масиву початкової НВ при умові ($y>1$) за допомогою повного ЛДК відбувається не більше ніж за $2^{m+1}-1$ кроків, де m розраховується за допомогою виразу $m=R(\frac{\log_2 n_1}{1+\log_2 y})$.

3) Загальна кількість всіх кінцевих вершин логічної структури (листів дерева розпізнавання) побудованої схеми класифікації буде однозначно визначати кінцеву потужність схеми методу дерева класифікації (моделей ЛДК/АДК).

4) Потужністю деякої УО (набору побудованих УО) для фіксованого кроку схеми методу АДК вважається загальна кількість навчальних пар $\left(x_i, f_R(x_i)\right)$ початкової НВ (підмножини початкової НВ) вигляду (6.15), які апроксимує (правильно класифікує) дана узагальнена ознака (послідовність узагальнених ознак).

5) У випадку слабкого розділення класів початкової НВ для схеми АДК процес побудови дерева класифікації збігається відносно масиву даних НВ та закінчується не більше ніж за M кроків, де $M-$ кількість усіх навчальних пар початкової НВ.

6) У випадку сильного розділення класів початкової НВ для схеми АДК, коли потужність побудованої УО (або набору УО) обмежена лише практичною можливістю самого алгоритму класифікації a_i та початковими параметрами НВ, схему (модель) АДК буде побудовано за t кроків, де величина t визначається співвідношенням (6.42).

Висновки до розділу 6

1. У розділі представлено загальну концепцію методу логічного дерева класифікації, яка полягає в покроковій апроксимації масиву початкових даних НВ набором ранжованих за інформативністю елементарних ознак (атрибутів) дискретних об‘єктів. Показано, що критерієм розгалуження в методах ЛДК залишається ефективність (інформативність) фіксованої елементарної ознаки (набору ознак та їх сполучень) відносно деякої частини даних початкової НВ, а важливим напрямком розвитку методів апроксимації даних НВ набором елементарних ознак є вибір ефективного критерію розгалуження в дереві.

2. Запропоновано схему апроксимації початкової НВ набором відсортованих елементарних ознак (оцінених за інформативністю), яка дає відносно компактні та ефективні структури моделей ЛДК. Уведено набір загальних показників якості моделі ЛДК, який фіксує найважливіші характеристики логічних дерев та може бути застосований в якості критерію оптимальності в процедурі побудови ЛДК та фінальному відборі з множини моделей ЛДК, а проведені практичні випробовування підтвердили працездатність запропонованих моделей ЛДК.

3. Запропоновано концепцію покрокової апроксимації масиву початкових даних НВ набором відібраних та оцінених незалежних алгоритмів класифікації та розпізнавання, яка дозволяє будувати різнотипні моделі АДК. Досліджено питання оцінки якості (ефективності, інформативності) набору алгоритмів класифікації (вершин структури дерева класифікації) в схемі методів АДК – відбору критерію розгалуження конструкції дерева класифікації.

4. Розроблено методи побудови моделей АДК двох типів, причому отримані дерева класифікації складаються з різних алгоритмів та методів розпізнавання і, в свою чергу, є новими алгоритмами (схемами) класифікації. Дані методи АДК при побудові моделей дерев класифікації дозволяють досягти хороших результатів та розширити коло прикладних задач

застосування, можливості побудови моделей класифікації, точність яких можна регулювати в процесі побудови (або будувати системи з наперед заданою точністю), можливості раціонального використання вже накопиченого потенціалу методів та алгоритмів розпізнавання та класифікації, причому загальна концепція методів АДК дає змогу будувати моделі дерев класифікації різних типів (вони не обмежуються запропонованими в даному дослідженні двома типами граф-схем), які базуються на простій ідеї апроксимації начальної навчальної вибірки набором автономних алгоритмів класифікації та представлення отриманої моделі у вигляді деякої деревоподібної схеми.

5. Розроблено обмежений метод побудови АДК, який спрямовано на добудову лише тих шляхів (ярусів) структури дерева класифікації, де є найбільша кількість помилок (всіх типів) класифікації. Такий підхід синтезу моделі розпізнавання дозволяє досить ефективно регулювати складність (точність) моделі дерева класифікації, що будується, причому доцільно його застосовувати в ситуаціях з обмеженнями щодо апаратних ресурсів інформаційної системи, обмеженнями точності та структурної складності моделі, обмеженнями структури, послідовності та глибини розпізнавання масиву даних НВ.

6. Досліджено питання збіжності процедури побудови моделей дерев класифікації представлених у роботі методів ЛДК/АДК для умов слабкого та сильного розділення класів початкової НВ, проведено відповідні числові оцінки, причому загальна кількість усіх кінцевих вершин логічної структури (листів дерева розпізнавання) побудованої схеми класифікації буде однозначно визначати кінцеву потужність схеми методу дерева класифікації (моделей ЛДК/АДК).

РОЗДІЛ 7
РОЗРОБКА ПРОГРАМНОГО ІНСТРУМЕНТАРІЮ ДЛЯ СТВОРЕННЯ ПРИКЛАДНИХ ПРОГРАМ НА ОСНОВІ МЕТОДІВ ДЕРЕВ КЛАСИФІКАЦІЇ

7.1. Схема функціональної оцінки важливості ознак та груп ознак

Забезпечуючи розв‘язок різнотипного набору практичних задач розпізнавання, дуже часто доводиться стикатися з такими принциповими питаннями:

- кодування вихідної інформації;
- знаходження важливості ознак та груп ознак;
- мінімізація вихідного опису;
- визначення якості кодування вихідної інформації;
- побудова множини алгоритмів розпізнавання та знаходження коректного (найбільш ефективного за набором параметрів) алгоритму для даної задачі, зважаючи на її особливості.

Відмітимо, що всі ці задачі доводиться вирішувати в рамках набору деяких фіксованих (зазвичай апаратних та прикладних) обмежень: час, пам‘ять, ефективність та інше.

На цьому етапі дослідження розглянемо більш детально дану проблематику – так, у задачі кодування вихідної інформації, якщо об’єкти з $I(l)$ представляють собою деякі точки $n-$ вимірного простору, то завжди при умові вихідної різниці між класами (заданої НВ) ми можемо розбити $n-$ вимірний простір, обмежений можливими значеннями ознак, гіперплощинами, на $n-$ мірні гіперпаралелепіпеди так, щоб точки з різних класів не попадали в один і той самий паралелепіпед. Після цього кожному гіперпаралелепіпеду в $n-$ вимірному просторі можна поставити у відповідність деякий двійковий код. Ці двійкові коди, в залежності від того куди попадають об’єкти з $I(l)$ та G, і визначає набір бінарних ознак задачі розпізнавання.

У питанні знаходження важливості ознак та груп ознак зафіксуємо: деяку початкову інформацію – нехай навчальна інформація $I(l)$ представлена у вигляді наборів об‘єктів відомої класифікації:

$$\left(x_1, f_R(x_1)\right), \dots, (x_m, f_R(x_m)) \tag{7.1}$$

Зауважимо, що тут $x_i \in G, f_R(x_i) \in \{0,1, \dots, k-1\}, (i = 1,2, \dots m)$, $m -$ кількість об‘єктів з $I(l)$, $f_R(x_i)$ – деяка скінчено-значна функція, що задає розбиття R множини G на класи (образи) $H_0, H_1, \dots, H_{k-1}$. Відношення $f_R(x_i) = l$, $(l = 1,2, \dots, k-1)$ означає $x_i \in H_l$, $x_i = \{x_{i_1}, x_{i_2}, \dots, x_{i_n}\}$, $x_{i_1} -$ значення $j -$ тової ознаки для об‘єкта x_i, $(j = 1,2, \dots, n)$, $n -$ кількість ознак в $I(l)$.

Визначимо на $I(l)$ наступні функціонали:

$$W(P_i) = \frac{1}{m}\sum_{j=0}^{k_i-1} \max_{0 \le m \le k-1} b_j^m \tag{7.2}$$

Зокрема m – кількість об‘єктів $I(l)$, k_j – кількість різних значень ознаки P_i, b_j^m – кількість значень P_i – тової ознаки в класі Ω_m, $(m = 0,1, \dots, k-1)$, $k -$ кількість класів $I(l)$.

$$W\left(P_i \backslash P_{i_1} = \eta_{i_1}, \dots, P_{i_\xi} = \eta_{i_\xi}\right) = \frac{1}{h}\sum_{j=0}^{k_i-1} \max_{0 \le m \le k-1} b_j^m \tag{7.3}$$

Зауважимо, що тут $i \neq i_1, \dots, i_\xi$. Функціонал (7.3) обчислюється аналогічно функціоналу (7.2) з урахуванням того, що беруться тільки ті об’єкти, для яких $P_{i_1} = \eta_{i_1}, \dots, P_{i_\xi} = \eta_{i_\xi}$, h – кількість таких об’єктів.

$$W\left(P_{i_1}, \dots, P_{i_\xi}\right) = \frac{1}{M}\sum_{j=0}^{|\Gamma|} \max_{0 \le m \le K} B_{\Delta_j}^m \tag{7.4}$$

Зауважимо, що тут $\Delta_j = (x_{i_1}, \dots, x_{i_\xi})$, $\left(x_{i_k} \in G, k = 1, \dots, \xi\right) -$ фіксований $j -$ товий об’єкт із множини Г, Г – множина об’єктів $I(l)$, що відрізняються між собою значеннями хоча б однієї ознаки, $|\Gamma|$ – кількість об’єктів (загальна потужність), що входять у множину Г (потужність множини Г), $B_{\Delta_j}^m$ – кількість Δ_j в класі G_m, $(m = 0,1, \dots, k-1)$, m – загальна кількість усіх об’єктів в $I(l)$. Відмітимо, що $\frac{1}{k} \le W(P_{i_1}, \dots, P_{i_\xi}) \le 1$, $k -$ кількість класів в $I(l)$, якщо

$W\left(P_{i_1},\dots,P_{i_\xi}\right)=1$, то ознаки $P_{i_1},\dots,P_{i_\xi}$ утворюють тест і, навпаки, якщо $P_{i_1},\dots,P_{i_\xi}$ утворюють тест, то функціонал $W\left(P_{i_1},\dots,P_{i_\xi}\right)=1$. Причому значення функціоналу $W\left(P_{i_1},\dots,P_{i_\xi}\right)$, помножене на 100%, характеризує найкращу якість можливого алгоритму класифікації (розпізнавання) на початковій інформації $I(l)$, побудованого за допомогою набору ознак $P_{i_1},\dots,P_{i_\xi}$. Відмітимо, що значення функціоналу $W\left(P_{i_1},\dots,P_{i_\xi}\right)$ буде вважатися важливістю набору ознак $P_{i_1},\dots,P_{i_\xi}$ на початковій інформації (вибірці) $I(l)$.

На наступному етапі дослідження приведемо один із можливих програмних алгоритмів знаходження важливості ознак та груп ознак за допомогою запропонованого функціонала (7.4) і його можливе використання для розв‘язку широкого кола задач розпізнавання та класифікації.

Звернемо увагу, що особливістю даного алгоритму є випадковий механізм вибору ознак та груп ознак для оцінки їх інформативності і так далі.

На початковому етапі зафіксуємо основні програмні масиви та змінні.

$N-$ кількість ознак у початковій $I(l)$;

$M-$ загальна кількість об‘єктів у $I(l)$;

$N2-$ загальна кількість класів у $I(l)$;

$OB[1..N,1...M]-$ масив для зберігання даних $I(l)$ у пам’яті комп‘ютера;

$KOB[1..N2]-$ масив для зберігання порядкових номерів останніх об‘єктів кожного класу з масиву $I(l)$;

$HB[1..MX]-$ масив для зберігання поліномів, необхідних для формування випадкових бітових рядків B, одиниці яких задають номера ознак, для яких в подальшому знаходиться їх важливість за допомогою запропонованого функціонала (7.4);

$MX-$ загальна кількість поліномів (кожний поліном степеня N задається у вигляді деякого бітового рядка).

Уведемо також набір масивів $GRP[1..N],KOB2[1..N2],IOB[1..M]$ – допоміжних програмних елементів.

Звернемо увагу, що в даному алгоритмі також використовуються тимчасові допоміжні змінні.

Змінною W задається значення груп ознак, номери яких зберігаються в масиві $GRP[1..N]$.

7.1.1. Схема розрахунку інформативності наборів ознак

Крок 1. Етап вводу початкових значень: $M, N, N2,\ OB[1..N, 1...M]$, $KOB[1..N2]$, $HB[1..MX]$, $IJB = 0$, $M2 = 2 * N$, $N5 = 64$.

Крок 2. Формування випадкового бітового рядка B. $IJB = IJB + 1$.

Крок 3. $GRP = 0, K1 = 0, IWJ = 0, N4 = 0, N3 = N5 - N + 1$.

Крок 4. $I = N3 - 1$.

Крок 5. $I = I + 1, N4 = N4 + 1$.

Крок 6. Якщо $i-$ товий біт рядка B дорівнює 1, то $K1 = K1 + 1$, $GRP[K1] = N4$, якщо $I < N3 + N$, то перейти на *Крок 5*.

Крок 7. $L2 = 0$, $IOB = 0$, $I = 0$.

Крок 8. $I = I + 1$. Якщо $IOB[I] = 1$, то перейти на *Крок 17*.

Крок 9. $I1 = 1$, $KOB2 = 0$, $J = 0$.

Крок 10. $J = J + 1$. Якщо $J = KOB[I1]$, то $I1 = I + 1$. Якщо $IOB[J] = 1$, то перейти на *Крок 14,* $K = 0$.

Крок 11. $K = K + 1$. Якщо $OB[J, GRP[K]] > OB[I, GRP[K]]$, перейти *Крок 14*.

Крок 12. Якщо $K < K1$, то перейти на *Крок 14*.

Крок 13. $IOB[J] = 1$, $KOB[I1] = KOB2[I1] + 1$.

Крок 14. Якщо $J < M$, то перейти на *Крок 10*.

Крок 15. Знайти максимальне значення в масиві $KOB2$ та видати його IW.

Крок 16. $IWJ = IWJ + IW$.

Крок 17. Якщо $I < M$, то перейти на *Крок 8*.

Крок 18. $W = IWJ/M$. Вивести значення GRP та W.

Крок 19. Якщо $IBJ < M2$, то перейти на *Крок 2*.

Крок 20. Очистити всі змінні та масиви, звільнити пам‘ять.

Крок 21. Кінець роботи алгоритму (END).

Слід відмити, що при незначній модифікації даний алгоритм можна використовувати для широкого кола задач розпізнавання образів:

- знаходження всіх тестів за даними $I(l)$;
- знаходження тестів фіксованої довжини;
- знаходження наперед заданої кількості тестів;
- знаходження деяких тестів за обмежений час;
- розв'язування задач геологічного прогнозування;
- визначення якості кодування інформації $I(l)$;
- мінімізація вихідного опису $I(l)$;
- знаходження ознак для розпізнавання образів;
- розв'язування деяких соціологічних задач тощо.

7.1.2. Набір задач розпізнавання на основі схеми обчислення інформативності ознак

На наступному етапі зупинимося більш детально на цих та деяких інших застосуваннях запропонованого алгоритму.

1. Знаходження всіх тестів за даними початкової НВ. Відмітимо, що ця задача, як зазначалося вище, вирішується за допомогою даного алгоритму, якщо на *Кроці 18* помістити умову перевірки змінної W на 1. Якщо $W = 1$, то вивести значення GRP, в іншому випадку – перейти на *Крок 19*. При цьому формування групи ознак $P_{i_1}, \dots, P_{i_\xi}$, яку перевіряють на тест, проходить випадковим чином. Для скорочення процедури перебору можна використовувати наступні тривіальні властивості тестів:

a) якщо деяка підмножина ознак не є тестом, то жодне її звуження (власна підмножина) також не представляє собою тест;

b) якщо деяка множина ознак представляє собою тест, то всяке її розширення також являється тестом.

Отже, ці властивості потрібно використовувати на *Кроці 2* представленого алгоритму при формуванні бітового рядка. Важливо, що формування бітового

рядка на *Кроці 2* може носити довільний характер, при цьому будемо отримувати різні алгоритми знаходження тестів.

2. Знаходження тестів фіксованої довжини. Найпростіший шлях для розв'язання даної задачі за допомогою запропонованого вище алгоритму – це *Крок 2* доповнити наступними умовами: якщо в бітовому рядку кількість одиниць не співпадає із заданою фіксованою довжиною тесту, то перейти на *Крок 19*. Можна замінити *Крок 2* в даному алгоритмі на схему формування бітового рядка з фіксованою кількістю одиниць. *Крок 18* залишається таким самим, як і при розв'язуванні попередньої задачі. Відмітимо, що можливі інші модифікації даного алгоритму, проте *Кроки 3-17* залишаються без змін.

3. Знаходження наперед заданої кількості тестів. Як вже підкреслювалось у другому розділі даного дослідження – при розв'язуванні деяких практичних задач знайти всі тести не представляється можливим, а, інколи, і нема в цьому необхідності. Тому виникає потреба в алгоритмах знаходження наперед заданої кількості тестів. Представлений вище алгоритм дозволяє вирішити таку задачу за наступною схемою: на *Кроці 18* необхідно ввести лічильник кількості знайдених тестів та при досягненні заданого числа перейти на *Крок 20*.

4. Знаходження тестів за обмежений час. Відмітимо, що ця задача вирішується так само, як і задача 3. На *Кроці 18* повинна бути умова перевірки використаного та наперед заданого часу (фіксованого часового інтервалу). Якщо використаний час перевищує або дорівнює заданим обмеженням, то перейти на *Крок 20*.

5. Знаходження важливості окремих ознак та груп ознак (фіксованої довжини, визначеної якості, за наперед заданий час). Ця задача вирішуються аналогічно описаним вище задачам 1 – 4 з урахуванням того, що на *Кроці 18* не потрібна перевірка змінної W на 1.

6. Розв'язання задач геологічного прогнозування. Як вже вказувалося в роботі [272], розв'язування цих задач часто зводиться до наступних трьох етапів:

a) обчислення ознак, що розділяє вагу матриці $T^{'}$ ($T^{'}$ – частковий випадок $I(l)$;

b) обчислення середньої ваги еталонів, що описують відповідні стани, $\overline{\beta_i}, 1 \le i \le r$;

c) обчислення ваги $\beta_i^{'}$ заданого еталону.

Зауважимо, що вирішення питання про приналежність еталону до тієї чи іншої групи зводиться до порівняння $\overline{\beta_i}$ та $\beta_i^{'}$, $1 \le i \le r$, при заданій точності $\overline{\beta_i}$ та $\beta_i^{'}$ – знаходяться за всіма тупиковими тестами. Можливість знаходження тестів за допомогою запропонованого алгоритму, та розв'язок представлених задач 1 – 5, дозволяє розширити область застосування тестових методів при розв'язуванні задач геологічного прогнозування. Причому важливість ознак можна знаходити за допомогою функціонала (7.3).

7. Визначення якості кодування інформації в $I(l)$. Часто при розв'язуванні задач розпізнавання образів в якості експериментального матеріалу відбирають весь масив даних, що вдається виміряти. При цьому виникає принципове питання виділення інформативних ознак та груп ознак. Відмітимо, що вирішення даного питання можливе за допомогою функціонала (7.3). Для побудови правил класифікації, в багатьох випадках, цю інформацію потрібно закодувати, наприклад, у вигляді двійкових, k – значних або дійсних векторів, оскільки багато алгоритмів розпізнавання застосовуються тільки для початкової інформації $I(l)$, представленої таким чином. Проте, як вже відмічалося, при кодуванні можливі помилки та великі втрати інформації. На основі функціонала (7.3) можна запропонувати один підхід до розв'язання даної задачі, та дати методику знаходження ефективних (в сенсі роздільної здатності класів) алгоритмів кодування інформації $I(l)$.

Нехай задано деякий алгоритм A, який інформацію $I(l)$ типу (7.1) переводить в $I(l)$ типу (7.5).

$$\left(\left(x_1^{'}, f_R(x_1^{'})\right), \dots, \left(x_m^{'}, f_R(x_m^{'})\right)\right) \qquad (7.5)$$

Зауважимо, що $x_i^{'} \in G^{'}, G^{'} \in G$ (в якості G може бути множина, що складається з нулів та одиниць, з $(0,1,\dots,K^{'}-1)$, де $K^{'}-$ деяке ціле число, $f_R(x_i^{'}) \in \{0,1,\dots,K-1\}, (i=1,2,\dots,m)$).

Нехай потрібно оцінити якість алгоритму A. Якщо під якістю кодування розуміти степінь збереження відмінностей між $H_0, H_1, \dots, H_{K-1}$, то ця задача вирішується наступним чином. Потрібно знайти значення функціоналу (7.3) для всіх ознак інформації $I(l)$ типу (7.4). Якщо це значення дорівнює 1, то алгоритм A вважається ефективним. Якщо $W\left(P_{i_1}, \dots, P_{i_\xi}\right) \leq 1$, то відмінність між $H_0, H_1, \dots, H_{K-1}$ не збережено в процесі кодування, тому A потрібно замінити іншим алгоритмом і для нього проробити ту ж саму процедуру. Таким чином можна знайти ефективні алгоритми кодування у вище визначеному сенсі.

Проте, при розв'язуванні деяких практичних задач цієї якості замало. Тому введемо інше можливе визначення якості кодування $I(l)$ типу (7.1). Під якістю кодування $I(l)$ типу (7.1) будемо розуміти степінь збереження тестів, що були раніше. Ця задача легко вирішується за допомогою функціоналу (7.3) за наступною алгоритмічною схемою:

1) На першому етапі потрібно знайти визначену кількість тестів для початкової інформації $I(l)$ типу (7.1).

2) На другому етапі, після процедури кодування потрібно перевірити степінь збереження тестів за $I(l)$ типу (7.5). Якщо відсоток тестів для $I(l)$ типу (7.5) є по відношенню до $I(l)$ типу (7.1) не меншим, то алгоритм кодування вважати якісним, інакше необхідно обрати інший алгоритм кодування та повторити вищеописану процедуру.

Відмітимо також, що можливі і інші підходи щодо оцінки якості кодування початкової інформації $I(l)$, наприклад, пов'язані з інформативністю та тестовістю (груп ознак). Зрозуміло, що не завжди потрібно шукати всі можливі тести $I(l)$, а перевірка на тестовість інформації $I(l)$ типу (7.5) за допомогою функціонала (7.3) відбувається дуже швидко. Важливо, що таким

же чином можна знаходити і кількість градацій для кожної фіксованої ознаки при кодуванні інформації $I(l)$.

8. Мінімізація вихідного опису $I(l)$. При розв'язуванні задач розпізнавання образів насамперед доводиться стикатися з проблемою опису об'єктів, які підлягають класифікації, тобто з вибором системи ознак, які характеризують ці об'єкти. Такий опис має бути мінімальним (містити як можна менше ознак), але разом з тим достатнім для вирішення поставленої задачі з потрібною точністю та надійністю.

Відмітимо, що дана задача вирішується разом із задачею пункту 7 шляхом відкидання найменш інформативних ознак та груп ознак, що знайдені за допомогою функціонала (7.3), та введення деяких вимог щодо кількості ознак, що зберігаються при виконанні ними деяких наперед заданих вимог.

9. Знаходження ознак на зображеннях. При розв'язуванні даної задачі описані вище функціонали можна використовувати для знаходження ознак на дискретних зображеннях.

10. Розв'язання соціологічних задач. Так, на основі функціоналу (7.3) розв'язувалася наступна задача з галузі психології: навчаюча інформація була представлена у вигляді (7.1), причому ознаки об'єктів приймали як якісні, так і кількісні значення (випадок різнотипних атрибутів з етапом зведення та кодування). Функція $f_R(x)$ задавала степінь вираження волі та приймала від трьох до п'яти значень. За допомогою функціонала (7.3) знаходилася важливість кожної ознаки окремо по відношенню до $f_R(x)$ та оцінювалася важливість різних груп ознак. Були виділені також найбільш інформативні ознаки і групи ознак, за якими будувалися різні правила класифікації.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) Часто розв'язок різнотипного набору практичних задач розпізнавання дискретних образів зводиться до проблематики оцінки важливості ознак та їх наборів (можливих комбінацій).

2) Важливим питанням залишається задача кодування початкової інформації $I(l)$. Однин із шляхів вирішення якої полягає в схемі розбиття $n-$ вимірного простору (який обмежений можливими значеннями ознак), гіперплощинами, на $n-$ мірні гіперпаралелепіпеди так, щоб елементи (об‘єкти) з різних класів не попадали в один і той самий паралелепіпед (задача геометричного обмеження) з присвоєнням кожному з даних геометричних об‘єктів фіксованого двійкового коду.

3) Представлений в даному підрозділі програмний алгоритм обчислення важливості ознак (груп ознак) за допомогою запропонованого функціонала дозволяє вирішувати широке коло наведених практичних задач.

7.2. Загальна схема програмної побудови структури ЛДК

Відмітимо, що довільне ЛДК (для бінарного випадку) можна досить просто представити в програмному форматі у вигляді наступних базових елементів – масивів:

1) $T[1..MN]$, $SLED[1..MN]$, $UPOD[1..MN]$, де MN – кількість вершин ЛДК;

2) $T[1..MN]$ – масив, що характеризує інформацію, яка знаходиться у вершинах ЛДК (перелік міток, атрибутів, вершин структури дерева);

3) $SLED[1..MN]$ – масив, що вказує на місце наступної вершини, яка знаходиться праворуч і є нащадком даної вершини в структурі ЛДК;

4) $UPOD[1..MN]$ – масив, що характеризує інформацію про місце лівого нащадка даної вершини.

Зауважимо, що тут $MN < 2N - 1$, де M – загальна кількість всіх об‘єктів початкової інформації $I(l)$. Відмітимо також, що крім вказаних масивів для побудови ЛДК у даній програмній реалізації використовуються набір наступних допоміжних масивів – $PR[1..N]$, $PR1[1..N]$, $SPR[1..N]$.

Отже, для програмної побудови ЛДК необхідно $M * N$ байт оперативної пам’яті (в самому простому випадку) для зберігання початкової інформації $I(l)$, а також $3 * MN * (6M - 1)$ байт – для зберігання масивів N, $SLED$, $UPOD$ та $3N$ байт для зберігання масивів PR, $PR1$, SRP (знову ж таки – байт тільки для найпростішого випадку). На наступному етапі розглянемо приклад та через нього представимо загальну схему алгоритму програмної побудови фіксованого ЛДК.

Приклад 2. Нехай маємо деяке ЛДК, яке має фіксовану чотириярусну структуру (рис. 7.1), причому дане дерево однозначно визначається наступною інформацією базових масивів T, $SLED$ та $UPOD$, представлених у табличній формі (табл. 7.1).

Слід зауважити, що типи всіх представлених масивів – це масиви байтів (зрозуміло, що це залежить від інформаційної ємності міток самого ЛДК,

типізації атрибутів логічного дерева, початкових умов самої задачі, типу та об‘єму початкової інформації $I(l)$).

Таблиця 7.1. Інформаційний вміст масивів T, $SLED$ та $UPOD$

T	7	5	1	0	3	1	2	1	0	0	4	1	0
$SLED$	0	3	0	5	0	7	0	9	0	11	0	13	0
$UPOD$	2	4	6	0	8	0	10	0	0	0	12	0	0

Зрозуміло, що в даному прикладі фактична структура отриманого ЛДК представляється за допомогою вмісту трьох основних масивів. Отже, маючи сформовані ці три масиви можна графічно відтворити структуру побудованого програмно ЛДК.

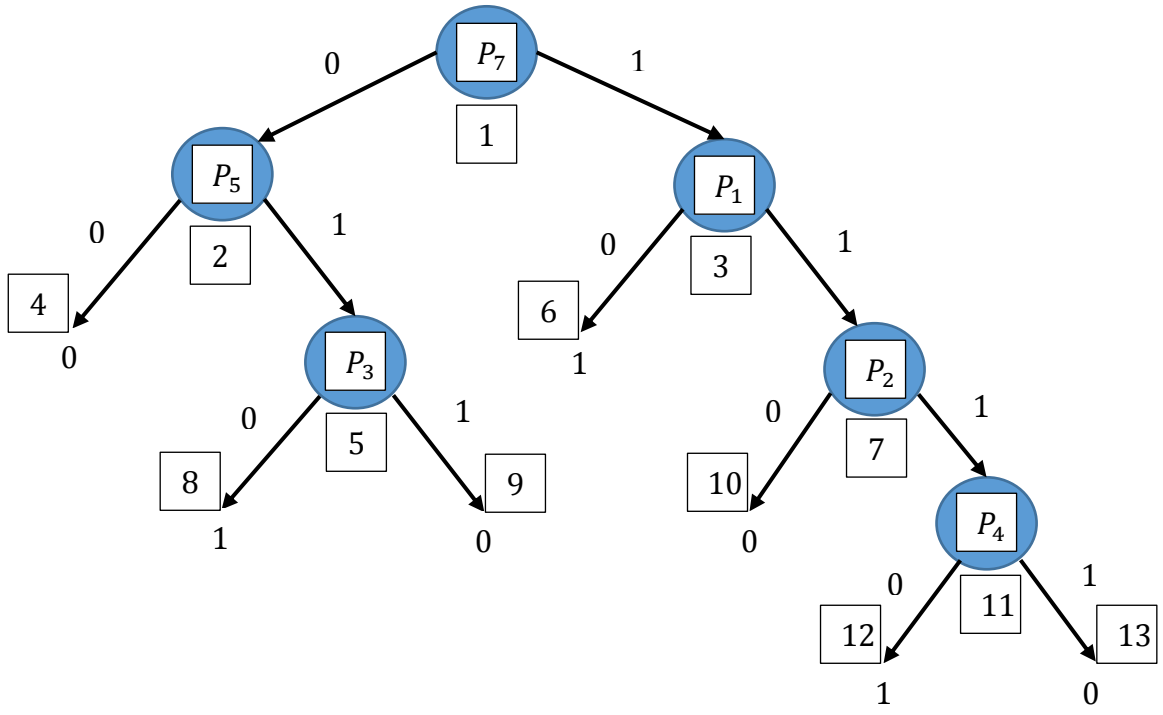


Рис. 7.1. Загальний вигляд початкового ЛДК.

Зважаючи на все вищезазначене, запропонуємо наступну алгоритмічну реалізацію побудови ЛДК за даними початкової інформації $I(l)$.

Загальна схема алгоритму побудови ЛДК за даними початкової НВ.

Крок 1. Вибір шляху в ЛДК, що не приводить в кінцеву вершину. $UPOD = 1, PR = 0, TT = 2, KI1 = 0, SLED = 0, MMM1 = M * 2 - 1.$

Крок 2. $TT = TT + 2$. Якщо $TT = 0$, то перейти на *Крок 9*.

Крок 3. $KI = KI + 1$. Якщо $UPOD[KI] \geq 0$, то перейти на *Крок 7*. $KII = KI$. Якщо $KII = 1$, то перейти на *Крок 9*.

Крок 4. Якщо $SLED[KII] > 0$, то: $(SRP[KI1 + 1] = 0,\ PAR = KI)$, в іншому випадку: $(SRP[KI1 + 1] = 1, PAR = KII - 1, KII = KII - 1)$.

Крок 5. $IK2 = KII + 1$.

Крок 6. $KI2 = KI2 - 1$. Якщо $UPOD[IK2] = PAR$, то: $(KI1 = KI1 + 1,$ $PR1\big[T[IK2]\big] = 1, PR1[KI1] = T[IK2], KII = IK2)$. Якщо $KII = 1$, то перейти на *Крок 9*. Якщо $KII = 0$, то перейти на *Крок 5*.

Крок 7. Якщо $KII < TT$, то перейти на *Крок 3*.

Крок 8. $IJKL = 0$.

Крок 9. Якщо $PR[1] = 0$ та $TT = 0$, то перейти на *Крок 13*.

Крок 10. (Етап обчислення значень функціонала (7.3)). Обчислити значення функціоналу для всіх P_i, що входять в область його визначення та знайти ознаку, яка має найбільшу інформативність (якість, важливість). Інформація щодо $P_{i_1}, \dots, P_{i_\xi}$ та $\eta_{i_1}, \dots, \eta_{i_\xi}$ знаходиться в масивах PR, SRP відповідно.

Крок 11. На основі *Кроку 10* заповнити відповідні значення масивів T, $SLED$, $UPOD$.

Крок 12. Перейти на *Крок 2*.

Крок 13. Етап розпізнавання НВ за побудованим ЛДК.

Крок 14. $I = I + 1, II = 1$.

Крок 15. $I1 = T[II]$. Якщо $UPOD[II] = 0$, то $(PASP[I] = I1$. Перейти на *Крок 16*). Якщо $TB[I, I1] = 0$, то $(II = UPOD[II]$. Перейти на *Крок 15*). $II = UPOD[II] + 1$. Перейти на *Крок 14*.

Крок 16. Якщо $I < M$, то перейти на *Крок 14*.

Крок 17. Очистити всі змінні та масиви, звільнити пам‘ять.

Крок 18. Закінчити роботу алгоритму (END).

Відмітимо, що в запропонованому алгоритмі заслуговують уваги наступні часові характеристики:

1) загальний час побудови результуючого ЛДК залежить від об'єму початкової інформації $I(l)$, тобто при збільшенні потужності початкової інформації $I(l)$ час на побудову ЛДК являється лінійною функцією від $I(l)$;

2) загальний час прийняття рішень (на основі програмно побудованої структури даного ЛДК) не перевищує n простих порівнянь, де $n-$ кількість ознак у початковій інформації $I(l)$ (фактичний час проходження за фіксованим шляхом у структурі ЛДК).

Зауважимо, що якщо в приведеному вище алгоритмі *Крок 10* замінити на виклик процедури генератора випадкових чисел (PRG) для вибору деякої ознаки, то даний алгоритм буде будувати множину випадкових дерев розпізнавання, над якими можна буде здійснювати операції замикання (питання ВДК вже піднімалося в попередніх розділах даного дослідження).

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) В одному з варіантів, загальну структуру довільного ЛДК можна досить просто представити в програмному форматі у вигляді трьох базових елементів (набору трьох масивів).

2) Відмітимо, що для програмної побудови ЛДК необхідно $M * N$ байт оперативної пам'яті (для простого випадку) для зберігання початкової інформації $I(l)$, а також $3 * MN * (6M - 1)$ байт – для зберігання трьох базових масивів та $3N$ байт для зберігання трьох допоміжних.

3) Зауважимо, що час побудови результуючого ЛДК залежить від об'єму початкової інформації $I(l)$ (являється лінійною функцією від $I(l)$), а час прийняття рішень за побудованим ЛДК не перевищує n простих порівнянь, де $n-$ кількість ознак у початковій інформації $I(l)$.

4) Відмітимо, що запропонований вище алгоритм дозволяє забезпечити ефективний механізм програмної побудови фіксованого ЛДК за набором деяких початкових даних (НВ).

7.3. Особливості програмної реалізації моделей ЛДК на основі селекції наборів елементарних ознак

Відмітимо, що станом на сьогодні день відомо біля трьох десятків готових ПС для побудови різних типів моделей дерев класифікації (дерев рішень) у вигляді структур ЛДК (RStudio, RulQuest, DTTL v1.5, RLQTree, DCT v7, Precision Tree System, Edraw, SHAIDS, Weka, ЛАСТАН, АУРОН та інші) та лише одна ПС, яка базується на концепції АДК (ОРІОН). Усі ці системи відрізняються прикладною спрямованістю задач, що розв‘язуються, методами та концептуальними засадами, різноманітним рівнем підтримки, причому багато з них знаходять у вільному (або частково вільному) доступі. Домінуючими підходами, як вже підкреслювалося раніше, є системи на основі методів CART (спрямованих для розв‘язку задач класифікації та регресивного аналізу), а також ПС на основі схеми C4.5 та її сучасних модифікацій (для розв’язання задач розпізнавання та класифікації) та ID3 – причому всі вони характеризуються наступними критеріями розгалуження та зупинки процедури побудови дерева:

1) ID3 схема базується на використанні обмеженого ентропійного критерію – структура ЛДК будується до тих пір, поки для кожної результуючої вершини (листа дерева) не залишаться лише об‘єкти одного фіксованого класу, або доки сама процедура розгалуження в дереві, що будується, дає зменшення початкового ентропійного критерію.

2) C4.5/C5.0 схема базується на відомому критерії *Gain-Ratio* (нормативний ентропійний критерій), причому в якості критерію зупинки процедури розгалуження (побудови дерева) використовується обмеження на кількість об‘єктів для результуючої вершини (листа структури ЛДК). Відмітимо, що процедура відсікання в структурі ЛДК проводиться за схемою *Error-Based Pruning*, яка базується на загальній оцінці здатності узагальнення для прийняття рішення щодо видалення гілок та вершин конструкції дерева класифікації. В даному підході обробка пропущених атрибутів (ознак) здійснюється за схемою, в якій ігноруються об‘єкти з відсутніми значеннями

в процедурі розрахунку критерію розгалуження структури ЛДК, а на наступному етапі відносить такі об‘єкти до обидвох піддерев із визначеними атрибутами.

3) CART схема в своїй роботі використовує критерії Джині, причому процедура і відсікання в структурі ЛДК проводиться за схемою *Cost-Complexity Pruning*, а для випадку наявних пропусків атрибутів використовується базова схема сурогатних предикатів.

Відмітимо, що алгоритм ID3 є однією з найпростіших схем для отримання дерев рішень із категоріальними класами та атрибутами (на основі ентропійного критерію), причому саме на основі ID3 і був написаний пізніше алгоритм C4.5. Хоча існує достатньо багато реалізаційних схем різних методів та підходів дерев рішень (дерев класифікації), однією з найбільш вживаних та такою, що забезпечує необхідну ефективність, є алгоритмічна схема C5.0 (подальший розвиток концепції алгоритму C4.5), причому вона стала фактично галузевим стандартом для побудови моделей дерев класифікації, оскільки підходить для більшості типів задач безпосередньо прикладного характеру в сегменті аналізу різнотипної інформації. В порівнянні з більш досконалими та складними моделями машинного навчання (наприклад концепцією нейронних мереж) дерева класифікації в рамках схеми C5.0 зазвичай забезпечують не гіршу ефективність та точність, але в той самий час більш підходять для зовнішнього аналізу та фінальної покрокової інтерпретації (виділенні правил класифікації), тобто є достатньо простими та зрозумілими для експерта та спостерігача [302-308].

Зрозуміло, що існують й інші схеми (алгоритми та їх програмні реалізації) побудови структур дерев класифікації – наприклад LightGBM платформа (бібліотека) градієнтного бустингу, що використовує алгоритм навчання на основі дерева, причому визначальною особливістю LightGBM є те, що структура дерева рішень будується вертикально, в той час як всі інші, згадані в даному дослідженні, роблять це горизонтально (крім структур АДК фіксованих типів) [114, 145, 163]. Це означає, що LightGBM будує дерево

рішень за кожним вузлом (листом), в той час як інші алгоритми будують конструкцію за структурними рівнями (ярусами). Відмітимо, що головною особливістю LightGBM є його спрямованість на масиви даних (НВ) великого та надвеликого об‘єму (можливість роботи з вибірками об‘ємом від 10000 об‘єктів і більше), а також дуже ефективний менеджер пам‘яті та даних. Іншою особливістю LightGBM є те, що він орієнтований в основному на точність результатів (моделі ЛДК будуються відповідно до заданої умовами задачі точності).

Відмітимо, що в фреймворку XGBoost реалізований механізм штрафування моделі, що будується, як на основі параметру регуляризації *LASSO (L1)*, так і *Ridge* – регуляризації *(L2)*, для того, щоб уникнути значного переускладнення фінальної структури дерева. Особливістю є те, що в даному фреймворку є можливість роботи з розрідженими даними (різного ступеня), в процесі навчання заповнюючи пропущені атрибути (ознаки, значення) в залежності від параметру втрат моделі, причому дозволяється робота з різними схемами розрідженості масивів навчальних даних. Варто звернути увагу, що XGBoost використовує метод зважених квантилів для того, щоб найбільш ефективно визначати оптимальні точки розгалуження (поділу множин) у разі роботи з великими масивами даних, причому в XGBoost реалізований модифікований метод фінальної крос – валідації на кожному кроці побудови структури дерева, моделі (тобто немає потреби в окремій реалізації цього пошуку з визначенням кількості циклів бустингу для кожного запуску) [309-314].

7.3.1. Програмна система побудови дерев класифікації DeTree

Звичайно, що представленими ПС та фреймворками не обмежуються програмні реалізації концепції дерев рішень. Так, в Ужгородському національному університеті на основі представленої вище в дослідженні схеми побудови ЛДК (селекції наборів елементарних ознак) було розроблено ПС DeTree, яка базується на концепції розгалуженого вибору ознак та дозволяє працювати з НВ великого та надвеликого об‘єму (рис. 7.2). Причому

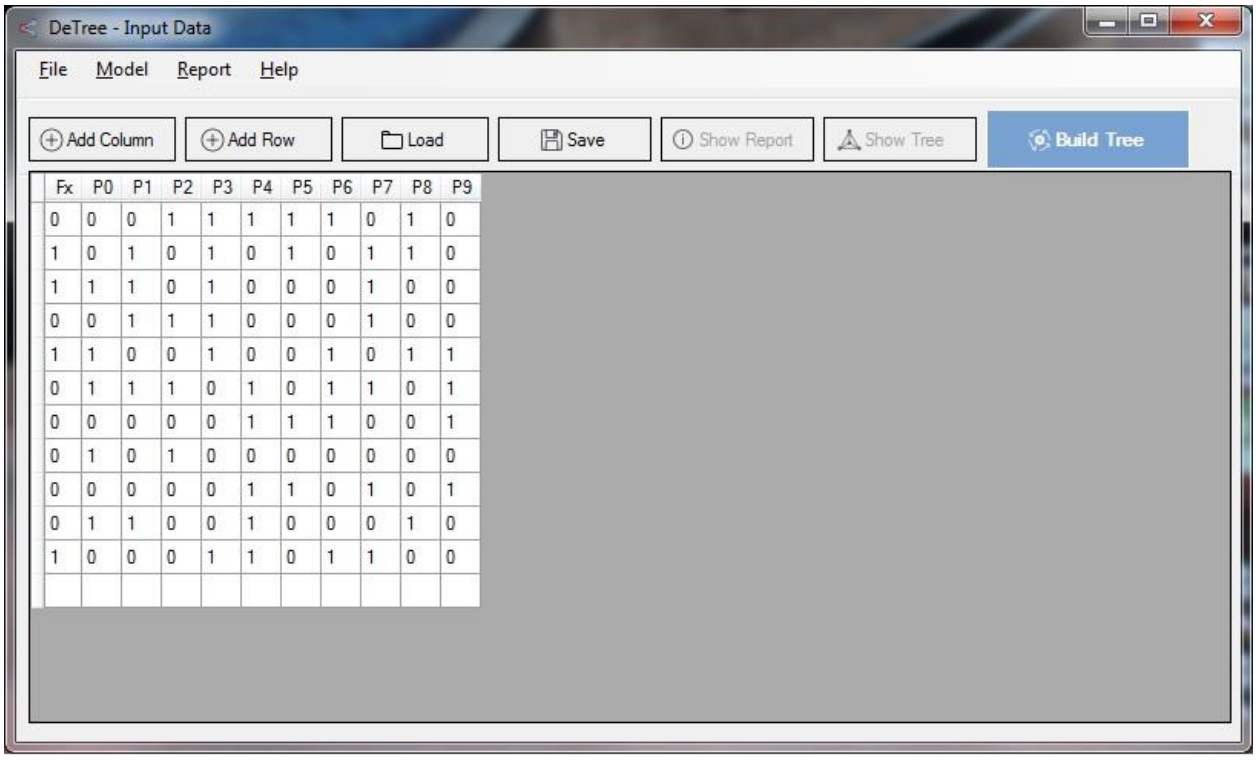
на початковому етапі проектування ПС ставилися наступні базові вимоги щодо загального функціоналу системи:

- значна увага приділялася оптимізації алгоритмів побудови моделі класифікації для досягнення максимальної швидкості – як генерації, так і роботи самої побудованої моделі ЛДК (*Runtime speed/ Operation speed*);
- вимога якісної та ефективної роботи з оперативною та постійною пам'яттю інформаційної системи в зв‘язку з спрямованістю на масиви початкових даних великого об‘єму;
- простий та зручний інтерфейс для оператора, обов‘язкова наявність автоматичного (заснованого на різних алгоритмах) та інтерактивного режиму генерації моделей ЛДК;
- вимога простого портування готової ПС на інші апаратно/програмні платформи в перспективі;
- Вимога можливості постпроцедурної корекції та донавчання побудованої структури дерева класифікації;
- Вимога можливості роботи з НВ вибірками великого та надвеликого об‘єму.

Відповідно до даних вимог було вирішено розділити програмний продукт на два базові проекти (компоненти) – *DeTreeBackend* та *DeTree,* причому для зручності компонент *DeTreeBackend* представляється як бекенд, а *DeTree* – як фронтенд. У даній схемі ПС *DeTree* компонент *DeTreeBackend* забезпечує функціонал усіх базових обчислень (усіх схем обробки даних та менеджера пам‘яті) та додатково містить у собі набір класів та алгоритмів низькорівневого функціоналу. Компонент *DeTree* в свою чергу реалізує повний функціонал роботи з боку користувача (інтерактив) та забезпечує роботу довідкової та сервісної служб. Якщо розглядати реалізаційну схему програмного продукту з точки зору патерну *MVC* (*Model-View Controller*), проєкт фронтенду (*DeTreeBackend*) виконує функції *View* та відповідає за базовий ввід та вивід даних, причому контролером в цій реалізаційній схемі виступає безпосередньо проєкт бекенду (*DeTree*). За основу базових потоків

даних в програмному продукті фіксувалася схема вводу та виводу даних з файлів для некритичних по швидкості ділянок коду (для максимальної економії оперативної пам‘яті). Така організація обчислень дозволяє достатньо ефективно розділяти логіку програмного забезпечення на окремі логічні підсистеми (компоненти).

В якості інструментарію розробки для компонента (*DeTree*) було обрано мову програмування C++ (з додатковою можливістю портування коду), причому такий вибір у значній мірі пояснюється можливість компіляції в нативний код та можливостю роботи з низькорівневими задачами – такими як робота з оперативною пам'яттю, функціонал вводу/виводу даних. Також такий вибір інструментарію розробки дозволяє забезпечити ефективну побудову основної логіки програми, а її базовий код відносно просто може бути портований на різні апаратно/програмні системи.



Fx	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9
0	0	0	1	1	1	1	1	0	1	0
1	0	1	0	1	0	1	0	1	1	0
1	1	1	0	1	0	0	0	1	0	0
0	0	1	1	1	0	0	0	1	0	0
1	1	0	0	1	0	0	1	0	1	1
0	1	1	1	0	1	0	1	1	0	1
0	0	0	0	0	1	1	1	0	0	1
0	1	0	1	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	1	0	1
0	1	1	0	0	1	0	0	0	1	0
1	0	0	0	1	1	0	1	1	0	0

Рис. 7.2. Загальний інтерфейс ПС DeTree.

Для компонента *DeTreeBackend* було обрано інструментарій C# з метою простоти створення графічних інтерфейсів для платформи Windows (середовище розробки *Microsoft Visual Studio*) – рис. 7.3.

Frontend Overview (DeTree). Відмітимо, що проєкт фронтенду складається з чотирьох основних вікон (*Forms*) та двох базових класів, причому форми забезпечують функції користувацького інтерфейсу та відображення різнотипної робочої інформації в процесі побудови структур ЛДК. Серед основних форм можна виділити наступні:

- форма поточного статусу роботи програми;
- форма базового вводу – виводу інформації задачі та даних інтерактивного режиму генерації ЛДК;
- форма візуалізації згенерованої структури моделі ЛДК для автоматичного та інтерактивного режиму роботи ПС;
- форма сервісної візуалізації проміжних процедур обробки даних у процесі побудови структури ЛДК.

В якості основного компоненту базового вводу даних обрано графічний елемент – *DataGrid*, що дозволяє досить гнучко та просто працювати з динамічними табличними даними (генерувати та корегувати таблиці різної розмірності та типу, виконувати первинні перевірки коректності вхідних даних та забезпечувати електронний підпис елементів табличних даних унікальними цифровими ідентифікаторами). Додатково в структурі компонента базового вводу даних для даної форми доступний функціонал завантаження та зберігання табличних даних у найбільш розповсюджених форматах даних (*cvs/cur/bin/dat/txt/html*), причому ця форма містить верхню панель інструментів відповідного завантаження/збереження даних, інструменти зміни типу/розмірності таблиці вхідних даних, інструменти візуалізації результатів обчислень у текстовому та графічному вигляді, а також безпосередньо інструмент генерації структури ЛДК. Відмітимо, що форма візуалізації побудованого дерева (структури ЛДК) відображає згенероване зображення з відрендереним деревом класифікації, причому доступний функціонал, який дає змогу застосовувати основні функції зміни масштабу зображення та інструменти збереження його в графічній формі (основних графічних форматах). Додатково вікно з візуалізатором процесу обчислень

проміжних даних задачі показує поточний лог розрахунку даних ЛДК на стороні компонента бекенду, а до функціоналу даної форми можна віднести можливість збереження опису обчислень у системний лог – файл для наступної перевірки (корекції параметрів моделі) та аналізу. Так, структура компонента фронтенду містить два базові класи: один з яких – для роботи з імпортом та експортом файлів у форматі даних (CSV, HTML та ін.), інший – для безпосереднього керування процесом обчислень на стороні бекенда. Відмітимо, що робота з бекендом проводиться безпосередньо за допомогою базового класу *Process* бібліотеки *System.Diagnostics*, саме за його допомогою створюється системний процес, який запускає компонент бекенду з набором основних аргументів командного рядка.

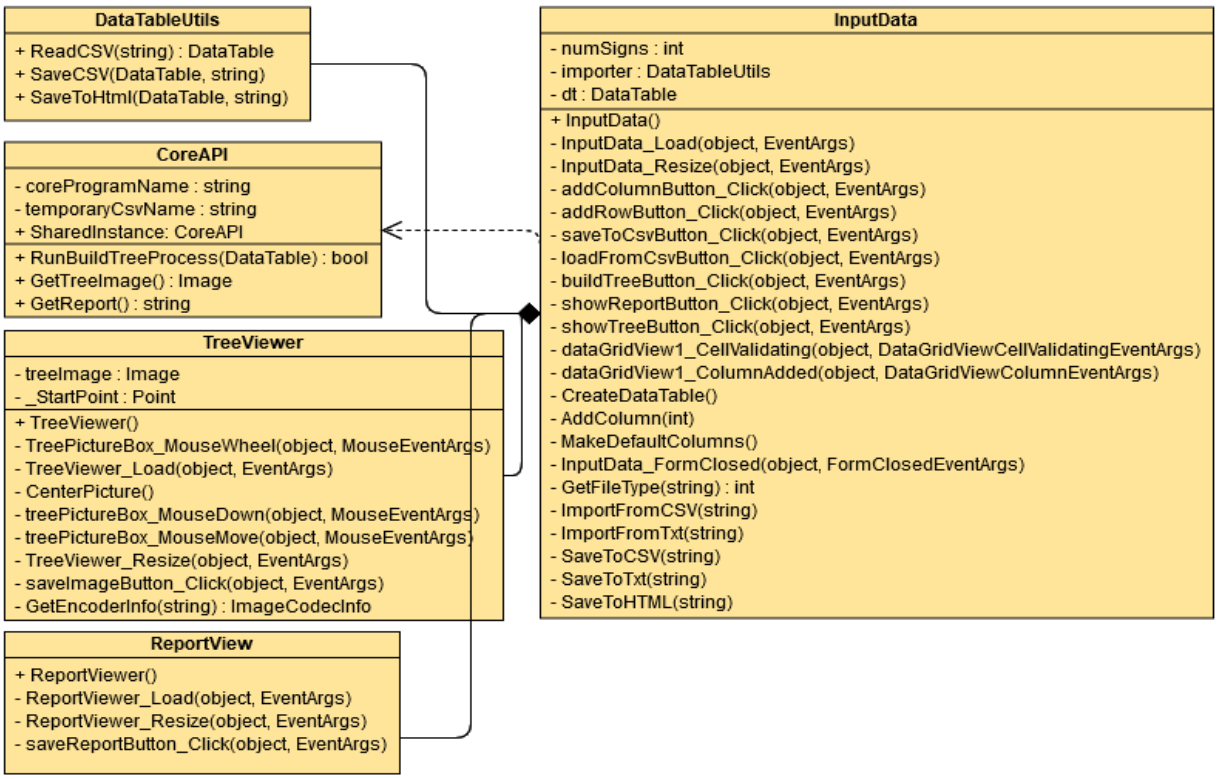


Рис. 7.3. UML – схема організації компонентів DeTree.

Так, перед запуском процесу побудови дерева класифікації, компонент фронтенду формує пакет вхідних даних – цей процес проводиться за допомогою конвертування даних графічного елемента *DataGrid* з типу *DataTable* в файл формату CSV, а після конвертування даних, процес побудови

дерева класифікації може завантажити файл на своїй стороні та провести необхідні обчислення (рис. 7.4).

Backend Overview (DeTreeBackend). Як вже наголошувалось вище – проект бекенду написаний на мові програмування C++ та є класичним консольним додатком, який забезпечує базовий функціонал усіх математичних обчислень даних (алгоритмічних схем), збір та роботу з інформацією на їх основі. Основний комунікаційний інтерфейс з нею реалізований, як варіант, через набори аргументів командного рядка, відповідно сформовані пакети даних, так і через графічний інтерфейс форми фронтенда. Відмітимо, що на даний момент бекенд складається з базових 14 класів та 12 структур даних. Для прикладу, клас *InputParser* описує загальну логіку роботи для аналізу та розбиття наборів вхідних аргументів командного рядка на токени, цей клас забезпечує можливість швидкої перевірки та обробки вхідних параметрів командного рядка ПС, перевірки коректності їх значень. Наприклад для завантаження даних ознак (атрибутів) у набори аргументів командного рядка використовується синтаксис вигляду *(-f <filename>)*, де відповідний клас парсеру перевіряє наявність аргументу вхідного файлу та дає можливість отримати безпосередньо значення шляху.

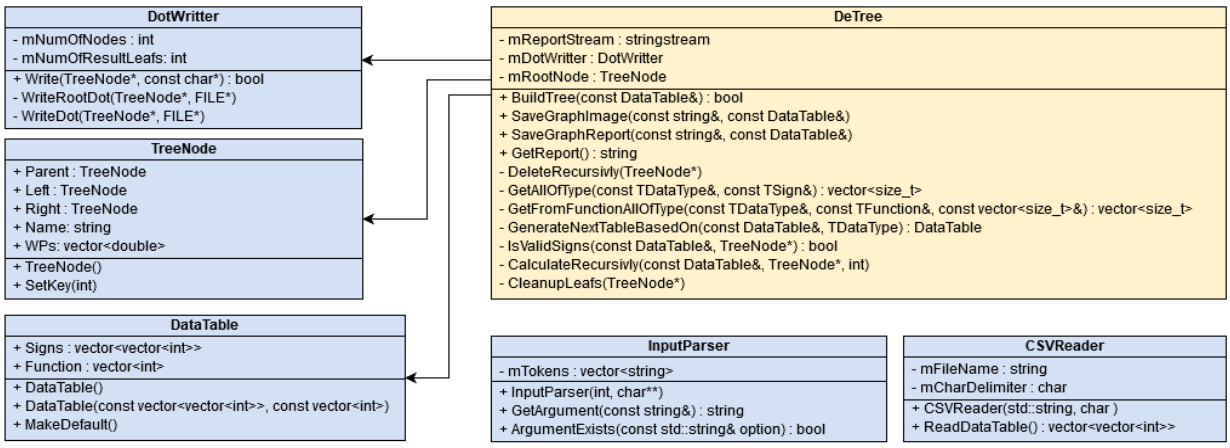


Рис. 7.4. UML – схема бекенду DeTree.

Після етапу обробки та аналізу набору вхідних параметрів та сформованих пакетів даних, наступає етап первинного завантаження масиву даних НВ. Основна схема інтерфейсу комунікації даних між компонентами

фронтенду та бекенду полягає через посилання абсолютних шляхів файлів даних у форматі CSV (в залежності від налаштувань ПС). Така схема комунікації при передачі даних між компонентами ПС пояснюється значною ефективністю, зручністю в можливості корекції даних та простоті процесу організації. Для завантаження даних в форматі CSV в компоненті бекенду реалізований відповідний клас *CSVReader,* який забезпечує повний функціонал задачі завантаження, базової валідації, корекції та формування внутрішнього представлення CSV таблиць масивів даних. Для прикладу, в найпростішому випадку дані представляються за допомогою внутрішнього псевдоніма *TDataTable,* який в свою чергу описується вектором векторів (*std::vector<std::vector<int>>*).

Після етапу завантаження, всіх процедур перевірки даних та їх трансформації у внутрішній формат ПС, система переходить до безпосередньої побудови дерева (структури ЛДК). На етапі проектування ПС всю внутрішню логіку, пов'язану з обчисленнями та аналізом даних, було винесено в базовий клас *DeTree*, причому для побудови дерева класифікації в структурі даного класу використовується основний метод *BuildTree*. Послідовність кроків побудови структури дерева класифікації (моделі ЛДК) можна визначити наступним порядком дій:

1) Підготовчий етап початкового вибору, ініціалізація базових параметрів та налаштувань ПС під реалії конкретної прикладної задачі відповідно до умов генерації структури (моделі) ЛДК – критеріїв розгалуження та критеріїв зупинки побудови дерева класифікації.

2) Початковий етап валідації наборів вхідних даних поточної задачі (перевірка на коректність різних типів даних НВ та ТВ), вибір та перевірка на коректність основних режимів (параметрів) роботи процедури генерації дерева класифікації.

3) Етап забезпечення первинних процедур запиту, виділення та розподілу оперативної пам‘яті системи під центральну вершину (*Root Node*),

вузли та переходи структури дерева класифікації (робота менеджера пам‘яті ПС).

4) Етап роботи рекурсивної процедури обрахунку набору величин інформативності (важливості) атрибутів (ознак) $W(P_i)$ відповідно до обраного автоматично або в інтерактивному режимі критерію розгалуження структури ЛДК.

5) Етап генерації структури ЛДК (вузлів дерева класифікації) відповідно до зафіксованих критеріїв розгалуження та зупинки побудови дерева класифікації.

6) Етап роботи процедури відсікання (оптимізації структури ЛДК) для мінімізації структурних компонентів (вузлів, блоків) побудованого дерева класифікації.

7) Етап фінальної перевірки основних параметрів побудованої структури ЛДК (моделі класифікації) в автоматичному або інтерактивному режимі залежно від початкових налаштувань ПС.

8) Етап постпроцедурного аналізу побудованої моделі (структури ЛДК), етап декомпозиції синтезованого дерева класифікації з виділенням та збереженням окремих правил класифікації.

Відмітимо, що процедура рекурсивного обчислення величин інформативності атрибутів базується на основному методі *CalculateRecursivly*, а весь функціонал вузлів конструкції ЛДК працює на основі базової структури вузла дерева (*Node*) *TreeNode*. У спрощеному випадку структура *TreeNode* складається із списку інформаційної оцінки атрибутів $W(P_i)$, літерального імені для компонента візуалізації фрагменту ЛДК та унікального цифрового ключа, причому в залежності від розташування ця структура містить набір посилань на батьківський, лівий та правий вузли (блоки) конструкції ЛДК (відповідні структури).

Функцію візуалізації побудованих структур ЛДК (генерації зображень як повних дерев класифікації, так і їх окремих компонентів) було покладено на формат даних *Dot*, який входить у відкритий програмний продукт *GraphViz*

(бібліотека з відкритим програмним кодом), саме на її основі будується графічне представлення дерева класифікації (візуалізації складних граф – схемних представлень) і в багатьох інших подібних програмних продуктах. Для побудови файлу в форматі *Dot* було розроблено базовий клас *DotWritter*, який формує текстове представлення файлу формату *Dot* та на основі рекурсивної процедури обчислення даних заповнює його інформацією на базі побудованого дерева класифікації (структури ЛДК).

На наступному етапі роботи ПС – після завершення всіх операції розрахунків даних, генерації всіх структур файлу формату *Dot* та формування пакетів даних, базовий клас *DeTree* зберігає всі побудовані набори даних у поточну робочу папку ПС у спеціальному найменуванні (форматі) та викликає зовнішню утиліту (компонент) програмного пакету *GraphViz*, яка на основі файлу формату *Dot* будує графічне (граф – схемне) представлення дерева класифікації (побудованої моделі класифікації). На наступному кроці – після успішного виконання всіх попередніх етапів функціонування ПС – компонент фронтенду відстежує подію коректного в плані формування пакетів даних та безпомилкового завершення процесу бекенду та переходить до процедури фінального завантаження та візуалізації всіх результатів розрахунків поточної задачі (моделі класифікації).

Звернемо увагу, що ПС DeTree має можливість збереження як графічного та параметричного представлення побудованої моделі дерева класифікації (структури ЛДК), так і допоміжних даних (проміжних етапів) синтезу конструкції дерева, що дозволяє провести ефективний аналіз самої процедури синтезу дерева класифікації (моделі) та знайти можливі варіанти його фінальної оптимізації (обрізки структури). Одним із варіантів аналізу побудованої структури ЛДК (моделі класифікації) може бути режим роботи ПС DeTree – *step-by-step model decomposition*) – який дозволяє провести поетапне виділення правил класифікації з конструкції ЛДК (шляхів в структурі дерева класифікації або фіксованих Т – опорних множин) для їх подальшого аналізу та збереження. Зрозуміло, що такий підхід буде

актуальний для прикладних задач класифікації з НВ великого та надвеликого об‘єму та у разі великої складності фінальної моделі класифікації (структури ЛДК) після етапу остаточної оптимізації, мінімізації (обрізки конструкції дерева).

7.3.2. Етап експериментальної перевірки ПС побудови ЛДК

Для перевірки побудованого програмного забезпечення використовувалась відома задача про тип лісового покриву (ліс – 581012 елементів масиву вибірки). Структура вибірки (НВ та ТВ) містить сім класів розбиття (типи можливого лісового покриву), причому об‘єкт класифікації представляється як послідовність 12 числових ознак та, додатково, двох багатозначних дискретних атрибутів. Зауважимо, що половина початкового масиву вибірки (а саме: 290506 об‘єктів відомої класифікації) відводилась для навчання системи, а інша частина – для тестування побудованих моделей ЛДК. Загальні дані про задачу та безпосередньо сам масив вибірки для перевірки можна отримати з ресурсу *UCI KDD Archive* (http://kdd.ics.uci.edu).

Відмітимо, що алгоритм C4.5 та його ідеологічний нащадок C5.0 були скомпільовані за допомогою відкритого компілятора GCC (для системи *Linux*) з набором однакових параметрів оптимізації та компіляції бінарного коду. В якості базової операційної системи використовувалася *Fedora Linux 23 (RedHat)*, а для емуляції та запуску *Win32 API* (розробленої ПС DeTree) прошарок *Wine* (бібліотека *libwine*). Всі заміри часу проводились в секундах, а системи побудови структур ЛДК було запущено на двох різних апаратних конфігураціях:

- *Configuration №1* – Intel Core I7 7700K / Ram 16 GB;
- *Configuration №2* – AMD FX8370 / Ram 16 GB.

Основні результати тестування приведено в порівняльних таблицях: табл. 7.2 - 7.4.

Відмітимо, що наведені алгоритмічні схеми C4.5, C5.0 та ПС DeTree можуть генерувати як класифікатори (набори правил класифікації), так і працездатні моделі (структури) ЛДК. Здебільшого в задачах аналізу даних

набори правил класифікації є кращим варіантом з точки зору експерта за рахунок простоти та наочності ніж структури ЛДК, але з іншого боку таке представлення даних є повільним і вимагає значно більшої оперативної пам'яті.

Таблиця 7.2. Порівняння схем побудови ЛДК за кількістю помилок класифікації, кількістю правил класифікації, та часом побудови ЛДК.

Алгоритмічна схема	*Загальна кількість помилок* - Er_{All}	*Загальна кількість правил класифікації* - Rt_{All}	*Загальний час генерації ЛДК* - T_{All}
C4.5	7.2%	5420	Config. №1 – 34 с. Config. №2 – 42 с.
C5.0	6.3%	4845	Config. №1 – 186 с. Config. №2 – 230 с.
DeTree	6.7%	5028	Config. №1 – 102 с. Config. №2 – 129 с.

Відмітимо, що побудовані схемою C5.0 набори правил (модель ЛДК) мають помітно нижчу частоту помилок в порівнянні з C4.5 та DeTree, причому мають однакову точність готової моделі, але набір правил у моделі C5.0 незначно але менший. Щодо часу генерації структур ЛДК (та правил класифікації) зафіксуємо, що алгоритм C4.5 набагато швидший за рахунок простоти схеми, достатньої оптимізації та обмежень щодо процедури фінальної обрізки (*pruning*). Так, витрати оперативної пам‘яті схеми C5.0 зазвичай на порядок менше ніж у C4.5: при побудові набору правил (моделі ЛДК) було використано 230 MB, а для C4.5 – витрачено 4.3 GB, що все одно більше ніж у ПС DeTree – 180 MB. Принциповою особливістю схеми C5.0 в порівнянні з іншими такими алгоритмами побудови дерев класифікації є наявність механізму бустингу (*booosting*). Під бустингом для схеми C5.0 будемо розуміти загальний метод генерації та заключного об'єднання декількох побудованих класифікаторів для підвищення точності класифікації.

Таблиця 7.3. Порівняння схеми бустингу для алгоритму C5.0.

Первинне ЛДК алгоритму C5.0	*ЛДК алгоритму C5.0 на основі бустингу*	*Набір первинних класифікаторів алгоритму C5.0*	*Набір класифікаторів алгоритму C5.0 на основі бустингу*
6.7%	3.8%	6.2%	3.6%

Зауважимо, що схема C5.0 підтримує прямий бустинг з будь-якою кількістю ітерацій, причому більша кількість ітерацій зазвичай призводить до подальших покращень якості синтезованих класифікаторів. Зрозуміло, що на створення гібридних класифікаторів (на основі процедури бустингу) витрачається значно більше часу, але виграш у додатковій точності моделі може виправдати додаткові витрати процесорного часу. Отже, процедуру бустингу завжди слід проводити, коли потрібна максимальна точність (якість) класифікації, особливо, коли первинні класифікатори вже дають непогану точність.

Зауважимо, що загальна схема процедури бустингу в алгоритмі C5.0 не є принципово складною, що дозволяє реалізацію подібних методик (алгоритмів та схем) навіть більшої складності (ефективності) для ПС DeTree та C4.5 у вигляді окремого програмного компоненту. Тим не менше механізм бустингу є якісним та ефективним атрибутом роботи системи порівняно з іншими подібними ПС побудови дерев рішень.

Таблиця 7.4. Порівняння схем побудови ЛДК за фіксованою точністю, кількістю вузлів та часом побудови структури ЛДК.

Алгоритмічна схема	*Загальна кількість помилок - Er_{All}*	*Загальна кількість вузлів ЛДК - Vt_{All}*	*Загальний час генерації ЛДК - T_{All}*
C4.5	6.8%	10167	Config. №1 – 57 с. Config. №2 – 43 с.
C5.0	6.8%	9201	Config. №1 – 62 с. Config. №2 – 51 с.
DeTree	6.7%	1012	Config. №1 – 50 с. Config. №2 – 47 с.

Звернемо увагу, що схеми C4.5, C5.0 та DeTree синтезують моделі ЛДК з аналогічною прогнозною точністю для запропонованої задачі (наборів НВ), але алгоритм C5.0 показує трохи більший час роботи при побудові дерев класифікації при майже однаковій кількості помилок на початковій вибірці, причому основні відмінності полягають у розмірах та структурах побудованих дерев класифікації і часу обчислень при побудові готової моделі – структура дерева C5.0 помітно менше, але витрати часу C5.0 є відносно більшими.

Звернемо увагу, що схема C5.0 включає в себе кілька нових функцій, таких як змінні (параметри) втрат на неправильну класифікацію, що фактично дозволяє вводити додаткові типи помилок класифікації з різною вартістю (якістю та ціною). В схемі C4.5 та ПС DeTree всі помилки розглядаються як рівні ціною (якістю, кінцевим впливом), але в прикладних задачах деякі особливі помилки класифікації (різних типів) можуть бути більш серйозними (важкими), ніж інші (звичайно в залежності від специфіки задачі). Так, алгоритм C5.0 дозволяє визначити (врахувати) окрему вартість (ціну впливу) для кожної помилки (довільного типу) моделі, що будується – якщо використовується ця опція (в схемі алгоритму), тоді будуються класифікатори для мінімізації очікуваних витрат на неправильну класифікацію, а не загальної частоти помилок структур ЛДК, причому самі помилки класифікації (випадки) також можуть мати неоднакове значення (ціну). В схемі C5.0 передбачено відповідний параметр (атрибут) ваги випадку (помилки класифікації), який кількісно та якісно визначає важливість кожного такого випадку (фактично схема C5.0 намагається мінімізувати зважену частоту помилок прогнозування).

Більшість сучасних ПС та комплексів інтелектуального аналізу даних характеризуються дуже високою універсальністю щодо розмірності та структури НВ – з сотнями і тисячами атрибутів (ознак). Принциповою особливістю схеми C5.0, яка відсутня в C4.5 та DeTree (але може бути додатково реалізована окремим модулем), є автоматичне відсіювання атрибутів (ознак) перед побудовою класифікатора, які характеризуються лише

незначною кореляцією (релевантністю). Для НВ великого та надвеликого об‘єму така початкова корекція масивів даних може призвести до зменшення складності класифікаторів та підвищення точності розпізнавання, а також часто може скоротити час, необхідний для побудови наборів правил класифікації. Звернемо також увагу на простоту використання системи DeTree та C5.0. Програмний інструмент RuleQuest (з відкритим кодом) дозволяє забезпечити процедури зчитування та інтерпретації класифікаторів схем See5/C5.0 (See5 – відкрита програмна реалізації алгоритму C5.0 на Java). Після того, як класифікатори (моделі дерев класифікації) були побудовані алгоритмами See5/C5.0, ця система дає можливість отримати до них доступ з інших незалежних ПС.

Отже, зважаючи на все вищезазначене, можна зафіксувати наступні пункти:

1) На сьогоднішній день відомі десятки ПС побудови різних типів моделей дерев класифікації (дерев рішень) у вигляді структур ЛДК та лише одна ПС, яка базується на концепції АДК, причому всі ці системи відрізняються прикладною спрямованістю задач, що розв‘язуються, методами та концептуальними засадами, різноманітним рівнем підтримки, причому багато з них є у вільному (або частково вільному) доступі.

2) Домінуючими підходами методів та схем дерев рішень є системи на основі методів CART (спрямованих для розв’язання задач класифікації та регресивного аналізу), ПС на основі схеми C4.5/C5.0 та їх сучасні модифікації/реалізації (для розв’язання задач розпізнавання та класифікації) та платформи (набори алгоритмів) прямого та градієнтного бустингу (бібліотеки LightGBM та XGBoost).

3) У зв‘язку з тим, що структури дерев класифікації після побудови за вибірками реальних даних великого об‘єму мають здебільшого складну для аналізу та неоднорідну за рівнями (ярусами) структуру, то принциповою проблемою залишається питання організації процедури оптимізації або обрізки (*pruning*) таких конструкцій. Значна ефективність того чи іншого

алгоритму або схеми дерев класифікації в значній мірі визначається ефективною реалізацією (ефективністю реалізованих алгоритмів) саме цього компонента.

4) Відмітимо, що, не зважаючи на високу ефективність в практичній площині, наявність якісного механізму оптимізації, мінімізації побудованих структур дерев класифікації, схема C5.0 не позбавлена і певних системних недоліків, які обов‘язково потрібно враховувати як при реалізації, так і при роботі з побудованими моделями ЛДК.

5) Фреймворки XGBoost та LightGBM – набір методів та алгоритмів концепції дерев рішень, які використовують принцип послідовного бустингу (в найпростішому випадку, схема побудови бінарного дерева рішень) на основі алгоритму градієнтного спуску, причому бібліотека XGBoost розглядається як вдосконалена версія фреймворку LightGBM через системну оптимізацію та вдосконалення алгоритмів. Схема XGBoost (найсучасніше переосмислення загальної концепції бустингу) базується на так званій схемі екстремального градієнтного бустингу та спрямована на значну програмну та апаратну оптимізацію обчислень при побудові моделей дерев класифікації.

6) ПС DeTree базується на концепції розгалуженого вибору ознак (поетапної селекції ознак) та дозволяє працювати з НВ різнотипної інформації широкого спектру прикладних задач, причому на початковому етапі проектування ПС ставилися базові вимоги щодо загального функціоналу системи – наявність ефективного менеджеру пам‘яті системи, спрямованість на роботу з масивами даних великого та надвеликого об‘єму, вимога ефективності та оптимізації коду системи, вимога простоти інтерфейсу оператора, вимога наявності автоматичного та інтерактивного режимів роботи системи, вимога кросплатформності готової системи, вимога корекції та донавчання готової моделі класифікації (структури ЛДК).

7) Одним із варіантів аналізу побудованої структури ЛДК (моделі класифікації) може бути режим роботи ПС DeTree (*step-by-step model decomposition*), який дозволяє провести поетапне виділення правил

класифікації з конструкції ЛДК для їх подальшого дослідження та збереження, причому зрозуміло, що такий підхід буде актуальний для прикладних задач класифікації з НВ великого та надвеликого об‘єму та у разі великої складності фінальної моделі класифікації (структури ЛДК) після етапу остаточної оптимізації, мінімізації (обрізки конструкції дерева).

7.4. Модель алгоритмічного дерева для задачі класифікації гідрографічних даних

Одним із можливих напрямків застосування концепції алгоритмічних дерев класифікації є задачі, пов‘язані з прогнозуванням та класифікацією паводкових явищ (на основі масивів метеорологічних та гідрографічних даних). Відомо, що паводкові явища Карпатського регіону завдають значної шкоди як економічному, так і екологічному сектору державного господарства та бізнесу: підтоплюють та виводять з господарського обороту значні сільськогосподарські угіддя, населені пункти, руйнують житловий сектор і промислові підприємства, греблі, транспортні комунікації, трапляються і людські жертви – що є актуально для Закарпатського регіону. Тому всебічне дослідження умов формування паводкових ситуацій у розрізі класифікації метеорологічних (гідрографічних) явищ при сучасних мінливих кліматичних умовах є необхідним етапом для подальшого обґрунтування нових методик для розрахунків та прогнозів подолання кризових екологічних ситуацій.

Відмітимо, що режим поверхневого водного стоку басейну річки Уж Закарпатського регіону характеризується значною територіальною нерівномірністю, обумовленою кліматичними, температурними факторами та впливом гірських масивів Карпат, які мають визначальний вплив для формування сезонних стоків. Для аналізу умов та причин формування паводкових явищ на річці Уж Закарпатської області в осінньо – весняний період використовувалися дані 2–х гідрологічних (спостережних) постів із періодом моніторингу від початку 1992 і по 2010 р. включно (на ділянці спостереження 30 км). Відмітимо, що за своїм водним режимом, річку Уж можна віднести до річок із паводковим режимом Карпатського підтипу, де паводкові ситуації спостерігаються переважно на часовому відрізку осінньо – весняного періоду. Зазвичай, такі паводкові ситуації називають паводками холодного періоду, причому вони спостерігаються з жовтня/листопада по квітень/травень. Зрозуміло, що з поправкою на сучасні кліматичні зміни – межі теплого і холодного періодів не є постійними з року в рік, та в основу

визначення їх початку і кінця було покладено в основному базові метеорологічні умови (температура повітря, види атмосферних опадів та стану снігового покрову).

У даному дослідженні головну увагу буде приділено басейну річки Уж загальною довжиною 133 км, загальна площа басейну якої складає 2750 км² (причому в межах України довжина – 107 км та загальна площа – 1950 км²). Спостерігаються перепади долини від 15 м (у верхів‘ях басейну) до 100 – 350 м, у пониззі перепад висот сягає від 2 до 2.5 км. Форма заплави має переривисту структуру, часто спостерігається асиметрія, завширшки 50 - 500 м, у нижній течії – до одного кілометра, причому річище звивисте, має помірну кількість розгалужень, відповідно у верхній та середній течії річки спостерігаються пороги, наявні формації невисоких водоспадів, та різноманітних форм островів у руслі річки. Так, середня ширина русла річки коливається в межах 15 – 35 м, причому в межах міста Ужгород може доходити до 140 м, загальний похил русла 7.35 м/км. Важливим параметром є пересічний водний стік, який сягає до 31 м³/с. Береги басейну ріки мають круту структуру (так і в межах міста) в середньому до 2.5 м, в окремих випадках пересічного рельєфу – до 10 м, дана формація має звичайну кам‘янисту структуру крім меж міста та ділянок біля берегів (спостерігається значне замулення). Визначальною характеристикою є основний спосіб водного живлення, який є загальним – сніговим (дощовим). Для басейну річки Уж важливим є те, що на відміну від паводків теплого періоду, які можуть мати місце в будь-який час року і виникають лише внаслідок надмірного випадання дощів і злив (поточних параметрів типових опадів), осінньо – весняні паводки мають змішану природу та утворюються в результаті накладання явищ сніготанення з одночасним випаданням дощів [315-317]. Важливою особливістю басейну ріки Уж, на відміну від інших річкових басейнів західного регіону України, є утворення паводкових явищ змішаного походження (на основі сніжних та дощових стоків, найчастіше з перевагою дощового компоненту), які відбуваються в більшості спостережень в осінньо

– весняний період року, причому такий режим водності зумовлений кліматичними особливостями Закарпаття, наявністю гірських масивів, руху атмосферних фронтів та інших факторів. Підкреслимо, що в басейнах рік Закарпатського регіону дощові (сніжно – дощові) паводкові явища різної інтенсивності та тривалості повторюються з періодичністю до 4 - 6 разів на календарний рік спостережень. З іншого боку слід відмітити, що внаслідок інтенсивного випадання снігу (відповідного процесу сходу снігу) для басейну річки Уж спостерігаються виключно снігові паводкові явища, причому слід зазначити, що річні максимуми тільки за рахунок талих неводних артерій Закарпатського регіону є доволі частим явищем. Так, річні максимуми паводків холодного періоду часто значно перевищують максимуми відповідно теплого періоду, причому паводкові явища холодного періоду більш тривалі в часі та відбуваються частіше, а максимальні водні витрати паводків холодного періоду більші за максимальні витрати теплого майже в 2.5 рази (звичайно в такій схемі по окремим рокам є певні виключення). Особливістю басейну річки Уж є те, що максимальна інтенсивність підйому води складає від 1.5 – 2.5 м на добу, в окремих випадках (1992 р.) даний показник може перевищувати і 3 м на добу.

7.4.1. Параметри задачі класифікації паводкових явищ річки Уж

Початкові параметри даної прикладної задачі класифікації паводкових явищ річки Уж представлені в табл. 7.5.

Таблиця 7.5. Початкові параметри задачі класифікації паводкових явищ басейну річки Уж.

Номер поста спостере-ження №	*Розмірність ознакового простору – N*	*Потужність масиву даних початкової НВ – M*	*Потужність масиву даних ТВ – S*	*Загальна кількість класів за розбиттям даних НВ – l*	*Відношення об'єктів різних класів НВ – $(H_1/H_2/H_3)$*
1	18	6889	500	3	73/105/6711
2	18	6736	500	3	68/97/6571

Загальний водний стік ріки треба розглядати не тільки як певний гідрологічний фактор, але і як геоморфологічний (це варто враховувати на

етапі моніторингу та вибору ознакового простору спостережень). Так, у табл. 7.6 представлено інформацію щодо паводкових явищ на об‘єкті спостереження по роках проведення моніторингу.

Таблиця 7.6. Паводкові явища для річки Уж за роками проведення моніторингу.

	Нейтральна зона (пост №1)	*Жовта зона (пост №1)*	*Червона зона (пост №1)*	*Нейтральна зона (пост №2)*	*Жовта зона (пост №2)*	*Червона зона (пост №2)*
1992	341	14	11	339	13	11
1993	327	6	3	323	5	3
1994	347	4	3	334	4	3
1995	354	6	4	352	5	3
1996	351	4	2	350	4	2
1997	356	5	4	349	5	4
1998	345	11	9	341	10	8
1999	357	5	3	342	4	3
2000	359	4	3	351	4	3
2001	360	3	2	358	3	2
2002	357	5	3	343	5	3
2003	353	7	5	350	6	4
2004	361	3	2	356	3	2
2005	358	4	3	352	3	3
2006	360	3	2	345	3	2
2007	357	5	3	340	5	3
2008	357	5	4	342	5	3
2009	355	6	3	351	5	2
2010	356	5	4	353	5	4

Відмітимо, що моніторинг басейну ріки має специфічні особливості (може бути навіть унікальним у межах певної місцевості спостереження), які обумовлені тим, що ріка є одним із досить динамічних об'єктів природи, причому вимагає в значній мірі індивідуального підходу (обумовленого географією, умовами формування, водним режимом тощо). Отже, моніторинг

русла ріки пов'язаний із великим простором (загальним басейном ріки) та значними витратами часу, зміною періодичності спостережень, причому до завдань посту спостереження входить повна ознакова характеристика не тільки виключно гідрологічних об‘єктів, а й загальних геолого-геоморфологічних умов басейну ріки. Відмітимо, що загальна модель паводкового явища описується на основі 18 ознак (атрибутів), які мають різну природу та формуються на основі багаторічних спостережень басейну річки Уж, причому серед даних ознак можна виділити наступні визначальні характеристики (набори параметрів) річкового басейну:

1) Загальна густина річкового покрову сектору спостереження – відношення довжини всіх поверхневих водостоків русла річки (в кілометрах) до величини загальної площі басейну (в квадратних кілометрах) у відповідному секторі спостереження. Відмітимо, що даний параметр розраховується окремо для кожного поста спостереження.

2) Коефіцієнт кривизни русла річки на ділянці спостереження є відношенням фактичної довжини ділянки річки (ділянки спостереження) до довжини по прямій від витоку до гирла.

3) Параметр типу річних терас – ознака, яка характеризує геологічні особливості будови русла річки на ділянці спостереження та приймає значення з набору $\{1, \dots, 6\}$.

4) Параметр середньої ширини русла (СШР) річки на ділянці спостереження – для басейну річки Уж характеризується різкими перепадами величин залежно від поста спостереження, де проводяться виміри.

5) Параметр умовного рівня води (УРВ) – зазвичай для річок в якості умовного рівня приймається середній рівень вод, що спостерігався протягом року в період, коли річка вільна від льодового покрову, причому цей рівень визначається як середнє арифметичне зі щоденних значень рівня за меженний період (від спаду повені до початку льодових явищ). Приблизно визначають період і висоту низького, постійного та проміжного рівня, який приймається в якості умовного (початкового), а на наступному етапі розраховується

перевищення робочого рівня над умовним. У випадку перевищення використовується як величина зростання для приведення всіх поточних спостережень до умовного рівня.

6) Параметр середньої глибини русла (СГР) річки на ділянці спостереження, для річки Уж характеризується досить великими розбіжностями для різних ділянок спостереження.

7) Параметр загального режиму ріки – ознака, яка характеризує особливості водного живлення та стоку русла річки на ділянці спостереження та приймає значення з набору $\{1, \dots, 10\}$.

8) Параметр температурного режиму ріки (ТРР) – ознака, яка характеризує особливості температурного режиму ріки на ділянці спостереження та приймає значення з набору $\{1, \dots, 50\}$.

9) Параметр середньої швидкості течії ріки (СШТ) – атрибут, який характеризує середній показник швидкості течії води в середині русла на ділянці спостереження.

10) Параметр максимальної поверхневої швидкості течії ріки – відрізняється від параметру СШТ та вимірюється на постійній основі для ділянки спостережень русла ріки.

11) Показник середніх добових витрат води річки на ділянці спостереження в розрахунку на 24 години (показник розраховується один раз на добу) – розрахунок може проводитись не на основі поточних вимірів, а на базі параметра СШТ.

12) Параметр типу рельєфу русла ріки – ознака, яка характеризує ступінь складності рельєфу русла ріки на ділянці спостереження та приймає значення з набору $\{1, \dots, 20\}$.

На основі набору представлених гідрографічних характеристик побудовано моделі класифікації паводкових явищ для річки Уж за 19-річний період (1992 - 2010) у вигляді структур (моделей) АДК. З цією метою використано ПС «Оріон ІІІ» для генерації автономних систем розпізнавання та класифікації, де алгоритмічна бібліотека системи нараховує 15 алгоритмів

(методів та схем розпізнавання). Основний масив НВ складався з об‘єктів (кожний з яких описується 18 ознаками) трьох базових класів, а на етапі екзамену побудована система класифікації (модель АДК) забезпечувала ефективне розпізнавання об‘єктів невідомої класифікації відносно цих трьох класів. Загальні параметри (характеристики) побудованих структур (моделей АДК) представлено в табл. 7.7.

Таблиця 7.7. Загальні параметри побудованих моделей АДК.

Номер побудованої моделі АДК	*Метод (алгоритм) побудови моделі АДК*	*Загальна кількість різних алгоритмів класифікації застосованих в АДК* - N_{All}	*Загальна кількість УО (наборів УО) в структурі АДК* O_{Uz}	*Загальна кількість усіх вершин (разом із результуючими) в структурі АДК*	*Загальний час генерації структури АДК поточної НВ*
Пост спостереження №1					
1	Метод повного АДК (тип I)	5 обмеження на послідовне використання по одному алгоритму	42	84	686 с.
2	Метод повного АДК (тип II)	5 обмеження на кількість генерацій УО на один крок побудови АДК	43	98	712 с.
3	Обмежений метод АДК (Z=10)	1 алг. гіперсфер	35	71	406 с.
4	Обмежений метод АДК (Z=5)	1 алг. гіперкуба	58	117	839 с.
5	Обмежений метод АДК (Z=3)	1 алг. гіпереліпс	37	75	442 с.
6	Метод повного АДК (тип I)	2 алг. гіперсфер алг. Гіперкуба	40	81	706 с.

Пост спостереження №2					
1	Метод повного АДК (тип I)	5 обмеження на послідовне використання по одному алгоритму	39	79	676 с.
2	Метод повного АДК (тип II)	5 обмеження на кількість генерацій УО на один крок побудови АДК	40	95	701 с.
3	Обмежений метод АДК (Z=10)	1 алг. гіперсфер	34	69	380 с.
4	Обмежений метод АДК (Z=5)	1 алг. гіперкуба	57	115	828 с.
5	Обмежений метод АДК (Z=3)	1 алг. гіперпаралелепіпед	48	98	793 с.
6	Метод повного АДК (тип I)	2 алг. гіперсфер алг. гіперкуба	38	77	695 с.

На початковому етапі роботи програмної системи навчальну вибірку було автоматично перевірено на коректність (пошук та видалення однакових об‘єктів різної належності – помилки першого роду). Зауважимо, що в масиві початкової навчаючої інформації переважали навчальні пари класу H_3 (об‘єкти ситуаційного стану нейтральної зони, зеленого маркеру), на другому місці зі значним відривом за кількістю знаходились навчальні пари класу H_2 (об‘єкти ситуаційного стану спостережної зони, жовтого маркеру) і на третьому місті знаходились безпосередньо навчальні парі паводкових явищ (об‘єкти червоного маркеру) – класу H_1. Звернемо увагу, що потужність класу H_2 незначно переважає потужність класу H_1, це пояснюється динамікою зміни паводкової ситуації в часі, яка може повертатися до нормального стану (нейтральної зони) – явищ класу H_3, а здебільшого переходить у кризовий стан

(червону зону паводкового явища) –клас H_1 (рис. 7.5). Масив НВ складався з 8391 об‘єктів (наборів відомої класифікації) для двох пунктів моніторингу на ділянці міста Ужгород, причому ефективність сконструйованої системи розпізнавання оцінювалася на тестовій вибірці об‘єму 500 об‘єктів на кожному з постів спостереження, причому масив ТВ представляв собою відокремлену частину початкової НВ (складався з дискретних об‘єктів відомої класифікації).

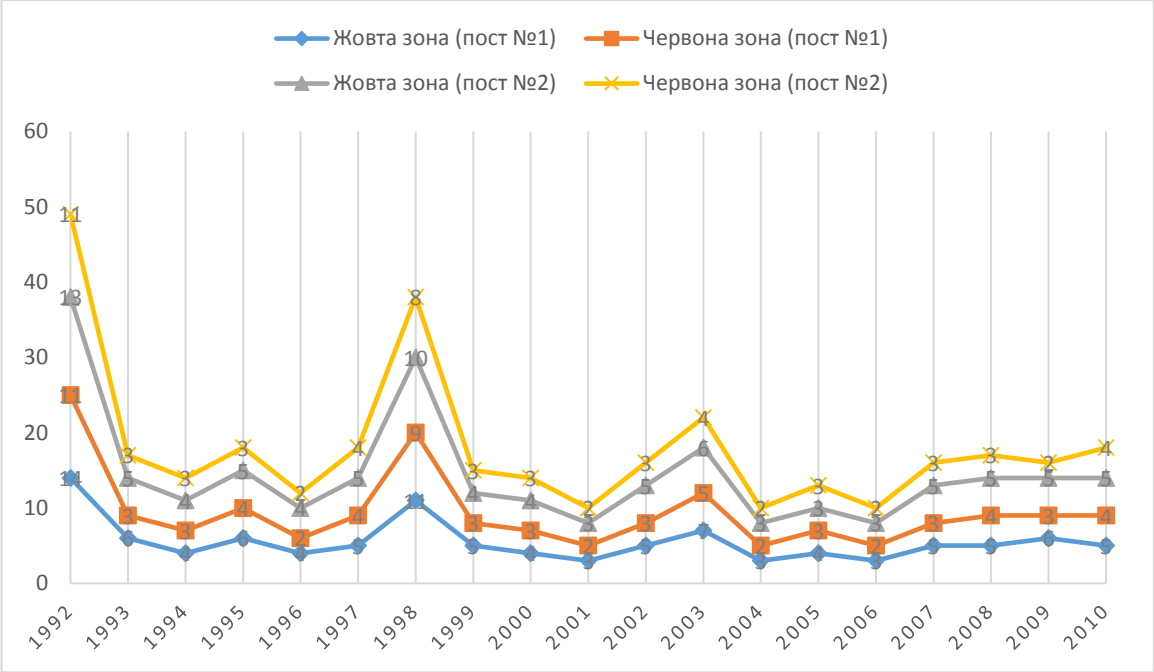


Рис. 7.5. Паводкові явища жовтої та червоної зони
річки Уж в період 1992 – 2010 рр.

Відмітимо, що такий об‘єм тестової вибірки недостатній для всебічного аналізу якості побудованих моделей (структур) дерев класифікації, але в зв‘язку з обмеженим характером самої НВ навіть така ТВ дає змогу оцінити та проаналізувати основні параметри синтезованих структур АДК. Побудовані структури (моделі) АДК забезпечили необхідний рівень точності та ефективності класифікації, заданих умовою прикладної задачі, необхідну швидкодію та витрати робочої пам‘яті системи, але показували різну структурну складність у побудовах дерев класифікації (параметрів складності конструкції ЛДК – наборів алгоритмів, наборів узагальнених ознак, кількості вершин структури дерева класифікації).

7.4.2. Експериментальна перевірка побудованих моделей класифікації

На основі запропонованого в межах даного дослідження методу АДК було побудовано моделі класифікації паводкових явищ річки Уж, причому структура АДК дозволяє досить ефективно регулювати складність (донавчати) моделі дерева класифікації, що будується, або будувати моделі дерев розпізнавання з наперед заданою точністю відповідно до умов задачі. Зрозуміло, що для порівняння та відбору конкретної моделі дерева класифікації з фіксованого набору необхідно виділити найбільш важливі їх характеристики (розмірність ознакового простору, кількість вершин, кількість переходів конструкції дерева тощо) та визначити їх похибку відносно масиву вхідних даних.

Важливим є аналіз критеріїв якості отриманих інформаційних моделей, які залежать від похибки моделі, потужності початкового масиву даних НВ та ТВ (кількість навчальних пар та розмірність ознакового простору задачі), кількості структурних параметрів моделі і так далі. Зрозуміло, що критично важливими параметрами побудованої моделі АДК, які необхідно мінімізувати, є помилки моделі відповідно на масивах даних НВ, ТВ та для кожного з класів (частин, підмножин початкової НВ), які задані початковою умовою поточної прикладної задачі. Відмітимо, що одним із найважливіших показників, який характеризує базові властивості отриманих моделей АДК є базовий показник узагальнення даних початкової НВ деревом класифікації (моделлю класифікації), який розраховується наступним чином:

$$I_{Main} = \frac{m*O_{Uz}}{Fr_{All}+V_{All}+N_{All}+2P_{All}}. \quad (7.6)$$

Даний показник узагальнення моделі дерева класифікації (структури АДК) відображає його базові параметри (характеристики) дерев класифікації та може бути застосований в якості критерію оптимальності в процедурі оцінки довільної деревоподібної схеми розпізнавання. Для довільної прикладної задачі важливо максимізувати параметр I_{Main} (показник узагальнення моделі АДК), що дозволяє домогтися оптимальної структури

дерева (моделі) класифікації та забезпечує фактично максимальний стиск даних початкової НВ (дає змогу представити масив початкових даних мінімальним за структурною складністю деревом). Так, важливим показником якості побудованої моделі у вигляді дерева класифікації з урахуванням параметрів структури моделі АДК є загальний інтегральний показник якості, представлений у наступній формі:

$$Q_{Main} = \frac{Fr_{All}}{O_{Uz} * \sum_i p_i} * e^{-\frac{Er_{All}}{M_{All}}}. \quad (7.7)$$

Відмітимо, що в функціоналах 7.6 – 7.7 параметри мають наступну інтерпретацію:

Er_{All} – загальна кількість помилок моделі АДК на масивах даних початкових тестової та навчальної вибірки – $Er_{All} = En_{tr} + Et_{tr}$;

M_{All} – загальна потужність цих двох масивів даних – $M_{All} = m + T$, де m та T – відповідні потужності НВ та ТВ;

Fr_{All} характеризує загальну кількість вершин отриманої моделі АДК з результуючими значеннями f_R (ФР, тобто листів дерева класифікації);

O_{Uz} представляє загальну кількість усіх узагальнених ознак (наборів УО) в структурі моделі АДК;

V_{All} представляє загальну кількість усіх типів вершин (крім результуючих ФР) у структурі моделі АДК;

N_{All} – загальна кількість різних автономних алгоритмів класифікації, що використовуються в моделі дерева класифікації;

P_{All} – загальна кількість переходів між вершинами (ярусами) в структурі побудованої моделі дерева класифікації.

Набір параметрів p_i для інтегральної оцінки якості моделі дерева класифікації представляє собою найважливіші характеристики дерева класифікації (відповідно до структур ЛДК/АДК), що оцінюється (наприклад кількість елементарних ознак або узагальнених ознак, що використовуються в моделі дерева класифікації, кількість переходів між вершинами, ярусами дерева класифікації тощо), і може бути розширений і на структури (методи)

ЛДК даної прикладної задачі. Відмітимо, що даний інтегральний показник якості моделі АДК буде приймати значення в межах нуля та одиниці, причому чим меншим він буде, тим гіршою буде якість побудованого дерева класифікації, а чим більшим буде показник, тим кращою буде отримана модель.

Таблиця 7.8. Порівняльна таблиця моделей класифікації (структур АДК) класифікації паводкових явищ річки Уж (постів №1-№2).

№	*Метод (схема) синтезу структури (моделі) дерева класифікації (ЛДК/АДК)*	*Інтегральний показник якості моделі дерева класифікації* Q_{Main}	*Загальна кількість помилок моделі на НВ та ТВ* Er_{All}
1	Метод повного АДК (тип I) (кількість алгоритмів – 5, обмеження на послідовне використання по одному алгоритму)	0,005821 0,005845	0 0
2	Метод повного АДК (тип II) (кількість алгоритмів – 5, обмеження на кількість генерацій УО на один крок побудови АДК)	0,004778 0,004712	0 0
3	Обмежений метод АДК (Z=10) (кількість алгоритмів – 1, алг. гіперсфер)	0,004464 0,004389	0 0
4	Обмежений метод АДК (Z=5) (кількість алгоритмів – 1, алг. гіперкуба)	0,004387 0,004228	12 13
5	Обмежений метод АДК (Z=3) (кількість алгоритмів – 1, алг. гіперпаралелепіпеда)	0,004256 0,004354	6 6
6	Обмежений метод АДК (Z=3) (кількість алгоритмів – 1, алг. гіпереліпса)	0,005582 0,005645	1 0
7	Метод повного АДК (тип I) (кількість алгоритмів – 2, алг. гіперсфер, алг. гіперкуба)	0,005790 0,005801	0 0

Запропонована інтегральна оцінка якості моделі дерева класифікації (структури АДК) відображає його базові параметри (характеристики), також може бути застосована в якості критерію оптимальності в процедурі оцінки довільної деревоподібної схеми розпізнавання (відповідно до параметрів моделі). Так, фрагмент основних результатів інтегральної оцінки якості

побудованих структур (моделей) АДК, приведених вище експериментів – порівняльних тестів методів побудови моделей АДК (структур дерев класифікації) на масиві даних даної прикладної задачі, представлений в табл. 7.8.

7.4.3. Напрямки вдосконалення побудованих моделей класифікації паводкових явищ

Побудовані дерева класифікації (моделі АДК) забезпечили необхідні якість та швидкість схем класифікації паводкових явищ річки Уж при достатньо компактній структурі самої конструкції дерева. Набори незалежних алгоритмів класифікації, які були відібрані для генерації груп УО також підтвердили свою ефективність у межах даної прикладної задачі. Можливим шляхом подальших досліджень може бути розширення переліку алгоритмів класифікації в схемі АДК, а також додаткові умови та обмеження щодо генерації наборів УО для кожного кроку схеми дерева класифікації (структури АДК).

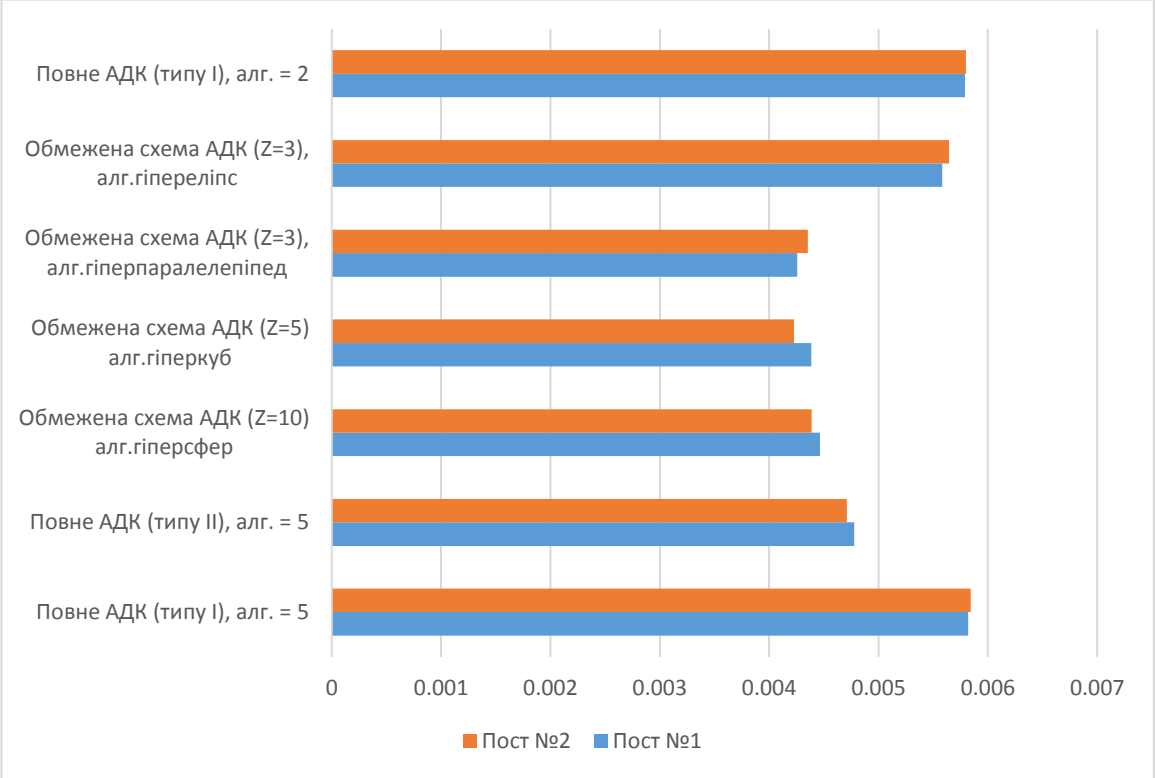


Рис. 7.6. Інтегральний показник якості побудованих моделей АДК.

Представлені моделі дерев класифікації можна застосовувати для оцінки загального стану басейну річки Уж (на ділянці спостереження) та виявлення ситуації червоної (паводкової) зони на основі поточних замірів постів спостережень. Відмітимо, що проведені практичні випробовування структур (моделей) АДК підтвердили працездатність математичного забезпечення та запропонованих методів та алгоритмів побудови АДК, розробленого програмного забезпечення, що дозволяє рекомендувати використання даного підходу (концепції моделей АДК) та його програмної реалізації для розв'язання широкого спектру прикладних задач класифікації та розпізнавання в практичній площині. Єдиним принциповим моментом, на який треба зважати при генерації моделей АДК, є те, що, зазвичай, витрати робочої пам'яті та процесорного часу інформаційної системи є значно більшими порівняно із структурами (моделями) ЛДК, причому в значній мірі це залежить від особливостей реалізації алгоритмів розпізнавання (класифікаторів), кількості алгоритмів у схемі АДК, схеми (типу моделі) структури АДК, яка генерується.

Отже, зважаючи на все вище зазначене, можна відмітити, що перспективним напрямком розв'язання даної задачі класифікації ситуаційного стану річки Уж, може бути також застосування методів, алгоритмів та схем побудови моделей ЛДК із можливістю їх порівняння (ефективності, структурної складності та швидкодії класифікації) з побудованими структурами (моделями) АДК, розширення об'єму бібліотеки алгоритмів розпізнавання (класифікаторів), оптимізації програмних реалізацій запропонованих в даному дослідженні методів побудови АДК (за взірець такої ефективної програмної оптимізації можна взяти еволюційний перехід від алгоритму C4.5 до C5.0), а також практичної апробації побудованих моделей дерев класифікації на множині даних НВ більшої кількості постів спостереження басейну річки Уж.

Висновки до розділу 7

1. У розділі розроблено алгоритмічну схему обчислення важливості ознак (груп ознак) за допомогою введеного в роботі функціонала, який дає можливість вирішувати широке коло практичних задач. Розглянуто питання кодування початкової інформації НВ, однин із шляхів вирішення якої полягає в схемі розбиття ознакового простору задачі на $n-$ мірні гіперпаралелепіпеди так, щоб елементи (об‘єкти) з різних класів не попадали в один і той самий паралелепіпед (задача геометричного обмеження) з присвоєнням кожному з даних геометричних об‘єктів фіксованого двійкового коду.

2. Розроблено ефективну алгоритмічну схему побудови дерев класифікації та відповідну комплексну ПС – DeTree, яка забезпечує ефективний механізм програмної побудови та фінального аналізу фіксованого ЛДК (набору ЛДК) за масивом деяких початкових даних (НВ).

3. Побудовано моделі дерев класифікації (структури АДК) на основі даних гідрографічного спостереження загального стану басейну річки Уж для виявлення ситуації червоної (паводкової) зони на основі поточних замірів постів спостережень.

4. Проведено практичні випробовування моделей АДК, які підтвердили працездатність математичного забезпечення та запропонованих методів та алгоритмів побудови дерев класифікації, розробленого програмного забезпечення, що дає змогу рекомендувати використання даного підходу (концепції моделей АДК) та його програмної реалізації для розв’язання широкого спектру прикладних задач класифікації та розпізнавання в практичній площині.

ВИСНОВКИ

У дисертаційній роботі на основі проведених досліджень вирішено актуальну науково-прикладну проблему розвитку теорії аналізу та синтезу дерев рішень, розробленню моделей, методів, прикладного інструментарію інтелектуального аналізу даних на основі логічних та алгоритмічних дерев класифікації з більшою точністю, зменшеною складністю моделей та підвищеною ефективністю класифікації дискретних об‘єктів.

При цьому отримано такі результати:

1. Виконано аналіз сучасного стану досліджень, методів побудови деревоподібних моделей класифікації (дерев рішень). Проведений аналіз показав велику сегментацію підходів відносно прикладних задач та відсутність єдиної методології раціонального використання накопиченого потенціалу методів та алгоритмів класифікації.

2. Уперше запропоновано комплексний метод побудови деревоподібних моделей класифікації, який базується на поетапній апроксимації масиву початкових даних НВ набором відібраних і оцінених незалежних алгоритмів розпізнавання та дає змогу будувати різнотипні моделі структур АДК на основі оцінки якості набору алгоритмів класифікації в схемі методів АДК, відбору критерію розгалуження конструкції дерева.

3. Уперше запропоновано метод Т – опорних множини, який полягає у відборі (фіксації) певного набору ознак разом зі своїми значеннями (правилом класифікації в структурі ЛДК) на основі інформації деякої початкової НВ, з можливістю наступної оцінки даних опорних множин за допомогою відповідних функціоналів, виділено основні способи задання систем Т – опорних множин, показано зв‘язок зі структурами ЛДК, що дозволяє дати нове формальне визначення алгоритму розпізнавання, ввести нові означення інформативності Т – опорних множин щодо довільного об‘єкту та класу, дослідити основні підмоделі алгоритму розпізнавання (на основі Т – опорних множин).

4. Уперше розроблено методи знаходження подібності конструкцій логічних дерев у задачах мінімізації їх структур та досліджено питання

критерію оптимальності регулярного логічного дерева, представлено схему оцінки складності структури регулярного логічного дерева, що дає можливість забезпечити механізм фінальної оптимізації (обрізки) побудованої структури моделі ЛДК.

5. Уперше розроблено методи оптимального розташування змінних (атрибутів, міток) конструкції ЛДК, які в більшості випадків дають оптимальне логічне дерево відносно його структури, що забезпечує можливість якісно провести процедуру фінальної обрізки побудованої структури ЛДК.

6. Розроблено метод обчислення впливу процедури перестановки ярусів у структурі регулярного логічного дерева на його загальну складність для бінарного випадку, що дає можливість досягти помітного ефекту зменшення складності логічного дерева, для фіксованого класу регулярних логічних конструкцій майже в 3 рази.

7. Вперше розроблено моделі та методи побудови структур АДК різних типів (обмежених структур АДК), де отримані дерева класифікації складаються з різних підходів, методів, алгоритмів розпізнавання, в свою чергу представляють собою нові схеми класифікації. Це дає змогу розширити область застосування, будувати моделі класифікації, точність яких можна регулювати в процесі побудови, або будувати системи з наперед заданою точністю, раціонально використовувати вже накопичений потенціал методів та алгоритмів теорії розпізнавання. Для обмеженої схеми швидкість синтезу моделі зросла на 20%.

8. Встановлено питання збіжності процедури побудови моделей дерев класифікації (запропонованих у дослідженні структур АДК) для умов слабкого та сильного розділення класів початкової НВ. Це забезпечує отримання параметрів моделі АДК максимальної складності.

9. Розроблено програмний інструментарій побудови структур ЛДК/АДК (моделей дерев класифікації різних типів), для розв’язання широкого спектру різнотипних прикладних задач розпізнавання образів, автоматизації процедури синтезу моделей класифікації різних типів

(структур ЛДК/АДК).

За результатами дисертаційного дослідження проведено експериментальне моделювання, апробацію розроблених моделей та методів на практичних задачах, а визначення ефективності запропонованих у роботі методів побудови АДК здійснено на основі інтегральних показників якості моделей. У представленому дослідженні запропоновано, обґрунтовано та теоретично досліджено нову організацію побудови моделей алгоритмів розпізнавання та класифікації – їх використання для автоматизації розробки реальних систем розпізнавання образів, розпізнавання дискретних об‘єктів, пов’язаних з обробкою великих масивів різнотипної інформації. Також в роботі вивчено властивості алгоритмів розпізнавання на основі моделей ЛДК та АДК.

Запропоновано одноманітну побудову схем розпізнавання цього класу дерев класифікації розроблено ефективні методи для обробки великих масивів різнотипної інформації. Введено нове поняття Т – опорної множини та на його основі побудовано і вивчено різні моделі алгоритмів розпізнавання. Доведено, що їх ємність скінчена, введено нове визначення алгоритму розпізнавання, нову операцію над цими алгоритмами, що дозволяє значно зменшити кількість операцій при побудові нових алгоритмів розпізнавання з набору відомих алгоритмів та методів теорії РО, отримано відповідні оцінки та властивості введеної операції, знайдено умови, коли множина алгоритмів розпізнавання утворює базис задачі. На основі цього запропоновано новий підхід до побудови експертних систем.

Відмітимо, що багато результатів із вище приведених пунктів перенесені для розв‘язку задач розпізнавання образів. Розроблено програмний інструментарій для розв‘язку задач розпізнавання образів. Розроблений в дисертаційному дослідженні підхід (на основі концепції структур АДК та модульного принципу) дає можливість методично одноманітно вирішувати багато прикладних задач аналізу даних, розпізнавання образів, розпізнавання дискретних об‘єктів із використанням накопиченого досвіду українських та зарубіжних вчених.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Wei Yi, Yang Chenhui, Yangand Hang, Xu Yaming. A Novel Signal Similarity Evaluation Algorithm and its Application in Irregular Shape Recognition. *Journal of Physics: Conference Series*. 2017. Vol. 910. Article number: 012033. DOI: 10.1088/1742-6596/910/1/012033

2. Ovezgeldyev A. O., Petrov K. E. Fuzzy-Interval Choice of Alternatives in Collective Expert Evaluation. *Cybernetics and Systems Analysis*. 2016. Vol. 52. P. 269–276. DOI: 10.1007/s10559-0169823-4

3. Kalashnikov V. V., Avramenko V. V., Kalashnykova N. I. Derivative disproportion functions for pattern recognition. *Unconventional modelling, simulation, and optimization of geoscience and petroleum engineering* / Watada J, Tan S. C., Vasant P. et al. (eds.). Berlin-Heidelberg : Springer-Verlag, 2018. P. 95–104.

4. Subbotin S. A., Gofman Ye. A. The fractal analysis of sample and decision tree model. *Радіоелектроніка, інформатика, управління*. 2020. № 1. С. 98–107.

5. Jafari M., Xu H. Intelligent Control for Unmanned Aerial Systems with System Uncertainties and Disturbances Using Artificial Neural Network. *Drones*. 2018. Vol. 3. P. 24–36.

6. Leoshchenko S., Oliinyk A., Skrupsky S., Subbotin S., Lytvyn V. Parallel Genetic Method for the Synthesis of Recurrent Neural Networks for Using in Medicine. *Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019).* 2019. P. 1–17. URL: https://dblp.org/rec/conf/cmis/LeoshchenkoOSSL19

7. Jaman S. F. I., & Anshari M. Facebook as marketing tools for organizations: Knowledge management analysis. *Dynamic perspectives on globalization and sustainable business in Asia*. IGI Global. 2019. P. 92-105.

8. Hu Z., Bodyanskiy Y., Tyshchenko O. Kohonen Maps and Their Ensembles for Fuzzy Clustering Tasks. *Self-Learning and Adaptive Algorithms for Business Applications*. Publisher: Emerald Publishing, 2019. P. 51–77. DOI:10.1108/978-1-83867-171-620191004

9. Bag of tricks for image classification with convolutional neural networks / He T. et al. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Long Beach, Ca, USA, June 16–20 2019. P. 558–567. DOI:10.1109/CVPR.2019.00065

10. Kushneryk P., Kondratenko Y., Sidenko I. Intelligent dialogue system based on deep learning technology. *15th International Conference on ICT in Education, Research, and Industrial Applications*: *PhD Symposium (ICTERI 2019: PhD Symposium)*. Kherson, Ukraine, 2019. Vol. 2403. P. 53–62.

11. Ryman-Tubb N. F., Krause P. and Garn W. How Artificial Intelligence and machine learning research impacts payment card fraud detection: A survey and industry benchmark. *Engineering Applications of Artificial Intelligence*. 2018. № 76. P. 130–157.

12. Matchnet: Unifying feature and metric learning for patch-based matching / X. Han et al. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015. P. 3279–3286.

13. Discriminative learning of deep convolutional feature point descriptors / E. Simo-Serra at. al. *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2015. P. 118–126.

14. He K., Zhang X., Ren S., and Sun J. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, Las Vegas, NV, USA, June 27–30, 2016. P. 770–778.

15. Hanin B. Which neural net architectures give rise to exploding and vanishing gradients? *Advances in Neural Information Processing Systems* : *Proceedings of the* 32nd International conference (NIPS-2018). P. 582–591.

16. Doll´ar P., Girshick R., He K., Hariharan B., Belongie S. Feature pyramid networks for object detection / T.-Y. Lin et al. *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, July 21–26 2017, Honolulu, HI, USA. 2017. P. 2117–2125.

17. Kotsovsky V., Geche F., Batyuk A. Finite generalization of the offline spectral learning. *Proceedings of the 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*, Lviv, Ukraine August 21–25, 2018. Lviv : Polytechnic National University, 2018. P. 356–360.

18. Deep 2D-Neural Network and its Fast Learning / Bodyanskiy Y. et al. *Proceedings of the 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*, Lviv, Ukraine August 21–25, 2018. Lviv : Polytechnic National University, 2018. P. 519–523.

19. Subbotin S. The instance and feature selection for neural network based diagnosis of chronic obstructive bronchitis. *Applications of Computational Intelligence in Biomedical Technology*. Cham : Springer, 2016. P. 215–228.

20. Sirichotedumrong W., Maekawa T., Kinoshita Y. Kiya H. Privacypreserving deep neural networks with pixel-based image encryption considering data augmentation in the encrypted domain. *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019. P. 674–678.

21. Makri E., Rotaru D., Smart N. P., Vercauteren F. Epic: Efficient private image classification (or: Learning from the masters). *Cryptographers' Track at the RSA Conference*. Springer, 2019. P. 473–492.

22. P3sgd: Patient privacy preserving sgd for regularizing deep cnns in pathological image classification / B. Wu et al. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Long Beach, Ca, USA, June 16–20 2019. P. 2099–2108.

23. Autoaugment: Learning augmentation strategies from data / E. D. Cubuk et al. *Proceedings of the IEEE conference on computer vision and pattern recognition*. Long Beach, Ca, USA, June 16–20 2019. P. 113–123.

24. Lupei M., Mitsa A., Repariuk V. and Sharkan V. Identification of authorship of Ukrainian-language texts of journalistic styleu sing neural networks. *Eastern-European Journal of Enterprise Technologies*. 2020. №1. P. 30–36. DOI: https://doi.org/10.15587/1729-4061.2020.195041

25. Bodyanskiy Y., Vynokurova O., Setlak G. and Pliss I. Hybrid neuro-neo-fuzzy system and its adaptive learning algorithm. *Computer Sciences and Information Technologies (CSIT)* : Xth International Scientific and Technical Conference, Lviv, September 14–17, 2015. Lviv. P. 111–114.

26. Bodyanskiy Y., Dolotov A., Peleshko D., Rashkevych Y. and Vynokurova **O**. Associative probabilistic neuro-fuzzy system for data classification under short training set

conditions. *Advances in Intelligent Systems and Computing*. Vol. 761. Heidelberg : Springer, 2019. P. 56–63.

27. Bodyanskiy Y., Dolotov A., Peleshko D., Rashkevych Y., Vynokurova O. Online time series changes detection based on neurofuzzy approach. *Predictive Maintenance in Dynamic Systems* / E. Lughofer, M. Sayed-Mouchaweh (eds.). Cham : Springer, 2019. P. 131–166.

28. Kotsovsky V., Geche F. and Batyuk A. On the computational complexity of learning bithreshold neural units and networks. *Advances in Intelligent Systems and Computing* / V. Lytvynenko et al. (eds.). Heidelberg : Springer, 2020. Vol. 1020. P. 189–202.

29. Morozov V., Kalnichenko O., Proskurin M., Mezentseva O. Investigation of Forecasting Methods the State of Complex ITProjects With Using Deep Learning Neural Networks. *Lecture notes in computational intelligence and decision making*. 2020. P. 261–280 (Series "Advances in intelligent systems and computing". Vol. 1020).

30. Morozov V., Kalnichenko O., Proskurin M. Methods of Proactive Management of Complex Projects Based on Neural Networks. *Proceedings of the 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications(IDAACS)*. IEEE. 2019. P. 964–969.

31. A Clever Approach to Develop an Efficient Deep Neural Network-Based IDS for Cloud Environments Using a Self-Adaptive Genetic Algorithm / Z. Chiba et al. *2019 International Conference on Advanced Communication Technologies and Networking (CommNet)*. Rabat, Morocco, 2019. P. 1–9.

32. Imamovic D., Babovic E. and Bijedic N. Prediction of mortality in patients with cardiovascular disease using data mining methods. *2020 19th International Symposium INFOTEH-JAHORINA (INFOTEH)*, East Sarajevo, Bosnia and Herzegovina, 2020. P. 1–4.

33. Oladunni T. and Sharma S. Hedonic Housing Theory – A Machine Learning Investigation. *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. Anaheim, CA, 2016. P. 522–527.

34. Lin S.-L. and Huang H.-W. Improving Deep Learning for Forecasting Accuracy in Financial Data. *Discrete Dynamics in Nature and Society*. 2020. Vol. 2020. P. 1–12.

35. Subbotin S. The special deep neural network for stationary signal spectra classification. *Proceedings of 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET 2018)*, Slavske. IEEE, Los Alamitos, 2018. P. 123–128. DOI: 10.1109/tcset.2018.8336170

36. Giatzitzoglou D. G., Sotiropoulos D. N., Tsihrintzis G. A. AIRS-x: An eXtension to the Original Artificial Immune Recognition Learning Algorithm. *2019 International Conference on Computer, Information and Telecommunication Systems (CITS)*. Beijing, China, 2019. P. 1–5.

37. Chawla C., Panwar D., Anand G.S., Bhatia M.P.S. Classification of computer generated images from photographic images using convolutional neural networks. *International Journal of Computer and Informational Engineering*. 2018. Vol. 12, № 10. P. 822–827.

38. Bodyanskiy Hu., Zh., Tyshchenko Ye., Shafronenko O. A. Fuzzy clustering of incomplete data by means of similarity measures. *Proceedings 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON), Track 6*. Lviv, Ukraine July 2–6, 2019. Lviv, 2019. P. 149–152.

39. Park J., Jang K., Yang S.-B. Deep neural networks for activity recognition with multisensor data in a smart home. *2018 IEEE 4th World Forum on Internet of Things (WFIoT)*. IEEE, Singapore, 2018. P. 155–160. URL: https://doi.org/10.1109/WFIoT.2018.8355147.

40. Gupta Y. Selection of important features and predicting wine quality using machine learning techniques. *Procedia Computer Science*. 2018. Vol. 125. P. 305–312. URL: https://doi.org/10.1016/j.procs.2017.12.041.

41. Rabcan J., Rusnak P., Subbotin S. Classification by fuzzy decision trees inducted based on Cumulative Mutual Information. *Proceedings of 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET 2018)*. IEEE, Los Alamitos, 2018. P. 208–212. DOI: 10.1109/tcset.2018.8336188

42. Rabcan J., Levashenko V., Zaitseva E., Kvassay M., Subbotin S. Application of Fuzzy Decision Tree for Signal Classification. *IEEE Transactions on Industrial Informatics*. 2019. Vol. 15(10). P. 5425–5434. DOI: 10.1109/tii.2019.2904845

43. Kamiński B., Jakubczyk M., Szufel P. A framework for sensitivity analysis of decision trees. *Central European Journal of Operations Research*. 2017. 26 (1). P. 135–159. DOI:10.1007/s10100-017-0479-6

44. Rabcan J., Levashenko V., Zaitseva E., Kvassay M., Subbotin S. Non-destructive diagnostic of aircraft engine blades by Fuzzy Decision Tree. *Engineering Structures*. 2019. Vol. 197. DOI: 10.1016/j.engstruct.2019.109396

45. Subbotin S., Kirsanova E. The regression tree model building based on a clusterregression approximation for data-driven medicine. *CEUR Workshop Proceedings*. 2018. Vol. 2255. P. 155–169.

46. Denisko D., Hoffman M. Classification and interaction in random forests. *Proceedings of the National Academy of Sciences of the United States of America*. 2018. Vol. 115, №8. P. 1690–1692. DOI:10.1073/pnas.1800256115.

47. Subbotin S. A random forest model building using a priori information for diagnosis. *CEUR Workshop Proceedings*. 2019. Vol. 2353. P. 962–973.

48. Jordan M. I., Mitchell T. M. Machine learning: trends, perspectives, and prospects. *Science*. 2015. Vol. 349(6245). P. 255–260.

49. Zeng X., Lu J. Decision Support Systems with Uncertainties in Big Data Environments. *Knowledge-Based Systems*. 2018. Vol. 143. P. 327.

50. Savenkov P. A. Using machine learning methods and algorithms in management decision support systems. *Journal of Science and Education*. 2019. Vol. 55. P. 23–25.

51. Журавлев Ю. И. Корректные алгебры над множествами некорректных (эвристических) алгоритмов. Ч. I. *Кибернетика*. 1977. №4. С. 14–21.

52. Журавлев Ю. И. Корректные алгебры над множествами некорректных (эвристических) алгоритмов. Ч. II. *Кибернетика*. 1977. №6. С. 21–27.

53. Журавлев Ю. И. Корректные алгебры над множествами некорректных (эвристических) алгоритмов. Ч. III. *Кибернетика*. 1978. №2. С. 35–43.

54. Журавлев Ю. И. Непараметрические задачи распознавания образов. *Кибернетика*. 1976. №6. С. 93–123.

55. Журавлев Ю. И., Никифоров В. В. Алгоритмы распознавания, основанные на вычислении оценок. *Кибернетика*. 1971. №3. С. 1–11.

56. Журавлев Ю. И. Об алгебраическом подходе к решению задач распознавания и классификации. *Проблемы кибернетики* : сб. статей. Москва : Наука, 1978. Вып. 33. С. 5–68.

57. Russell S., Norvig P. Artificial intelligence: a modern approach. Prentice Hall, Upper Saddle River. 2009. 420 p.

58. Luger G. F. Artificial Intelligence. Structures and Strategies for Complex Problem Solving. London : Pearson Education, 2011. 380 p.

59. Dopico J. R., de la Calle J. D., Sierra A. P. Encyclopedia of artificial intelligence. Information Science Reference. New York, 2009. 1120 p.

60. Boulesteix A.-L., Janitza S., Kruppa J., König I. R. Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. 2012. Vol. 2, № 6. P. 493–507.

61. Geurts P., Irrthum A., Wehenkel L. Supervised learning with decision tree-based methods in computational and systems biology. *Molecular Biosystems*. 2009. Vol. 5, № 12. P. 1593–1605.

62. Utgoff P. E. Incremental induction of decision trees. *Machine learning*. 1989. Vol. 4, № 2. P. 161–186. DOI:10.1023/A:1022699900025

63. Hyafil L., Rivest R. L. Constructing optimal binary decision trees is npcomplete. *Information Processing Letters*. 1976. Vol. 5, № 1. P.15–17.

64. Субботин С. А. Методы синтеза моделей количественных зависимостей в базисе деревьев регрессии, реализующих кластер-регрессионную аппроксимацию по прецедентам. *Радіоелектроніка, інформатика, управління*. 2019. № 3. С. 76–85.

65. Jensen R., Shen Q. Computational intelligence and feature selection: rough and fuzzy approaches. Hoboken : John Wiley & Sons, 2008. 300 p.

66. Субботин С. А. Построение деревьев решений для случая малоинформа тивных признаков. *Радіоелектроніка, інформатика, управління*. 2019. № 1. С. 122–131.

67. Повхан І. Ф. Питання структурної складності для випадку регулярного логічного дерева. *Scientific and technical progress in European countries and the contribution of higher education institutions* : collective monograph. Riga : Izdevnieciba "Baltija Publisher", 2020. 308 p.

68. Povhan I. General scheme for constructing the most complex logical tree of classification in pattern recognition discrete objects. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2019. Vol. 11. С. 73–80.

69. Povhan I. Question of the optimality criterion of a regular logical tree based on the concept of similarity. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2020. Vol. 13. С. 19–27. DOI: https://doi.org/10.30970/eli.13.2

70. Повхан І. Ф. Задача загальної оцінки складності максимального побудованого логічного дерева класифікації. *Вісник Національного технічного університету «Харківський політехнічний інститут»* : зб. наук. пр. Серія: Інформатика та моделювання. 2019. №13 (1338). С. 104–117.

71. Повхан І. Ф. Питання оцінки ефекту перестановки ярусів логічного дерева максимальної складності для бінарного випадку. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2020. №2 (480). С. 99–107.

72. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Проблема оцінки складності логічних дерев розпізнавання та загальний метод їх оптимізації. *Восточно-европейский журнал передовых технологий*. 2011. №6/4 (54). С. 24–28.

73. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Загальна оцінка мінімізації деревоподібних логічних структур. *Восточно-европейский журнал передовых технологий*. 2012. №1/4 (55). С. 29–32.

74. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Дослідження стійкості максимального логічного дерева відносно перестановки ярусів. *Восточно-европейский журнал передовых технологий*. 2012. №2/4 (56). С. 15–22.

75. Повхан І. Ф. Питання загальної складності процедури побудови логічного дерева. *Проблеми інформатики та моделювання* : матеріали XX наук.-техн. конф., Харків-Одеса, 16–21 верес. 2020 р. Харків : НТУ “ХПІ”, 2020. С. 68–74.

76. Muller K. R., Mika S., Ratsch G. An introduction to kernelbased learning algorithms. *IEEE Transactions on Neural networks*. 2001. Vol. 12(2). P. 181–202.

77. Duda R. O., Hart P. E., Stork D. G. Pattern Classification. Second ed. New York, John Wiley & Sons, 2001. 654 p.

78. Jauregi Iztueta E., Lazkano E., Martinez-Otzeta J. M., Sierra B. Visual Approaches for Handle Recognition. *Springer Tracts in Advanced Robotics*. 2008. Vol. 44. P. 313–322.

79. Wang H., & Hong M. Online ad effectiveness evaluation with a two-stage method using a Gaussian filter and decision tree approach. *Electronic Commerce Research and Applications*. 2019. Vol. 35. Article 100852. DOI: 10.1016/j.elerap.2019.100852.

80. García S., Luengo J., Herrera F. Data Preprocessing in Data Mining. Springer, Switzerland, 2015. 526 p.

81. Big data preprocessing: methods and prospects / García S., Ramírez-Gallego S., Luengo J. et al. *Big Data Analytics.* Springer. 2016. Vol. 1(9). DOI: 10.1186/s41044-016-0014-0

82. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф., Повхан Л. С. Вычисление важности дискретных признаков (анализ некоторых подходов). *Восточно-европейский журнал передовых технологий*. 2010. №5/4 (47). С. 71–75.

83. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф., Поліщук В. В. Групова та індивідуальна оцінка важливості бульових аргументів. *Вісник Національного технічного університету «Харківський політехнічний інститут»*: зб. наук. пр. 2011. №53. С. 57–64.

84. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. The importance of discrete signs. *Перспективні шляхи й напрями вдосконалення освітньої системи у світлі Болонського процесу* : зб. наук. доп. за матеріалами XX Міжнар. наук.-практ. конф., Ужгород (Україна) – Кошице (Словаччина) – Мішкольц (Угорщина), 16–19 листоп. 2010 р. Ужгород : ЗакДУ, 2011. Вип. 2(21), Ч. 1. С. 46–56.

85. Василенко Ю. А., Мошкола В. П., Повхан І. Ф. Модульний синтез схем класифікації та розпізнавання. *Сучасні тенденції інформаційних технологій* : матеріали IV міжнар. наук. конф., Прага, 15–31 берез. 2008 р. Прага, 2008. С. 32–34.

86. Василенко Ю. А., Василенко Ю. А., Бунда В. В., Бунда С. О., Лавер О. Г., Повхан І. Ф. Универсальный подход к анализу и обработке специальной информации. *Інновації в навчальному процесі вищих навчальних закладів: міжнародний та національний досвід* : зб. наук. ст. за матеріалами XV міжнар. наук.-практ. конф., Сніна (Словацька Республіка), 6–9 листоп. 2007 р. Ужгород : Ліра, 2008. С. 99–106.

87. Василенко Ю. А., Василенко Ю. А., Бунда В. В., Повхан І. Ф., Чедреки Я. Н. Математическое конструирование распознающих систем для дискретных объектов. *Лісабонська стратегія як визначальний чинник європейської інтеграції в галузі освіти і науки* : матеріали XVI Міжнар. наук.-практ. конф. Ужгород – Гирляни, 6–9 трав. 2008 р. Ужгород : Ліра, 2008. С. 79–83.

88. Василенко Ю. А., Завілопуло А. М., Повхан І. Ф., Повхан Л. С. Синтез систем розпізнавання на основі схем-агентів. *Інститут електронної фізики Національної академії наук України – 2011*: матеріали міжнар. конф. молодих учених, Ужгород, 24–27 трав. 2011 р. Ужгород : Мистецька Лінія, 2011. С. 187–188.

89. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Автоматизация построения систем классификации на основе схем-агентов. *Математическое моделирование, оптимизация и информационные технологии* : материалы 3-й междунар. конф., Кишинев, 19–23 марта 2012 г. Кишинев : Академия транспорта, информатики и коммуникаций, 2012. С. 444–446.

90. Повхан І. Ф., Лавер В. О. Алгоритми побудови логічних дерев класифікації в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), №4. С. 100–106.

91. Povkhan I., Lupei M., Kliap M., Laver V. The issue of efficient generation of generalized features in algorithmic classification tree methods. *International Conference on Data Stream Mining and Processing*: DSMP 2020 Data Stream Mining & Processing, Springer, Cham, 2020. P. 98–113.

92. Povkhan I., Lupei M. The algorithmic classification trees. *Proceedings of the "2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP)"*, August 21–25 2020, Lviv, Ukraine. Lviv, 2020. P. 37–44.

93. Lupei M., Mitsa A., Povkhan I., Sharkan V. Determining the eligibility of candidates for a vacancy using artificial neural networks. *Proceedings of the "2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP)"*, August 21–25 2020, Lviv, Ukraine. Lviv, 2020. P. 18–23.

94. Повхан І. Ф. Логічні дерева класифікації в задачах штучного інтелекту : монографія. Ужгород : Поліграфцентр-Ліра, 2019. 276 с.

95. Povhan I. F. Logical recognition tree construction on the basis a step-to-step elementary attribute selection. *Radio Electronics, Computer Science, Control*. 2020. № 2. P. 95–106.

96. Povhan I. Designing of recognition system of discrete objects. *Proceedings of the "2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP)"*, August 23–27 2016 Lviv, Ukraine. Lviv, 2016. P. 226–231.

97. Повхан І. Ф. Питання побудови деревоподібних моделей розпізнавання образів. *Вісник Національного технічного університету «Харківський політехнічний інститут»* : зб. наук. пр. Серія: Інформатика та моделювання. 2019. №28 (1353). С. 39–57.

98. Пастор Н. Е., Повхан І. Ф. Генерація схем розпізнавання дискретних об'єктів на онові апроксимації навчаючої вибірки. *Радіоелектроніка та молодь в XXI столітті* : матеріали XXIII міжнар. молодіжного форуму, Харків, 16–18 квіт. 2019 р. Харків, 2019. Т. 5. С. 140–142.

99. Повхан І. Ф. Поняття функції та алгебраїчної схеми розпізнавання в задачах класифікації дискретних об'єктів. *Вчені записки Таврійського національного університету імені Вернадського*. Серія: Технічні науки. 2019. Т. 30 (69), №2. С. 171–177.

100. Повхан І. Ф. Проблема функціональної оцінки навчальної вибірки в задачах розпізнавання дискретних об'єктів. *Вчені записки Таврійського національного університету імені Вернадського*. Серія: Технічні науки. 2018. Т. 29 (68), №6. С. 217–222.

101. Han J., Kamber M., Pei J. Data Mining: Concepts and Techniques. Morgan Kaufmann, USA, 2011. 743 p.

102. Harley J. B., Sparkman D. Machine learning and NDE: Past, present, and future. *AIP Conference Proceedings*. 2019. Vol. 2102(1). DOI:10.1063/1.5099819

103. Povhan I. Logical classification trees in recognition problems. *Kwartalnik Naukowo-Techniczny: Informatyka Automatyka Pomiary w gospodarce o ochronie srodowiska*. Krakow, 2020. №2. P. 12–16. DOI: http://doi.org/10.35784/iapgos.927

104. Повхан І. Ф. Методи логічних дерев класифікації в задачах штучного інтелекту. *Priority directions of science development*: Abstracts of II International Scientific and Practical Conference, Lviv (Ukraine), 25–26 November 2019. Lviv, 2019. P. 213–218.

105. Murphy K. Machine Learning: A Probabilistics Perspective. The MIT Press, Cambridge, Massachusetts, 2012. 423 p.

106. Bishop C. M. Pattern recognition and machine learning. New York : Springer, 2014. 375 p.

107. Encyclopedia of survey research methods / Lavrakas, P. J. (ed.). Sage Publications, Thousand Oaks, 2008. 648 p.

108. Haro-García A., Cerruela-García G., García-Pedrajas N. Instance selection based on boosting for instance-based learners. *Pattern Recognition*, Elsevier. 2019. Vol. 96. DOI: 10.1016/j.patcog.2019.07.004

109. Povhan I. Generation of classification schemes in the form of logical trees in discrete object recognition problems. *Modern Problems of Mathematical Modeling, Automated Control and Information Technologies MCIT-2019* : International Scientific and Practical Conference, Rivne, Noveber 14–16 2019. Рівне, 2019. P. 195–200.

110. Повхан І. Ф. Проблематика методів логічних дерев класифікації в задачах розпізнавання. *Science, engineering and technology: global and current trends*: Proceedings of the international scientific and practical conference, Prague (Czech Republic), December 27–28 2019. Prague, 2019. P. 28–32.

111. Do we need hundreds of classifiers to solve real world classification problems? / M. Fernandez-Delgado et al. *Journal of Machine Learning Research*. 2014. Vol. 15(1). P. 3133-3181.

112. Повхан І. Ф. Особливості випадкових логічних дерев класифікації в задачах розпізнавання образів. Вчені записки Таврійського національного університету імені Вернадського. Серія: Технічні науки. 2019. Т. 30 (69), №5. С. 138–143.

113. Hamidzadeh J., Monsefi R., Sadoghi Yazdi H. IRAHC: Instance Reduction Algorithm using Hyperrectangle Clustering. *Pattern Recognition*, Elsevier. 2015. Vol. 48(5). P. 1878–1889. DOI: 10.1016/j.patcog.2014.11.005

114. Повхан І. Ф. Моделі алгоритмів розпізнавання у вигляді логічних дерев класифікації. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2019. №1 (475). С. 156–163.

115. Kuncheva L. I. Combining pattern classifiers: methods and algorithms. John Wiley & Sons, Inc., Hoboken, New Jersey, 2014. 384 p.

116. Polikar R. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*. 2006. Vol. 6(3). P. 21–45. DOI:10.1109/mcas.2006.1688199

117. Sammut C., Webb G. Encyclopedia of Machine Learning. 2nd Edition. New York : Springer, 2017. 1335 p.

118. Povkhan I. A constrained method of constructing the logic classification trees on the basis of elementary attribute selection. *CEUR Workshop Proceedings*: Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020), Zaporizhzhia, Ukraine, April 15–19, 2020. Zaporizhzhia, 2020. Vol. 2608. P. 843–857. CEUR Workshop Proceedings (CEUR-WS.org).

119. Kotsiantis S. B. Bagging and boosting variants for handling classifications problems: a survey. *The Knowledge Engineering Review*. Cambridge University Press. 2013. Vol. 29(1). P. 78–100. DOI:10.1017/s0269888913000313

120. Повхан І. Ф. Побудова систем прогнозування економічних явищ на основі концепції логічного дерева класифікації. *Математичні методи, моделі та інформаційні технології в економіці* : матеріали VI міжнар. наук.-метод. конф., Чернівці, 18–19 квіт. 2019 р. Чернівці : Друк-Арт, 2019. С. 134–136.

121. Haykin S. O. Neural Networks and Learning Machines. Pearson, 2008. 528 p.

122. Bodyanskiy Y., Peleshko D., Vynokurova O. and Y. Tatarinova. Architecture of hybrid generalized additive neuro-fuzzy system in modelling technological process. *Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)* : 13th International Conference, Lviv, February 24–27 2015. Lviv, 2015. P. 333–335.

123. Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. СПб. : Питер, 2018. 480 с.

124. Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение = Deep Learning. Москва : ДМК-Пресс, 2017. 652 с.

125. Bishop Christopher M. Neural networks for pattern recognition. Oxford : Clarendon Press. 1995. 580 p. ISBN 978-0198538493, OCLC 33101074.

126. Izonin I., Tkachenko R., Peleshko D., Rak T. and Batyuk D. Learning-based image super-resolution using weight coefficients of synaptic connections. *Xth International Scientific and Technical Conference «Computer Sciences and Information Technologies (CSIT»)*, 2015. Lviv, 2015. P. 25–29.

127. Wilson Halsey. Artificial intelligence. Grey House Publishing, 2018. 786 p. ISBN 978-1682178676.

128. Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction. The MIT Press Cambridge, Massachusetts London, England, 2012. 338 p.

129. Batch Normalized Recurrent Neural Networks / C. Laurent et al. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 20–25 2016, Shanghai, China. IEEE : Curran Associates, Inc., 2016. P. 2657–2661.

130. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups / G. Hinton et al. *IEEE Signal Processing Magazine*. 2012. Vol. 29, №. 6. P. 82–97.

131. Deng L. A Tutorial Survey of Architectures, Algorithms, and Applications for Deep Learning. APSIPA Transactions on Signal and Information Processing. 2014. 438 p.

132. Gal Y. Uncertainty in Deep Learning: Ph.D. thesis. University of Cambridge, 2016. 280 p.

133. Koskimaki H., Juutilainen I., Laurinen P., Roning J. Two-level clustering approach to training data instance selection: a case study for the steel industry. *Neural*

Networks : International Joint Conference (IJCNN-2008), Hong Kong, 1–8 June 2008 : proceedings. Los Alamitos: IEEE, 2008. P. 3044–3049. DOI: 10.1109/ijcnn.2008.4634228

134. Quinlan J. R. Induction of Decision Trees. *Machine Learning*. 1986. №1. P. 81–106.

135. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Berlin : Springer, 2008. 768 p.

136. Classification and regression trees / L. L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone. Boca Raton : Chapman and Hall/CRC, 1984. 368 p.

137. Повхан І. Ф. Питання гнучкості логічних дерев класифікації в задачах розпізнавання образів. *Технічні науки та технології*. 2019. №3 (17). С. 131–140.

138. Karimi K., Hamilton H. J. Generation and Interpretation of Temporal Decision Rules. *International Journal of Computer Information Systems and Industrial Management Applications*. 2011. Vol. 3. P. 314–323.

139. Повхан І. Ф. Модульна концепція побудови дерев класифікації. *Цифрова економіка та інформаційні технології* : матеріали міжнар. наук.-практ. конф., Київ, 15–16 квіт. 2020 р. Київ, 2020. С. 87–90.

140. Vtogoff P. E. Incremental Induction of Decision Trees. *Machine Learning*. 1989. № 4. P. 161−186.

141. Srikant R., Agrawal R. Mining generalized association rules. *Future Generation Computer Systems*. 1997. Vol. 13, №2. P. 161–180.

142. Deng H., Runger G., Tuv E. Bias of importance measures for multi-valued attributes and solutions. *Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN)*, Finland, June 14–17. 2011. P. 293–300.

143. Kaminski B., Jakubczyk M., Szufel P. A framework for sensitivity analysis of decision trees. *Central European Journal of Operations Research*. 2017. №26 (1). P. 135–159.

144. Kotsiantis S. B. Supervised Machine Learning: A Review of Classification Techniques. *Informatica*. 2007. №31. P. 249–268.

145. Повхан І. Ф. Загальна концепція алгоритмічного дерева класифікації в задачах розпізнавання образів. *Інформаційні технології у житті молодих науковців*

Закарпаття : матеріали V наук.-практ. конф., Ужгород, 7–8 листоп. 2019 р. Ужгород : УжНУ, 2019. С. 58–62.

146. Повхан І. Ф. Питання представлення неповністю визначених багатозначних логічних функцій у вигляді логічних дерев в задачах класифікації. *Topical issues of the development of modern science*: Abstracts of IV International Scientific and Practical Conference, Sofia (Bulgaria), 11–13 December 2019. Sofia, Bulgaria, 2019. P. 390–396.

147. Inductive Logic Programming. *13th International Conference, ILP 2003* / Horvath Tamas; Yamamoto Akihiro (Eds.), Szeged, September/October, 2003 : Lecture Notes in Computer Science (2835). Szeged, Hungary, 2003.

148. Miyakawa M. Criteria for selecting a variable in the construction of efficient decision trees. *IEEE Transactions on Computers*. 1989. Vol. 38, № 1. P. 130–141.

149. Painsky A., Rosset S. Cross-validated variable selection in tree-based methods improves predictive performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2017. Vol. 39, № 11. P. 2142–2153. DOI:10.1109/tpami.2016.2636831

150. Kaftannikov I. L., Parasich A. V. Decision Tree's Features of Application in Classification Problems. *Bulletin of the South Ural State University. Ser. Computer Technologies, Automatic Control, Radio Electronics*. 2015. Vol. 15, №. 3. P. 26–32. DOI: 10.14529/ctcr150304

151. Breiman L. Bagging predictors. *Machine Learning*. 1996. Vol. 24. P. 123–140.

152. Breiman L. Random Forests. *Machine Learning*. 2001. Vol. 45(1). P. 5–32. DOI: 10.1023/A:1010933404324

153. Маценов А. А. Комитетный бустинг: минимизация числа базовых алгоритмов при простом голосовании. *Математические методы распознавания образов* : сб. докладов 13-й Всероссийской конф. (ММРО-13), г. Зеленогорск, 30 сент.–6 окт. 2007. Москва : МАКС Пресс, 2007. С. 180–183.

154. Mason L., Bartlett P., Baxter J. Direct Optimization of Margins Improves Generalization in Combined Classifiers. *Proceedings of the 1998 conference on Advances in Neural Information Processing Systems II.* MIT Press, 1999. P. 288–294.

155. Freund Y., Schapire R. E. Experiments with a New Boosting Algorithm. *Proceedings Thirteenth of the International Conference on Machine Learning (ICML'96)*. Morgan Kaufmann Publishers Ins., 1996. P. 148–156.

156. Флах П. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных / пер. с англ. А. А. Слинкина. Москва : ДМК Пресс, 2015. 400 с.

157. Bauer E., Kohavi R. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. *Machine Learning*. 1999. P. 105–139.

158. Gopika S., Vanitha Dr. M. Machine learning Approach of Chronic Kidney Disease Prediction using Clustering Technique. *International Journal of Innovative Research in Science, Engineering and Technology*. 2017. Vol. 6, №. 7. P. 14488–14496.

159. Kazemi Y., Mirroshandel S. A. A novel method for predicting kidney stone type using ensemble learning. *Artificial Intelligence in Medicine*. 2017. Vol. 79, № 3. P. 1696–1707.

160. Hunt E. B., Marin J. & Stone P. J. Experiments in induction. Oxford, England : Academic Press. 1966. P. 287.

161. Повхан І. Ф. Особливості синтезу узагальнених ознак при побудові систем розпізнавання за методом логічного дерева. *Інформаційні технології та комп'ютерне моделювання ІТКМ-2019* : матеріали міжнар. наук.-практ. конф., Івано-Франківськ – Яремче, 20–25 трав. 2019. Івано-Франківськ, 2019. С. 169–174.

162. Povhan I. Generation of elementary signs in the general scheme of the recognition system based on the logical tree. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2019. Vol. 12. С. 20–29.

163. Повхан І. Ф. Метод побудови алгоритмічного дерева другого типу на основі апроксимації навчальної вибірки набором алгоритмів класифікації. *Технічні науки та технології* : наук. журн. Чернігів, 2020. №2 (20). С. 126–139.

164. Ian H. Witten, Eibe Frank and Mark A. Hall. Data Mining: Practical Machine Learning Tools and Techniques. 3rd Edition. Morgan Kaufmann Publishers Ins., 2011. 664 p. ISBN 9780123748560.

165. Wang J. Encyclopedia of Data Warehousing and Mining. Second Edition. Hershey : Information Science Reference, 2009. 2227 p.

166. Poncelet Pascal, Masseglia Florent, and Teisseire Maguelonne (eds.) Data Mining Patterns: New Methods and Applications. Hershey, New York : Information Science Reference, 2007. 307 p.

167. Murphy Chris. Is Data Mining Free Speech? *Information Week*: *12*. 16 May 2011.

168. Nisbet Robert, Elder John, Miner Gary. Handbook of Statistical Analysis & Data Mining Applications. Academic Press/Elsevier, 2009. 564 p.

169. Lambodar J., Narendra K. Distributed Data Mining Classification Algorithms for Prediction of Chronic Kidney Disease. *International Journal of Emerging Research in Management and Technology*. 2015. Vol. 4, № 11. P. 110–180.

170. Ramya S., Radha N. Diagnosis of Chronic Kidney Disease Using Machine Learning Algorithms. *International Journal of Innovative Research in Computer and Communication Engineering.* 2016. Vol. 4, № 1. P. 812–820.

171. Yoruk U., Hargreaves B. A., Vasanawala S. S. Automatic Renal Segmentation for MR Urography Using 3D-GrabCut and Random Forests. *International Society for Magnetic Resonance in Medicine*. 2017. Vol. 79, № 3. P. 1696–1707.

172. J. Ross Quinlan. C4.5. Programs for Machine learning. Morgan Kaufmann Publishers, 1993. 272 p.

173. Диев В. С. Управленческие решения: неопределенность, модели, интуиция. Новосибирск, 1998. 218 с.

174. Елманова Н. Построение деревьев решений. *КомпьютерПресс*. 2003. № 12. С. 186.

175. Burell J. How the machine ‘thinks’: Understanding opacity in machine learning algorithms. *Big Data & Society*. 2016. Vol. 3. № 1. P. 138–146.

176. Long W. J., Griffith J. L., Selker H. P. D’Agostino R. B. A Comparison of Logistic Regression to Decision-Tree Induction in a Medical Domain. *Computers and Biomedical Research*. 1993. Vol. 26. P. 74–97.

177. Shin Y. Application of Boosting Regression Trees to Preliminary Cost Estimation in Building Construction Projects. *Computational Intelligence and Neuroscience*. 2015. P. 201–209.

178. Макленнен, Дж., Танг Ч., Криват Б. Microsoft SQL Server 2008: Data Mining – интеллектуальный анализ данных. СПб. : БХВ – Петербург, 2009. 720 с.

179. Барсегян А. А., Куприянов М. С., Степаненко В. В., Холод И. И. Методы и модели анализа данных: OLAP и Data Mining. СПб. : БХВ-Петербург, 2004. 336 с.

180. Lior Rokach, Maimon O. Data mining with decision trees: theory and applications. World Scientific Publishing Co Inc., 2008. 244 p. ISBN 978-9812771711.

181. Shalev-Shwartz Shai, Ben-David Shai. Decision Trees. *Understanding Machine Learning*: *From Theory to Algorithms*. Cambridge University Press, 2014. P. 250–257.

182. Wang Fulton, Rudin Cynthia. Falling Rule Lists (PDF). *Journal of Machine Learning Research*. 2015. Vol. 38.

183. Witten Ian, Frank Eibe, Hall Mark. Data Mining. Burlington, MA: Morgan Kaufmann, 2011. 664 p.

184. Rodriguez J. J., Kuncheva L. I., Alonso C. J. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2006. Vol. 28, № 10. DOI:10.1109/TPAMI.2006.211. — PMID 16986543.

185. Ron Rivest. Learning Decision Lists. *Machine Learning*. 1987. Vol. 3, № 2. DOI:10.1023/A:1022607331053.

186. Strobl C., Malley J., Tutz G. An Introduction to Recursive Partitioning: Rationale, Application and Characteristics of Classification and Regression Trees, Bagging and Random Forests. *Psychological Methods*. 2009. Vol. 14, № 4. DOI:10.1037/a0016973. – PMID 19968396.

187. Rokach L., Maimon O. Top-down induction of decision trees classifiers-a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*. 2005. Vol. 35, №4. DOI:10.1109/TSMCC.2004.843247.

188. Max Bramer. Principles of Data Mining : (Undergraduate Topics in Computer Science). Springer-Verlag, 2007. 344 p. ISBN 978-1-84628-765-7, DOI:10.1007/978-1-84628-766-4.

189. Gareth J., Witten D., Hastie T., Tibshirani R. An Introduction to Statistical Learning. New York : Springer, 2015. ISBN 978-1-4614-7137-0.

190. Dinesh Mehtaa, Vijay Raghavan. Decision tree approximations of Boolean functions. *Theoretical Computer Science*. 2002. Vol. 270, № 1–2. С. 609–623. DOI:10.1016/S0304-3975(01)00011-1.

191. Laurent Hyafil, Rivest R. L. Constructing Optimal Binary Decision Trees is NP-complete. *Information Processing Letters*. 1976. Vol. 5, № 1. С. 15–17. DOI:10.1016/0020-0190(76)90095-8.

192. Murthy S. Automatic construction of decision trees from data: A multidisciplinary survey. *Data Mining and Knowledge Discovery*. 1998. Vol. 2, №4. P. 345–389.

193. Irad Ben-Gal, Alexandra Dana, Niv Shkolnik, Gonen Singer. Efficient Construction of Decision Trees by the Dual Information Distance Method. *Quality Technology & Quantitative Management*. 2014. Vol. 11, № 1. С. 133–147.

194. Andreas M. Brandmaier, Timo von Oertzen, John J. McArdle, Ulman Lindenberger. Structural equation model trees. *Psychological Methods*. 2012. Vol. 18, №1. С. 71–86. DOI:10.1037/a0030001. – PMID 22984789.

195. Papagelis A., Kalles D. Breeding Decision Trees Using Evolutionary Techniques. *Machine Learning* : Proceedings of the Eighteenth International Conference (ICML), June 28–July 1 2001. Morgen Kaufmann Publishers, 2001. P. 393–400.

196. Rodrigo C. Barros, Basgalupp M. P., Carvalho A. C. P. L. F., Alex A. Freitas. A Survey of Evolutionary Algorithms for Decision-Tree Induction. *IEEE Transactions on Systems, Man and Cybernetics*. 2012. Vol. 42, № 3. P. 291–312. DOI:10.1109/TSMCC.2011.2157494.

197. Hugh A. Chipman, Edward I. George, Robert E. McCulloch. Bayesian CART model search. *Journal of the American Statistical Association*. 1998. Vol. 93, №443. С. 935–948. DOI:10.1080/01621459.1998.10473750.

198. Barros R. C., Cerri R., Jaskowiak P. A., Carvalho A. C. P. L. F. A bottom-up oblique decision tree induction algorithm. *Proceedings of the 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011)*. 2011. P. 450–456. ISBN 978-1-4577-1676-8, DOI:10.1109/ISDA.2011.6121697.

199. Martin N. F. and England J. W. Mathematical Theory of Entropy. 2011, Cambridge University Press, UK. 286 p. (Encyclopedia of Mathematics and its Applications. Vol. 12).

200. Montgomery D. C. Design and Analysis of Experiments. 5 th edition. New York : Wiley, 2008. 696 p.

201. Василенко Ю. А., Ващук Ф. Г. Распознавание дискретных объектов. *Науковий вісник Ужгородського державного інституту інформатики, економіки і права*. 1999. № 3. С. 32–37.

202. Василенко Ю. А., Ващук Ф. Г. Логики-алгебраический подход к обработке обучающей выборке. *Науковий вісник Ужгородського державного інституту інформатики, економіки і права*. 1997. №1. С. 4–10.

203. Vasilenko Y. A., Kuhayivsky A. I., Papp I. O., Vasilenko E. Yu. Construction and optimization of recongnizing systems. *Інформаційні технології і системи* : наук.-техн. журн., Львів. 1999. Т. 1, № 1. С. 122–125.

204. Василенко Ю. А., Василенко Е. Ю., Повхан І. Ф. Концептуальна основа систем розпізнавання образів на основі метода розгалуженого вибору ознак. *European Journal of Enterprise Technologies*. 2004. Vol. 7, № 1. P. 13–15.

205. Василенко Ю. А., Василенко Е. Ю., Повхан І. Ф. Автоматична побудова автономних систем розпізнавання образів на основі методу РВО. *"Наука і освіта 2004"* : матеріали VII Міжнар. наук.-практ. конф. (Дніпропетровськ). 2004. Т. 77. С. 32–34.

206. Василенко Ю. А., Василенко Е. Ю., Повхан І. Ф. Метод розгалуженого вибору ознак в математичному конструюванні багаторівневих систем розпізнавання образів. *Штучний Інтелект*. 2003. №7. С. 246–249.

207. Василенко Ю. А., Василенко Е. Ю., Славік С. С., Повхан І. Ф. Автоматизована побудова СР образів на основі комплексних математичних моделей. *Наука і освіта 2003* : матеріали VI Міжнар. наук.-практ. конф., (Дніпропетровськ). 2003. Т. 30. С. 30–32.

208. Повхан І. Ф. Генерація узагальнених ознак в методі розгалуженого вибору ознак. *Штучний Інтелект*. 2004. № 3. С. 516–522.

209. De Mántaras R. L. A distance-based attribute selection measure for decision tree induction. *Machine Learning*. 1991. № 6. P. 81–92.

210. Goodman R. M. and Smyth P. Decision tree design from a communication theory standpoint. *IEEE Transactions on Information Theory*. 1988. Vol. 34, №5. P. 979–994.

211. Hancock T., Jiang T., Li M. and Tromp J. Lower bounds on learning decision lists and trees. *Information and Computation*. 1996. №126. P. 114–122.

212. Murthy S. K. and Salzberg S. Decision Tree Induction: How Effective Is the Greedy Heuristic? *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, Montreal, Kanada, August 20–21 1995. AAAI Press, 1995. P. 222–227.

213. Page D. and Ray S. Skewing. An efficient alternative to lookahead for decision tree induction. *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, August 9–15 2003, Acapulko, Mexico. Publisher Not Avail, 2003. P. 601–612.

214. Yang J. and Li Y. P. Orthogonal relief algorithm for feature selection. *Lecture Notes in Computer Science*. 4113. 2006. P. 227–234.

215. Finnson H. Generalized Monte-Carlo Tree Search Extensions for General Game Playing. *Proceedings 26th AAAI Conference on Artificial Intelligence*, July 22–26 2012, Toronto, Canada. Published by The AAAI Press, 2012. P. 1550–1556.

216. The Grand Challenge of Computer Go: Monte Carlo Tree Search and Extensions / S. Gelly et al. *Communications of the ACM*. 2012. Vol. 55, № 3. P. 106–113.

217. Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, 2016. URL: http:// www.deeplearningbook.org.

218. Liang M., Hu X. Recurrent Convolutional Neural Network for Object Recognition. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 7–12 2015, Boston, MA, USA. IEEE : Curran Associates, Inc., 2015. P. 3367–3375.

219. A Survey of Monte Carlo Tree Search Methods / C. B. Browne et al. *IEEE Transactions on Computational Intelligence and AI in Games*. March 2012. Vol. 4, № 1. P. 1–43.

220. Mastering the Game of Go with Deep Neural Networks and Tree Search / D. Silver et al. *Nature*. 2016. Vol. 529, № 7587. P. 484–489.

221. Naumenko I., Myronenko M., Piatachenko V. Information-extreme learning of on-board system for recognition of ground vehicle. *Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019)*, Zaporizhzhia, Ukraine, April 15–19 2019. Zaporizhzhia, 2019. P. 121–132.

222. Dovbysh A. S., Moskalenko V. V., Rizhova A. S. Information-Extreme Method for Classification of Observations with Categorical Attributes. *Cibernetica and Systems Analysis*. 2016. Vol. 52, №2. P. 45–52. DOI: 10.1007/s10559-016-9818-1

223. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework / Alcalá-Fdez J., Fernandez A., Luengo J. at. al. *Journal of multiple-valued logic and soft computing*. 2010. Vol. 17, № 4. P. 255–287.

224. Zhou Z. H. Ensemble Methods Foundations and Algorithms. Chapman & Hall/CRC Press, 2012. 218 p.

225. Yoav F., Schapire R., Abe N. A Short Introduction to Boosting. *Journal of Japanese Society For Artificial Intelligence*. 1999. Vol. 14, № 5. P. 771–780.

226. Subbotin S., Oliinyk A. The dimensionality reduction methods based on computational intelligence in problems of object classification and diagnosis. *Recent Advances in Systems, Control and Information Technology* / R. Szewczyk, M. Kaliczyńska (eds.). Cham : Springer, 2017. P. 11–19. (Advances in Intelligent Systems and Computing. Vol. 543).

227. Subbotin S. A. Methods of sampling based on exhaustive and evolutionary search. *Automatic Control and Computer Sciences*. 2013. Vol. 47, № 3. P. 113–121. DOI: 10.3103/s0146411613030073

228. Повхан І. Ф. Загальна схема алгоритму усунення помилок розпізнавання в логічних деревах класифікації. *Technical Sciences: History, The Present Time, The Future, Eu Experience (informatics and cybernetics, electronics, radio engineering and communications, automation and computer technology, mechanic engineering, transport*:

Proceedings of the international scientific and practical conference, Wloclawek (Republic of Poland), September 27–28 2019. Wloclawek, 2019. P. 42–45.

229. Повхан І. Ф. Задача апроксимації вибірки дискретних наборів геометричними об‘єктами. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), № 3. С. 136–142.

230. Aggarwal C. C. Data Mining: Text Book. Springer, 2015. 734 p.

231. A comparative study of decision tree ID3 and C4.5 / B. Hssina, A. Merbouha, H. Ezzikouri, M. Erritali. *International Journal of Advanced Computer Science and Applications (IJACSA)*. 2014. P. 13–19. (Special Issue on Advances in Vehicular Ad Hoc Networking and Applications).

232. Rokach L., Maimon O. Data Mining with decision trees: Theory and Applications. 2nd Edition. Singapore : World Scientifc Publishing Co. Pte. Ltd, 2015. 305 p.

233. Zaki M. J., Meira W. Data Mining and Analysis: Fundamental Concepts and Algorithms. New York : Cambridge University Press, 2014. 660 p.

234. Witten I. H., Frank E. Data Mining. Practical Machine Learning Tools and Techniques. Second Edition. San Francisco : Elsevier Inc., 2005. 558 p.

235. Venugopal K. R., Srinivasa K. G., Patnaik L. M. Soft Computing for Data Mining Applications. Berlin Heidelberg : Springer-Verlag, 2009. 354 p. (Studies in Computational Intelligence. Vol. 190).

236. The Top Ten Algorithms in DataMining / edited by Xindong Wu, Vipin Kumar. Taylor & Francis Group, LLC, 2009. 2006 p.

237. Nilsson N. J. Introduction to machine learning. An early draft of a proposed textbook. Stanford : Stanford University, 1998. 188 p.

238. McNamee P., Celona J. Decision Analysis for the Professional. Fourth edition. SmartOrg, Inc., 2008. 342 p.

239. Harrington P. Machine Learning in Action. Shelter Island : Manning Publications Co., 2012. 354 p.

240. Han J., Kamber M., Pei J. Data Mining Concepts and Techniques. Third Edition. Morgan Kaufmann Publishers is an imprint of Elsevier, 2012. 740 p.

241. Berry L., Linoff G. Data Mining Techniques For Marketing, Sales, and Customer Relationship Management. Second Edition. Indianapolis, Indiana : Wiley Publishing, Inc., 2004. 672 p.

242. Berry M., Browne M. Lecture Notes in Data Mining. Singapore : World Scientific Publishing Co., 2006. 237 p.

243. An Introduction to Statistical Learning with Applications in R / G. James, D. Witten, T. Hastie, R. Tibshirani. Springer New York Heidelberg Dordrecht London. 2013. 418 p.

244. XGBoost Documentation. URL: https://xgboost.readthedocs.io/en/latest

245. Алгоритм XGBoost. URL: https://medium.com/nuances-of-programming/

246. XGBoost. URL: https://en.wikipedia.org/wiki/XGBoost

247. XGBoost is an optimized distributed gradient boosting library. URL: https://github.com/dmlc/xgboost

248. Dmlc XGBoost. URL: https://techcave.ru/posts/81-sozdaem-pervuyu-xgboost-model-na-python-s-ispolzovaniem-scikit-learn.html

249. XGBoost Bauman National Library. URL: https://ru.bmstu.wiki/XGBoost

250. CRAN – Package xgboost. URL: https://cran.r-project.org/web/packages/xgboost/index.html

251. How to Use XGBoost for Time Series Forecasting. URL: https://machinelearningmastery.com/xgboost-for-time-series-forecasting/

252. RAPIDS and XGBoost. URL: https://rapids.ai/xgboost.html

253. Tree Boosting With XGBoost. URL: http://pzs.dstu.dp.ua/DataMining/boosting/bibl

254. Boosting performance with XGBoost. URL: https://towardsdatascience.com/boosting-performance-with-xgboost-b4a8deadede7

255. XGBoost – H2O Documentations. URL: https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/xgboost.html

256. Welcome to LightGBM documetations. URL: https://lightgbm.readthedocs.io/en/latest/

257. LightGBM vs XGBoost. URL: https://edwvb.blogspot.com/2017/06/lightgbm-vs-xgboost.html

258. Understanding LightGBM Parameters (and How to Tune Them). URL: https://towardsdatascience.com/understanding-lightgbm-parameters-and-how-to-tune-them-6764e20c6e5b

259. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. URL: https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf

260. LightGbmExtensions.LightGbm Method. URL: https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.lightgbmextensions.lgbm

261. LightGBM 3.0.0. URL: https://www.nuget.org/packages/LightGBM/

262. LightGBM (Light Gradient Boosting Machine). URL: https://www.geeksforgeeks.org/lightgbm-light-gradient-boosting-machine/

263. Ranking with LightGBM Models. URL: https://docs.vespa.ai/documentation/lightgbm.html

264. LightGBM: An Effective and Scalable Algorithm for Prediction of Chemical Toxicity–Application to the Tox21 and Mutagenicity Data Sets. URL: https://pubs.acs.org/doi/10.1021/acs.jcim.9b00633

265. LightGBM on the GPU – The Kernel Trip. URL: https://www.thekerneltrip.com/machine-learning/lgbmgpu/

266. Census income classification with LightGBM. URL: https://slundberg.github.io/shap/notebooks/Census%20income%20classification%20with%20LightGBM.html

267. The Top 32 Lightgbm Open Source Projects. URL: https://awesomeopensource.com/projects/lightgbm

268. Gradient Boosting with Scikit-Learn, XGBoost, LightGBM, and CatBoost. URL: https://machinelearningmastery.com/gradient-boosting-with-scikit-learn-xgboost-lightgbm-and-catboost/

269. Python API — LightGBM documentation. URL: http://devdoc.net/bigdata/LightGBM-doc-2.2.2/Python-API.html

270. An Overview of LightGBM. URL: https://www.avanwyk.com/an-overview-of-lightgbm/

271. Лбов Г. С. Методы обработки разнотипных экспериментальных данных. Новосибирск : Наука, 1981. 160 с.

272. Геология и математика / Воронин Ю. А. и др. Новосибирск : Наука, 1970. 225 с.

273. Распознавание образов в задачах оценки сейсмической опасности. URL: https://cyberleninka.ru/article/n/raspoznavanie-obrazov-v-zadachah-otsenki-seysmicheskoy-opasnosti/viewer

274. Коллективы решающих правил. URL: http://www.codenet.ru/progr/alg/ai/htm/gl3_9.php

275. Коллективы решающих правил, основанные на учёте их условий компетентности. URL: https://studopedia.org/8-27055.html

276. Theodoridis S., Koutroumbas C. Pattern Recognition. 4th Edition. Elsevier Inc., 2009. 840 p.

277. Esmaeili M., Rahmati M. Creating of Multiple Classifier Systems by Fuzzy Decision Making in Human-Computer Interface Systems. *IEEE International Fuzzy Systems Conference*, 23-26 July 2007, London, UK. Publisher : Piscataway, IEEE, 2007. P. 1–7.

278. Мазуров В. Д. Метод комитетов в задачах оптимизации и классификации. Москва : Наука, 1990. 248 с.

279. Pardo M., Sberveglieri G. Learning from data: a tutorial with emphasis on modern pattern recognition methods. *Sensors Journal, IEEE*. 2002. Vol. 2, №3. P. 203–217.

280. Горелик А. Л., Гуревич И. Б., Скрипкин В. А. Современное состояние проблемы распознавания: некоторые аспекты. Москва : Радио и связь, 1985. 160 с.

281. Tresp V. Committee machines. *Handbook for Neural Network Signal Processing* / edited by Yu Hen Hu, Jenq-Neng Hwang. CRC Press, 2001. P. 135–151.

282. Савченко В. В., Савченко А. В. Принцип минимального информационного рассогласования в задаче распознавания дискретных объектов. *Известия вузов России. Радиоэлектроника*. 2005. Вып. 3. С. 10–18. ISSN 1993-8985.

283. Савченко А. В. Выбор параметров алгоритма распознавания изображений на основе коллектива решающих правил и принципа максимума апостериорной вероятности. *Компьютерная Оптика*. 2012. Т. 36, № 1. С. 116–122.

284. Коллективы решающих правил. URL: https://helpiks.org/2-84877.html

285. Коллектив решающих правил при оценке состояния здоровья человека. URL: https://www.twirpx.com/file/2590309/

286. Повхан І. Ф. Модифікований метод побудови дерев класифікації. *Комплексне забезпечення якості технологічних процесів та систем* : матеріали X міжнар. наук.-практ. конф., Чернігів, 29–30 квіт. 2020 р. Чернігів, 2020. Т. 2. С. 167–169.

287. Повхан І. Ф. Питання представлення дискретних зображень в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), №6. С. 154–159.

288. Nabney I. T. NETLAB: Algorithms for pattern recognition. Springer, 2004. 330 p.

289. Алексанян А. А., Журавлев Ю. И. 0б одном подходе к вопросу построения эфективных алгоритмов распознавания. *Журнал вычислительной математики и математической физики*. 1985. Т. 25, № 2. С. 283–291.

290. Дюсембаев А. Е. Синтез корректных алгоритмов в замыкании распознающих алгоритмов с представительными наборами и системами опорных множеств. *Журнал вычислительной математики и математической физики*. 1983. Т. 23, № 6. С. 1487–1496.

291. Murthy S. K., Kasif S. and Salzberg S. A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*. August 1994. № 2. P. 1–33.

292. Perner P. Improving the accuracy of decision tree induction by feature preselection. *Applied Artificial Intelligence.* 2001. Vol. 15, № 8. P. 747–760.

293. Polikar R. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*. 2006. Vol. 6, № 3. P. 21–45.

294. Rodriguez J. J., Kuncheva L. I. and Alonso C. J. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2006. Vol. 28, № 10. P. 1619–1630.

295. Rokach L., Maimon O. Feature set decomposition for decision trees. *Journal of Intelligent Data Analysis*. 2005. Vol. 9, № 2. P. 131–158.

296. Shilen S. Nonparametric classification using matched binary decision trees. *Pattern Recognition Letters*. 1992. № 13. P. 83–87.

297. Skurichina M. and Duin R. P. W. Bagging, boosting and the random subspace method for linear classifiers. *Pattern Analysis and Applications*. 2002. Vol. 5, № 2. P. 121–135.

298. Блох А. Ш. Граф-схемы и их применение. Минск : Вышэйшая школа, 1975. 304 с.

299. Загоруйко Н. Г. Методы распознавания и их применение. Москва : Сов. радио, 1972. 206 с.

300. Марр Д. Зрение. Информационный подход к изучению представления и обработки зрительных образов / пер. с англ. Москва : Радио и связь, 1987. 400 с.

301. Орлов В. А. Граф-схемы алгоритмов распознавания. Москва : Наука, 1982. 118 с.

302. Алгоритм C4.5 URL: http://datascientist.one/algorithm-c4-5/

303. Деревья решений — C4.5 математический. URL: https://loginom.ru/blog/decision-tree-c45-1

304. Algorithm C4.5 URL: https://wiki.loginom.ru/articles/algorithm-C-4-5.html

305. What is the C4.5 algorithm and how does it work. URL: https://towardsdatascience.com/what-is-the-c4-5-algorithm-and-how-does-it-work-2b971a9e7db0

306. C5.0 Classification Models. URL: https://cran.r-project.org/web/packages/C50/vignettes/C5.0.html

307. C5.0 Decision Trees and Rule-Based Models. URL: https://topepo.github.io/C5.0/reference/C5.0.html

308. C5.0 An Informal Tutorial. URL: https://www.rulequest.com/see5-unix.html

309. C5.0.default. URL: https://www.rdocumentation.org/packages/C50/versions/0.1.3.1/topics/C5.0.default

310. Determining Creditworthiness for Loan Applications Using C5.0 Decision Trees. URL: https://rpubs.com/cyobero/C50

311. Методы построения деревьев решений в задачах классификации в Data Mining. URL: https://ami.nstu.ru/~vms/lecture/data_mining/trees.htm

312. Градиентый бустинг. URL: https://neurohive.io/ru/osnovy-data-science/gradientyj-busting/

313. Overview of Boosting Methods. URL: http://www.machinelearning.ru/wiki/images/9/9a/fonarev.overview_of_boosting_methods.pdf

314. Деревья принятия решений и градиентный бустинг. URL: https://logic.pdmi.ras.ru/~sergey/teaching/mlhse17/16-xgboost.pdf

315. Паводок на Закарпатті 1998 року. URL: https://uk.wikipedia.org/wiki/Паводок_на_Закарпатті_1998_року

316. Гидрологические исследования реки. URL: https://collectedpapers.com.ua/ru/methodology-of-field-physical-and-geographical-studies/gidrologichni-doslidzhennya-richki

317. Гідрологія суші, моніторинг і стан довкілля. URL: https://uhmi.org.ua/pub/np/253/18_Sosedko+Luk.pdf

Додаток А
Матеріали про впровадження результатів роботи

УКРАЇНА
МІНІСТЕРСТВО ОСВІТИ І НАУКИ

ЗАКАРПАТСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ІНФОРМАТИКИ

UKRAINE
MINISTRY OF EDUCATION AND SCIENCE

TRANSCARPATHIAN STATE UNIVERSITY
INFORMATION SCIENCES FACULTY

88015, Закарпатська область, м. Ужгород,
вул. Заньковецької, 89А
Тел.: (03122) 515-24, 508-00, 514-34, факс (0312) 61-25-35
E-mail:ZakDU@ukrpost.net

89A, Zanykovetska str.,
Uzhhorod, Transcarpathia, 88015
Tel.: (03122) 515-24, 508-00, 514-34, fax (0312) 61-25-35

№ "17" від 07.09 2011р.

ДОВІДКА
про використання досліджень, моделей та методів Повхана Ігоря Федоровича

Набір моделей дерев класифікації, методів та програмного забезпечення (ОРІОН ІІІ) досліджень к.т.н. Повхана Ігоря Федоровича використовувались при виконанні науково – дослідної роботи "Моделювання та передбачення надзвичайних ситуацій в Карпатському регіоні та країнах Центрально - Східної Європи", номер державної реєстрації роботи – 0106V00285, категорія роботи – фундаментальні дослідження (КПКВ 2201020), 01 Фундаментальні дослідження з найважливіших проблем природничих, суспільних і гуманітарних наук факультету інформатики ЗакДУ в задачах прогнозування паводкових явищ Карпатського регіону.

В.о. декана факультету інформатики **Ю.Ю. Білак**

Богачова О.Р.
(0312) 65-52-50

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
вул.Підгірна, 46, м. Ужгород, Закарпатська область, 88000
тел: (0312) 61-33-21, 42-99-89 факс: (0312) 61-33-96
e-mail: official@uzhnu.edu.ua Код ЄДРПОУ 02070832

31.08.20 № 2449/01-14 На № _____ від _____

Довідка
про використання в ДВНЗ "Ужгородський національний університет"
наукових результатів докторської дисертаційної роботи
доцента, кандидата технічних наук
Повхана Ігоря Федоровича

Моделі, алгоритми, методи та програмне забезпечення дисертаційного дослідження Повхана І.Ф. впроваджені в навчальний процес факультету інформаційних технологій ДВНЗ "УжНУ". Зокрема, при підготовці навчально – методичних комплексів з дисциплін «Методи та засоби штучного інтелекту», «Програмування в системах абстрактних об'єктів і задачі штучного інтелекту», «Теорія алгоритмів», «Технології програмування та створення програмних продуктів», «Пакети прикладних програм», «Організація баз даних і баз знань» спеціальностей 121 "Інженерія програмного забезпечення" 122 "Комп'ютерні науки", 126 "Інформаційні системи та технології".

Результати досліджень, приведених у дисертаційні роботі Повхана І.Ф., використовуються студентами при виконанні комплексу курсових, кваліфікаційних, дипломних та магістерських робіт.

Проректор з наукової роботи Студеняк І.П.

ІнфоСфера
ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ

- Комп'ютери
- Копіювальні апарати
- Печатки та штампи
- Сервіс

Офіційний представник дистрибютора TOSHIBA в Україні

Україна, 88000, м. Ужгород, вул. Кошицька, 7, тел./факс: (312) 61-38-78, 61-66-62
E-mail: infosphere@ukrpost.net

Довідка
впровадження результатів докторської дисертаційної роботи доцента ДВНЗ "Ужгородський національний університет" кандидат технічних наук
Повхана Ігоря Федоровича

Основним напрямком дисертаційного дослідження Повхана І.Ф. є розробка моделей і методів систем класифікації ситуаційних явищ широкого спектру. Основний об'єм досліджень проведений протягом 2014 – 2019 років Повханом І.Ф., показали, що математичне, алгоритмічне та програмне забезпечення дозволяє ефективно систематизувати та виявити основні типи та класи апаратних відмов комп'ютерної техніки.

Моделі, методи та програмне забезпечення, запропоновані Повханом І.Ф. були використані ТОВ «Інфосфера» при діагностиці, сервісному обслуговуванні та ремонті продукції категорії комп'ютерних систем. Використання запропонованих у дисертаційному дослідженні Повхана І.Ф. моделей та методів забезпечує ефективну процедуру виявлення та усунення апаратних збоїв. Це доведено неодноразовим застосуванням зазначеного підходу.

Директор ТОВ «Інфосфера» С. В. Куруца

МЕДІА СЕРВІС
КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ

КП "МЕДІА-СЕРВІС"
Код: 30459003
Юрид. адреса: 88000, м. Ужгород, вул. Бічна, 1
Пошт. адреса: 88018 м. Ужгород, вул. Бічна, 1
Телефон/факс: (0312) 67-90-99, 67-90-97
Електронна адреса: office@medias-ua.com
Р/р 26005015000285 в АКБ КомІнвестБанк МФО 312248
ІПН 304590007010, свідоцтво платника ПДВ № 200066833

ДОВІДКА
впровадження результатів докторської дисертаційної роботи доцента ДВНЗ "Ужгородський національний університет" кандидат технічних наук
ПОВХАНА ІГОРЯ ФЕДОРОВИЧА

№16, 11 вересня 2020 р.

Дисертаційне дослідження Повхана І.Ф. спрямоване на розробка загальних моделей і методів систем класифікації/прогнозування ситуаційних явищ в різних галузях практичного застосування. Представлені моделі дерев класифікації Повхана І.Ф., показали, що математичне, алгоритмічне та програмне забезпечення дозволяє виявити основні типи апаратних відмов комп'ютерної техніки (давати рекомендації, щодо їх усунення), на основі чого забезпечувати ефективний механізм сервісного обслуговування. Так набір моделей, методів та програмне забезпечення побудови моделей класифікації, запропоновані Повханом І.Ф. були використані ТОВ "Медіа Сервіс" при діагностиці, сервісному обслуговуванні та ремонті широкого спектру комп'ютерних систем. Представлене програмне забезпечення та набір моделей АДК гарантує ефективну процедуру виявлення та усунення апаратних збоїв. Даний факт доведено неодноразовим застосуванням зазначеного інструментарії в сервісному центрі компанії.

Директор КП "Медіа-Сервіс" І. О. Матяшов

КОЛЕКТИВНЕ ПІДПРИЄМСТВО «МЕДІА-СЕРВІС» код 30459003 УКРАЇНА УЖГОРОД

Закарпатська обласна громадська організація
Патріотичний оборонно-спортивний центр «Вітязь»
м. Ужгород, пл. Ш.Петефі, 21, тел / факс: (0312) 616554, (0312) 615537 код 26031417
р/р: 26006016756364, Укрексімбанк м. Ужгорода, МФО 312226

Довідка

про використання результатів досліджень моделей систем класифікації на основі концепції дерев рішень кандидат технічних наук

Повхана Ігоря Федоровича

Результати досліджень систем класифікації на основі моделей дерев рішень, проведених кандидатом технічних наук Повхан Ігорем Федоровичем успішно застосовуються у діяльності Закарпатської обласної громадської організації Патріотичний оборонно-спортивний центр "Вітязь". Моделі, методи та програмне забезпечення запропоновані І. Повхан, використовуються при регламентних роботах парашутних систем для аналізу загального стану технічних засовів, виявлення та усунення певних типів відмов.

Так розроблене програмне Повхана Ігоря Федоровича (модель класифікації ситуаційного стану) використовувалось для аналізу статичних лог – даних пристроїв безпеки парашутних систем типу Cypres та Vigil I.

18 травня 2012 р.

Голова громадської організації
Патріотичний оборонно-спортивний центр "Вітязь" О.Ю. Юшков

УЖГОРОДСЬКА МІСЬКА РАДА
ВИКОНАВЧИЙ КОМІТЕТ

пл. Поштова, 3, м. Ужгород, 88000; тел.: (0312) 61-70-71, тел./факс: (0312) 61-51-91
E-mail: umr@rada-uzhgorod.gov.ua; сайт: www.rada-uzhgorod.gov.ua; код ЄДРПОУ 04053699

03-12/391 № 04.09.2020 На № _____ від _____

Довідка

про використання результатів докторської дисертаційної роботи доцента ДВНЗ «Ужгородський національний університет», кандидата технічних наук Повхана Ігоря Федоровича

Управління економічного розвитку міста Ужгородської міської ради часто зустрічається із потребою вирішення задач ранжування та класифікації показників та об'єктів економічного характеру при визначенні поточного політико-економічного розвитку м.Ужгород.

Для ефективного розв'язку набору таких задач було використано підходи запропоновані в дисертаційному дослідженні Повхана І.Ф., які зводять вирішення даних проблем до задач класифікації та прогнозування. Зокрема, були використані моделі та методи на основі концепції дерев класифікації (алгоритмічних дерев класифікації).

Використання запропонованих у дисертаційній роботі Повхана І.Ф. моделей, методів та алгоритмів забезпечує високу ефективність при вирішенні задач класифікації та прогнозування економічних явищ в умовах впровадження нових підходів розвитку.

Заступник міського голови **Василь ГОМОНАЙ**

Павло Логвінов 612000

Сертифікат
58E2D9E7F900307B040000007DA427005C018300
Підписувач Гомонай Василь Васильович
Дійсний з 02.04.2020 8:51 по 02.04.2022 8:51

Виконавчий комітет
Ужгородської міської ради

03-12/391 від 04.09.2020

Додаток В

Авторське свідоцтво, участь в проектах

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ДЕПАРТАМЕНТ ІНТЕЛЕКТУАЛЬНОЇ ВЛАСНОСТІ

Україна, МСП 03680, Київ-35, вул. Урицького, 45
Тел.: (044) 494-06-06 Факс: (044) 494-06-67

РІШЕННЯ

ПРО РЕЄСТРАЦІЮ АВТОРСЬКОГО ПРАВА НА ТВІР

Державний департамент інтелектуальної власності розглянув заяву
Закарпатський державний університет, вул. Заньковецької, 89-А, м. Ужгород, Закарпатська обл., 88015

(повне ім'я фізичної або повне офіційне найменування юридичної особи, адреса)

заявка від **30.09.2009** **№ 31283**

про реєстрацію авторського права на твір і прийняв рішення зареєструвати авторське право на службовий твір **Комп'ютерна програма "Програмний комплекс (ОРІОН II) класифікації явищ та дискретних об'єктів"("ОРІОН II"); Повхан Ігор Федорович; Закарпатський державний університет**

(вид, повна, скорочена (за наявності) назва твору, повне ім'я, псевдонім (за наявності) автора (ів), повна офіційна назва роботодавця)

Внесення відомостей до Державного реєстру свідоцтв про реєстрацію авторського права на твір та видача свідоцтва будуть здійснені за умови сплати збору за оформлення і видачу свідоцтва про реєстрацію авторського права на твір відповідно до п.3 постанови Кабінету Міністрів України від 27 грудня 2001 року № 1756 "Про державну реєстрацію авторського права і договорів, які стосуються права на твір".

Якщо протягом трьох місяців від дати одержання заявником рішення про реєстрацію авторського права на твір Державний департамент не одержав документ про сплату збору за оформлення і видачу свідоцтва у розмірі та порядку, визначених законодавством, або копію документа, що підтверджує право на звільнення від сплати зазначеного збору, заявка вважається відхиленою і реєстрація авторського права та публікація відомостей про реєстрацію Державним департаментом не проводиться.

Голова Державного департаменту інтелектуальної власності **М.В. Паладій**

Міністерство освіти і науки України * Державний департамент інтелектуальної власності * Ідентифікаційний код 21716978

УКРАЇНА

Міністерство освіти і науки України

Державний департамент інтелектуальної власності

СВІДОЦТВО

про реєстрацію авторського права на твір

№ 31160

Комп'ютерна програма "Програмний комплекс (ОРІОН II) класифікації явищ та дискретних об'єктів" ("ОРІОН II")

(вид, назва службового твору)

Автор(и) **Повхан Ігор Федорович**

(повне ім'я, псевдонім (за наявності))

Авторські майнові права належать **Закарпатський державний університет, вул. Заньковецької, 89-А, м. Ужгород, Закарпатська обл., 88015**

(повне ім'я фізичної та/або повне офіційне найменування юридичної особи, адреса)

Дата реєстрації 30.11.2009

Голова Державного департаменту інтелектуальної власності М.В.Паладій

М.П.

PANEURÓPSKA VYSOKÁ ŠKOLA

Pan European University

FAKULTA INFORMATIKY

Faculty of Informatics

Bratislava, Slovenská republika

CERTIFIKÁT

No.14/2017

potvrdzuje, že

pán **Ihor Povkhan**

bol účastníkom medzinárodného projektu

„Inovatívne metódy vzdelávania na podporu partnerstiev – InovEduc“

realizovaného v rokoch 2015 – 2017.

Projekt č. CBC01008 bol podporený z Nórskych fondov a štátneho rozpočtu Slovenskej republiky v rámci programu SK08 Cezhraničná spolupráca.

Hlavný riešiteľ projektu:

Paneurópska vysoká škola - Slovensko

Partneri projektu

Centrum pre európsku politiku - Slovensko
Carpathia Užhorod - Ukrajina
Generálny konzulát Slovenskej republiky v Užhorode - Slovensko
Zakarpatský inštitút postdiplomového pedagogického vzdelávania - Ukrajina
Užhorodská národná univerzita - Ukrajina
Evanjelická spojená škola v Prešove – Slovensko
Lingvistické gymnázium T. H. Ševčenka Užhorod – Ukrajina
Metodicko pedagogické centrum Bratislava - Slovensko
Ústav etnológie Slovenskej akadémie vied, Bratislava – Slovensko
Imsa Knowledge Company AS - Nórsko

Bratislava 30. apríl 2017

.....

doc. RNDr. Eugen Ružický, CSc., dekan
Fakulta informatiky Paneurópska vysoká škola

„Granty EHP a Nórska – Spoluprácou k spoločným hodnotám“
„Slovensko – Ukrajina: Spolupráca naprieč hranicou“
www.eeagrants.sk

Додаток С

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ ТА ВІДОМОСТІ ПРО АПРОБАЦІЮ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЇ

1. Повхан І. Ф. Логічні дерева класифікації в задачах штучного інтелекту : монографія. Ужгород : Поліграфцентр - Ліра, 2019. 276 с.
2. Повхан І. Ф. Питання структурної складності для випадку регулярного логічного дерева. *Scientific and technical progress in European countries and the contribution of higher education institutions* : collective monograph. Riga: Izdevnieciba “Baltija Publisher”, 2020. 308 p.
3. Povhan I. F. Logical recognition tree construction on the basis a step-to-step elementary attribute selection. *Radio Electronics, Computer Science, Control*. 2020. № 2. P. 95–106.
4. Povkhan I. F. The general concept of the methods of algorithmic classification trees. *Radio Electronics, Computer Science, Control*. 2020. № 3. P. 108–121.
5. Povhan I. F. Limited method for the case of algorithmic classification tree. Radio Electronics, Computer Science, Control. 2020. № 4. P. 106-118.
6. Povhan I. Designing of recognition system of discrete objects. *Proceedings of the “2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP)”*, August 23–27 2016 Lviv, Ukraine. Lviv, 2016. P. 226–231.
7. Povkhan I., Lupei M. The algorithmic classification trees. *Proceedings of the “2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP)”*, August 21–25 2020, Lviv, Ukraine. Lviv, 2020. P. 37–44.
8. Lupei M., Mitsa A., Povkhan I., Sharkan V. Determining the eligibility of candidates for a vacancy using artificial neural networks. *Proceedings of the “2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP)”*, August 21–25 2020, Lviv, Ukraine. Lviv, 2020. P. 18–23.
9. Povkhan I., Lupei M., Kliap M., Laver V. The issue of efficient generation of generalized features in algorithmic classification tree methods. *International Conference on Data Stream Mining and Processing*: DSMP 2020 Data Stream Mining & Processing, Springer, Cham, 2020. P. 98–113.

10. Povkhan I. Classification models of flood-related events based on algorithmic trees. *Eastern-European Journal of Enterprise Technologies.* 2020, Vol. 6, No 4, (108 (2020)). P. 58-68. DOI: https://doi.org/10.15587/1729-4061.2020.219525

11. Povhan I. General scheme for constructing the most complex logical tree of classification in pattern recognition discrete objects. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2019. Vol. 11. С. 73–80.

12. Povhan I. Generation of elementary signs in the general scheme of the recognition system based on the logical tree. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2019. Vol. 12. С. 20–29.

13. Povhan I. Question of the optimality criterion of a regular logical tree based on the concept of similarity. *Електроніка та інформаційні технології* : зб. наук. пр. Lviv, 2020. Vol. 13. С. 19–27. DOI: https://doi.org/10.30970/eli.13.2

14. Повхан І. Ф. Особливості синтезу узагальнених ознак при побудові систем розпізнавання за методом логічного дерева. *Інформаційні технології та комп'ютерне моделювання ІТКМ-2019* : матеріали міжнар. наук.-практ. конф., Івано- Франківськ – Яремче, 20–25 трав. 2019. Івано-Франківськ, 2019. С. 169–174.

15. Povhan I. Generation of classification schemes in the form of logical trees in discrete object recognition problems. *Modern Problems of Mathematical Modeling, Automated Control and Information Technologies MCIT-2019*: International Scientific and Practical Conference, Rivne, Noveber 14–16 2019. Рівне, 2019. P. 195–200.

16. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф., Поліщук В. В. Групова та індивідуальна оцінка важливості бульових аргументів. *Вісник Національного технічного університету «Харківський політехнічний інститут»*: зб. наук. пр. 2011. №53. С. 57–64.

17. Повхан І. Ф. Задача загальної оцінки складності максимального побудованого логічного дерева класифікації. *Вісник Національного технічного університету «Харківський політехнічний інститут»* : зб. наук. пр. Серія: Інформатика та моделювання. 2019. №13 (1338). С. 104–117.

18. Повхан І. Ф. Питання побудови деревоподібних моделей розпізнавання образів. *Вісник Національного технічного університету «Харківський політехнічний інститут»*: зб. наук. пр. Серія: Інформатика та моделювання. 2019. №28 (1353). С. 39–57.

19. Povhan I. Logical classification trees in recognition problems. *Kwartalnik Naukowo-Techniczny: Informatyka Automatyka Pomiary w gospodarce o ochronie srodowiska*. Krakow, 2020. №2. P. 12–16. DOI: http://doi.org/10.35784/iapgos.927

20. Повхан І. Ф. Питання гнучкості логічних дерев класифікації в задачах розпізнавання образів. *Технічні науки та технології*. 2019. №3 (17). С. 131–140.

21. Повхан І. Ф. Метод побудови алгоритмічного дерева другого типу на основі апроксимації навчальної вибірки набором алгоритмів класифікації. *Технічні науки та технології* : наук. журн. Чернігів, 2020. №2 (20). С. 126–139.

22. Повхан І.Ф. Питання синтезу дискретних зображень в задачах розпізнавання образів. *Вісник Вінницького політехнічного інституту* : наук. журн. Вінниця, 2020. №4 (151). С. 50 – 57.

23. Повхан І. Ф. Моделі алгоритмів розпізнавання у вигляді логічних дерев класифікації. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2019. №1 (475). С. 156–163.

24. Повхан І. Ф. Питання оцінки ефекту перестановки ярусів логічного дерева максимальної складності для бінарного випадку. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2020. №2 (480). С. 99–107.

25. Повхан І. Ф. Питання оптимізації логічного дерева шляхом перестановки структурних елементів. *Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова*. 2020. №3 (481). С. 91–101.

26. Повхан І. Ф. Проблема функціональної оцінки навчальної вибірки в задачах розпізнавання дискретних об‘єктів. *Вчені записки Таврійського*

національного університету імені Вернадського. Серія: Технічні науки. 2018. Т. 29 (68), №6. С. 217–222.

27. Повхан І. Ф. Поняття функції та алгебраїчної схеми розпізнавання в задачах класифікації дискретних об‘єктів. *Вчені записки Таврійського національного університету імені Вернадського*. Серія: Технічні науки. 2019. Т. 30 (69), №2. С. 171–177.

28. Повхан І. Ф. Задача апроксимації вибірки дискретних наборів геометричними об‘єктами. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), № 3. С. 136–142.

29. Повхан І. Ф., Лавер В. О. Алгоритми побудови логічних дерев класифікації в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), №4. С. 100–106.

30. Повхан І. Ф. Особливості випадкових логічних дерев класифікації в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Серія: Технічні науки. 2019. Т. 30 (69), №5. С. 138–143.

31. Повхан І. Ф. Питання представлення дискретних зображень в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2019. Т. 30 (69), №6. С. 154–159.

32. Повхан І. Ф. Питання однозначного покриття зображень прямокутниками в задачах розпізнавання образів. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2020. Т. 31 (70), №1. С. 119-124.

33. Повхан І. Ф. Моделі дерев класифікації паводкових явищ річки Уж Закарпатського регіону. *Вчені записки Таврійського національного університету імені Вернадського*. Київ, 2020. Т. 31 (70), №5. С. 100-107.

34. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф., Повхан Л. С. Вычисление важности дискретных признаков (анализ некоторых подходов). *Восточно-европейский журнал передовых технологий*. 2010. №5/4 (47). С. 71–75.

35. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Проблема оцінки складності логічних дерев розпізнавання та загальний метод їх оптимізації. *Восточно-европейский журнал передовых технологий*. 2011. №6/4 (54). С. 24–28.

36. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Загальна оцінка мінімізації деревоподібних логічних структур. *Восточно-европейский журнал передовых технологий*. 2012. №1/4 (55). С. 29–32.

37. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Дослідження стійкості максимального логічного дерева відносно перестановки ярусів. *Восточно-европейский журнал передовых технологий*. 2012. №2/4 (56). С. 15–22.

38. Василенко Ю. А., Мошкола В. П., Повхан І. Ф. Модульний синтез схем класифікації та розпізнавання. *Сучасні тенденції інформаційних технологій* : матеріали IV міжнар. наук. конф., Прага, 15–31 берез. 2008 р. Прага, 2008. С. 32–34.

39. Василенко Ю. А., Повхан І. Ф. Автоматизація побудови схем класифікації. *Теорія прийняття рішень : IV Міжнародна школа – семінар*, Ужгород, 29 вересня – 4 жовтня 2008 р. Ужгород, 2008. С. 137-138.

40. Василенко Ю. А., Василенко Ю. А., Бунда В. В., Бунда С. О., Лавер О. Г., Повхан І. Ф. Универсальный подход к анализу и обработке специальной информации. *Інновації в навчальному процесі вищих навчальних закладів: міжнародний та національний досвід* : зб. наук. ст. за матеріалами XV міжнар. наук.-практ. конф., Сніна (Словацька Республіка), 6–9 листоп. 2007 р. Ужгород : Ліра, 2008. С. 99–106.

41. Василенко Ю. А., Василенко Ю. А., Бунда В. В., Повхан І. Ф., Чедреки Я. Н. Математическое конструирование распознающих систем для дискретных объектов. *Лісабонська стратегія як визначальний чинник європейської інтеграції в галузі освіти і науки* : матеріали XVI Міжнар. наук.-практ. конф. Ужгород – Гирляни, 6–9 трав. 2008 р. Ужгород : Ліра, 2008. С. 79–83.

42. Василенко Ю. А., Завілопуло А. М., Повхан І. Ф., Повхан Л. С. Синтез систем розпізнавання на основі схем-агентів. *Інститут електронної фізики*

Національної академії наук України – 2011: матеріали міжнар. конф. молодих учених, Ужгород, 24–27 трав. 2011 р. Ужгород : Мистецька Лінія, 2011. С. 187–188.

43. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. The importance of discrete signs. *Перспективні шляхи й напрями вдосконалення освітньої системи у світлі Болонського процесу* : зб. наук. доп. за матеріалами XX Міжнар. наук.-практ. конф., Ужгород (Україна) – Кошице (Словаччина) – Мішкольц (Угорщина), 16–19 листоп. 2010 р. Ужгород : ЗакДУ, 2011. Вип. 2(21), Ч. 1. С. 46–56.

44. Василенко Ю. А., Ващук Ф. Г., Повхан І. Ф. Автоматизация построения систем классификации на основе схем-агентов. *Математическое моделирование, оптимизация и информационные технологии* : материалы 3-й междунар. конф., Кишинев, 19–23 марта 2012 г. Кишинев : Академия транспорта, информатики и коммуникаций, 2012. С. 444–446.

45. Повхан І.Ф. Побудова систем прогнозування економічних явищ на основі концепції логічного дерева класифікації. *Математичні методи, моделі та інформаційні технології в економіці :* матеріали VI міжнародної науково-методичної конференції, Чернівці, 18-19 квітня 2019 р. Чернівці : Друк –Арт, 2019. С. 134-136.

46. Повхан І. Ф. Загальна схема алгоритму усунення помилок розпізнавання в логічних деревах класифікації. *Technical Sciences: History, The Present Time, The Future, Eu Experience (informatics and cybernetics, electronics, radio engineering and communications, automation and computer technology, mechanic engineering, transport*: Proceedings of the international scientific and practical conference, Wloclawek (Republic of Poland), September 27–28 2019. Wloclawek, 2019. P. 42–45.

47. Повхан І. Ф. Проблематика методів логічних дерев класифікації в задачах розпізнавання. *Science, engineering and technology: global and current trends*: Proceedings of the international scientific and practical conference, Prague (Czech Republic), December 27–28 2019. Prague, 2019. P. 28–32.

48. Повхан І. Ф. Особливості реалізацій моделей дерев класифікації на основі селекції ознак. *Science, engineering and technology: global trends, problems and solutions*: Proceedings of the international scientific and practical conference (Part 1), Prague (Czech Republic), September 25–26 2020. Prague, 2020. P. 74-79.

49. Повхан І. Ф. Модульна концепція побудови дерев класифікації. *Цифрова економіка та інформаційні технології* : матеріали міжнар. наук.-практ. конф., Київ, 15–16 квіт. 2020 р. Київ, 2020. С. 87–90.

50. Повхан І. Ф. Методи логічних дерев класифікації в задачах штучного інтелекту. *Priority directions of science development*: Abstracts of II International Scientific and Practical Conference, Lviv (Ukraine), 25–26 November 2019. Lviv, 2019. P. 213–218.

51. Повхан І. Ф. Питання представлення неповністю визначених багатозначних логічних функцій у вигляді логічних дерев в задачах класифікації. *Topical issues of the development of modern science*: Abstracts of IV International Scientific and Practical Conference, Sofia (Bulgaria), 11–13 December 2019. Sofia, Bulgaria, 2019. P. 390–396.

52. Повхан І. Ф. Модифікований метод побудови дерев класифікації. *Комплексне забезпечення якості технологічних процесів та систем* : матеріали X міжнар. наук.-практ. конф., Чернігів, 29–30 квіт. 2020 р. Чернігів, 2020. Т. 2. С. 167–169.

53. Пастор Н. Е., Повхан І. Ф. Генерація схем розпізнавання дискретних об‘єктів на онові апроксимації навчаючої вибірки. *Радіоелектроніка та молодь в XXI столітті* : матеріали XXIII міжнар. молодіжного форуму, Харків, 16–18 квіт. 2019 р. Харків, 2019. Т. 5. С. 140–142.

54. Повхан І. Ф. Загальна концепція алгоритмічного дерева класифікації в задачах розпізнавання образів. *Інформаційні технології у житті молодих науковців Закарпаття* : матеріали V наук.-практ. конф., Ужгород, 7–8 листоп. 2019 р. Ужгород : УжНУ, 2019. С. 58–62.

55. Повхан І. Ф. Питання загальної складності процедури побудови логічного дерева. *Проблеми інформатики та моделювання* : матеріали XX наук.-техн. конф., Харків-Одеса, 16–21 верес. 2020 р. Харків : НТУ “ХПІ”, 2020. С. 68–74.

56. Povkhan I. A constrained method of constructing the logic classification trees on the basis of elementary attribute selection. *CEUR Workshop Proceedings*: Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020), Zaporizhzhia, Ukraine, April 15–19, 2020. Zaporizhzhia, 2020. Vol. 2608. P. 843–857. CEUR Workshop Proceedings (CEUR-WS.org).

57. Повхан І. Метод дерева алгоритмів для задачі класифікації геологічних даних. Перспективні напрямки сучасної електроніки, інформаційних і комп‘ютерних систем: Тези доповідей на V Всеукраїнській науково – практичній конференції MEICS – 2020, Дніпро, 25-27 листопада 2020 р. Дніпро, 2020. С. 20–22.

58. Повхан І. Моделі класифікації паводкових явищ в Закарпатському регіоні. Комп'ютерні технології, інновації, проблеми, рішення: Тези доповідей III Всеукраїнської науково - технічної конференції, Житомир, 26-27 листопада 2020 р. Житомир, 2020. С. 75-79.