НАЦІОНАЛЬНА
АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ПРОГРАМНИХ СИСТЕМ

ПРОБЛЕМИ ПРОГРАМУВАННЯ

науковий журнал

№ 2-3 ***СПЕЦІАЛЬНИЙ ВИПУСК*** ***2024***

Заснований у березні 1999 р.

ЗМІСТ

Паралельне програмування

Машинне навчання та нейронні мережі

Великі дані (Big Data)та Аналітика даних (Data Science)

Лінгвістичні системи

Моделі та засоби систем баз даних і знань

Програмування для комп'ютерних мереж та інтернет

Свідоцтво про державну реєстрацію КВ № 7490 від 01.07.2003

Науковий журнал "Проблеми програмування" занесений до переліку наукових фахових видань України, в яких можуть публікуватися основні результати дисертаційних робіт.

ISSN 1727-4907

NATIONAL ACADEMY
OF SCIENCES OF UKRAINE
INSTITUTE OF SOFTWARE SYSTEMS

PROBLEMS OF PROGRAMMING

scientific journal

№ 2-3 ***special issue*** ***2024***

Founded in March, 1999

CONTENT

ПЕРЕДМОВА

Спеціальний випуск журналу «Проблеми програмування» присвячений матеріалам Чотирнадцятої міжнародної науково-практичної конференції з програмування ***УкрПРОГ'2024,*** що вже вчотирнадцяте відбувається у Кібернетичному центрі НАН України (м. Київ). За висновками програмного комітету конференції ***УкрПРОГ'2024*** до публікації у журналі рекомендовано статті, класифіковані редколегією журналу за такими тематичними напрямками:

Теоретичні і методологічні основи програмування
Комп'ютерне моделювання
Архітектура програмного забезпечення
Інструментальні засоби та середовища програмування
Мови програмування
Методи і засоби програмної інженерії
Прикладне програмне забезпечення
Інформаційні системи
Програмні засоби захисту інформації
Агентно-орієнтовані інформаційні системи
Паралельне програмування
Машинне навчання та нейронні мережі
Великі дані (Big Data) і Аналітика даних (Data Science)
Лінгвістичні системи
Моделі та засоби систем баз даних і знань
Програмування для комп'ютерних мереж та Інтернет
Експертні та інтелектуальні інформаційні системи, штучний інтелект
Правові та соціальні аспекти програмування
Інформатизація наукових досліджень
Інформатизація освіти
Освітні та навчальні аспекти програмування
Онтологічний аналіз, Semantic Web та менеджмент знань у відкритому просторі

Як і в минулі роки, журнал «Проблеми програмування» створює можливості для публікації наукових матеріалів, що висвітлюють найважливіші досягнення у галузі програмування і програмної інженерії, визначають головні напрямки їх розвитку.

Приносимо щиру подяку за співпрацю всім авторам, членам програмного й організаційного комітету, членам редакційної групи журналу та організаторам конференції.

Голова конференції **УкрПРОГ'2024**
Генеральний директор Кібернетичного
Центру НАН України
Академік НАН України

І.В. СЕРГІЄНКО

Головний редактор журналу
«Проблеми програмування»,
Співголова програмного і
організаційного комітетів
Член-кореспондент НАН України

І.П. СІНІЦИН

ЧОТИРНАДЦЯТА МІЖНАРОДНА
НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ
З ПРОГРАМУВАННЯ

УкрПРОГ'2024

14-15 травня 2024 р.
Київ, Україна
Кібернетичний центр
Національної академії наук України

ОРГАНІЗАТОРИ КОНФЕРЕНЦІЇ

НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КІБЕРНЕТИЧНИЙ ЦЕНТР НАЦІОНАЛЬНОЇ АКАДЕМІЇ НАУК УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

КООРДИНАЦІЙНА РАДА

СЕРГІЄНКО І.В. – академік НАН України,
голова координаційного комітету
Андон П.І. (Україна), Богданов В.І. (Україна), Згуровський М.З. (Україна),
Сініцин І.П. (Україна), Хіміч О.М. (Україна)

ПРОГРАМНИЙ КОМІТЕТ

Андон П.І. (Україна) – голова Програмного комітету
Сініцин І.П. – співголова
Шевченко В.Л. – співголова

Анісімов А.В. (Україна)
Балабанов О.С. (Україна)
Горлач С.П. (Німеччина)
Дорошенко А.Ю. (Україна)
Глибовець М.М. (Україна)
Жебка В.В. (Україна)
Летичевський О.А. (Україна)
Коршун Н.В. (Україна)
Ільченко М.Ю. (Україна)
Куссуль Н.М. (Україна)
Поперешняк С.В. (Україна)
Провотар О.І. (Україна)
Хлухі (Словаччина)
Кривий С.Л. (Україна)
Нікітченко М.С. (Україна)
Панкратова Н.Д. (Україна)
Пашко С.В. (Україна)
Погорілий С.Д. (Україна)
Редько В.Н. (Україна)
Рогушина Ю.В. (Україна)
Ромео Л. (Італія)
Салєм А.Б.М. (Єгипит)
Сергієнко І.В. (Україна)
Шинкаренко В.І. (Україна)
Складанний П.М. (Україна)
Соколов В.Ю. (Україна)
Сидоров М.О. (Україна)
Пасічник В.В. (Україна)
Теленик С.Ф. (Україна)
Єршов С.В. (Україна)

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

Дергильова О.В. (Україна) – голова, Поркалова О.М. (Україна) – співголова,
Янченко О.С. (Україна), Старостін А.М. (Україна), Гребенніков А.Б. (Україна),
Салата М.В. (Україна), Грішанова І.Ю. (Україна), Єгоров В.О. (Україна)

УДК 004.4 http://doi.org/10.15407/pp2024.02-03.011

В.І. Шинкаренко, О.В. Макаров

ГЕНЕТИЧНИЙ АЛГОРИТМ ДЛЯ СТРУКТУРНОЇ АДАПТАЦІЇ АЛГОРИТМІВ СОРТУВАННЯ

Застосовано конструктивізм для формування коду алгоритму сортування. Представлено метаалгоритм генерації програмного коду. Для генерації використовуються частини існуючих алгоритмів сортування і допоміжні утиліти. Використано генетичний алгоритм для вибору алгоритму з максимальною часовою ефективністю у заданих умовах використання. Використання стандартного генетичного алгоритму стикається з проблемою, пов'язаною з різною кількістю елементарних дій по сортуванню, що призводить до використання хромосом різної довжини. Для рішення проблеми запропоновано представлення хромосоми у формі бінарного дерева. Кожен вузол має гени початку, кінця і двох вузлів-нащадків. Для формування алгоритму, який гарантовано буде сортувати масив, усі кінцеві вузли (листя) включають ген фінального сортування у кінець початкової послідовності генів. Даний ген декодується викликом існуючого алгоритму сортування, який гарантовано виконає сортування. Операції кросоверу та мутацій виконуються на хромосомах у формі бінарного дерева. Схрещення виконується за допомогою обміну вузлами дерева. Реалізовані механізми кодування і декодування алгоритму сортування із хромосоми. Для декодування і формування відповідного алгоритму сортування виконується лінеріалізація: формування текстового представлення за алгоритмом обходу дерева у глибину. Фітнес функція визначається як середній час сортування випадково сформованих масивів для сортування (для всіх хромосом однакові масиви) у деякому стабільному середовищі з урахуванням певних особливостей цих масивів. Передбачено застосування інших фітнес функцій пов'язаних з кількістю обчислень, порівнянь або перестановок. Розроблене програмне забезпечення має застосовуватись у процесі адаптації алгоритмів сортування до стабільних потоків вхідних даних та середовищ використання.

Ключові слова: алгоритм сортування, сортування, конструктивізм, генетичний алгоритм, хромосома, бінарне дерево.

V.I. Shinkarenko, O.V. Makarov

GENETIC ALGORITHM FOR STRUCTURAL ADAPTATION OF SORTING ALGORITHMS

Constructivism was applied to form the sorting algorithm code. The meta-algorithm of program code generation is presented. Parts of existing sorting algorithms and auxiliary utilities are used for generation. A genetic algorithm was used to select the algorithm with the maximum time efficiency under the given conditions of use. The use of a standard genetic algorithm faces a problem associated with a different number of elementary sorting operations, which leads to the use of chromosomes of different lengths. To solve the problem, a representation of the chromosome in the form of a binary tree is proposed. Each node has start genes, end genes, and two child nodes. To form an algorithm that is guaranteed to sort the array, all end nodes (leaves) include the final sorting gene at the end of the start genes sequence. This gene is decoded by calling the existing sorting algorithm, which is guaranteed to perform sorting. Crossover and mutation operations are performed on chromosomes in the form of a binary tree. Crossover is performed using the exchange of tree branches. Mechanisms of coding and decoding of the sorting algorithm from chromosome have been implemented. Decoding and formation of a suitable sorting algorithm is performed using linearization: formation of a textual representation using a depth-first tree traversal algorithm. The fitness function is defined as the average time of sorting randomly generated arrays for sorting (identical arrays for all chromosomes) in some stable environment, considering certain features of these arrays. It is possible to use other fitness functions related to the number of calculations, comparisons, or permutations. The developed software should be used for adaptation of sorting algorithms to stable input data streams and environments.

Key words: sorting algorithm, sorting, constructivism, genetic algorithm, chromosome, binary tree.

Вступ

Із появою Інтернету і цифрової технології взагалі, стрімко зростають обсяги даних, що генеруються та зберігаються. З цифровізацією більшості сфер життя – від бізнесу та науки до особистих комунікацій – виникає потреба в ефективному управлінні цими даними. Зростання обсягів даних ставить перед нами виклики, пов'язані зі збереженням, обробкою, аналізом та інтерпретацією.

У такому контексті актуальність ефективних алгоритмів сортування стає критичною. Для ефективної роботи з великими обсягами даних необхідно швидко та ефективно впоратися з їх обробкою. Навіть найпростіші операції, такі як сортування, можуть стати часо- та ресурсо- затратними, якщо використовуються неефективні методи.

Ефективні алгоритми сортування дозволяють швидко опрацьовувати великі обсяги даних, що є критичним для багатьох застосувань. У деяких мережевих пристроях сортування даних може використовуватися для оптимізації обробки пакетів даних та керування мережевим трафіком. Системи керування базами даних (СКБД) використовують алгоритми сортування для виконання запитів, об'єднання даних та інших операцій [1]. У розподілених системах зберігання даних, таких як Hadoop або Apache Spark [2], алгоритми сортування використовуються для обробки великих обсягів інформації та забезпечення швидкодії.

Еволюція алгоритмів сортування є захоплюючим шляхом в історії обчислювальної науки, який почався з простих, але ефективних методів. Наприклад, сортування бульбашкою (Bubble sort) або вставками (Insertion sort) із обчислювальною складністю $O(n^2)$ [3]. У подальшому з'явились більш складні та оптимізовані алгоритми, більш придатні для сортування великих обсягів даних. Такі як швидке сортування (Quick sort) чи сортування злиттям (Merge sort), із обчислювальною складністю у середньому $O(n * log(n))$. У подальшому зростання вимог до стабільності і швидкості сортування привели до появи комбінованих алгоритмів. Найвідомішими представниками яких є Timsort [4] – симбіоз сортувань вставками та злиттям. Інтроспективне сортування (Introsort) [5] починає із швидкого сортування, потім за певних умов переходить на пірамідальне сортування (Heapsort) і викликає сортування вставками для невеликих послідовностей.

Комбіновані алгоритми поєднують у собі переваги складових для підвищення ефективності. У [6] розглядається новітній підхід до формування, перетворення та аналізу конструкцій за допомогою операцій зв'язування, підстановки, виводу тощо.

Для формування структур (складових та їх взаємного розташування) алгоритмів сортування застосований підхід конструктивно-продукційного моделювання. Це забезпечує вирішення наступних задач:

- створення нових алгоритмів сортування із частин існуючих;
- адаптація алгоритмів сортування до сортованих даних;
- адаптація структур даних в оперативній пам'яті.

В даній роботі не розглядається теоретична модель, а лише за цією моделлю її практичне застосування.

Мета

Метою даної роботи є розробка генетичного алгоритму [7] для структурної адаптації алгоритмів сортування. Структурна адаптація алгоритмів сортування полягає у формуванні адаптованого алгоритму з частин відомих алгоритмів таким чином, щоб він був не гіршим за часовими показниками від інших алгоритмів сортування в деякому стабільному середовищі використання.

Особливістю генетичного алгоритму для даної задачі є формування хромосом невизначеної довжини і складу з можливістю як кодування, так і декодування у алгоритм сортування.

Конструювання алгоритмів сортування

Для конструювання алгоритмів сортування будемо використовувати частини існуючих загальновідомих алгоритмів сортування. Використовувались у поточній версії програми такі базові (атомарні) операції з даних алгоритмів:

- швидке сортування (Quick sort). Розділення масиву даних на дві частини відносно обраного елемента, який називається опорним (pivot). Усі елементи, менші за опорний, переставляються ліворуч від нього, а всі більші елементи – праворуч;
- сортування вставками (Insertion sort). Вставка одного елемента із невідсортованої частини масиву у відсортований підмасив;
- сортування вибором (Selection sort). Пошук мінімального або максимального елемента у невідсортованому підмасиві і перестановка;
- шейкерне сортування (Cocktail shaker sort). Прохід масивом у прямому або зворотному напрямку і перестановка усіх пар елементів, що стоять у зворотному порядку;
- сортування злиттям (Merge sort). Злиття двох відсортованих масивів у один.

Також можливою операцією є умовне розбиття масиву на дві частини і сортування кожної окремо. Після того, як кожна частина буде відсортована необхідно виконати злиття у єдиний відсортований масив.

Модель передбачає додавання інших базових операцій. Це можуть бути частини існуючих алгоритмів сортування. Наприклад, вставка елемента із певним кроком, використана у сортуванні Шелла. Також доцільне використання детермінованих і стохастичних передобробок [6, 8], або

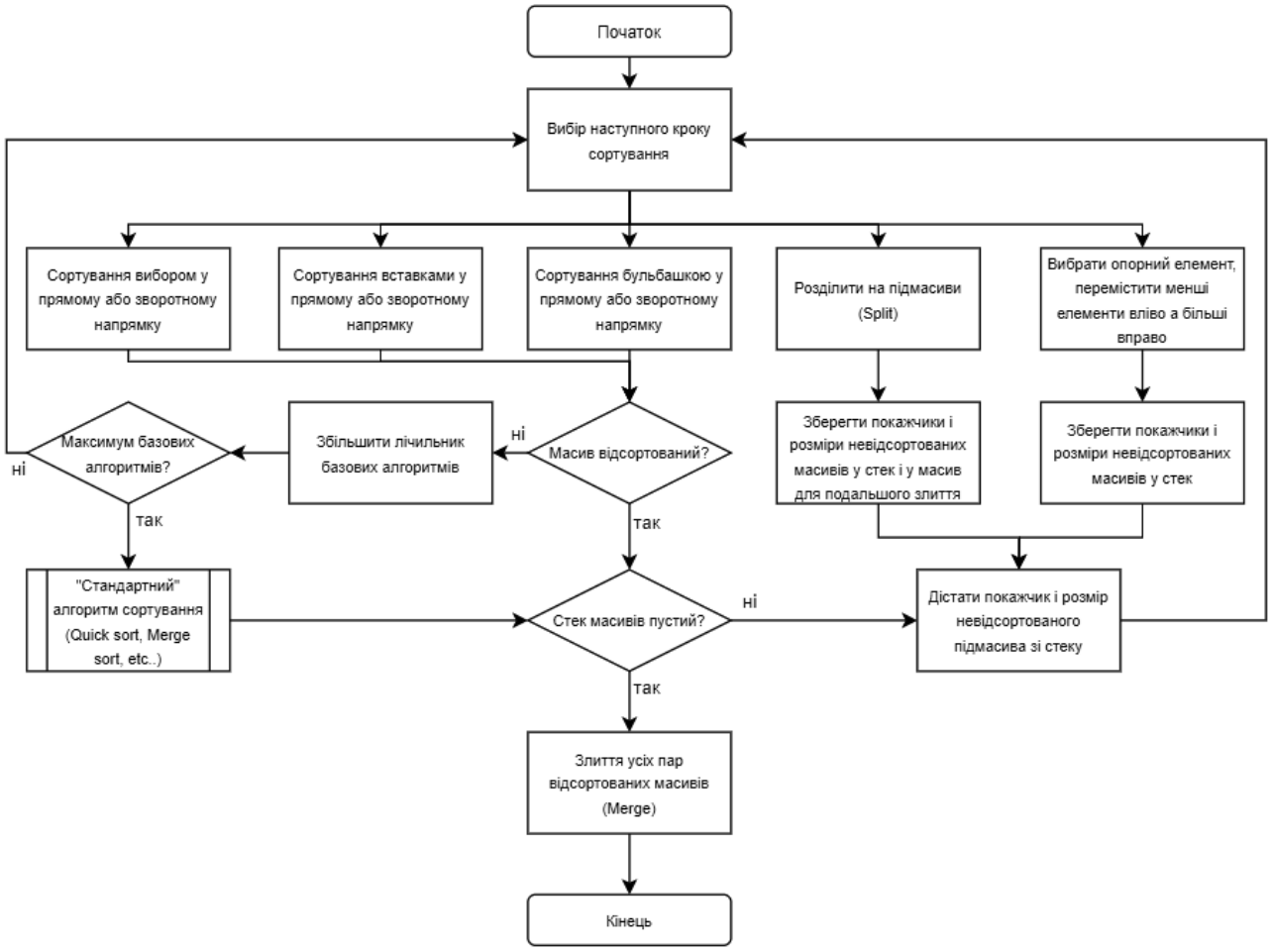


Рис. 1. Блок-схема метаалгоритму генерації програмного коду

їхніх складових операцій. Це може підвищити ефективність кінцевого алгоритму.

Правила формування алгоритму. Для формування алгоритму (рис. 1) складові базові операції вибираються випадково. Таким чином можливі будь-які комбінації атомарних алгоритмів і, як результат, безліч унікальних конструйованих алгоритмів сортування.

Деякі базові алгоритми мають спеціальні вимоги до їхнього використання. У разі, якщо вимоги не виконуються, генерація буде неможливою або конструйований алгоритм не виконуватиме повне сортування масиву. Наприклад, під час використання розбиття масиву на дві частині і сортування кожної частини окремо, має викликатись операція злиття в один відсортований масив.

Розглянемо проблеми з комбінацією алгоритмів сортування. Атомарні частини алгоритмів сортування вибором і бульбашкою розділяють масив на невідсортовану (або ще не проаналізовану) частину та відсортовану. Якщо виконуються проходи у прямому та зворотному напрямку, то масив ділиться на дві відсортовані частини на початку і в кінці масиву та невідсортовану. Через особливості логіки алгоритмів відсортовані частини мають у собі елементи строго менші (на початку) або більші (у кінці масиву), ніж ті, що лишаються у невідсортованій серединній частині. Якщо використовувати проходи тільки в один бік, то отримаємо класичне сортування вибором або бульбашкою. У цьому випадку масив буде повністю відсортований. Якщо використовувати проходи у різних напрямках, то відсортовані частини на початку і в кінці будуть збільшуватись доки не зустрінуться і утворять єдиний відсортований масив.

Для класичного сортування вставками – коли усі вставки виконуються тільки у ліву (або тільки у праву) частину – також результатом роботи буде повністю відсортований масив. Але, якщо виконувати вставки у початок і кінець масиву, то будемо отримувати відповідно два відсортованих масиви. Але немає жодних гарантій, що елементи правого масиву строго більші або рівні елементам з лівого. Для об'єднання двох відсортованих підмасивів у один необхідно викликати операцію злиття (Merge).

Додаткова складність виникає у процесі комбінування алгоритмів, описаних вище. Припустимо, що сортується масив довжиною N. Після виконання M операцій вибору в початок або проходів сортуванням бульбашкою у зворотному напрямку гарантовано матимемо відсортований підмасив [0, M-1] зліва, усі елементи якого строго менші або рівні решті елементів у невідсортованій частині. Якщо тепер виконати вставку елемента під індексом M у ліву частину, то матимемо відсортований масив [0, M], але тепер його елементи не будуть менші, ніж ті, що лишились у невідсортованій частині.

Подальші виклики операцій вибору можуть бути не ефективні через те, що найменший елемент із невідсортованої частини масиву може виявитись меншим, ніж той, що додався сортуванням вставкою. І ліва відсортована частина після додавання цього елемента стає невідсортованою, що означає неефективність створеного алгоритму.

Для сортування бульбашкою одна операція вставки не є критичною. Якщо елемент доданий вставкою під індексом M більший, ніж наступний (M+1), що буде доданий сортуванням бульбашкою, то на першому кроці вони поміняються місцями і підмасив буде відсортований. Однак, якщо була виконана вставка і хоча б одна операція вибору, то сортування бульбашкою втрачає ефективність, і масив може лишитись у невідсортованому стані (рис. 2).

2	7	4	3	1	0	Бульбашкою
0	**2**	7	4	3	1	Вибором
0	1	**7**	4	3	2	Вставкою
0	1	7	**4**	3	2	Вибором
0	1	7	2	**3**	4	**Ліва частина невідсортована**

Рис. 2. Приклад невірної послідовності базових алгоритмів

Для вирішення вищезгаданої проблеми введемо нові атомарні операції – злиття зліва (ML) і злиття справа (MR). Для відстежування викликів операцій вставки будуть використовуватись додаткові індикатори. Якщо було використано операцію вставки у початок, відповідний індикатор буде встановлено. Перед наступним викликом операцій сортування бульбашкою або вибором для поточного масиву, потрібно буде спочатку викликати операцію злиття зліва. Тобто зберегти покажчики відсортованої частини зліва і решти масиву для подальшого злиття. Для аналогічної ситуації у кінці масиву буде викликатись операція злиття справа.

Застосування генетичного алгоритму для вибору найбільш ефективного алгоритму сортування

Хромосому представимо послідовністю генів – базових алгоритмів сортування та допоміжних утиліт. Для представлення хромосоми у текстовому вигляді задаємо текстове представлення кожному гену. Таким чином хромосома буде мати послідовність генів, розділених символом - «,». Через те, що кожен ген являє собою якусь функцію, то використаємо для текстового представлення її абревіатуру. Розглянемо існуючі гени:

- BSB (Bubble sort backward) – один прохід масивом із кінця у початок із перестановкою пар елементів, що йдуть у зворотному порядку;
- BSF (Bubble sort forward) – те саме, що і BSF, тільки прохід від початку у кінець масиву;
- FSB (Find swap biggest element) – пошук найбільшого елемента у невідсортованій частині масиву і перестановка на поточне місце;
- FSS (Find swap smallest element) – те саме що і FSB, тільки виконується пошук найменшого елемента;
- SEEI (Single element end insertion) – вставка поточного елемента у відсортовану частину в кінці масиву;
- SEI (Single element insertion) – вставка поточного елемента у відсортовану частину на початку масиву;
- SIS (Split in subarrays) – розділення невідсортованої частини масиву на дві рівні частини для подальшого сортування кожної з них окремо. Також зберігання покажчиків та розмірів масивів для подальшого виконання операції злиття;
- FS (Final sort) – один із загальновідомих алгоритмів сортування, який гарантовано виконає сортування;
- PUA (Pop unsorted array) – дістати покажчик і розмір наступної невідсортованої частини для сортування. Виконується для кожної частини збереженої операцією SIS;
- ESS (End subarray sort) – допоміжна операція, яка закриває фігурні дужки, відкриті попередніми операціями для перевірки індикатора відсортованості. Виконується для кожної операції PUA після послідовності генів сортування;
- ML (Merge left) – збереження покажчиків на відсортовану частину на початку та решту масиву для подальшого злиття в один відсортований масив;
- MR (Merge right) – те ж саме, що і ML тільки для відсортованої частини в кінці масиву;

Приклад сформованої хромосоми у текстовому вигляді:
SEEI, MR, FSB, SIS, PUA, BSF, FSS, FS, ESS, PUA, SEI, ML, FSS, FS, ESS, ESS.

Для обмеження довжини хромосоми, введемо певні обмеження.

На початку сортування вхідного масиву глибина дорівнює одному. У процесі розбиття масиву і переходу до сортування будь-якої із його частин, глибина збільшується на одиницю. Таким способом обмежується кількість розбивань масиву на частини.

Окремим параметром обмежується кількість базових операцій сортування для кожного підмасиву. При досягненні максимального значення, викликається розбиття або фінальне сортування.

Генетичний алгоритм для генерації алгоритмів сортування. Кожен індивідуум представляє собою алгоритм сорту-

вання. Генами представлені атомарні операції-складові існуючих алгоритмів сортування та допоміжні утиліти.

Функцією придатності, яка оцінює якість кожного індивідуума, буде часова ефективність. Але можуть бути і такі: кількість порівнянь, кількість перестановок тощо. Або зважена комбінація таких чинників.

Генерація наступної популяції здійснюється за допомогою схрещування і мутацій. Схрещування здійснює обмін частинами між двома батьками для створення нового індивідуума.

Мутація випадковим чином змінює деякі елементи генетичної послідовності. Заново генеруються деякі ділянки хромосоми і випадково вибираються атомарні операції.

Для наступного покоління обираються індивідууми на основі їхньої придатності. Індивіди з більшою придатністю мають більше шансів вижити і брати участь у створенні нових індивідів.

Певна кількість індивідів з найкращими показниками переноситься в наступне покоління без змін. Інші формуються схрещуванням індивідів існуючої популяції. Індивіди для схрещення можуть вибиратись випадково, або за певними правилами. Також можливий варіант із схрещенням кращих індивідів за правилами, а решти – випадково. Для урізноманітнення популяції додається певна кількість індивідів, згенерованих випадковим чином, так само, як була створена найперша популяція.

Встановлюються параметри генетичного алгоритму, такі як розмір популяції, ймовірність мутації, кількість поколінь тощо. Визначається умова зупинки, наприклад, максимальна кількість поколінь або досягнення певного рівня придатності.

Проводиться декілька ітерацій генетичного пошуку, оптимізуються параметри і функція придатності для покращення результатів.

Застосування генетичного алгоритму до генерації алгоритмів сортування дозволяє автоматично еволюціонувати рішення, призначені для різних сценаріїв, та адаптувати їх до умов, що постійно змінюються.

Представлення хромосоми у вигляді дерева

Вхідний масив може мати будь-який відсоток відсортованості. Як тільки дані у масиві будуть відсортовані, доцільно завершити виконання для уникнення погіршення часової ефективності. Відстежування такої ситуації реалізуємо використанням ознаки відсортованості масиву, яка буде перевірятись після кожної базової операції сортування.

Постійне оновлення і перевірки ознаки відсортованості масиву вводить додаткові правила та обмеження у процес генерації хромосоми. У коді кожна наступна перевірка додає новий рівень вкладеності і область видимості, обмежену фігурними дужками “{“ та “}”. Перевірка ознаки відсортованості і відкриття області видимості має у собі кожний ген, що представляє базову операцію сортування. Відповідно кінцева ділянка хромосоми повинна мати фігурні дужки, що закриваються у кількості рівній відкритим дужкам. При розбитті масиву на підмасиви і сортуванні кожного окремо, матимемо послідовність базових операцій сортування і відповідну кількість закритих дужок. Оптимальною структурою даних для представлення вище описаного підходу є бінарне дерево (рис. 3).

Кожен вузол представляє собою частину хромосоми (послідовність генів) що відповідає сортуванню певної частини масиву. Окремо зберігаються гени початку і кінця. Вузли нащадків створюються під час розбиття масиву і сортування частин масиву окремо. Наприклад, під час розбиття масиву на дві рівні частини створюються два вузла нащадків. Кожний створений вузол матиме частину хромосоми, що реалізує сортування відповідної частини масиву. Масив генів кінця включатиме у собі злиття відсортованих масивів і ген завершення сортування поточного масиву, що включає закриття фігурних дужок і обнуління змінних.

Генерація алгоритму із хромосоми повинна відбуватись за принципом обходу

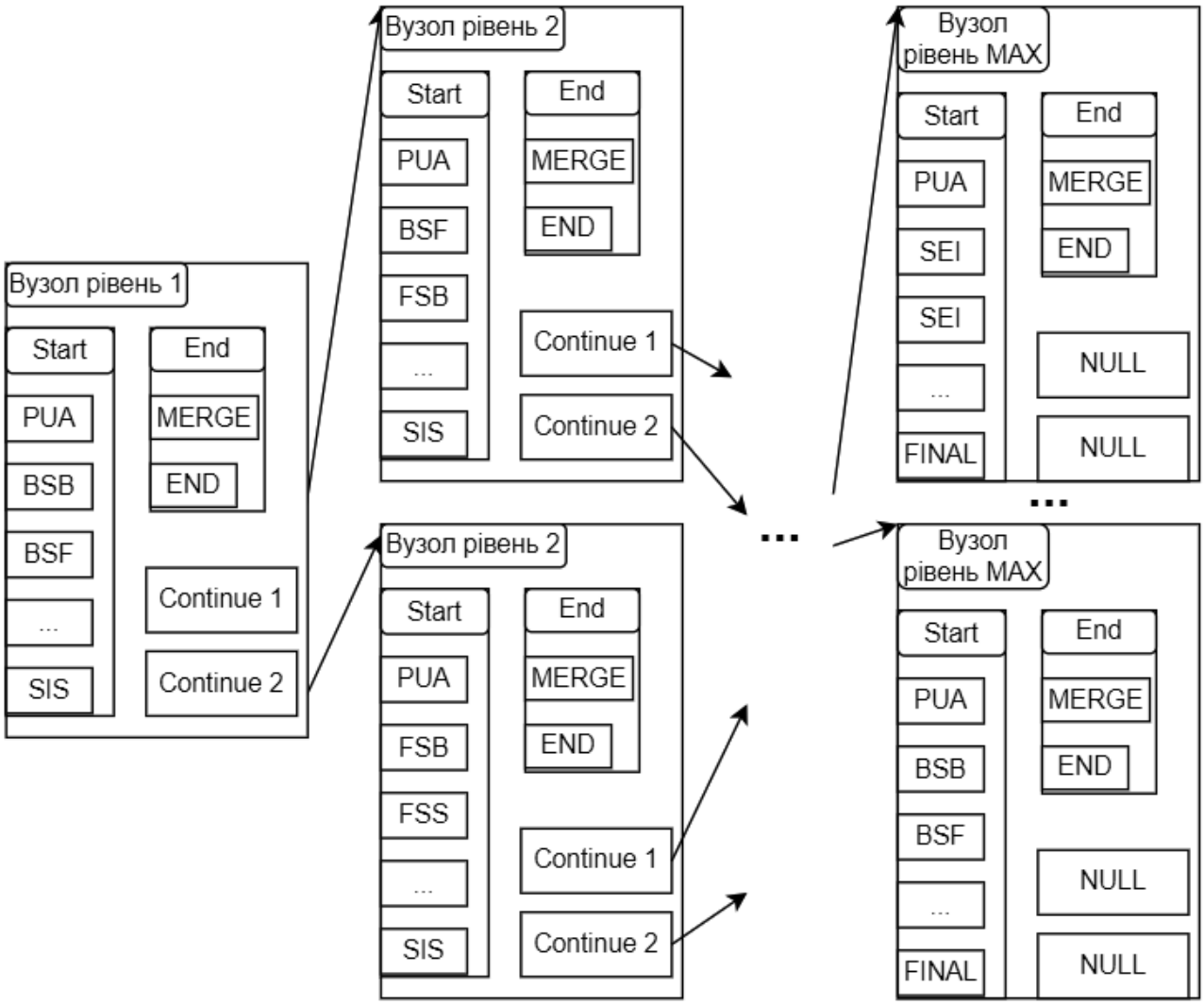


Рис. 3. Схема хромосоми у формі дерева

дерева вглиб [9]. Тобто для кожного вузла гени ідуть у наступній послідовності:

- гени початку;
- гени першого дитячого вузла;
- гени другого дитячого вузла;
- гени кінця.

Операція схрещення деревовидних хромосом виконується як обмін вузлами. Для збереження глибини дерева-хромосоми обмін може бути проведений тільки між вузлами відповідного рівня. Однак через те, що вершини на останньому рівні мають ген фінального сортування, а це гарантовано відсортує масив, обмін можливо проводити між вузлами різних рівнів.

Висновки

Розроблений генетичний алгоритм для структурної адаптації алгоритмів сортування. Його застосування дозволяє здійснити структурну адаптацію і вибрати найбільш ефективні та пристосовані до стабільних середовищ використання.

Представлений підхід до формування хромосоми у вигляді дерева дозволяє вирішити наявні проблеми. Він дозволяє ефективно формувати хромосоми, результатом декодування яких буде алгоритм, який гарантовано виконує сортування.

Формалізована модель формування алгоритмів сортування засобами конструктивно-продукційного моделювання є до-

сить об'ємною та буде представлена окремою статтею.

Розроблене програмне забезпечення має застосовуватись у процесі адаптації алгоритмів сортування до стабільних потоків вхідних даних та середовищ використання.

Література

1. А.Ю. Берко, О.М. Верес, В.В. Пасічник, Системи баз даних та знань. Книга 2, Магнолія-2006, 2013.
2. O. Mendelevitch, C. Stella, D. Eadline, Practical Data Science with Hadoop and Spark, Addison-Wesley, 2016.
3. D. Knuth, The Art Of Computer Programming, vol. 3: Sorting And Searching, Addison-Wesley, 1973.
4. Timsort. Accessed: 08.07.2023. https://svn.python.org/projects/python/trunk/Objects/listsort.txt.
5. D.R. Musser, Introspective Sorting and Selection Algorithms, *Software: Practice and Experience*, 1997, № 27 (8), pp.983–993. doi: 10.5555/261387.261395.
6. В.І. Шинкаренко, А.Ю. Дорошенко, О.А. Яценко, В.В. Разносілін, К.К. Галанін, Двокомпонентні алгоритми сортування, *Проблеми програмування,* 2022, № 3-4. С. 32-41. doi: 10.15407/pp2022.03-04.032.
7. J.H. Holland, Adaptation in Natural and Artificial Systems, The MIT Press, 1992.
8. В.І. Шинкаренко, О.В. Макаров, Дослідження ефективності деяких детермінованих методів передобробки сортування даних, *Проблеми програмування,* 2023, № 4, С. 3-14. doi: 10.15407/pp2023.04.003.
9. R. Tarjan, Depth-First Search and Linear Graph Algorithms, *SIAM Journal on Computing*, 1972, № 1(2). doi: 10.1137/0201010.

References

1. A.Yu. Berko, O.M. Veres, V.V. Pasichnik, Database and knowledge systems. Book 2, Magnolia-2006, 2013.
2. O. Mendelevitch, C. Stella, D. Eadline, Practical Data Science with Hadoop and Spark, Addison-Wesley, 2016.
3. D. Knuth, The Art Of Computer Programming, vol. 3: Sorting And Searching, Addison-Wesley, 1973.
4. Timsort. Accessed: 08.07.2023. https://svn.python.org/projects/python/trunk/Objects/listsort.txt.
5. D.R. Musser, Introspective Sorting and Selection Algorithms, in: Software: Practice and Experience, 1997, 27 (8), pp.983–993. doi: 10.5555/261387.261395.
6. V.I. Shinkarenko, A.Yu. Doroshenko, O.A. Yatsenko, V.V. Raznosilin, K.K. Galanin, Bicomponent sorting algorithms, in: Problems of programming, 2022, № 3-4, pp.32-41. doi: 10.15407/pp2022.03-04.032. [in Ukrainian]
7. J.H. Holland, Adaptation in Natural and Artificial Systems, The MIT Press, 1992.
8. V.I. Shinkarenko, O.V. Makarov, Study of the effectiveness of some deterministic preprocessing methods of data sorting, in: Problems of programming, 2023, № 4, pp.3-14. doi: 10.15407/pp2023.04.003. [in Ukrainian]
9. R. Tarjan, Depth-First Search and Linear Graph Algorithms, in: SIAM Journal on Computing, 1972, № 1(2). doi: 10.1137/0201010.

Одержано: 14.04.2024
Внутрішня рецензія отримана: 26.04.2024
Зовнішня рецензія отримана: 29.04.2024

Про авторів:

Шинкаренко Віктор Іванович,
доктор технічних наук, професор,
https://orcid.org/0000-0001-8738-7225

Макаров Олексій Вікторович,
аспірант,
https://orcid.org/0009-0003-0921-155X

Місце роботи авторів:

Український державний університет
науки і технологій,
49010, Україна, Дніпро,
вул. академіка Лазаряна, 2.
E-mail: office@ust.edu.ua
https://ust.edu.ua/

УДК 510.64 http://doi.org/10.15407/pp2024.02-03.019

О.С. Шкільняк, С.С. Шкільняк

МОДАЛЬНІ ЛОГІКИ ЧАСТКОВИХ КВАЗІАРНИХ ПРЕДИКАТІВ З РІВНІСТЮ ТА СЕКВЕНЦІЙНІ ЧИСЛЕННЯ ЦИХ ЛОГІК

Робота присвячена дослідженню нових класів програмно-орієнтованих логічних формалізмів модального типу – чистих першопорядкових модальних логік часткових квазіарних предикатів зі знятою умовою монотонності та збагачених предикатами рівності. Апарат модальних логік використовується для опису й моделювання різноманітних предметних областей, систем штучного інтелекту, інформаційних та програмних систем. Обмеження класичної логіки предикатів, на якій базуються традиційні модальні логіки, зумовлюють актуальність проблеми побудови нових, програмно-орієнтованих логічних формалізмів. Такими є композиційно-номінативні модальні логіки, які синтезують можливості традиційних модальних логік і логік часткових квазіарних предикатів. Важливим їх класом є транзиційні модальні логіки (ТМЛ), які відбивають аспект зміни й розвитку предметних областей. Чисті першопорядкові ТМЛ названо TML^{Q}. В роботі запропоновано два різновиди TML^{Q} з рівністю: з предикатами строгої рівності $\equiv_{xy}$, їх названо $\mathrm{TML}^{Q\equiv}$, та слабкої рівності $=_{xy}$, їх названо $\mathrm{TML}^{Q=}$. Для елімінації кванторів в логіках немонотонних предикатів потрібні спеціальні предикати-індикатори наявності у вхідних даних компоненти з відповідним предметним іменем. Використання цих предикатів є характерною особливістю $\mathrm{TML}^{Q\equiv}$ та $\mathrm{TML}^{Q=}$. Тотальні предикати-індикатори визначають наявність чи відсутність компоненти з певним іменем, а часткові предикати-індикатори визначають лише наявність такої компоненти. Тотальні предикати-індикатори *Ez* фігурують в $\mathrm{TML}^{Q\equiv}$ як спеціальні 0-арні композиції, часткові предикати-індикатори вже присутні в $\mathrm{TML}^{Q=}$ як предикати $=_{xx}$. Ще одна особливість $\mathrm{TML}^{Q\equiv}$ та $\mathrm{TML}^{Q=}$ – використання композицій розширеної реномінації. В роботі описано семантичні моделі та мови $\mathrm{TML}^{Q\equiv}$ та $\mathrm{TML}^{Q=}$. Увагу акцентовано на властивостях, пов'язаних з предикатами рівності, описано особливості заміни рівних в $\mathrm{TML}^{Q\equiv}$ та в $\mathrm{TML}^{Q=}$. Для цих логік визначено низку відношень логічного наслідку для множин специфікованих станами формул. На цій семантичній основі для досліджених логік запропоновано відповідні числення секвенційного типу.
Ключові слова: модальна логіка, предикат, логічний наслідок, секвенційне числення

O.S. Shkilniak, S.S. Shkilniak

MODAL LOGICS OF PARTIAL QUASIARY PREDICATES WITH EQUALITY AND SEQUENT CALCULI OF THESE LOGICS

The aim of the work is to study new classes of program-oriented logical formalisms of the modal type – pure first-order modal logics of partial quasiary predicates without monotonicity condition and enriched with equality predicates. Modal logics can be used to describe and model various subject areas, artificial intelligence systems, information and software systems. The limitations of the classical predicate logic on which traditional modal logics are based determine the relevance of the problem of introducing new program-oriented logical formalisms. Such are composition-nominative modal logics, which synthesize facilities of traditional modal logics and logics of partial quasiary predicates. One of their important classes are transitional modal logics (TML), they reflect the aspect of change and evolution of subject areas. We denote pure first-order TML by TML^{Q}. In this paper two types of TML^{Q} with equality are considered: $\mathrm{TML}^{Q\equiv}$ (with strong equality predicates $\equiv_{xy}$), and $\mathrm{TML}^{Q=}$ (with weak equality predicates $=_{xy}$). For quantifier elimination in logics of non-monotonic predicates special predicates which indicate whether a component with a corresponding name has a value in the input data are required. The use of these predicates is a characteristic feature of both $\mathrm{TML}^{Q\equiv}$ and $\mathrm{TML}^{Q=}$. Total indicator predicates determine the presence or absence of a component with a certain name, while partial indicator predicates signalize only the presence of such a component. Thus, total indicator predicates are introduced as special parametric 0-ary compositions *Ez* in $\mathrm{TML}^{Q\equiv}$, and partial indicator predicates are represented in $\mathrm{TML}^{Q=}$ as predicates $=_{xx}$. Another feature of $\mathrm{TML}^{Q\equiv}$ and $\mathrm{TML}^{Q=}$ is the use of the extended renomination compositions. In this paper we describe semantic models and languages of $\mathrm{TML}^{Q\equiv}$ and $\mathrm{TML}^{Q=}$. Particular attention is paid to the properties related to equality predicates, substitution of equals in $\mathrm{TML}^{Q\equiv}$ and $\mathrm{TML}^{Q=}$ is described. A number of logical consequence relations for these logics are defined on sets of formulas specified with states. On this semantic basis, the corresponding sequent type calculi are proposed for the investigated logics.
Key words: modal logic, predicate, logical consequence, sequent calculus

Вступ

Апарат модальних логік успішно використовується для опису й моделювання різноманітних предметних областей, систем штучного інтелекту, інформаційних та програмних систем (див., напр., [1, 2]). Темпоральні логіки застосовуються для моделювання динамічних систем, специфікації та верифікації програм [2–4], на базі цих логік розроблено низку систем і мов специфікацій. Епістемічні логіки з великим успіхом використовуються для опису систем штучного інтелекту, інформаційних та експертних систем. Водночас обмеження класичної логіки предикатів, яка лежить в основі традиційних модальних логік, роблять вельми актуальною задачу побудови нових, програмно-орієнтованих логічних формалізмів модального типу. Такими є композиційно-номінативні модальні логіки (КНМЛ), які поєднують можливості традиційних модальних логік [3] і композиційно-номінативних логік часткових квазіарних предикатів [5–8]. Найважливішим класом КНМЛ є транзиційні модальні логіки (ТМЛ), які відбивають аспект зміни й розвитку предметних областей. Такі логіки вивчались, зокрема, в [9, 10]. Традиційні модальні логіки природним чином розглядаються в межах ТМЛ.

Метою пропонованої роботи є дослідження нових класів програмно-орієнтованих модальних логік – чистих першопорядкових ТМЛ часткових квазіарних предикатів зі знятою умовою монотонності (еквітонності) та збагачені предикатами рівності. Виділено [8] два різновиди таких предикатів: слабкої рівності $=_{xy}$ та строгої рівності $\equiv_{xy}$. Чисті першопорядкові ТМЛ зі знятою умовою монотонності назвемо $\mathrm{TML}^{\mathrm{Q}}$. $\mathrm{TML}^{\mathrm{Q}}$ з предикатами строгої рівності назвемо $\mathrm{TML}^{\mathrm{Q}\equiv}$, а $\mathrm{TML}^{\mathrm{Q}}$ з предикатами слабкої рівності назвемо $\mathrm{TML}^{\mathrm{Q}=}$. $\mathrm{TML}^{\mathrm{Q}}$ без предикатів рівності досліджено в [9, 10]. $\mathrm{TML}^{\mathrm{Q}\equiv}$ та $\mathrm{TML}^{\mathrm{Q}=}$ вивчаються в цій роботі.

Для елімінації кванторів в логіках немонотонних предикатів потрібні спеціальні предикати-індикатори наявності у вхідних даних компоненти з відповідним предметним іменем. Використання предикатів-індикаторів є характерною особливістю $\mathrm{TML}^{\mathrm{Q}}$. Тотальні предикати-індикатори визначають наявність чи відсутність компоненти з певним іменем, а часткові предикати-індикатори визначають лише наявність компоненти з певним іменем. Тотальні предикати-індикатори *Ez* фігурують, зокрема, в [5–7, 9]. Предикати *Ez* можна виразити як $\exists y \equiv_{xy}$, проте доцільніше задати їх явно як спеціальні 0-арні композиції, як зроблено в $\mathrm{TML}^{\mathrm{Q}\equiv}$ Водночас часткові предикати-індикатори вже присутні в $\mathrm{TML}^{\mathrm{Q}=}$, такими є предикати $=_{xx}$.

В роботі описано семантичні моделі та мови $\mathrm{TML}^{\mathrm{Q}}$, описано особливості заміни рівних в $\mathrm{TML}^{\mathrm{Q}\equiv}$ та $\mathrm{TML}^{\mathrm{Q}=}$, визначено низку відношень логічного наслідку для множин специфікованих станами формул та наведено їх властивості, На цій семантичній основі для $\mathrm{TML}^{\mathrm{Q}}$ з рівністю запропоновано низку числень секвенційного типу.

Поняття, які в цій роботі не визначаються, тлумачитимо в сенсі [5–9].

1. Транзиційні модальні системи

Центральним для КНМЛ є поняття композиційно-номінативної модальної системи (КНМС). Такі системи є моделями світів розгляду модальних логік. КНМС – це об'єкт вигляду $\boldsymbol{M} = (Cms, Ds, Im)$. Тут:

– *Cms* – композиційна модальна система, задає семантичні аспекти світу;

– *Ds* – дескриптивна система, вона визначає стандартні дескрипції; зазвичай це множина *Fm* формул мови КНМЛ;

– *Dns* – денотаційна система, вона визначає значення стандартних дескрипцій на семантичних моделях; зазвичай для цього використовується відображення *Im* інтерпретації формул на станах світу.

Композиційна модальна система – це об'єкт вигляду $Cms = (St, \boldsymbol{R}, Pr, C)$, де:

– *St* – множина станів світу;

– $\boldsymbol{R}$ – множина відношень на *St* вигляду $\mathrm{R} \subseteq St \times St^n$;

– *Pr* – множина предикатів на *St*;

– *C* – множина композицій на *Pr*.

КНМС далі будемо трактувати як об'єкти вигляду $\boldsymbol{M} = ((St, \boldsymbol{R}, Pr, C), Fm, Im)$.

Для чистих першопорядкових КНМС множину St конкретизуємо як множину алгебраїчних систем (структур) вигляду $\alpha = (A_\alpha, Pr_\alpha)$, де A_α – множина базових даних стану α, Pr_α – множина квазіарних предикатів ${}^V A_\alpha \to \{T, F\}$, названих *предикатами стану* α.

Предикати вигляду ${}^V A \to \{T, F\}$, де $A = \bigcup_{\alpha \in S} A_\alpha$, назвемо *глобальними*.

Важливим класом КНМЛ є транзиційні модальні логіки (ТМЛ), вони відбивають аспект зміни й розвитку предметних областей, описуючи переходи від одного стану світу до іншого. В основі ТМЛ лежить поняття транзиційної модальної системи (ТМС), такі системи є найважливішим класом КНМС. Чисті першопорядкові ТМС назвемо $\mathrm{TMS}^{\mathrm{Q}}$.

ТМС – це КНМС, у яких множина $\boldsymbol{R}$ складається з відношень вигляду $\mathrm{R} \subseteq St \times St$. Ці відношення трактуємо як відношення переходу на станах.

Окремими випадками ТМС є загальні транзиційні, темпоральні, мультимодальні системи (див. [9, 10]).

ТМС, в яких $\boldsymbol{R}$ складається з єдиного бінарного відношення $\triangleright$, а базовою модальною композицією є $\Box$ ("необхідно"), названо *загальними* (GMS).

ТМС, в яких $\boldsymbol{R}$ складається з єдиного бінарного відношення $\triangleright$, а базовими модальними композиціями є $\Box\uparrow$ ("завжди буде") і $\Box\downarrow$ ("завжди було"), названо *темпоральними* (TmMS).

ТМС із множиною відношень $\boldsymbol{R} = \{\triangleright_i \mid i \in I\}$ і базовими модальними композиціями M_i, $i \in I$, в яких кожному $\triangleright_i \in \boldsymbol{R}$ зіставлено відповідну композицію M_i, названо *мультимодальними* (MMS). В MMS дія кожної M_i аналогічна дії $\Box$, але тільки щодо свого відношення $\triangleright_i$, $i \in I$.

Для GMS традиційно задають похідну композицію $\Diamond$ ("можливо"): $\Diamond P$ означає $\neg\Box\neg P$. Для TmMS задають похідні композиції $\Diamond\uparrow$ ("колись буде") та $\Diamond\downarrow$ ("колись було"): $\Diamond\uparrow P$ означає $\neg\Box\uparrow\neg P$, а $\Diamond\downarrow P$ означає $\neg\Box\downarrow\neg P$.

Базовими загальнологічними композиціями для $\mathrm{TMS}^{\mathrm{Q}}$ вважаємо логічні зв'язки $\neg$ та $\vee$, композиції реномінації $\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}$ та квантифікації $\exists x$. Для $\mathrm{TMS}^{\mathrm{Q}}$ з рівністю до них додаємо спеціальні 0-арні композиції – предикати рівності. $\mathrm{TMS}^{\mathrm{Q}}$ з предикатами строгої рівності $\equiv_{xy}$ назвемо $\mathrm{TMS}^{\mathrm{Q}\equiv}$, а $\mathrm{TMS}^{\mathrm{Q}}$ з предикатами слабкої строгої рівності $=_{xy}$ назвемо $\mathrm{TMS}^{\mathrm{Q}=}$.

Нагадаємо, що V-A-квазіарним предикатом називаємо [5] часткову функцію вигляду $Q : {}^V A \to \{T, F\}$. Тут ${}^V A$ – множина всіх V-A-іменних множин $\{T, F\}$ – множина істиннісних значень. V і A трактуємо як множини предметних імен (змінних) і предметних значень.

V-A-іменну множину (V-A-ІМ) визначають [5] як однозначну функцію вигляду $V \to A$. Подаємо V-A-ІМ у вигляді $[v_i \mapsto a_i]_{i \in I}$, де $v_i \in V$, $a_i \in A$, $v_i \neq v_j$ при $i \neq j$.

Для V-A-ІМ вводимо операції $\|_{-Z}$, де $Z \subseteq V$, накладення ∇, реномінації та розширеної реномінації (див. [5, 7, 8]).

Кожний V-A-квазіарний предикат Q задаємо двома множинами: область *істинності* $T(Q) = \{d \mid Q(d) = T\}$ та область *хибності* $F(Q) = \{d \mid Q(d) = F\}$.

Предикат Q *однозначний*, або *P-предикат*, якщо $T(Q) \cap F(Q) = \varnothing$;

Q *неспростовний*, якщо $F(Q) = \varnothing$;

Q *виконуваний*, якщо $T(Q) \neq \varnothing$.

Далі розглядаємо саме однозначні V-A-квазіарні предикати.

В класі P-предикатів маємо 3 константних:

– Q *тотожно істинний* (позн. T), якщо $F(Q) = \varnothing$ та $T(Q) = {}^V A$;

– Q *тотожно хибний* (позн F), якщо $T(Q) = \varnothing$ та $F(Q) = {}^V A$;

– Q *тотально невизначений* (позн. $\perp$, якщо $T(Q) = F(Q) = \varnothing$.

P-предикат Q *еквітонний*, якщо $(Q(d)\downarrow$ та $d \subseteq d') \Rightarrow Q(d')\downarrow = Q(d)$.

$x \in V$ *неістотне* для предиката Q, якщо $d_1 \|_{-x} = d_2 \|_{-x} \Rightarrow Q(d_1) = Q(d_2)$.

Визначення базових загальнологічних композицій $\neg$, $\vee$, $\exists x$, $\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}$ квазіарних предикатів наведено в [7]

Предикати рівності трактуємо як спеціальні 0-арні композиції, беручи до уваги їх загальнологічний статус. Виділено

[6] два різновиди цих предикатів: слабкої (з точністю до визначеності) рівності $=_{\{x,y\}}$ та строгої (точної) рівності $\equiv_{\{x,y\}}$.

Задаємо $=_{\{x,y\}}$ та $\equiv_{\{x,y\}}$ так:

$T(=_{\{x,y\}}) = \{d \mid d(x)\downarrow, d(y)\downarrow \text{ та } d(x) = d(y)\}$,

$F(=_{\{x,y\}}) = \{d \mid d(x)\downarrow, d(y)\downarrow \text{ та } d(x) \neq d(y)\}$;

$T(\equiv_{\{x,y\}}) = \{d \mid d(x)\downarrow, d(y)\downarrow \text{ та } d(x) = d(y)\} \cup$
$\cup \{d \mid d(x)\uparrow \text{ та } d(y)\uparrow\}$,

$F(\equiv_{\{x,y\}}) = \{d \mid d(x)\downarrow, d(y)\downarrow, d(x) \neq d(y)\} \cup$
$\cup \{d \mid d(x)\downarrow, d(y)\uparrow \text{ або } d(x)\uparrow, d(y)\downarrow\}$.

Окремим випадком $=_{\{x,y\}}$ та $\equiv_{\{x,y\}}$, якщо x та y збігаються, є $=_{\{x\}}$ та $\equiv_{\{x\}}$.

Предикати $=_{\{x,y\}}$, $=_{\{x\}}$ та $\equiv_{\{x,y\}}$, $\equiv_{\{x\}}$ більш звично позначаємо $=_{xy}$, $=_{xx}$ та $\equiv_{xy}$, $\equiv_{xx}$. Отже, $=_{xy}$ та $=_{yx}$ – це один і той же предикат, $\equiv_{xy}$ та $\equiv_{yx}$ – теж один і той же предикат.

Предикати $\equiv_{xy}$ та $\equiv_{xx}$ тотальні немонотонні; $=_{xy}$ та $=_{xx}$ часткові еквітонні.

Спеціальні 0-арні композиції – предикати-індикатори – визначають наявність у вхідних даних компоненти зі вказаним іменем. Тотальні предикати-індикатори встановлюють *наявність* чи *відсутність* такої компоненти, часткові предикати-індикатори встановлюють лише *наявність* цієї компоненти.

Тотальні предикати-індикатори Ez немонотонні, їх задаємо (див. [7]) так:

$T(Ez) = \{d \mid d(z)\downarrow\}$;

$F(Ez) = \{d \mid d(z)\uparrow\}$.

Частковими предикатами-індикаторами в $\text{TMS}^{Q=}$ є еквітонні предикати $=_{zz}$. Справді, маємо $T(=_{zz}) = \{d \mid d(z)\downarrow\} = T(Ez)$ та $F(=_{zz}) = \{d \mid d(z)\downarrow \text{ та } d(z) \neq d(z)\} = \varnothing$.

Таким чином, маємо такі різновиди TMS^{Q} з рівністю: $\text{GMS}^{Q\equiv}$, $\text{TmMS}^{Q\equiv}$, $\text{MMS}^{Q\equiv}$ для $\text{TMS}^{Q\equiv}$ та $\text{GMS}^{Q=}$, $\text{TmMS}^{Q=}$, $\text{MMS}^{Q=}$ для $\text{TMS}^{Q=}$.

Опишемо мову $\text{GMS}^{Q\equiv}$. Алфавіт мови: множина V предметних імен (змінних); множина Ps предикатних символів; множина $\{\neg, \vee, R^{\bar{v},\bar{u}}_{\bar{x},\perp}, \exists x, \equiv_{xy}, Ex\}$ символів базових загальнологічних композицій; множина $Ms = \{\Box\}$ символів базових модальних композицій. Множина Fr формул мови визначається так:

Fa) $Ps \subseteq Fr$;

F≡) $\{Ex \mid x \in V\} \subseteq Fr$ та $\{\equiv_{xy} \mid x, y \in V\} \subseteq Fr$;

Fp) $\Phi, \Psi \in Fr \Rightarrow \neg\Phi \in Fr$ та $\vee\Phi\Psi \in Fr$;

FR) $\Phi \in Fr \Rightarrow R^{\bar{v},\bar{u}}_{\bar{x},\perp}\Phi \in Fr$;

F∃) $\Phi \in Fr \Rightarrow \exists x\Phi \in Fr$;

F□) $\Phi \in Fr \Rightarrow \Box\Phi \in Fr$.

Формули вигляду $p \in Ps$, Ex, $\equiv_{xy}$ назвемо атомарними.

Множини гарантовано неістотних для формул імен задаємо функцією $\nu : Fr \to 2^V$ (див. [7, 9]).

Тип $\text{GMS}^{Q\equiv}$ визначається розширеною сигнатурою $\sigma = (Ps, \nu)$ та властивостями відношення $\triangleright$.

Задамо відображення інтерпретації Im формул на станах світу. Спочатку задаємо $Im : Ps \times St \to Pr$, водночас має бути $Im(p, \alpha) \in Pr\alpha$ (базові предикати є предикатами станів). Символи композицій (зокрема, символи Ex та $\equiv_{xy}$) інтерпретуються як відповідні композиції (зокрема, відповідні предикати-індикатори та предикати рівності). Продовжимо таке Im до відображення $Fm \times St \to Pr$ наступним чином:

Ip) $Im(\neg, \alpha) = \neg(Im(\Phi, \alpha))$;

$Im(\vee\Phi\Psi, \alpha) =$
$\vee(Im(\Phi, \alpha), Im(\Psi, \alpha))$;

IR) $Im(R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Phi), \alpha) = \mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}(Im(\Phi, \alpha))$;

I∃) $Im(\exists x\Phi, \alpha)(d) =$

$$= \begin{cases} T, \text{ якщо існує } a \in A_\alpha : Im(\Phi, \alpha)(d\nabla x \mapsto a) = T, \\ F, \text{ якщо } Im(\Phi, \alpha)(d\nabla x \mapsto a) = F \text{ для всіх } a \in A_\alpha, \\ \text{невизначене в усіх інших випадках.} \end{cases}$$

I□) $Im(\Box\Phi, \alpha)(d) =$

$$= \begin{cases} T, \text{ якщо } Im(\Phi, \delta)(d) = T \text{ для всіх } \delta \in S : \alpha \triangleright \delta, \\ F, \text{ якщо існує } \delta \in S : \alpha \triangleright \delta \text{ та } Im(\Phi, \delta)(d) = F, \\ \text{невизначене в усіх інших випадках.} \end{cases}$$

Якщо для $\alpha \in S$ не існує β: $\alpha \triangleright \beta$, то $Im(\Box\Phi, \alpha)(d)\uparrow$ для кожного $d \in {}^V A$.

Предикати, які є значеннями немодалізованих формул (при їх побудові не використовуються символи MS), належать до предикатів станів.

Предикати, які є значеннями модалізованих формул, належать до глобальних.

TMS записуємо як $\boldsymbol{M} = (St, \boldsymbol{R}, A, Im)$.

Наступні визначення даються однаково для всіх описаних різновидів TMS^{Q}.

Предикат $Im(\Phi, \alpha)$ – значення формули Φ у стані α – позначаємо Φ_α.

Формула Ф *неспростовна в стані* α (позначаємо $\alpha \models \Phi$), якщо Φ_α – неспростовний предикат.

Формула Ф *неспростовна в* TMS $\boldsymbol{M}$ (позн.$\boldsymbol{M} \models \Phi$), якщо для всіх $\alpha \in St$ предикат Φ_α є неспростовним.

Нехай $\mathcal{M}$ – клас TMS певного типу.

Формула Ф $\mathcal{M}$*-неспростовна* (позн. $^{\mathcal{M}}\models \Phi$), якщо $\boldsymbol{M} \models \Phi$ для всіх TMS $\boldsymbol{M} \in \mathcal{M}$.

Залежно від умов, накладених на відношення $\triangleright$, можна визначати різні класи GMS$^{Q\equiv}$. Традиційними є випадки, коли $\triangleright$ може бути рефлексивним, симетричним чи транзитивним. Тоді в назві GMS$^{Q\equiv}$ пишемо символ *R*, *T* чи *S*. Отримуємо такі класи: *R*-GMS$^{Q\equiv}$, *T*-GMS$^{Q\equiv}$, *S*-GMS$^{Q\equiv}$, *RT*-GMS$^{Q\equiv}$, *RS*-GMS$^{Q\equiv}$, *TS*-GMS$^{Q\equiv}$, *RTS*-GMS$^{Q\equiv}$.

Мова GMS$^{Q=}$ визначається аналогічно мові GMS$^{Q\equiv}$ з такими відмінностями. Множина символів базових загальнологічних композицій – це $\{\neg, \vee, R^{\bar v,\bar u}_{\bar x,\perp}, \exists x, =_{xy}\}$. У визначенні множини формул замість п. F$\equiv$ маємо $\{=_{xy} \mid x,y \in V\} \subseteq Fr$. Відповідно визначається відображення інтерпретації.

Опишемо мову TmMS$^{Q\equiv}$. Алфавіт мови – це алфавіт мови GMS$^{Q\equiv}$ із тією відмінністю, що множина символів базових модальних композицій $Ms = \{\Box\uparrow,\ \Box\downarrow\}$. Множину *Fm* формул мови задаємо за пп. Fa, F$\equiv$, Fp, FR, F$\exists$ для мови GMS$^{Q\equiv}$, а замість п. F$\Box$ маємо:

F$\Box\uparrow\downarrow$) $\Phi \in Fr \Rightarrow \Box\uparrow\Phi \in Fr$ та $\Box\downarrow\Phi \in Fr$.

Визначаючи відображення *Im* замість I$\Box$ записуємо I$\Box\uparrow\downarrow$ (див. [9, 10]).

Мова TmMS$^{Q=}$ визначається подібно до мови TmMS$^{Q\equiv}$ з такими відмінностями, які має мова GMS$^{Q=}$ щодо мови GMS$^{Q\equiv}$.

Залежно від умов, накладених на $\triangleright$, визначаємо різні класи GMS$^{Q=}$, TmMS$^{Q\equiv}$, TmMS$^{Q=}$ так, як це зроблено для GMS$^{Q\equiv}$.

У такий спосіб визначаємо і мови MMS$^{Q\equiv}$ та MMS$^{Q=}$.

Залежно від того, як задається значення $\Phi_\delta(d)$ за умови $d \notin {}^V A_\delta$, виділено [10]) два різновиди TMS: із *сильною* умовою визначеності на станах та із *загальною* умовою визначеності на станах. Сильна умова є занадто обмежувальною, вона також порушує [10] еквітонність предикатів при дії модальної композиції. Загальна умова набагато природніша, вона задається так:

$\Phi_\delta(d) = \Phi_\delta(d_\delta)$ для всіх $d \in {}^V A$ та $\delta \in St$

Тут d_δ – це позначення для IM $[v \mapsto a \in d \mid a \in A_\delta]$.

За умови $d \notin {}^V A_\delta$ маємо $\Phi_\delta(d) = \Phi_\delta(d_\delta)$. Це означає,що предикати стану δ "відчувають" лише компоненти $v \mapsto a$ з базовими даними $a \in A_\delta$.

Взаємодія модальних композицій із реномінаціями та кванторами досліджена в [10]. Стисло опишемо її для GMS^{Q}.

Теорема 1. Для всіх $\Phi \in Fr$, $d \in {}^V A$ маємо $R^{\bar v,\bar u}_{\bar x,\perp}(\Box\Phi)(d) = \Box(R^{\bar v,\bar u}_{\bar x,\perp}(\Phi))(d)$.

Отже, символи *Ms* можна проносити через символи реномінації.

Теорема 2. Формули вигляду $\exists x \Box\Phi \to \Box\exists x\Phi$, $\Box\forall x\Phi \to \forall x\Box\Phi$, $\Diamond\forall x\Phi \to \forall x\Diamond\Phi$, $\exists x\Diamond\Phi \to \Diamond\exists x\Phi$ спростовуються в GMS^{Q}, водночас вони неспростовні в GMS^{Q} еквітонних предикатів;

2) формули вигляду $\Box\exists x\Phi \to \exists x\Box\Phi$, $\forall x\Diamond\Phi \to \Diamond\forall x\Phi$, $\forall x\Box\Phi \to \Box\forall x\Phi$, $\Diamond\exists x\Phi \to \exists x\Diamond\Phi$ спростовні в GMS^{Q} еквітонних предикатів.

Теореми 1 та 2 відповідним чином формулюються для TmMS^{Q} та MMS^{Q}.

Розглянемо специфічні властивості GMS^{Q}, пов'язані з предикатами рівності. В TmMS^{Q} і MMS^{Q} ці властивості формулюються аналогічно.

Твердження 1. 1) для кожної GMS$^{Q\equiv}$ $\boldsymbol{M}$ маємо $\boldsymbol{M} \models \equiv_{xx}$ та $\boldsymbol{M} \models \Box\equiv_{xx}$;

2) для кожної GMS$^{Q=}$ $\boldsymbol{M}$ маємо $\boldsymbol{M} \models =_{xx}$ та $\boldsymbol{M} \models \Box=_{xx}$.

Справді, $F(\equiv_{xx}) = \varnothing$ та $F(=_{xx}) = \varnothing$.

Теорема 3. Формули $=_{xy} \to \Box=_{xy}$ та $\Box=_{xy} \to =_{xy}$ неспростовні в GMS$^{Q=}$.

Зокрема, $=_{xx} \to \Box=_{xx}$ та $\Box=_{xx} \to =_{xx}$ неспростовні в GMS$^{Q=}$. Водночас

Теорема 4. 1) Формули $\equiv_{xy} \to \Box\equiv_{xy}$ та $\Box\equiv_{xy} \to \equiv_{xy}$ спростовні в GMS$^{Q\equiv}$;

2) формули $Ex \to \Box Ex$ та $\Box Ex \to Ex$ спростовні в GMS$^{Q\equiv}$.

Теореми 3 та 4 засвідчують істотні відмінності $GMS^{Q\equiv}$ та $GMS^{Q=}$. Ще одним підтвердженням цього є

Теорема 5. 1) Формула $=_{xy} \& \Box=_{xz} \to \Box=_{yz}$ неспростовна в $GMS^{Q=}$;

2) формула $\equiv_{xy} \& \Box\equiv_{xz} \to \Box\equiv_{yz}$ спростовна в $GMS^{Q\equiv}$.

2. Відношення логічного наслідку

Відношення логічного наслідку в TMS задаємо на множині формул, специфікованих (відмічених) іменами станів, або специфікованих станами формул.

Специфікована іменем стану формула має вигляд Φ^{α}, де Φ – формула мови, $\alpha \in S$ – її специфікація, S – певна множина імен станів світу. Специфікація вказує на стан світу, в якому розглядається формула.

Множина специфікованих станами формул Σ із множиною специфікацій S *узгоджена* із TMS $M = (St, \boldsymbol{R}, A, Im)$, якщо задана ін'єкція S у St.

На множині специфікованих станами формул введемо відношення неспростовнісного, істиннісного, хибнісного та сильного логічного наслідку, або логічного *IR*-наслідку, *T*-наслідку, *F*-наслідку, *TF*-наслідку. Такі відношення відповідають однойменним відношенням в логіках квазіарних предикатів (див. [5–7]).

Нехай Δ та Γ – множини специфікованих станами формул. Надалі запис вигляду $\Gamma_M|=_* \Delta$ за умовчанням передбачає узгодженість Γ та Δ із TMS M.

Δ є *IR*-наслідком Γ в узгодженій із ними TMS $\boldsymbol{M}$ (позн. $\Gamma_M|=_{IR} \Delta$), якщо для всіх $d \in {}^{V}A$ маємо: $\Phi_{\alpha}(d) = T$ для всіх $\Phi^{\alpha} \in \Gamma \Rightarrow \Psi_{\beta}(d) \neq F$ для деякого $\Psi^{\beta} \in \Delta$.

Δ є *логічним IR-наслідком* Γ відносно TMS певного типу $\mathscr{M}$ (позн. $\Gamma^{\mathscr{M}}|=_{IR} \Delta$), якщо $\Gamma_M|=_{IR} \Delta$ для всіх $M \in \mathscr{M}$.

Δ є *T*-наслідком Γ в узгодженій із ними TMS $\boldsymbol{M}$ (позн. $\Gamma_M|=_T \Delta$), якщо для всіх $d \in {}^{V}A$ маємо: $\Phi_{\alpha}(d) = T$ для всіх $\Phi^{\alpha} \in \Gamma \Rightarrow \Psi_{\beta}(d) = T$ для деякого $\Psi^{\beta} \in \Delta$.

Δ є *логічним T-наслідком* Γ відносно TMS певного типу $\mathscr{M}$ (позн. $\Gamma^{\mathscr{M}}|=_T \Delta$), якщо $\Gamma_M|=_T \Delta$ для всіх $M \in \mathscr{M}$.

Δ є *F*-наслідком Γ в узгодженій із ними TMS $\boldsymbol{M}$ (позн. $\Gamma_M|=_F \Delta$), якщо для всіх $d \in {}^{V}A$ маємо: $\Psi_{\beta}(d) = F$ для всіх $\Psi^{\beta} \in \Delta \Rightarrow \Phi_{\alpha}(d) = F$ для деякого $\Phi^{\alpha} \in \Gamma$.

Δ є *логічним T-наслідком* Γ відносно TMS певного типу $\mathscr{M}$ (позн. $\Gamma^{\mathscr{M}}|=_F \Delta$), якщо $\Gamma_M|=_F \Delta$ для всіх $M \in \mathscr{M}$.

Δ є *TF*-наслідком Γ в узгодженій із ними TMS M (позн. $\Gamma_M|=_{TF} \Delta$), якщо $\Gamma_M|=_T \Delta$ та $\Gamma_M|=_F \Delta$.

Δ є *логічним TF-наслідком* Γ відносно TMS певного типу $\mathscr{M}$ (позн. $\Gamma^{\mathscr{M}}|=_{TF} \Delta$), якщо $\Gamma_M|=_{TF} \Delta$ для всіх $M \in \mathscr{M}$.

Тоді маємо: $\Gamma^{\mathscr{M}}|=_{TF} \Delta \Leftrightarrow \Gamma^{\mathscr{M}}|=_T \Delta$ та $\Gamma^{\mathscr{M}}|=_F \Delta$.

Немодальні властивості цих відношень повторюють відповідні властивості однойменних відношень для множин формул традиційної логіки квазіарних предикатів (див. [5–8]). Це такі властивості.

1) Властивості декомпозиції формул $\neg\neg_L$, $\neg\neg_R$, $\vee_L$, $\vee_R$, $\neg\vee_L$, $\neg\vee_R$, а також властивості $\neg_L$, $\neg_R$ для $|=_{IR}$ (див. [5]).

2) Властивості спрощення та еквівалентних перетворень, пов'язані з реномінаціями; вони отримуються на основі властивостей R, $R\perp I$, $R\perp U$, $R\perp R$, $R\perp\neg$, $R\perp\vee$, $R\uparrow$ для предикатів предикатів (див. [7, 8]).

3) Властивості, пов'язані з елімінацією кванторів; для $GMS^{Q\equiv}$ вони повторюють властивості $\exists R_{\perp L}$, $\neg\exists R_{\perp R}$, $\exists R_{\perp}v_R$, $\neg\exists R_{\perp}v_L$ (див. [8]), в $GMS^{Q=}$ маємо аналогічні властивості $\exists R_{\perp L=}$, $\neg\exists R_{\perp R=}$, $\exists R_{\perp}v_{R=}$, $\neg\exists R_{\perp}v_{L=}$, де замість Ez пишемо $=_{zz}$; до них в $GMS^{Q\equiv}$ додаємо властивості E-розподілу Ed та первісного означення Ev (див. [8]), а в $GMS^{Q=}$ додаємо аналогічні властивості $\downarrow_{=}d$ та $\downarrow_{=}v$, де замість Ez пишемо $=_{zz}$.

4) Властивості спрощення, пов'язані з предикатами-індикаторами в $GMS^{Q\equiv}$; це властивості, індуковані властивостями предикатів $R_{\perp E}$, $R_{\perp Ev}$ та властивість El_{RE}.

5) Властивості, пов'язані з предикатами $\equiv_{xy}$ в $GMS^{Q\equiv}$; це властивості спрощення, індуковані властивостями предикатів $R_{\perp}\equiv_{xx}$, $R_{\perp}\equiv_0$, $R_{\perp}\equiv_1$, $R_{\perp}\equiv_2$, $R_{\perp}\equiv_{1E}$, $R_{\perp}\equiv_{2E}$; властивості елімінації константних

формул $El_{\equiv}$ та $ElR_{\equiv}$; властивості транзитивності $Tr_{\equiv}$ та заміни рівних $\equiv R_{\perp} r_L$, $\equiv \neg R_{\perp} r_L$, $\equiv R_{\perp} r_R$, $\equiv \neg R_{\perp} r_R$.

6) Допоміжні властивості в $GMS^{Q_{\equiv}}$: зняття $\neg$ при перенесенні $\neg\Phi$ з лівої частини відношення наслідку у праву і навпаки для символів Ez, $\equiv_{xy}$ та їхніх реномінацій.

7) Властивості, пов'язані з предикатами $=_{xy}$ та $=_{xx}$ в $GMS^{Q_{=}}$; це властивості спрощення, індуковані властивостями предикатів $R_{\perp}=_{zz}$, $R_{\perp}=_{zz0}$ $R_{\perp}=_0$, $R_{\perp}=_1$, $R_{\perp}=_2$; властивості $El_{R=L}$, $El_{R=R}$ елімінації $\perp$-формул та властивість $El_{=L}$ елімінації $_{pT}$-формул $=_{zz}$; властивості транзитивності $Tr_{=}$ та заміни рівних $=R_{\perp} r_L$, $=R_{\perp} r_R$.

Наведемо умови гарантованої наявності відповідного відношення $\Gamma \mid\!=_* \Delta$.

Відношення $\mid\!=_{IR}$ в $GMS^{Q_{=}}$:
$C \vee C_{Rf=} \vee C\perp_L \vee C\perp_R$.

Відношення $\mid\!=_{IR}$ в $GMS^{Q_{\equiv}}$:
$C \vee CF \vee C_{Rf\equiv} \vee C_{E\equiv L} \vee C_{E\equiv R} \vee CT_{\equiv}$.

Відношення $\mid\!=_{T}$ в $GMS^{Q_{\equiv}}$:
$C \vee CL \vee CF \vee C_{Rf\equiv} \vee C_{E\equiv L} \vee C_{E\equiv R} \vee CT_{\equiv}$.

Відношення $\mid\!=_{F}$ в $GMS^{Q_{\equiv}}$:
$C \vee CR \vee CF \vee C_{Rf\equiv} \vee C_{E\equiv L} \vee C_{E\equiv R} \vee CT_{\equiv}$.

Відношення $\mid\!=_{TF}$ в $GMS^{Q_{\equiv}}$:
$C \vee CLR \vee CF \vee C_{Rf\equiv} \vee C_{E\equiv L} \vee C_{E\equiv R} \vee CT_{\equiv}$.

Зазначені окремі умови вигляду C* означають таке для відношення $\Gamma \mid\!=_* \Delta$:

C) існує формула Φ: $\Phi \in \Gamma$ та $\Phi \in \Delta$;

C) існує формула Φ: $\Phi \in \Gamma$ та $\Phi \in \Delta$;

CL) існує формула Φ: $\Phi \in \Gamma$ та $\neg\Phi \in \Gamma$;

CR) існує формула Φ: $\Phi \in \Delta$ та $\neg\Phi \in \Delta$;

CLR) існують формули Φ, Ψ:
$\Phi, \neg\Phi \in \Gamma$ та $\Psi, \neg\Psi \in \Delta$;

CF) існує формула $R^{\bar{v},\bar{u},z}_{\bar{x},\perp,\perp}(Ez) \in \Gamma$;

$C_{Rf\equiv}$) існує формула $\equiv_{xx} \in \Delta$;

$C_{E\equiv L}$) існують формули $\equiv_{xy}, Ex, Ey$:
$\equiv_{xy}, Ex \in \Gamma$ та $Ey \in \Delta$;

$C_{E\equiv R}$) існують формули $\equiv_{xy}, Ex, Ey$:
$\equiv_{xy}, Ex, Ey \in \Delta$;

$CT_{\equiv}$) існує формула $R^{\bar{v},\bar{u},x,z}_{\bar{x},\perp,\perp,\perp}(\equiv_{xz}) \in \Delta$;

$C_{Rf=}$) існує формула $=_{xx} \in \Delta$;

C$\perp_L$) існує формула $R^{\bar{v},\bar{u},x}_{\bar{w},\perp,\perp}(=_{xy}) \in \Gamma$;

C$\perp_R$) існує формула $R^{\bar{v},\bar{u},x}_{\bar{w},\perp,\perp}(=_{xy}) \in \Delta$.

3. Секвенційні числення

Семантичною основою побудови для ТМЛ числень секвенційного типу є властивості відношення логічного наслідку для множин специфікованих станами формул. Специфікації мають вигляд $\alpha\mid\!-$ чи $\alpha\!-\!\mid$, де α – ім'я стану Секвенції трактуємо як множини таких формул. Виділяючи $\mid\!-$-формули та $-\!\mid$-формули, секвенції позначаємо $\mid\!-\Gamma-\!\mid\Delta$.

Секвенції збагачуємо збудованими на момент виведення множинами відношень на станах. Нехай M – схема моделі світу, тобто збудоване на цей момент відношення досяжності, записане для імен станів. Збагачені секвенції записуємо Σ // M.

Ми пропонуємо секвенційні числення, які формалізують відношення $_{M=}\mid\!=_{IR}$ в $GMS^{Q_{=}}$ та відношення $_M\mid\!=_{IR}$, $_M\mid\!=_T$, $_M\mid\!=_F$, $_M\mid\!=_{TF}$ в $GMS^{Q_{\equiv}}$. Числення для відношення $_{M=}\mid\!=_{IR}$ в $GMS^{Q_{=}}$ назвемо $C^{GQ=IR}$, а числення для $_M\mid\!=_{IR}$, $_M\mid\!=_T$, $_M\mid\!=_F$, $_M\mid\!=_{TF}$ в $GMS^{Q_{\equiv}}$ назвемо $C^{GQ\equiv IR}$, $C^{GQ\equiv T}$, $C^{GQ\equiv F}$, $C^{GQ\equiv TF}$.

Секвенційне числення задається базовими секвенційними формами і умовами замкненості секвенції.

Виведення в секвенційних численнях має вигляд дерева, вершинами якого є секвенції. Правилами виведення секвенційних числень є секвенційні форми, вони індукуються властивостями відношень логічного наслідку. Аксіомами секвенційного числення є замкнені секвенції. Для замкненої секвенції $\mid\!-\Gamma-\!\mid\Delta$ має виконуватись умова $\Gamma \mid\!= \Delta$. Секвенційне дерево замкнене, якщо кожний його лист – замкнена секвенція. Секвенція Σ вивідна, якщо існує замкнене секвенційне дерево з коренем Σ, таке дерево – виведення секвенції Σ.

Умови замкненості секвенції $\mid\!-\Gamma-\!\mid\Delta$ задаються наведеними вище.умовами гарантованої наявності відповідного відношення логічного наслідку $\Gamma \mid\!=_* \Delta$.

Охарактеризуємо детальніше числення $C^{GQ\equiv T}$, $C^{GQ\equiv F}$, $C^{GQ\equiv TF}$. Вони мають од-

накові базові секвенційні форми, відрізняються різними умовами замкненості секвенції. Ці форми можна розбити на групи.

Секвенційні форми, які не пов'язані з модальностями, є аналогами відповідних секвенційних форм в численнях логік квазіарних предикатів $C_{\perp}^{Q\equiv T}$, $C_{\perp}^{Q\equiv F}$, $C_{\perp}^{Q\equiv TF}$ (див. [7]). Ці форми індукуються властивостями декомпозиції формул; властивостями, пов'язаними з елімінацією кванторів; властивостями спрощення та еквівалентних перетворень, пов'язаними з реномінаціями; властивостями спрощення, пов'язаними з предикатами-індикаторами; властивостями, пов'язаними з предикатами $\equiv_{xy}$.

Наведемо для прикладу форми, які є аналогами форм $\vdash\neg\vee$, $\dashv\vdash R_{\perp}\vee$, $\vdash R_{\perp EV}\dashv\vdash$ та $\equiv R_{\perp}r$

$$\vdash\neg\vee\ \frac{\alpha\vdash\neg\Phi,\ \vdash\neg\Psi,\Sigma\ //M}{\alpha\vdash\neg(\Phi\vee\Psi),\Sigma\ //M};$$

$$\dashv\vdash R_{\perp}\vee\ \frac{\alpha\dashv\vdash R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Phi)\vee R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Psi),\Sigma\ //\ M}{\alpha\dashv\vdash R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Phi\vee\Psi),\Sigma\ //\ M};$$

$$\vdash R_{\perp EV}\ \frac{\alpha\vdash Ey,\Sigma\ //M}{\vdash\alpha\vdash R^{\bar{v},\bar{u},z}_{\bar{x},\perp,y}(Ez),\Sigma\ //M};$$

$$\dashv\vdash\equiv R_{\perp}r$$

$$\frac{\alpha\vdash\equiv_{xy},\ \alpha\dashv\vdash R^{\bar{v},\bar{u},z}_{\bar{w},\perp,x}(p),\ \alpha\dashv\vdash R^{\bar{v},\bar{u},z}_{\bar{w},\perp,y}(p),\Sigma\ //M}{\alpha\vdash\equiv_{xy},\ \dashv\vdash R^{\bar{v},\bar{u},z}_{\bar{w},\perp,x}(p),\Sigma\ //M}.$$

До цих форм додаються секвенційні форми, пов'язані з модальностями. Це форми пронесення модальності через реномінацію $\vdash R_{\perp}\Box$, $\dashv\vdash R_{\perp}\Box$, $\vdash\neg R_{\perp}\Box$, $\dashv\vdash\neg R_{\perp}\Box$, та форми елімінації модальних операторів (детально описані в [8]). Для прикладу:

$$\vdash R\Box\ \frac{\alpha\vdash\Box R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Phi),\Sigma\ //M}{\alpha\vdash R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Box\Phi),\Sigma\ //M};$$

$$\dashv\vdash\neg R\Box\ \frac{\alpha\dashv\vdash\neg\Box R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Phi),\Sigma\ //M}{\alpha\dashv\vdash\neg R^{\bar{v},\bar{u}}_{\bar{x},\perp}(\Box\Phi),\Sigma\ //M}.$$

Для пропонованих модальних секвенційних числень справджуються теореми коректності та повноти. Доведення теорем повноти базується на теоремах про побудову контрмоделі за незамкненим шляхом у секвенційному дереві.

Теорема повноти формулюється однотипно для кожного з розглянутих числень. У цьому формулюванні відношенням $M\models_{IR}$, $M\models_{IR}$, $M\models_{T}$, $M\models_{F}$, $M\models_{TF}$ відповідають секвенційні числення $C^{GQ=IR}$, $C^{GQ\equiv IR}$, $C^{GQ\equiv T}$, $C^{GQ\equiv F}$, $C^{GQ\equiv TF}$.

Теорема 6 (коректності та повноти). $\Gamma\models_{*}\Delta\Leftrightarrow$ секвенція $\vdash\Gamma\dashv\vdash\Delta$ вивідна в численні C*.

Детальний опис пропонованих секвенційних числень буде зроблено в наступних роботах.

Висновки

Досліджено програмно-орієнтовані логічні формалізми модального типу – чисті першопорядкові модальні логіки часткових немонотонних квазіарних предикатів. Запропоновано різновиди таких логік з предикатами строгої рівності та слабкої рівності. Описано семантичні моделі та мови цих логік. Увагу акцентовано на властивостях, пов'язаних із предикатами рівності, описано особливості заміни рівних. Визначено низку відношень логічного наслідку для множин специфікованих станами формул. На цій семантичній основі для досліджених логік запропоновано відповідні числення секвенційного типу.

Література

1. S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum (eds), Handbook of Logic in Computer Science, Vol. 1–5, Oxford University Press, 1993–2000.
2. D. Bjorner, M.C. Henson (eds), Logics of Specification Languages, EATCS Series, Monograph in Theoretical Computer Sciens, Springer, 2008.
3. V. Goranko, Temporal Logics, Cambridge University Press, 2023.
4. F. Kröger, S. Merz. Temporal logic and state systems, Springer Science & Business Media, 2008, 436 p.
5. М.С. Нікітченко, О.С. Шкільняк, С.С. Шкільняк, Чисті першопорядкові логіки квазіарних предикатів, Проблеми програмування, 2016, № 2–3, С. 73–86.
6. С.С. Шкільняк, Першопорядкові композиційно-номінативні логіки з предикатами слабкої та строгої рівності, Проблеми програмування. 2019, № 3, С. 28–44.

7. O. Shkilniak, S. Shkilniak. First-Order Sequent Calculi of Logics of Quasiary Predicates with Extended Renominations and Equality, UkrPROG′2022, CEUR Workshop Proceedings (CEUR-WS.org), 2023, pp. 3–18.
8. М.С. Нікітченко, О.С. Шкільняк, С.С. Шкільняк, Секвенційні числення першопорядкових логік часткових предикатів з розширеними реномінаціями та композицією предикатного доповнення, Проблеми програмування, 2020, № 2–3, С. 182–197.
9. О.С. Шкільняк. Модальні логіки немонотонних часткових предикатів. Вісник Київського ун-ту. Серія: фіз.-мат. науки. 2015. Вип. 3. С. 141–147.
10. О.С. Шкільняк. Транзиційні модальні логіки немонотонних квазіарних предикатів. Комп'ютерна математика. 2014, В. 2. С. 99–110.

References

1. S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum (eds), Handbook of Logic in Computer Science, Vol. 1–5, Oxford University Press, 1993–2000.
2. D. Bjorner, M.C. Henson (eds), Logics of Specification Languages, EATCS Series, Monograph in Theoretical Computer Sciens, Springer, 2008.
3. V. Goranko, Temporal Logics, Cambridge University Press, 2023.
4. F. Kröger, S. Merz. Temporal logic and state systems, Springer Science & Business Media, 2008.
5. M. Nikitchenko, O. Shkilniak, S. Shkilniak, Pure first-order logics of quasiary predicates, in Problems in Progamming, 2016, No 2–3. pp. 73–86.
6. S. Shkilniak, First-order composition-nominative logics with predicates of weak equality and of strong equality, in Problems in Progamming, 2019, No 3, pp. 28–44.
7. O. Shkilniak, S. Shkilniak. First-Order Sequent Calculi of Logics of Quasiary Predicates with Extended Renominations and Equality, UkrPROG′2022, CEUR Workshop Proceedings (CEUR-WS.org), 2023, pp. 3–18.
8. M. Nikitchenko, O. Shkilniak, S. Shkilniak, Sequent calculi of first-order logics of partial predicates with extended renominations and composition of predicate complement, in Problems in Progamming, 2020, No 2–3, pp. 182–197.
9. O. Shkilniak, Modal logics of non-monotone partial predicates, in Bulletin of Taras Shevchenko National University of Kyiv, Series Physics & Mathematics, 2015, **3**, pp. 141–147.
10. O. Shkilniak, Transitional modal logics of non-monotone quasiary predicates, in Computer Mathematics, 2014, **2**, pp. 99–110.

Одержано: 10.04.2024
Внутрішня рецензія отримана: 21.04.2024
Зовнішня рецензія отримана: 29.04.2024

Про авторів:

[1]*Шкільняк Оксана Степанівна*,
к.ф.-м.н., доцент.
http://orcid.org/0000-0003-4139-2525.

[2]*Шкільняк Степан Степанович*,
д. ф.-м. н., професор
http://orcid.org/0000-0001-8624-5778.

Місце роботи авторів:

[1]Київський національний
університет імені
Тараса Шевченка,
тел. (+38) (044) 259-05-11
E-mail: oksana.sh@knu.ua
Сайт: http://csc.knu.ua

[2]Київський національний
університет імені
Тараса Шевченка,
Тел. (+38) (044) 521-33-45
E-mail: ss.sh@knu.ua,
Сайт: http://csc.knu.ua

УДК 004.65 http://doi.org/10.15407/pp2024.02-03.028

І.М. Лисенко

ЧИСЛЕННЯ РЯДКІВ ДЛЯ МУЛЬТИМНОЖИННОЇ ТАБЛИЧНОЇ АЛГЕБРИ

Дана робота становить логічне продовження досліджень, присвячених актуальній проблемі розробки теоретичних основ реляційних (табличних) баз даних. Питання використання мультимножин у табличних базах даних є важливим і актуальним, зважаючи на те, що багато мов запитів, орієнтованих на роботу з базами даних, потребують реляційну модель, яка передбачає мультимножинну семантику. Це обумовлено наявністю прикладних задач, особливістю яких є множинність і повторюваність даних. Наприклад, це соціологічні опитування різних груп населення, розрахунки на ДНК та ін. У цьому контексті розглядається питання побудови числення рядків для мультимножинної табличної алгебри, в якій поняття таблиці уточнюється, використовуючи поняття мультимножини. У роботі проведено формалізацію числення рядків для мультимножинної табличної алгебри: визначено алфавіт, синтаксис термів, атомів і формул числення рядків; введено множину дозволених формул, використовуючи концепцію вільних і зв'язаних рядків; також вводяться поняття схеми та множини атрибутів, з якими рядок зустрічається у формулах. Дано означення виразу числення рядків для мультимножинної табличної алгебри, згідно з яким – це мультимножина рядків, що задовольняють визначену дозволеною формулою умову. У статті наведено правила, за якими визначається кількість дублікатів рядків у результуючій мультимножині. Інший важливий результат полягає у доведенні того, що побудоване числення рядків є не менш виразним, ніж мультимножинна таблична алгебра. Це дослідження відкриває нові можливості для розвитку теорії баз даних і може бути корисним для спеціалістів у сфері інформаційних технологій та баз даних. Воно сприяє глибшому розумінню принципів побудови запитів, що є важливим аспектом у сучасній комп'ютерній науці та індустрії.
Ключові слова: реляційні бази даних мультимножина, мультимножинна таблична алгебра, числення рядків.

I.M. Lysenko

TUPLE CALCULUS FOR MULTISET TABLE ALGEBRA

This paper is a logical continuation of research devoted to the actual problem of developing the theoretical foundations of table (relational) databases. The issue of using multisets in table databases is important and relevant. Many database-oriented languages require a relational model with multiset semantics. There are many applied problems, the feature of which is multiplicity and repeatability of data. For example, these are sociological polls of different population groups, calculations on DNA, and others. In this context, the question of constructing a tuple calculus for a multiset table algebra is considered, in which the concept of a table is refined using the concept of a multiset. In the article, the formalization of tuple calculus for multiset table algebra is carried out. The alphabet, and the syntax of terms, atoms, and formulas are defined. A set of legal formulas is introduced through the concept of the free and bound variable. The concept of a scheme and set of attributes with which a tuple variable occurs in a formula are also introduced. The definition of tuple calculus expression for multiset table algebra is given, according to which it is a multiset of tuples that satisfy the condition defined by the legal formula. The article provides rules for determining the number of tuple duplicates in the resulting multiset. Another important result consists in proving that constructed tuple calculus is as expressive as multiset table algebra. This research opens up new possibilities for database theory development and may be useful for information technology and database professionals. It contributes to a deeper understanding of construction query principles, an important aspect of modern computer science and industry.
Key words: relation databases, multiset, multiset table algebra, tuple calculus.

Вступ

Реляційне числення є основою більшості реляційних мов запитів, оскільки орієнтоване лише на очікуваний результат. Водночас реляційна алгебра передбачає побудову реляційного виразу та виконання операцій. Існує два основних підходи до реляційного числення:

1) числення кортежів (Е. Кодд) – оперує рядками таблиць [1];

2) числення доменів (М. Лакруа, А. Піротт) – фокусується на доменах таблиць [2].

У роботі [3] розглядається таблична алгебра нескінченних (скінченних) таблиць, яка суттєво узагальнює та розширює класичну реляційну алгебру Кодда. Для даної алгебри побудовано числення на кортежах (доменах) та доведено еквівалентність табличної алгебри та даних числень. Методологічну основу вказаних досліджень складає композиційний підхід до програмології.

Крім того у роботі [3] побудовано мультимножинну табличну алгебру, яка розширює можливості баз даних за рахунок використання мультимножин. Природно постає питання про розробку числення для даної алгебри.

Мета дослідження – побудувати числення рядків для мультимножинної табличної алгебри та показати, що воно є не менш виразним, ніж мультимножинна таблична алгебра.

1. Мультимножинна таблична алгебра

Почнемо з розгляду ключових термінів мультимножин на основі джерел [4, 5]. Нехай U – довільна множина. Під мультимножиною α з основою U розумітимемо відображення вигляду $\alpha : U \to N$, де $N = \{1, 2, ...\}$ – множина натуральних чисел.

Позначимо через $\boldsymbol{D}$ – універсум елементів основ мультимножин. Характеристична функція мультимножини α – це функція вигляду $\chi_\alpha : \boldsymbol{D} \to Z_+$, значення якої задаються кусковою схемою:

$$\chi_\alpha(d) = \begin{cases} \alpha(d), \text{ якщо } d \in \operatorname{dom}\alpha, \\ 0, \text{ інакше;} \end{cases}$$

для всіх $d \in \boldsymbol{D}$.

Порожньою мультимножиною $\varnothing_m$ називається мультимножина, основа якої – це порожня множина.

1-мультимножинами називаються мультимножини, областю значень яких є одноелементна множина вигляду $\{1\}$.

Основні поняття та твердження мультимножинної табличної алгебри розглянемо, спираючись на монографію [3].

Розглянемо дві множини: $\boldsymbol{A}$ – множина атрибутів та $\boldsymbol{D}$ – універсальний домен. Схемою будемо називати довільну скінченну множину атрибутів $R \subseteq \boldsymbol{A}$. Рядком схеми R будемо називати іменну множину на парі R, $\boldsymbol{D}$, проекція якої за першою компонентою рівна R. Використаємо наступні позначення: $S(R)$ – множина всіх рядків схеми R, а S – множина всіх рядків.

Пара $\langle \psi, R \rangle$, де ψ – довільна (можливо нескінченна) мультимножина, а R – схема таблиці, називається таблицею мультимножинної табличної алгебри.

Множина всіх таблиць схеми R позначається як $\Psi(R)$, а множина всіх таблиць – $\Psi = \bigcup\limits_{R \subseteq \boldsymbol{A}} \Psi(R)$.

Через $Occ(s, \psi)$ позначають кількість дублікатів рядка s у мультимножині ψ. Мультимножину ψ також записують як $\{s_1^{n_1}, ..., s_k^{n_k}, ...\}$, де $n_i = Occ(s_i, \psi)$, $i = 1, 2, \ldots$, а $\Theta(\psi) = \{s_1, ..., s_k, ...\}$ – основа мультимножин ψ.

Мультимножинною табличною алгеброю називають алгебру $\langle \Psi, \Omega_{P,\Xi} \rangle$, де Ψ – множина всіх таблиць, $\Omega_{P,\Xi} = \left\{ \bigcup_{All}^{\Psi,R}, \bigcap_{All}^{\Psi,R}, \setminus_{All}^{\Psi,R}, \sigma_{p,R}, \pi_{X,R}, \underset{R_1,R_2}{\otimes}, Rt_{\xi,R}, \sim_R \right\}_{X,R,R_1,R_2 \subseteq \boldsymbol{A}}^{p \in P, \xi \in \Xi}$ – сигнатура, P, Ξ – множини параметрів.

Має місце наступне твердження.

Твердження 1. Будь-який вираз мультимножинної табличної алгебри можна замінити еквівалентним йому виразом, який використовує лише операції селекції, з'єднання, проєкції, об'єднання, різниці та перейменування.

2. Побудова числення рядків для мультимножинної табличної алгебри

Основою реляційного числення є числення предикатів першого порядку. Побудову числення рядків для мультимножинної табличної алгебри почнемо з визначення алфавіту. Алфавіт числення рядків для мультимножинної табличної алгебри складають:

- множина атрибутів $\boldsymbol{A}$ (тобто множина імен атрибутів) і універсальний домен $\boldsymbol{D}$;
- множина предметних змінних, тобто змінних рядків $x_1, x_2, \ldots$;
- множина предметних констант $d_1, d_2, \ldots$;
- множина функціональних символів $f_1^{n_1}, f_2^{n_2}, \ldots,\ n_i \geq 1$;
- множина предикатних символів $p_1^{m_1}, p_2^{m_2}, \ldots,\ m_i \geq 1$;
- множина символів сталих таблиць – $\langle \alpha, R \rangle$;
- множина символів змінних таблиць – $\langle X, R \rangle$;
- знаки логічних операцій $\neg$, $\wedge$, $\vee$ та квантори $\exists, \forall$;
- знаки пунктуації – дужки та кома $(,)$.

Область інтерпретації предметних констант – це універсальний домен $\boldsymbol{D}$, а область інтерпретації предметних змінних – це множина всіх рядків на домені $\boldsymbol{D}$.

Надалі в тексті використовуємо наступні позначення: $\boldsymbol{x}$ – синтаксична змінна, областю зміни якої є змінні: $\boldsymbol{f}$ ($\boldsymbol{p}$) – синтаксична змінна, областю зміни якої є функціональні (предикатні) символи; $\boldsymbol{d}$ – синтаксична змінна, областю зміни якої є константи; $\boldsymbol{\mathcal{A}}$ – синтаксична змінна, областю зміни якої є атрибути.

Серед слів, записаних за допомогою символів алфавіту, виділяють терми та формули. Сформулюємо означення цих синтаксичних об'єктів індукцією за їхньою довжиною.

Дамо означення *термів* числення рядків для мультимножин:

a) $\boldsymbol{d}$ – терм;

b) $\boldsymbol{x}(\boldsymbol{\mathcal{A}})$ – терм;

c) якщо $\boldsymbol{f}$ – n-арний функціональний символ, а $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$ – терми, то $\boldsymbol{f}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n)$ – терм;

d) ніяких інших термів, крім зазначених в пунктах a)-c) немає.

Надалі в тексті $\boldsymbol{u}$ – це синтаксична змінна, областю зміни якої є терми. Областю інтерпретації термів є універсальний домен $\boldsymbol{D}$.

Дамо означення *формул* числення рядків для мультимножин. Почнемо з *атомарних формул* (атомів), які бувають трьох типів:

a1') нехай $\langle \alpha, R \rangle$ – стала таблиця, а $\boldsymbol{x}$ – змінний рядок, тоді $\alpha_R(\boldsymbol{x})$ – атом, який (за інтерпретації) означає, що рядок $\boldsymbol{x} \in \langle \alpha, R \rangle$;

a1") нехай $\langle X, R \rangle$ – змінна таблиця, а $\boldsymbol{x}$ – змінний рядок, тоді $X_R(\boldsymbol{x})$ – атом, який (за інтерпретації) означає, що рядок $\boldsymbol{x} \in \langle X, R \rangle$;

a2) нехай $\boldsymbol{p}$ – m-арний предикат на універсальному домені $\boldsymbol{D}$, а $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ – терми, тоді $\boldsymbol{p}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m)$ – атом.

Побудуємо формули із атомів, використовуючи логічні зв'язки $\neg$, $\wedge$, $\vee$, квантори $\exists, \forall$ та дужки $(,)$.

f1. Будь-який атом є формулою.

f2. Якщо $\boldsymbol{P}$ і $\boldsymbol{Q}$ – формули, тоді вирази $(\neg \boldsymbol{P}), (\boldsymbol{P} \wedge \boldsymbol{Q}), (\boldsymbol{P} \vee \boldsymbol{Q})$ також є формулами.

f3. Нехай $\boldsymbol{x}$ – змінний рядок, $\boldsymbol{P}$ – формула, $R \subseteq \boldsymbol{A}$ – схема, тоді вирази $\exists \boldsymbol{x}(R)\boldsymbol{P}$ і $\forall \boldsymbol{x}(R)\boldsymbol{P}$ також є формулами.

f4. Якщо $\boldsymbol{P}$ – формула, тоді вираз $(\boldsymbol{P})$ – також формула.

f5. Інших формул, крім зазначених в пунктах f1-f4 немає.

Будемо використовувати $\boldsymbol{P}$, $\boldsymbol{Q}$ та $\boldsymbol{G}$ як синтаксичні змінні, областю зміни яких є формули.

Визначимо клас дозволених формул числення рядків для мультимножинної табличної алгебри, використовуючи поняття вільних та зв'язаних рядків, схеми $scheme(\boldsymbol{x},\boldsymbol{P})$ змінного рядка $\boldsymbol{x}$ та множини атрибутів $attr(\boldsymbol{x},\boldsymbol{P})$, з якими рядок $\boldsymbol{x}$ зустрічається у формулах.

Вирази $scheme(\boldsymbol{x},\boldsymbol{P})$ та $attr(\boldsymbol{x},\boldsymbol{P})$ визначені, якщо рядок $\boldsymbol{x}$ має хоча б одне вільне входження у формулу $\boldsymbol{P}$. Крім того має місце включення $attr(\boldsymbol{x},\boldsymbol{P}) \subseteq scheme(\boldsymbol{x},\boldsymbol{P})$, якщо дані вирази визначені.

Виділимо клас, так званих, дозволених формул числення рядків мультимножинної табличної алгебри.

Почнемо з термів:

1) якщо $\boldsymbol{u} = \boldsymbol{d}$, то $attr(\boldsymbol{x},\boldsymbol{u}) = \varnothing$;
2) якщо $\boldsymbol{u} = \boldsymbol{x}(\mathcal{A})$, то $attr(\boldsymbol{x},\boldsymbol{u}) = \{\mathcal{A}\}$, а $attr(\boldsymbol{x},\boldsymbol{y}(\mathcal{A})) = \varnothing$, якщо $\boldsymbol{x} \neq \boldsymbol{y}$;
3) якщо $\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{u}_1,...,\boldsymbol{u}_n)$ де $\boldsymbol{u}_i$ – терми, то $attr(\boldsymbol{x},\boldsymbol{u}) = \bigcup_{i=1}^{n} attr(\boldsymbol{x},\boldsymbol{u}_i)$.

Іншими словами, $attr(\boldsymbol{x},\boldsymbol{u})$ – це множина атрибутів, які повинна мати схема рядка $\boldsymbol{x}$.

Нехай $\boldsymbol{P}$ – атомарна формула, тоді

a1') якщо $\boldsymbol{P} = \alpha_R(\boldsymbol{x})$, то єдине входження змінного рядка $\boldsymbol{x}$ є вільним у формулі $\boldsymbol{P}$ і $scheme(\boldsymbol{x},\boldsymbol{P}) = attr(\boldsymbol{x},\boldsymbol{P}) = R$;

a1") аналогічно, якщо $\boldsymbol{P} = X_R(\boldsymbol{x})$, то єдине входження змінного рядка $\boldsymbol{x}$ є вільним у формулі $\boldsymbol{P}$ і $scheme(\boldsymbol{x},\boldsymbol{P}) = attr(\boldsymbol{x},\boldsymbol{P}) = R$;

a2) якщо $\boldsymbol{P} = \boldsymbol{p}(\boldsymbol{u}_1,...,\boldsymbol{u}_m)$, де $\boldsymbol{u}_i$ – терми, $\boldsymbol{x}_1,...,\boldsymbol{x}_k$ – усі змінні цих термів, тоді входження цих змінних рядків є вільними у формулі $\boldsymbol{P}$, при цьому схема $scheme(\boldsymbol{x}_i,\boldsymbol{P})$ не визначена, а $attr(\boldsymbol{x}_i,\boldsymbol{P}) = \bigcup_{j=1}^{m} attr(\boldsymbol{x}_i,\boldsymbol{u}_j)$, $i = 1,...,k$.

Атоми завжди дозволені. Побудову дозволених формул числення рядків для мультимножинної табличної алгебри проведемо індукцією за довжиною формули. Нехай $\boldsymbol{G}$ і $\boldsymbol{Q}$ дозволені формули.

f1. Якщо $\boldsymbol{P} = \neg\boldsymbol{G}$, то формула $\boldsymbol{P}$ – дозволена, а входження змінних у формулі $\boldsymbol{P}$ будуть вільними або зв'язаними залежно від того, вільні або зв'язані входження цих змінних у формулі $\boldsymbol{G}$. Якщо рядок $\boldsymbol{x}$ входить у формулу $\boldsymbol{G}$ вільно, то мають місце рівності $scheme(\boldsymbol{x},\boldsymbol{P}) \simeq scheme(\boldsymbol{x},\boldsymbol{G})$ і $attr(\boldsymbol{x},\boldsymbol{P}) = attr(\boldsymbol{x},\boldsymbol{G})$, де $\simeq$ – узагальнена рівність, яка означає, що обидві частини рівності або невизначені, або визначені та мають рівні значення.

f2. Якщо $\boldsymbol{P} = \boldsymbol{G} \wedge \boldsymbol{Q}$ або $\boldsymbol{P} = \boldsymbol{G} \vee \boldsymbol{Q}$, то входження змінних у формулі $\boldsymbol{P}$ будуть вільними або зв'язаними, залежно від того, вільні або зв'язані входження цих змінних у підформулах $\boldsymbol{G}$ або $\boldsymbol{Q}$. Припустимо, що змінний рядок $\boldsymbol{x}$ входить у підформули $\boldsymbol{G}$ та/або $\boldsymbol{Q}$ вільно. Для схеми та множини атрибутів, з якими рядок $\boldsymbol{x}$ зустрічається у формулах маємо три випадки:

a. Схеми формул $scheme(\boldsymbol{x},\boldsymbol{G})$ та $scheme(\boldsymbol{x},\boldsymbol{Q})$ визначені. Формула $\boldsymbol{P}$ дозволена, якщо виконується рівність $scheme(\boldsymbol{x},\boldsymbol{G}) = scheme(\boldsymbol{x},\boldsymbol{Q})$. Покладемо за означенням $scheme(\boldsymbol{x},\boldsymbol{P}) = scheme(\boldsymbol{x},\boldsymbol{G})$.

b. Схема визначена тільки для однієї з підформул. Припустимо, що схема $scheme(\boldsymbol{x},\boldsymbol{G})$ визначена, а схема $scheme(\boldsymbol{x},\boldsymbol{Q})$ – невизначена. Формула $\boldsymbol{P}$ дозволена, якщо виконується включення $attr(\boldsymbol{x},\boldsymbol{Q}) \subseteq scheme(\boldsymbol{x},\boldsymbol{G})$. Покладемо за означенням $scheme(\boldsymbol{x},\boldsymbol{P}) = scheme(\boldsymbol{x},\boldsymbol{G})$.

c. Схема невизначена для обох підформул. Формула $\boldsymbol{P}$ є дозволеною, але $scheme(\boldsymbol{x},\boldsymbol{P})$ теж невизначена.

Для будь-якого з розглянутих випадків a - c $attr(\boldsymbol{x},\boldsymbol{P}) = attr(\boldsymbol{x},\boldsymbol{G}) \cup attr(\boldsymbol{x},\boldsymbol{Q})$.

f3. Якщо $\boldsymbol{P} = \exists\boldsymbol{x}(R)\boldsymbol{G}$ і змінний рядок $\boldsymbol{x}$ входить у підформулу $\boldsymbol{G}$ вільно, то

формула $\boldsymbol{P}$ – дозволена. При визначеності схеми $scheme(\boldsymbol{x},\boldsymbol{G})$ та за умови виконання включення $attr(\boldsymbol{x},\boldsymbol{G}) \subseteq R$ повинна виконуватися рівність $scheme(\boldsymbol{x},\boldsymbol{G}) = R$. Оскільки змінна $\boldsymbol{x}$ не входить вільно у формулу $\boldsymbol{P}$, то $scheme(\boldsymbol{x},\boldsymbol{P})$ і $attr(\boldsymbol{x},\boldsymbol{P})$ невизначені. Якщо $\boldsymbol{y} \neq \boldsymbol{x}$, то будь-яке входження змінної $\boldsymbol{y}$ в $\boldsymbol{P}$ вільне або зв'язане, залежно від того, вільне або зв'язане входження $\boldsymbol{y}$ в $\boldsymbol{G}$. Якщо $\boldsymbol{y}$ входить у $\boldsymbol{P}$ вільно, то $scheme(\boldsymbol{y},\boldsymbol{P}) \simeq scheme(\boldsymbol{y},\boldsymbol{G})$ і $attr(\boldsymbol{y},\boldsymbol{P}) = attr(\boldsymbol{y},\boldsymbol{G})$.

f4. Якщо $\boldsymbol{P} = \forall \boldsymbol{x}(R)\boldsymbol{G}$ і змінний рядок $\boldsymbol{x}$ входить у підформулу $\boldsymbol{G}$ вільно, то формула $\boldsymbol{P}$ – дозволена, а всі визначення і обмеження аналогічні випадку f3.

f5. Якщо $\boldsymbol{P} = (\boldsymbol{G})$, то формула $\boldsymbol{P}$ – дозволена, а вільні та зв'язані входження змінних, схема і множина атрибутів, з якими змінний рядок зустрічається у формулах, залишаються такими як і для підформули $\boldsymbol{G}$.

Іншими словами, рівність $attr(\boldsymbol{x},\boldsymbol{P}) = R$, означає, що для конкретної інтерпретації формули $\boldsymbol{P}$, коли змінна $\boldsymbol{x}$ набуває значення у вигляді рядка s схеми R', повинне виконуватися включення $R \subseteq R'$.

Дозволеність формули забезпечує її коректну інтерпретацію при розгляді виразів числення рядків для мультимножинної табличної алгебри.

Вирази числення рядків для мультимножинної табличної алгебри мають вигляд $\{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$, де

1. формула $\boldsymbol{P}$ – дозволена;
2. змінна $\boldsymbol{x}$ – єдина змінна, яка входить у формулу $\boldsymbol{P}$ вільно;
3. якщо $scheme(\boldsymbol{x},\boldsymbol{P})$ визначена, то $scheme(\boldsymbol{x},\boldsymbol{P}) = R$, інакше, $attr(\boldsymbol{x},\boldsymbol{P}) \subseteq R$
4. n – кількість дублікатів рядка $\boldsymbol{x}$.

Варто підкреслити, що результатом виконання запиту, визначеного виразом $\{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$, є мультимножина рядків, які описує вираз $\boldsymbol{P}(\boldsymbol{x})$.

Нехай формула $\boldsymbol{P}(\boldsymbol{x})$ – дозволена, $R \subseteq \boldsymbol{A}$. Підставивши конкретний рядок s схеми R замість $\boldsymbol{x}$ у формулу $\boldsymbol{P}$ отримаємо формулу $\boldsymbol{P}(s/\boldsymbol{x})$, значення якої визначається шляхом модифікації кожного атома з $\boldsymbol{P}$ за наступними правилами:

a1') нехай рядок $\boldsymbol{x}$ підформули $\alpha_{R'}(\boldsymbol{x})$ вільний у формулі $\boldsymbol{P}$. За означенням дозволеної формули маємо включення $R' \subseteq R$. Атом $\alpha_{R'}(\boldsymbol{x})$ істинний у разі підстановки конкретного рядка s замість змінної $\boldsymbol{x}$, якщо $s \mid R' \in t$, інакше атом $\alpha_{R'}(\boldsymbol{x})$ хибний

a1") нехай рядок $\boldsymbol{x}$ підформули $X_{R'}(\boldsymbol{x})$ вільний у формулі $\boldsymbol{P}$. Аналогічно випадку a1'), має місце включення $R' \subseteq R$. Атом $X_{R'}(\boldsymbol{x})$ істинний у разі підстановки конкретного рядка S замість змінної $\boldsymbol{x}$, якщо $s \mid R' \in \bar{X}$, де $\bar{X} \in P(S(R'))$ – значення змінної таблиці;

a2) нехай рядок $\boldsymbol{x}$ підформули $\boldsymbol{p}(\boldsymbol{u}_1,...,\boldsymbol{u}_m)$ вільний у формулі $\boldsymbol{P}$, тоді замінимо $\boldsymbol{x}(A_i)$ на $d_i \in \boldsymbol{D}$, де $\langle A_i, d_i \rangle \in s$ (d_i значення атрибута A_i в рядку s) при підстановці конкретного рядка s замість змінної $\boldsymbol{x}$. Атом $\boldsymbol{p}(\boldsymbol{u}_1,...,\boldsymbol{u}_m)$ буде істинний, якщо предикат $\boldsymbol{p}$ – істинний на відповідних значеннях, інакше атом буде хибний.

Інтерпретацією формули є множина значень істинності всіх атомів цієї формули. Припустимо, що формула $\boldsymbol{P}$ – дозволена без вільних змінних. Визначимо інтерпретацію формули $\boldsymbol{P}$ для кожного випадку.

f1. Якщо $\boldsymbol{P} = \neg \boldsymbol{G}$, то в $\boldsymbol{G}$ не повинно бути вільних змінних. $\boldsymbol{P}$ істинна формула, коли підформула $\boldsymbol{G}$ хибна, і хибна, коли підформула $\boldsymbol{G}$ істинна.

f2. Якщо $\boldsymbol{P} = \boldsymbol{G} \wedge \boldsymbol{Q}$ або $\boldsymbol{P} = \boldsymbol{G} \vee \boldsymbol{Q}$, то в підформулах $\boldsymbol{G}$ та $\boldsymbol{Q}$ не повинно бути вільних змінних. При $\boldsymbol{P} = \boldsymbol{G} \wedge \boldsymbol{Q}$, формула $\boldsymbol{P}$ істинна тоді, коли підформули $\boldsymbol{G}$ та $\boldsymbol{Q}$ одночасно істинні, та хибна у всіх інших випадках. При $\boldsymbol{P} = \boldsymbol{G} \vee \boldsymbol{Q}$, формула $\boldsymbol{P}$ хибна тоді, коли підформули $\boldsymbol{G}$ і $\boldsymbol{Q}$ одноча-

сно хибні, та істинна у всіх інших випадках.

f3. Якщо $\boldsymbol{P} = \exists \boldsymbol{x}(R)\boldsymbol{G}$, то $\boldsymbol{x}$ – єдина змінна, яка входить у підформулу $\boldsymbol{G}$ вільно. Формула $\boldsymbol{P}$ істинна, якщо знайдеться принаймні один рядок $s \in S(R)$ такий, що формула $\boldsymbol{G}(s/\boldsymbol{x})$ отримана при підстановці s замість $\boldsymbol{x}$ – істинна, інакше $\boldsymbol{P}$ – хибна.

f4. Якщо $\boldsymbol{P} = \forall \boldsymbol{x}(R)\boldsymbol{G}$, то $\boldsymbol{x}$ – єдина змінна, яка входить у підформулу $\boldsymbol{G}$ вільно. Формула $\boldsymbol{P}$ істинна, якщо для кожного рядка $s \in S(R)$ формула $\boldsymbol{G}(s/\boldsymbol{x})$, отримана в результаті підстановки, буде істинна, інакше формула $\boldsymbol{P}$ хибна.

f5. Якщо $\boldsymbol{P} = (\boldsymbol{G})$, то формула $\boldsymbol{P}$ істинна, коли підформула $\boldsymbol{G}$ істинна і хибна, коли підформула $\boldsymbol{G}$ хибна.

Нехай $E = \{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$ – вираз числення рядків для мультимножинної табличної алгебри. Значенням виразу E назвемо таблицю $\langle \varphi, R \rangle$ мультимножинної табличної алгебри, яка складається з рядків $s \in S(R)$ таких, що формула $\boldsymbol{P}(s/\boldsymbol{x})$ істинна, а кількість дублікатів рядка s в таблиці $\langle \varphi, R \rangle$ визначається так:

1) нехай $\boldsymbol{P} = \alpha_R(\boldsymbol{x})$, тоді $n = Occ(s,\varphi) = Occ(s,\alpha)$;

2) нехай $\boldsymbol{P} = X_R(\boldsymbol{x})$, тоді $n = Occ(s,\varphi) = Occ(s,X)$;

3) нехай $\boldsymbol{P} = \boldsymbol{p}(\boldsymbol{u}_1,...,\boldsymbol{u}_m)$, де $\boldsymbol{u}_j$ – терми, $j = \overline{1,m}$, $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_k$ – всі змінні цих термів, тоді маємо

$$n = Occ(s,\varphi) = \sum_{s' \in \Theta(\psi), s' \mid R' = s \atop \boldsymbol{p}(u_1,\ldots,u_m) = true} Occ(s',\psi),$$

де $\langle \psi, R \rangle$ – таблиця до якої будується запит, $R' = \bigcup_{j=1}^{m} attr(\boldsymbol{x}_i, \boldsymbol{u}_j)$, $i = \overline{1,k}$;

4) нехай $\boldsymbol{P} = \neg \boldsymbol{G}$ та формула $\boldsymbol{G}$ породжує m дублікатів рядка s, тоді $n = Occ(s,\varphi) = Occ(s, C(\psi)) \dot{-} m$, де $C(\psi)$ – мультимножина таблиці $C(\langle \psi, R \rangle)$, яка є насиченням таблиці $\langle \psi, R \rangle$, що є значенням виразу $\{\boldsymbol{y}(R) \mid \boldsymbol{G}(\boldsymbol{y})\}$, а зрізана різниця $\dot{-}$ розуміється в звичайному сенсі $x \dot{-} y = \begin{cases} x - y, \text{якщо } x \geq y \\ 0, \text{якщо } x < y \end{cases}$;

5) нехай $\boldsymbol{P} = \boldsymbol{G} \wedge \boldsymbol{Q}$ та формула $\boldsymbol{G}$ породжує k дублікатів рядка s, а формула $\boldsymbol{Q}$ породжує m дублікатів рядка s, тоді $n = Occ(s,\varphi) = \min(k,m)$;

6) нехай $\boldsymbol{P} = \boldsymbol{G} \vee \boldsymbol{Q}$ та формула $\boldsymbol{G}$ породжує k дублікатів рядка s, а формула $\boldsymbol{Q}$ породжує m дублікатів рядка s, тоді $n = Occ(s,\varphi) = k + m$;

7) нехай $\boldsymbol{P} = \exists \boldsymbol{x}(R)\boldsymbol{G}$ та формула $\boldsymbol{G}$ породжує k дублікатів рядка s, тоді $n = Occ(s,\varphi) = k$;

8) нехай $\boldsymbol{P} = \forall \boldsymbol{x}(R)\boldsymbol{G}$ та формула $\boldsymbol{G}$ породжує k дублікатів рядка s, тоді $n = Occ(s,\varphi) = k$;

9) нехай $\boldsymbol{P} = (\boldsymbol{G})$ та формула $\boldsymbol{G}$ породжує k дублікатів рядка s, тоді $n = Occ(s,\varphi) = k$.

Твердження 2. Якщо F – вираз мультимножинної табличної алгебри, то можна ефективно побудувати еквівалентний йому вираз E числення рядків для мультимножинної табличної алгебри.

Доведення. Згідно Твердження 1 у процесі доведення досить розглянути вирази мультимножинної табличної алгебри, які міститимуть тільки операції об'єднання, різниці, селекції, проєкції, з'єднання та перейменування. Доведемо дане твердження методом математичної індукції за числом операцій у виразі F.

База індукції. У цьому випадку вираз F не містить операцій. Це або стала, або змінна таблиці.

Нехай $F = \langle \alpha, R \rangle$ – стала таблиця, де α – мультимножина рядків схеми R. Покладемо $E = \{\boldsymbol{x}^n(R) \mid \alpha_R(\boldsymbol{x})\}$.

Нехай $F = \langle X, R \rangle$ – змінна таблиця, тоді $E = \{\boldsymbol{x}^n(R) \mid X_R(\boldsymbol{x})\}$. □

Крок індукції. Припустимо, що твердження виконується для всіх виразів мультимножинної табличної алгебри, які міс-

тять менше i операцій. Розглянемо вираз F, який містить i операцій.

Випадок 1 (об'єднання). $F = F_1 \bigcup_{All}^{R} F_2$, де вирази F_1 і F_2 мають менше i операцій. Тоді існують вирази числення рядків $\{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$ і $\{\boldsymbol{x}^m(R) \mid \boldsymbol{Q}(\boldsymbol{x})\}$, які еквівалентні F_1 і F_2 відповідно. Значеннями цих виразів є таблиці, в які рядок $\boldsymbol{x}$ входить n та m разів відповідно. Покладемо E рівним $\left\{\boldsymbol{x}^k(R) \mid \boldsymbol{P}(\boldsymbol{x}) \vee \boldsymbol{Q}(\boldsymbol{x})\right\}$, де кількість дублікатів рядка $\boldsymbol{x}$ у вихідній таблиці дорівнює $k = n + m$. □

Випадок 2 (різниця). $F = F_1 \setminus_{All}^{R} F_2$, де вирази F_1 і F_2 мають менше i операцій. Нехай $\{\boldsymbol{x}^n(R_1) \mid \boldsymbol{P}(\boldsymbol{x})\}$ і $\{\boldsymbol{x}^m(R) \mid \boldsymbol{Q}(\boldsymbol{x})\}$ – вирази числення рядків, еквівалентні F_1 і F_2 відповідно. Значеннями цих виразів є таблиці, в які рядок $\boldsymbol{x}$ входить n та m разів. Покладемо E рівним $\left\{\boldsymbol{x}^k(R) \mid \boldsymbol{P}(\boldsymbol{x}) \wedge \neg \boldsymbol{Q}(\boldsymbol{x})\right\}$, причому кількість дублікатів рядка $\boldsymbol{x}$ у вихідній таблиці дорівнює $k = n \dot{-} m$. □

Випадок 3 (селекція). $F = \sigma_{\tilde{p}, R}\left(F_1\right)$, де вираз F_1 має менше i операцій. Тоді існує вираз $\{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$, еквівалентний F_1. Покладемо $E = \left\{\boldsymbol{x}^k(R) \mid \boldsymbol{P}(\boldsymbol{x}) \wedge \boldsymbol{p}(\boldsymbol{x}(A_1),...,\boldsymbol{x}(A_m))\right\}$, де $R = \{A_1,...,A_m\}$ схема таблиці, що є значенням виразу F_1, а кількість дублікатів рядка $\boldsymbol{x}$ у вихідній таблиці не змінюється, тому $k = n$. Предикат-параметр селекції заданий наступним чином $\tilde{p}(s) = T \Leftrightarrow \boldsymbol{p}(s(A_1),...,s(A_m)) = T, s \in S(R)$, де $\boldsymbol{p}$ – m-арний предикатний символ. □

Випадок 4 (проєкція). $F = \pi_{X, R}(F_1)$, де вираз F_1 має менше i операцій. Тоді існує вираз числення рядків $\{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$, еквівалентний F_1. Значенням цього виразу є таблиця, до якої рядок $\boldsymbol{x}$ входить n разів. Покладемо E рівним $\{\boldsymbol{y}^k(X \bigcap R) \mid \exists \boldsymbol{x}(R)(\boldsymbol{P}(\boldsymbol{x}) \wedge \bigwedge_{A \in X \bigcap R} \boldsymbol{y}(A) = \boldsymbol{x}(A))\}$, де кількість дублікатів рядка $\boldsymbol{y}$ у вихідній таблиці дорівнює $k = \sum_{\bigwedge_{A \in X \bigcap R} \boldsymbol{y}(A) = \boldsymbol{x}(A)} n$. □

Випадок 5 (з'єднання). $F = F_1 \underset{R_1, R_2}{\otimes} F_2$. Тоді існують вирази числення рядків $\{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$ і $\{\boldsymbol{x}^m(R) \mid \boldsymbol{Q}(\boldsymbol{x})\}$, які еквівалентні F_1 і F_2 відповідно. Значеннями цих виразів є таблиці, до якиїх рядок $\boldsymbol{x}$ входить n разів, а рядок $\boldsymbol{y}$ – m разів відповідно. Покладемо вираз E рівним $\{\boldsymbol{z}^k(R_1 \bigcup R_2) \mid \exists \boldsymbol{x}(R_1) \exists \boldsymbol{y}(R_2)(\boldsymbol{P}(\boldsymbol{x}) \wedge \boldsymbol{Q}(\boldsymbol{y}) \wedge$, $\wedge \bigwedge_{A \in R_1} \boldsymbol{z}(A) = \boldsymbol{x}(A) \wedge \bigwedge_{A \in R_2} \boldsymbol{z}(A) = \boldsymbol{y}(A))\}$ причому кількість дублікатів рядка $\boldsymbol{z}$ у вихідній таблиці дорівнює $k = n \times m$. □

Випадок 6 (перейменування). $F = Rt_{\xi, R}(F_1)$, де $\xi : \boldsymbol{A} \tilde{\rightarrow} \boldsymbol{A}$ – ін'єктивна функція, що здійснює перейменування атрибутів. Нехай $\{\boldsymbol{x}^n(R) \mid \boldsymbol{P}(\boldsymbol{x})\}$ – вираз числення рядків, еквівалентний F_1. Значенням цього виразу є таблиця, до якої рядок $\boldsymbol{x}$ входить n разів. Покладемо E рівним $\{\boldsymbol{y}^k(R_2) \mid \exists \boldsymbol{x}(R)(\boldsymbol{P}(\boldsymbol{x}) \wedge \bigwedge_{C \in R \setminus dom\xi} \boldsymbol{y}(C) = \boldsymbol{x}(C) \wedge$ $\wedge \bigwedge_{A \in R \bigcap dom\xi} \boldsymbol{x}(A) = \boldsymbol{y}(\xi(A)))\}$, де $R_2 = R \setminus dom\xi \bigcup \xi[R]$, а кількість дублікатів рядка $\boldsymbol{y}$ у вихідній таблиці дорівнює n. □ □

3. Зв'язок з мовою запитів SQL

Фундаментальним об'єктом даних у мові SQL є не класичне відношення Е. Кодда, а скоріше таблиця. Причому таблиці SQL містять, власне кажучи, не множини, а мультимножини рядків, тобто

допускаються повторення елементів. Основні оператори SQL не є реляційними операторами у сенсі цього терміну, а є аналогами реляційних операторів, які призначені для роботи з мультимножинами. При створенні нової таблиці за допомогою запиту, система SQL, як правило, не видаляє дублікати рядків, а повертає результат, в якому один і той же рядок може зустрічатися декілька разів. Так, щоб виключити появу дублікатів, за оператором SELECT потрібно поставити ключове слово DISTINCT.

Продемонструємо доцільність побудови числення рядків для мультимножинної табличної алгебри на наступному прикладі.

Розглянемо таблицю <*Scores*, *R*>, де схема R={№, Name, Topic 1, Topic 2, Topic 3, Quiz} (див. Табл. 1).

Таблиця 1

Таблиця <*Scores*, *R*>

№	Name	Topic 1	Topic 2	Topic 3	Quiz
1.	Баков А.	5	15	14	16
2.	Бойко І.	6	14	15	16
3.	Борода К.	7	17	20	20
4.	Геранов О.	9	20	19	20
5.	Кайдан Ю.	5	18	15	18
6.	Кузенко Є.	6	19	13	20
7.	Кулак П.	4	8	9	16

Відповідь на питання «Які бали отримали за тест (Quiz) перші п’ять студентів?» на мові SQL матиме вигляд:

```
SELECT Quiz
FROM Scores
LIMIT 5;
```

Таблиця-результат <Quiz, R_1>, де схема R_1={Quiz}, міститиме значення-дублікати (див. Табл. 2).

Таблиця 2

Таблиця <Quiz, R_1>

Quiz
16
16
20
20
18

Реалізувати цей запит в термінах класичного числення рядків неможливо, оскільки результатом його виконання буде множина рядків, а не мультимножина (як очікується), тобто дублікати не будуть враховуватися в результат.

У численні рядків для мультимножинної табличної алгебри вираз еквівалентний даному запиту матиме вигляд: $\{x^n(\text{Quiz}) \mid \exists y(R)(Scores(y) \wedge (y(№) = 1 \vee y(№) = 2 \vee y(№) = 3 \vee y(№) = 4 \vee y(№) = 5) \wedge \wedge x(\text{Quiz}) = y(\text{Quiz}))\}$.

Результат, який він описує, аналогічний результату отриманому при виконанні відповідного запиту на мові SQL.

Отже, як видно з прикладу, побудоване числення рядків для мультимножинної табличної алгебри дозволяє адекватно формалізувати мови запитів, зокрема SQL, врахувавши мультимножинну семантику закладену в їхній основі.

Висновки

У статті запропоновано числення рядків для мультимножинної табличної алгебри. Визначено алфавіт, синтаксис термів, атомів і формул числення рядків. Використовуючи концепцію вільних і зв'язаних рядків, поняття схеми та множини атрибутів, з якими рядок зустрічається у формулах, введено клас дозволених формул. Показано, що запропоноване числення рядків є не менш виразним, ніж мультимножинна таблична алгебра. На прикладі продемонстровано доцільність побудови числення рядків для мультимножинної табличної алгебри, враховуючи те, що мови за-

питів орієнтовані на роботу з базами даних передбачають повторюваність елементів у таблиці.

Проблематика наступних досліджень полягає у встановленні відповідного дуального результату.

Література

1. Codd E.F. Relational Completeness of Data Base Sublanguages. *Data Base Systems*. New York: Prentice-Hall, 1972. P. 65-98
2. Lacroix M., Pirotte A. Domain-oriented Relational Languages. Proc. 3rd Int. Conf. on Very Large Data Bases. Tokyo, October, 1977. P. 370-378.
3. Буй Д. Б., Глушко І. М. Числення на розширення сигнатур табличних алгебр: монографія. Ніжин: НДУ ім. М. Гоголя, 2016. 151 с.
4. Реляційні бази даних: табличні алгебри та SQL-подібні мови / В. Н. Редько, Ю. Й. Брона, Д. Б. Буй, С. А. Поляков. Київ: Видавничий дім "Академперіодика", 2001. 198 с.
5. Богатирьова Ю. О. Теорія мультимножин та її застосування: дис. Канд. фіз.-мат. наук: 01.05.03. К., 2011. 113с.

References

1. E.F. Codd Relational Completeness of Data Base Sublanguages, in: Data Base Systems (1972) 65-98.
2. Lacroix M., Pirotte A. Domain-oriented Relational Languages, in: Proceedings of. 3rd Int. Conf. on Very Large Data Bases., 1977, pp. 370-378.
3. V.N. Redko, et al. Relational Databases: Table Algebras and SQL-like Language. Kyiv: Publishing house Academperiodica, 2001. [in Ukrainian]
4. D.B Buy, I.M. Glushko, Calculi and extensions of table algebras signature. Nizhyn: NDU im. M. Gogol, 2016. [in Ukrainian]
5. J.A. Bogatyreva Multisets theory and its applications. Ph.D. thesis, Kyiv National Taras Shevchenko University, 2011. [in Ukrainian]

Одержано: 12.02.2024
Внутрішня рецензія отримана: 19.02.2024
Зовнішня рецензія отримана: 08.03.2024

Про авторів:
Лисенко Ірина Миколаївна,
к.ф.-м.н., доцент.
http://orcid.org/0000-0003-2549-5356.

Місце роботи автора:
Ніжинський державний
університет імені Миколи Гоголя,
E-mail: iryna.glushko@ndu.edu.ua
Сайт: http://www.ndu.edu.ua/

УДК 681.3 http://doi.org/10.15407/pp2024.02-03.037

Д.В. Рагозін, А.Ю. Дорошенко

МОДЕЛЮВАННЯ ПРОДУКТИВНОСТІ ПАМ'ЯТІ ВІДЕОКАРТ ДЛЯ LLM-НЕЙРОМЕРЕЖ

У статті розглянуто характеристики швидкодії нейромережевих алгоритмів типу Generative pre-trained transformer, відомих також як Large Language Model, на сучасній відеокарті з метою визначення обмежень на використання таких нейромереж на мобільних обчислювальних платформах. Нейромережі цього класу цікаві як засіб ухвалення рішень, але на великому корпусі вивчених текстів їхня швидкодія уповільнюється і швидко зростає кількість необхідних обчислюваних ресурсів, тому корисно визначити, чи можливе використання таких нейромереж, сформованих на звуженому корпусі текстів і на пристроях з відносно невеликою обчислювальною потужністю. Дослідження характеристик здійснювалося за допомогою симулятора GPGPUSim, який може бути вільно сконфігурований як віртуальна відеокарта будь-якої потужності. Оскільки алгоритм базований на послідовності множення матриць, а його швидкодія на сучасних обчислювачах обмежена швидкодією підсистеми пам'яті, була досліджена статистика роботи кеш-пам'яті відеопроцесора різних рівнів та її взаємодія з основною пам'яттю системи. За допомогою GPGPUSim зібрано статистику для різних варіантів конфігурації мережі Generative pre-trained transformer версії 2, від конфігурації small до конфігурації xl. Аналіз статистики доступів до кеш-па'мяті другого рівня, промахів при доступі до кеш-пам'яті, а також кількості доступів у основну динамічну пам'ять показує, що у середніх конфігураціях кількість промахів у кеш другого рівня перевищує 7-8%, що досить багато. Це свідчить, що наявного об'єму кеш-пам'яті другого рівня не вистачає для функціонування основних алгоритмів і наявний досить значний обмін з основною пам'яттю. Натомість мінімальна конфігурація small може бути обрахована з мінімальним залученням ресурсів, тому може бути використана у перспективі як система прийняття рішень на платформах з невеликими ресурсами у разі зменшення корпусу текстів. А це відкриває цікаві перспективи для використання нейромереж цього типу для автономного ухвалення рішень.

Ключові слова: відеокарта, large language model, швидкодія, нейромережа, обчислювальні ресурси

D.V.Rahozin, A.Y. Doroshenko

MODELING VIDEOCARD MEMORY PERFORMANCE FOR LLM NEURAL NETWORKS

The paper covers the analysis of performance characteristics of neural network-based algorithms class Generative pre-trained transformer, also known as Large Language Model, for contemporary videocards. The goal is to check the application limitations for this class of neural networks for mobile computing platforms. This network class is interesting for use as control system tool, but for the bigger text corpuses the network performance degrades and the number of used computer resources grows quickly, so we need to explore if this type of network, but based on a smaller text corpus, is a feasible tool for devices with comparatively low computing capability. The performance investigation was performed with the help of GPGPUSim simulator, which can be freely configured as a virtual videocard of any computing capability. As this neural network computations are based on the calculation of a sequence of matrix multiplications, and its performance is limited by the memory bandwidth, we analyze the behavior statistics of the different cache memory levels of the videocard processor and the cache interaction with the main memory. With the help of GPGPUSim we have gathered statistics for different Generative pre-trained transformer version 2 configurations, from small to xl configurations. The level 2 cache memory access statistics, level 2 cache memory access misses, number of accesses to main memory show that even for the middle-level network configurations the number of level 2 cache memory misses exceeds the 7-8% level. This number looks high and this evidence shows that the size of the cache memory is quite small for executing this neural network configuration, also there is the substantial traffic from cache memory to main memory. Although, the minimal so-called small configuration can be computed faster and with moderated resources, and so can be used further as a part of decision-making system for computing platforms with moderate performance and resources for the case of limited text corpuses. This opens good enough perspectives for using this type of neural networks for autonomous decision making.

Key words: videocard, large language model, computing performance, neural network, computational capability

Вступ

В останні 10 років розвиток мікроелектроніки практично привів до виникнення комп'ютерної інженерної галузі, наближеної до поняття «штучний інтелект». Якщо 7-8 років тому складні завдання розпізнавання образів або сегментації зображення у реальному часі вважалися за фантастику навіть для спеціалізованих потужних відеопроцесорів, то зараз навіть середні, а не флагманські мобільні обчислювальні платформи, такі як, System-on-Chip (SoC) від Qualcomm серії Snapdragon, вирішують ці завдання в реальному часі. Втім, новітні нейромережі типу LLM[1] кидають виклик наявній потужності обчислювачів. Розвиток мікроелектроніки за законом Мура прогнозує експоненціальне збільшення кількісті транзисторів у рамках мікросхем SoC і впровадження ще більш потужних нейромереж на мобільній обчислювальні платформи у найближчі роки. Однак, за прогнозами складні LLM буде застосовуватися обмежено.

У цій статті ми досліджуємо LLM-нейромережі з точки зору їх застосовності на мобільних платформах. Ми дослідимо найцікавіші характеристики нейромережі – швидкодію доступу до пам'яті і ефективність роботи багаторівневої пам'яті відеокарти для наявних алгоритмів. Більша кількість транзисторів принципово дозволяє мати додаткові обчислювальні пристрої, але їхня робота неможлива без ефективного доступу до пам'яті системи. Потенційний масивний обчислювальний паралелізм відеокарти може нівелюватися складнощами доступу до пам'яті, не зважаючи на наявні апаратні пристрої для прискорення доступу до пам'яті або приховування затримок під час доступу до пам'яті. Аналіз за допомогою спеціалізованих симуляторів відеокарт дозволяє оцінити використання ієрархії пам'яті та можливості застосування алгоритму на певних конфігураціях відеокарт.

Обчислення сучасних нейромереж

Практично всі сучасні нейромережі базуються на операціях множення матриць [2], що дозволяє успішно формалізувати обчислювальні процеси, коли нейромережа подається як послідовність обчислювальних блоків, які обробляють матриці. Якщо колись множення матриць було обчислювально складною операцією, оскільки потребувало арифметичного блоку з обробкою плаваючої коми, то зі збільшенням кількості транзисторів конвеєризоване множення чисел із плаваючою комою обчислюється за 1 машинний цикл навіть у дешевих процесорах ARM. З іншого боку, кожне множення потребує доступу до двох комірок пам'яті з умовних матриць A та B, що вже проблематично виконати за 1 цикл, за винятком архітектур DSP, які зараз застосовуються нечасто і лише як спеціалізовані співпроцесори.

Таким чином, ефективність подібного сорту алгоритмів повністю залежить від підсистеми пам'яті: від ефективності алгоритму, який має заощаджувати доступи до пам'яті, та від правильного керування доступом до ієрархічної пам'яті відеокарти. У більшості випадків програмування алгоритмів такі оптимізації не проводяться.

Питання, пов'язані з керуванням пам'яттю відеокарти для старіших нейромереж типу YolovX описані в [3]. У наш час за два роки може змінюватися покоління відеокарт, тому подібні за структурою нейромережі вже давно виконують свої функції у реальному часі, і фокус зацікавлення зміщений на більш обчислювально важкі системи, зокрема, нейромережі сегментації [4] або Large Language Models – LLM[1].

Обчислювальні особливості нейромереж типу LLM

Останнім часом нейромережі класу LLM асоціюються із застосунками типу ChatGPT, які викликають широкі дискусії щодо методів використання як серед професіоналів, так і серед пересічних користувачів. Фактично застосунок складається із семантичної мовної моделі і машини виводу, яка генерує (добудовує, розширює або скорочує) тексти за певним запитом.

Машина виводу може добудовувати різну семантику з тих зв'язків, які означені у мовній моделі. Зв'язки у LLM побудовані за корпусом текстів, який покриває певну область людської діяльності. Для великих моделей, що базуються на великих корпусах текстів, або на великому обсязі текстів програмного забезпечення, побудова відповідей на запити нагадує роботу уявного штучного інтелекту, хоча результат такої роботи й не генерує нових знань (без специфічного програмування з боку користувача), а є лише редукцію або розширенням тексту за допомогою семантичних зв'язків.

У разі звуження корпусу текстів, над якими визначається модель для LLM, семантична модель фактично стає довідником з певної царини знань, і для застосунків у невеликій предметній області може буде використана для побудови реакцій [5] на стан зовнішнього середовища. LLM може розглядатися як аналог старого поняття «експертної системи», яка була більш формалізована, але давала лише ті відповіді, на які була запрограмована. LLM дозволяє мати більше свободи у реагуванні на ситуації реального життя.

Як приклад LLM можна навести NanoGPT[6], для якого є декілька мовних моделей, пояснення та приклади генерації мовних моделей за наборами текстів, а також вихідні коди машини виводу текстів, обсяг якої складає близько 800 рядків мовою C. Це дозволяє зацікавленим особам ознайомитися на тільки з алгоритмікою LLM, а й змоделювати швидкодію LLM.

Обчислювальний приклад

Як обчислювальний приклад розглянемо GPT2, описаний Basil Hosmer в [7], і більш детально в [8]. Використано GPT2 small configuration з проєкту Андрія Карпати NanoGPT [6] з параметрами: *layers* = 12, *heads* = 12, *embed* = 768. Параметр *embed* далі змінюватиметься для аналізу відповідно до розширених конфігурацій GPT2 у рамках здійснення обчислювальних експериментів.

У LLM цього типу основне обчислювальне навантаження відбувається в Attention Layer, найбільш інтенсивно використовується обчислення *attention*: наступна послідовність шести матричних множень [7,9], де знаком `@`, як і в оригінальному тексті, позначено матричне множення:

Таблиця 1.

```
(1)  Q = input @ wQ
(2)  K_t = wK_t @ input_t
(3)  V = input @ wV
(4)  attn = sdpa(Q @ K_t)
(5)  head_out = attn @ V
(6)  out = head_out @ wO
```

У наступній таблиці подано розмірності оброблюваних масивів та ємність пам'яті необхідна для їхнього зберігання. Базовий тип даних – 4-х байтове значення з плаваючою комою, стандартне для відеокарти. Конфігурація GPT2 – «*small*». Всі масиви розташовані у пам'яті відеокарти.

Масив	Стовп.	Рядків	Пам'ять, Кбт
Input_t	768	256	768
Input	256	768	768
wQ	768	64	192
Q	256	64	64
wK_t	64	768	192
wV	768	64	192
wO	64	786	192

Додатково використано допоміжні масиви у пам'яті відеокарти:

Масив	Стовп.	Рядків	Пам'ять, Кбт
K_t	64	256	64
V	256	64	64
Attn	256	256	256
head_out	256	64	64
Out	256	768	768

Сумарна пам'ять масивів моделі *small* складає 3.5Мбт, що, зважаючи на розмір кешів сучасних мобільних відеокарт, є досить великим об'ємом для кеш-пам'яті, але досить невеликим відносно всієї оперативної пам'яті відеокарти. У цьому випадку складність обчислювальної задачі залежить не від відносно маленького наразі об'єму використаної пам'яті, а від кількості доступів до пам'яті, оскільки кожна операція множення вимагає два доступи до

кеш-пам'яті, незалежно від способу оптимізації алгоритму. Всі способи програмної оптимізації намагаються зробити ефективнішим використання саме кеш-пам'яті, адже кошт доступу до кеш-пам'яті вважається мінімальним і в більшості випадків оптимізації не піддається. Але пропускна здатність кеш-пам'яті теж лімітована. Зважаючи на масований паралелізм, коли сотні потоків виконання можуть дати несподівані ефекти для пропускної здатності контролерів пам'яті від кеш першого рівня до динамічної пам'яті, нам необхідно використовувати спеціальне програмне забезпечення, наприклад, GPGPUSim [10].

Симуляція масованого паралелізму за допомогою GPGPUSim

На відміну від давньої звичної парадигми багатоядерного процесора (наприклад Intel Core, AMD Ryzen), відеокарта пропонує масований паралелізм (тисячі потоків виконання замість десятків у багатоядерному процесорі) за рахунок спрощення обчислювальних ядер, ускладнення ієрархії пам'яті та адаптованої концепції програмування. Зазначимо, що у випадку програмування відеокарти немає ускладнень технології паралельного програмування відносно програмування багатоядерного процесора, оскільки для отримання якісного паралельного коду необхідно знати тонкощі синхронізації пам'яті та роботи кеш-пам'яті для обох типів архітектур. GPGPUSim повністю і точно (максимум 5% відхилення по продуктивності від апаратної реалізації [10]) моделює роботу відеокарт Nvidia, що дозволяє досить точно оцінити продуктивність алгоритму, вплив оптимізацій на продуктивність та вплив апаратних особливостей на роботу алгоритму. Модель GPGPUSim гнучко налаштовується від параметрів конвеєризації арифметико-логічних пристроїв до часових діаграм доступу до мікросхем динамічної пам'яті. Так користувач може змінити необхідним для себе чином конфігурацію відеокарти і змоделювати виконання програми на будь-якій модифікації відеокарти.

Основною цікавинкою GPGPUSim є моделювання складної ієрархії пам'яті, оскільки повністю модельована часова діаграма доступу до всіх рівнів кеш і динамічної пам'яті, всі пристрої та черги синхронізації при доступі до кеш-пам'яті. Для нашого дослідження основними цікавими цифрами є обсяг доступу до кеш другого рівня (кеш L2), оскільки він практично найбільше впливає на швидкодію. Кеш першого рівня (L1), кеш третього рівня (L3) та доступ до основної пам'яті мають дещо менше, оскільки кеш L1 більш локально обслуговує шейдери, кеш L3 повільніший, а поведінка L2 дає більш релевантну інформацію.

GPGPUSim є прозорим для користувача, вбудовується в систему на базі Linux стандартним чином і використовується для запуску вже наявних програм для відеокарт, реалізованих за допомогою технології CUDA від Nvidia. Для цього GPGPUSim має власну версію бібліотеки *cudart*, яка підміняє собою як API CUDA, так і внутрішні механізми роботи драйвера за допомогою стандартних можливостей наявного завантажувача виконаних файлів формату ELF, які використовуються в усіх модифікаціях Linux. Симулятор формує шейдери, моделює стан пам'яті, API CUDA і OpenCL. Тому з точки зору користувача програма виконується на відеокарті, але вкрай повільно. Інформація про симуляцію перенаправляється до текстового файлу.

В нашому випадку використана Ubuntu 18, компілятор GCC версії 7.5.x для збірки симулятора і модельної програми, пакет утиліт (*nvcc* та інше) від CUDA 10.1. Була використана модель відеокарти QV100, яка має пасивне охолодження, тому за сценаріями використання більш схожа на мобільні відеокарти, і нема ніяких підстав вважати, що за кілька років обчислювальна потужність мобільних відеокарт не наблизиться до цієї моделі.

Для реалізації алгоритму, вказаного у табл. 1, була реалізована CUDA-програма, яка базована на прикладах множення матриць *matMul* з прикладів (*samples*), наданих до CUDA 10.1. Система збірки прикладу була модифікована для GPGPUSim шляхом додавання вказівок

стандартному лінкеру для використання бібліотеки *cudart* з комплекту GPGPUSim. Далі програма запускалася за допомогою GPGPUSim, дані про роботу кеш виділялися з файлу результатів симуляції.

Результати симуляції

Основною метою симуляції було дослідження алгоритму, представленого у табл. 1, спочатку на даних моделі GPT2 *small* і подальшого збільшення параметрів *layers*/*heads*/*embed* до моделей *large*/*xl*. Хоча алгоритм множення матриць є найбільш вивченим за всю історію досліджень, комплексні операції з матрицями на платформі з масованим паралелізмом з урахуванням складного кеша 2-го рівня цікаві з точки зору як ефективності роботи пам'яті, оскільки правильно використані кеші можуть працювати як на паралельну видачу даних, так і тих явищ у роботі кешу, які виникають у разі збільшення розміру даних і конфліктів у кеш, і, зважаючи на складність алгоритму, більш-менш точне їх обчислення неможливе без моделювання.

У наступних таблицях наведені результати для різних значень *embed*: 768, 1280 і 1600, використаних у різних конфігураціях GPT2 [6]. Стовпчик # означає номер множення у табл. 1, *L2 access* – кількість доступів до кеш пам'яті L2, *L2 misses* – кількість промахів у L2 (що додає сильну затримку у подачі даних), *L2 misses* % - відносна кількість промахів. Літера *M* означає мільйон.

Таблиця 2

Доступ у пам'ять для *embed* = 768

#	L2 access	L2 misses	L2 misses %
1	1.23M	0.053M	4.3%
2	1.59M	0.098M	6.2%
3	2.82M	0.108M	3.8%
4	2.83M	0.108M	3.8%
5	2.97M	0.108M	3.8%
6	2.98M	0.108M	3.6%

Для моделі GPT2 *small* ми бачимо, що в цілому відносна кількість промахів у L2 є великою лише для стадії 2 алгоритму, оскільки для великої кількості алгоритмів «прийнятні» цифри відносних непопадань у L2 це 2-3%. Більший відсоток призводить до деградації швидкодії відносно наявних можливостей арифметико-логічних пристроїв комп'ютера. Аналізуємо далі.

Таблиця 3

Доступ у пам'ять для *embed* = 1280

#	L2 access	L2 misses	L2 misses %
1	3.22M	0.190M	5.9%
2	4.11M	0.305M	7.4%
3	7.46M	0.592M	7.9%
4	7.47M	0.597M	8.0%
5	7.61M	0.615M	8.1%
6	7.62M	0.615M	8.1%

Набір даних збільшився у 1.67 рази, відсоток промахів у кеш значно збільшився, особливо на останніх стадіях (4-6). Для стадій 2-6 спостерігаємо дуже великий відсоток промахів, понад 7, що призводить до значних сповільнень алгоритму, адже кошт промаху у десятки разів перевищує кошт вибірки даних з кешу L2. Проаналізуємо велику модель *xl* з GPT2.

Таблиця 4

Доступ у пам'ять для *embed*=1600

#	L2 access	L2 misses	L2 misses %
1	5.42M	0.315M	5.8%
2	6.96M	0.655M	9.4%
3	12.4M	1.08M	8.7%
4	12.5M	1.08M	8.7%
5	12.5M	1.1M	8.8%
6	12.5M	1.1M	8.8%

Порівняно із табл. 3 відсоток промахів збільшився, але принципово дані табл. 3 і 4 вказують на те, ще така конфігурація алгоритму перевищує продуктивність такої конфігурації процесора.

Наведемо також розбивку доступу до пам'яті на читання і запис для *embed*=768 і 1280.

Таблиця 5

#	*embed*=768		*embed*=1600	
	L2 RD	L2 WR	L2 RD	L2 WR
1	1.17M	0.074M	5.1M	0.32M
2	1.44M	0.147M	6.4M	0.64M
3	2.6M	0.221M	11.5M	0.96M
4	2.61M	0.222M	11.4M	0.96M
5	2.74M	0.23M	11.5M	0.96M
6	2.75M	0.23M	11.7M	0.97M

Ці цифри типові для матричних операцій, коли на велику кількість операцій читання маємо небагато операцій запису.

Дані табл. 2, 3 і 4 фактично відрізняються лише однією розмірністю масивів *embed*, використаних в алгоритмі в табл. 1. Всього цей алгоритм використовує три розмірності, і було вирішено як обчислювальний експеримент збільшити третю з матричних розмірностей з 64 до 96.

Таблиця 6

Доступ у пам'ять для *embed*=1600 і розмірності рядків Q і 96

#	L2 access	L2 misses	L2 misses %
1	5.42M	0.297M	5.5%
2	7.59M	0.646M	8.5%
3	13.01M	1.07M	8.2%
4	13.03M	1.08M	8.3%
5	13.17M	1.09M	8.3%
6	13.19M	1.09M	8.3%

Порівнюючи табл. 4 і 6 зазначимо, що, незважаючи на більший обсяг пам'яті, з яким оперує алгоритм у конфігурації з табл. 6, відносна кількість промахів у L2 кеш у відсотках стає меншою. Оскільки алгоритм досить складний, немає сенсу робити теоретичні розвідки щодо поведінки кешу, але симулятор дозволяє виявляти такі цікаві ефекти. Важливо, що симулятор моделює поведінку контролеру пам'яті між запусками шейдерів і таким чином може вказувати на оптимальні сценарії запуску шейдерів, коли виконані шейдери ефективно використовують результати попередньо виконаних шейдерів.

Зібрана у табл. 2-6 інформація може піддаватися вільній інтерпретації, але підкріпимо дані симуляції кешу L2 даними щодо кількості доступів до динамічної пам'яті. Трафік у динамічну пам'ять (DRAM) скерований контролером кеш пам'яті і є необхідним для збереження даних, які не вміщуються у кеш. Зазначимо, що GPGPUSim симулює DRAM за допомогою повного точного опису доступу до пам'яті у прив'язці до реального часу. Таким чином, трафік у DRAM корелює з кількістю промахів у L2 кеш, але не у лінійній залежності.

Таблиця 7.

Кількість читань та записів DRAM

	embed = 1280		*embed* = 1600	
	Reads	Writes	Reads	Writes
1	17.8K	42.7K	38.5K	156K
2	29K	124K	55K	462K
3	111K	383K	157K	834K
k	115K	387K	162K	834K
5	131K	403K	177K	853K
6	131K	403K	177K	853K

Окремо зазначимо, що трафік у DRAM для *embed*=768 нульовий. Це пояснюється тим, що у разі відсутності запитів на виділення кешу (які генеруються під час роботи інших шейдерів, які виконують свої певні алгоритми) контролер кеш не робитиме запитів на збереження даних у динамічній пам'яті окрім специфічних синхронізацій. Коли використаний обсяг пам'яті не вміщується в кеш, з'являється трафік у DRAM. Оскільки алгоритм з табл. 1 виконується багато раз, трафік у пам'ять відповідно зростає.

Наступний рис. 1 ілюструє залежність промахів у кеш-пам'ять від параметра *embed*, із зростанням якого зростає розмір масивів.

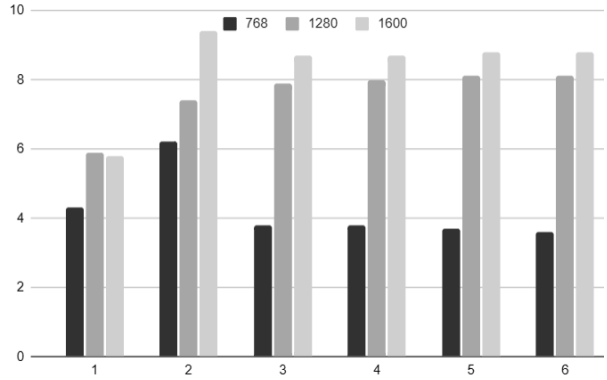


Рис. 1. Залежність промахів у кеш пам'ять від параметра *embed*.

Якість роботи алгоритму GPT2 прямо залежить від величини параметру *embed*. Якщо ми звернемося до більш потужних моделей, що обробляють дуже великі корпуси текстів[11] (наприклад, частину мережі Internet), то розмір наборів даних вимагає для тренування гігабайти пам'яті, а для виводу – 16Гбайт оператив-

ної пам'яті. Проте метою більш великих моделей є наближення до людського стилю викладу інформації, а GPT2 має меншу ресурсоємність і може продукувати вивід на відносно малопотужних процесорах. Відносно невелика швидкість генерації відповідей (десятки токенів на секунду) на сучасному процесорі Intel Core фактично вимагає переміщувати код на відеопроцесор і підлаштовувати розмір моделі до об'єму наявної пам'яті.

Розвитком цієї роботи можуть кілька напрямів. По-перше, визначення необхідних обчислювальних потужностей, необхідних для запуску алгоритму в цілому, оскільки темою цієї статті були найбільш обчислювально важкі місця. Це дозволяє визначити типовий час відповіді у режимі реального часу. По-друге, визначення прийнятного розміру корпусу професійних (спеціалізованих), за допомогою якого GPT2 дозволяє генерувати або звужувати текст, дозволяє визначити ліміти застосовності алгоритмів такого типу на мобільних пристроях. Таке дослідження стикається зі складнощами, оскільки моделі на основі вірогідностей важко тестувати, складно визначити якість проробки моделлю запиту, необхідно визначити обмеження на запити. Але це окреме дослідження ресурсоємності алгоритмів LLM показує, що ці алгоритми можуть використовуватися не лише на великих і потужних обчислювальних системах.

Висновки

У статті розглянуто аналіз продуктивності нейромереж типу LLM для сучасної відеокарти. Показано, що невеликі конфігурації моделі GPT2 можуть бути розміщені на обчислювальних пристроях такого типу, що відкриває можливості використання цього типу нейромереж для автономної роботи на мобільних пристроях.

References

1. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L .Jones, A. Gomez, L. Kaizer, I. Polosukhin. Attention is all you need. // In proc. 31st Conf. on Neural Information Processing Systems (NIPS), Dec. 2017, pp. 6000-6010.
2. S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro B, et al. CuDnn: efficient primitives for deep learning. arXiv preprint: arXiv:14100759. (2014). [Accessed 17/03/2024]
3. Rahozin D., Doroshenko A. (2022) Performance Model for Convolutional Neural Networks. In: Shkarlet S. et al. (eds) Mathematical Modeling and Simulation of Systems. MODS Lecture Notes in Networks and Systems, vol 344. Springer, pp. 239-251. DOI: doi.org/10.1007/978-3-030-89902-8_19
4. Y. Guo, Y. Liu, T. Georgiou et al. A review of semantic segmentation using deep neural networks. Int J Multimed Info Retr 7, 87–93 (2018). doi.org/10.1007/s13735-017-0141-z
5. S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan and Y Cao. ReAct: Synergizing Reasoning and Acting in Language Models. (2022) arxiv.org/abs/2210.03629. [Accessed 17/03/2024]
6. A. Karpathy A. NanoGPT. Available from: github.com/karpathy/nanoGPT [Accessed 13/04/2024]
7. B. Hosmer. Inside the Matrix: Visualizing Matrix Multiplication, Attention and Beyond. (25 Sept 2023) Available from: pytorch.org/blog/ inside-the-matrix/ [Accessed 13/03/2024]
8. A. Golden, S. Hsia, F. Sun, B. Acun, B. Hosmer, Y. Leeet al. Generative AI Beyond LLMs: System Implications of Multi-Modal Generation. (2023) Av ailable from: arxiv.org/abs/2312.14385 [Accessed 13/04/2024]
9. M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, B. Catanzaro. Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism. (2020) Available at: arxiv.org/abs/1909.08053 [Accessed 13/04/2024]
10. J. Lew, D. Shah, S. Pati, S. Cattell, M. Zhang et all. Analyzing Machine Learning Workloads Using a Detailed GPU Simulator (ISPASS 2019) Available at: arxiv.org/abs/1811.08933 [Accessed 13/04/2024]
11. T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal et all. Language Models are Few-Shot Learners. // In Proc. of 34th Conf. on Neural Information Processing Systems (Dec 2020), Vancouver, Canada. P. 1877-1901. Doi: 10.5555/3495724.3495883

Одержано: 05.04.2024
Внутрішня рецензія отримана: 14.04.2024
Зовнішня рецензія отримана: 20.04.2024

Про авторів:

Рагозін Дмитро Васильович,
кандидат технічних наук,
старший науковий співробітник.
http://orcid.org/0000-0002-8891-7002.

Дорошенко Анатолій Юхимович, доктор фізико-математичних наук, професор,
завідувач відділу теорії комп'ютерних обчислень
http://orcid.org/0000-0002-8435-1451

Місце роботи авторів:

Інститут програмних систем
НАН України,
03187, м. Київ-187,
проспект Академіка Глушкова, 40.

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»,
проспект Перемоги 37
E-mail: Dmytro.Rahozin@gmail.com

УДК 004.942 http://doi.org/10.15407/pp2024.02-03.045

В.П. Сизоненко

МОДИФІКОВАНА МОДЕЛЬ АГРЕГОВАНОЇ МЕРТВОЇ ЗОНИ НА ПРИКЛАДАХ ПЕРЕНОСУ РАДІОНУКЛІДІВ У ПРИРОДНИХ ГІДРОДИНАМІЧНИХ СИСТЕМАХ

У роботі представлено результати, отримані під час детального дослідження моделі агрегованої мертвої зони, призначеної для опису поздовжнього перенесення і розсіювання розчинених речовин у русловому потоці. Ця модель ґрунтується на новому підході до опису адвекції та дисперсії, який дає змогу адекватно відтворювати спостережувані у природних гідродинамічних системах концентрації розчинених речовин із високим ступенем точності. Замість того, щоб моделювати концентрацію розчиненої речовини безперервно як за відстанню, так і за часом уздовж водотоку, модель агрегованої мертвої зони використовує підхід "чорної скриньки" і розглядає концентрацію на виході камери (з агрегованої мертвої зони) як функцію концентрації на вході камери та поточного часу. Такий підхід значно скорочує час обчислень і зменшує вимоги до обсягу вихідних і граничних даних. Наведено математичний апарат розширеної моделі агрегованої мертвої зони, призначеної для аналізу транспортування неконсервативного радіоактивного забруднення в реальних водних об'єктах, з урахуванням можливої взаємодії радіонукліда зі зваженими намулами і шаром донних відкладень. Рівняння запропонованої моделі складають систему звичайних диференційних рівнянь із запізнілим аргументом. Показано результати моделювання поширення ^{3}H в результаті викидів від 14 ядерних реакторів у русловій ділянці р. Луара протягом півроку із щогодинною дискретністю. Наведено результати моделювання розповсюдження різких викидів ^{90}Sr у Київському водосховищі, що мали місце в 1999 р. внаслідок Чорнобильської катастрофи. Моделювання виконано з добовою дискретністю. Проведено порівняння отриманих модельних значень концентрацій радіонуклідів і даних вимірювань. Запропонована модель має порівняльну простоту, значно менші вимоги до кількості початкових і граничних даних, дуже короткий час, необхідний для проведення розрахунків.

Ключові слова: модель, агрегована мертва зона, адвекція, дисперсія, радіонукліди, річка, водосховище.

V.P. Sizonenko

MODIFIED MODEL OF THE AGGREGATED DEAD ZONE ON EXAMPLES OF RADIONUCLIDE TRANSFER IN NATURAL HYDRODYNAMIC SYSTEMS

This paper presents the results obtained during a detailed study of the aggregate dead zone model designed to describe the longitudinal transport and dispersion of dissolved substances in a channel flow. This model is based on a new approach to the description of advection and dispersion, which allows to adequately reproduce the concentrations of solutes observed in natural hydrodynamic systems with a high degree of accuracy. Instead of modelling the dissolved solute concentration continuously in both distance and time along the watercourse, the aggregate dead zone model uses a black box approach and considers the concentration at the chamber outlet (from the aggregate dead zone) as a function of the concentration at the chamber inlet and the current time. This approach significantly reduces the computational time and reduces the requirements for the amount of initial and boundary data. The mathematical apparatus of the extended model of the aggregated dead zone is presented, designed to analyse the transport of non-conservative radioactive contamination in real water bodies, taking into account the possible interaction of the radionuclide with suspended sediments and a layer of bottom sediments. The equations of the proposed model are a system of ordinary differential equations with a delayed argument. The results of modelling the distribution of ^{3}H as a result of releases from 14 nuclear reactors in the Russian section of the Loire River for six months with an hourly discreteness are presented. The results of modelling the propagation of sudden ^{90}Sr releases in the Kyiv reservoir, which occurred in 1999 as a result of the Chornobyl disaster, are presented. The modelling was carried out with a daily discreteness. A comparison of the obtained model values of radionuclide concentrations and measurement data was carried out. The proposed model has a comparative simplicity, much lower requirements for the amount of initial and boundary data, and very little time required for calculations.

Keywords: model, aggregated dead zone, advection, dispersion, radionuclides, river, reservoir.

Вступ

Моделювання розсіювання радіонуклідів у водних об'єктах є частиною більш загальної проблеми моделювання якості води, включно з моделюванням гідродинамічних гідравлічних процесів і моделюванням перенесення забруднень, зумовленого також хімічними та біологічними процесами [1, 2].

Традиційний підхід до вивчення процесів переносу в річках полягає в застосуванні адвекційно-дисперсійного рівняння (**A**dvection **D**ispersion **E**quation - ADE). Теорія цього підходу була спочатку розроблена Тейлором для потоку в трубі, а згодом була поширена на канали Елдером і Фічером [3-5] і дає математичний опис у термінах рівняння дифузії Фіка. До теперішнього часу моделі, що описують міграцію радіонуклідів у річкових потоках, використовують саме такий підхід [2, 6 - 10].

Із кінця 1960-х років, однак, було встановлено, що адвекційно-дисперсійні рівняння не описують спостережувані профілі розсіювання. Найважливішим із них є передбачення гаусових просторових розподілів концентрації, тоді як спостережувані розподіли концентрації незмінно асиметричні з довгими хвостами [11 - 15].

У природних каналах або інших гідродинамічних системах, наявність нерівностей у руслі або на березі каналів, чи-то наявність будь-яких застійних областей була відома як периферійні мертві зони. Загалом було помічено, що мертві зони можуть бути виявлені в різних частинах природних потоків. Наприклад, у руслі або на березі, як величезні перешкоди, дерева, дерев'яні уламки, скельні породи та форми русла. Таким чином, забруднення може затримуватися і повільно вивільнятися із зазначених зон. Було встановлено, що існування цих зон породжує довгий хвіст спостережуваного розподілу концентрації, а також неоднорідність дисперсійної дії в різних частинах поля течії. Надалі було зроблено припущення, що агрегований ефект багатьох мертвих зон у межах розглянутої ділянки можна вважати еквівалентним ефекту однієї великої мертвої зони. Слід зазначити, що "мертві зони", які розглядаються в рамках цієї моделі, складаються не тільки з фізично ідентифікованих застійних ділянок потоку, пов'язаних із нерегулярностями в руслі, а радше трактуються ширше й охоплюють будь-які особливості. До таких особливостей належать перехідні турбулентні вихори і вторинні циркуляції в різних просторових і часових масштабах.

Було зроблено різні спроби модифікувати теорію, або зробити її більш придатною до спостережуваного дисперсійного процесу [16 - 19].

ADZ модель

Агрегована модель мертвої зони (**A**ggregated **D**ead **Z**one - ADZ) - транспортна модель, яку віднесено до категорії механістичних моделей, що ґрунтуються на даних, і яка позбавлена недоліків ADE. У припущеннях моделі ADZ умови адвекції та дисперсії розділені на окремі процеси — спочатку адвекція, а потім миттєва дисперсія. У моделі ADZ будь-яка гідравлічна система представлена шляхом поділу на множину взаємопов'язаних ділянок. Ця модель передбачає, що розчинена речовина переноситься через усю ділянку за рахунок поршневого потоку (тобто перенесення без дисперсії), після чого вона проходить через єдину камеру моментального і повного змішання, що має агрегований ефект усіх мертвих зон у межах ділянки (тобто дисперсія без перенесення) і зазнає відповідних фізичних, хімічних і біологічних змін. Тож отримуємо камерну модель з агрегованою мертвою зоною із запізненням - часом перенесення [20 - 24][1].

Модель ADZ передбачає стійкий рух течії, за якого витрата Q та об'єм V є сталими. Однак, з огляду на те, що гідрологічні дані зазвичай відбирають через регулярні інтервали часу, у межах яких їх вважають

[1] Аналогічні припущення було свого часу незалежно сформульовані в [25].

сталими, можна вважати, що припущення моделі ADZ виконуються на кожному з цих інтервалів (інтервали сталості даних є загальноприйнятою практикою під час дослідження водних об'єктів), а кінцеві результати одного інтервалу будуть початковими для наступного. Замість того, щоб моделювати концентрацію розчиненої речовини безперервно як за відстанню, так і за часом уздовж водотоку, як у випадку моделі ADE, модель ADZ використовує підхід "чорної скриньки" і розглядає концентрацію на виході камери (в агрегованій мертвій зоні) як функцію концентрації на вході камери та поточного часу. Це значно скорочує час обчислень і зменшує вимоги до обсягу вихідних і граничних даних.

UNDBE модель

Запропонована модель UNDBE (**UN**-**D**riking **BE**aconman) становить розширення моделі ADZ для реальних водних об'єктів із неконсервативним радіоактивним забрудненням. З огляду на припущення моделі ADZ, розширення моделі, яке передбачає врахування радіоактивного розпаду і взаємодії радіонукліда зі зваженими намулами і донними відкладеннями, система рівнянь моделі UNDBE для окремої камери на інтервалі сталості даних набуває вигляду системи диференціальних рівнянь із запізнілим аргументом T_R [29]. Час транспортування T_R - залежить від швидкості течії та довжини камери. Його динаміка визначається шляхом калібрування гідрологічної моделі водного об'єкта. Для розв'язання диференціальних рівнянь із запізнілим аргументом використовувалась програма RETARD [27][2].

За умови використання запропонованої моделі необхідно визначити сім параметрів - K_s, τ_s , τ_{ds}, K_d, τ_{sb} , τ_{dsb}, n.

K_d, K_s - коефіцієнти розподілу в системі вода-дно і вода- зважені намули відповідно,

τ_s , τ_{ds} - часові значення адсорбційних і десорбційних процесів для системи вода - зважені намули,

τ_{sb}, τ_{dsb} - часові значення адсорбційних і десорбційних процесів для системи вода – шар донних відкладень, що бере участь в обміні з розчином;

1/n - частина об'єму (ADZ), яка перебуває біля витоку.

Ці параметри визначаються методом параметричної ідентифікації за даними вимірювань.

Слід зазначити, що параметри, отримані методом параметричної ідентифікації за даними вимірювань, є діючими, а не отриманими шляхом аналізів у лабораторних умовах, хоча, як показує практика, вони досить близькі.

Якщо потрібно моделювати поширення забруднення, що дуже слабко взаємодіє зі звислими намулами (у цьому випадку не суттєві концентрації звислих намулів та концентрації забруднення на звислих намулах)

Отже, в цьому випадку в процесі використання запропонованої моделі знадобиться визначення лише чотирьох параметрів – K_d, τ_{sb}, τ_{dsb}, n. До таких забруднень належать ^{90}Sr і ^{3}H.

Необхідно підкреслити, що в рівняннях моделі концентрації є середніми не в об'ємі всієї камери, а лише середніми в 1/n частині об'єму ADZ, яка перебуває біля витоку. Таким чином, модель UNDBE визначає концентрації в конкретному місці - біля витоку камери і не дає можливості розраховувати концентрації в проміжних областях камери. Тому при поділі водойми на камери витоки камер доцільно розташовувати так, щоб вони збігалися з найбільш важливими для аналізу місцями водойми, які цікавлять. За наявності достатньо великих притоків, що мають істотний вплив на процеси водообміну, розбиття на камери слід здійснювати так, щоб початок камери збігався з місцем впадіння такого притоку.

Під час проведення параметричної ідентифікації за даними вимірювань застосовано програму, яка реалізує метод RALG (Пошук екстремуму не-диференційованої багатовимірної функції. Розробка Інституту Кібернетики ім. В.М. Глушкова) [28, 29].

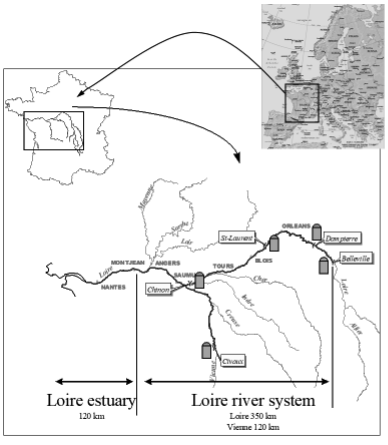
[2] Програму було модифіковано, що дозволило прискорити розрахунки в 1200 разів.

Моделювання поширення ^{3}H

Комп'ютерна реалізація моделі UNDBE (Реалізована на мові FORTRAN.) використовувалась для моделювання перенесення тритію (^{3}H) у руслі річки Луари (Франція) у процесі виконання міжнародного проєкту EMRAS [30]. Дані вимірювань було надано DIREN Centre (Direction Régionale de l'Environnement Centre - Управління Регіонального Центру Навколишнього Середовища) і EDF (Electricité de France - Електроенергетична компанія Франції).

У басейні р. Луари розташовано п'ять атомних електростанцій (Рис. 1), що мають 14 атомних реакторів, у результаті функціонування яких відбуваються різкі викиди ^{3}H у воду.

Потрібно було визначити динаміку концентрацій ^{3}H в 11 пунктах уздовж русла р. Луари (Beaulieu, Gien, Ouzouer, Orléans, Beaugency, Nouan, Tours, La Chapelle, Bertignolles, Angers, MontJean) з годинною дискретністю, протягом півроку з 1 липня по 31 грудня 1999 р.



	Abscissa (m)	Name
	532820	Belleville
1	532880	RejetBelleville
2	544780	ChatillonSurLoire
3	559600	Gien
4	569100	RejetDampierre
5	575030	Ouzouer
6	612660	SaintDenisHotel
7	627140	Orleans
8	633530	Orleans
9	637180	SaintPryve
10	638340	SaintMesmin
11	643420	Mareau
12	651010	MeungSurLoire
13	658830	Beaugency
14	668360	RejetStLaurent
15	672290	Nouan
16	691100	ChausseeSaintVictor
17	709770	Onzain
18	727610	Amboise
19	747400	Tours
20	750990	Tours
21	768190	Cher
22	774770	Langeais
23	790740	LaChapelle
24	793410	Indre
25	796260	Avoine
26	797280	RejetChinon
27	800210	Bertignolles
28	804030	Vienne
29	831010	Gennes
30	856140	Angers
31	856560	PontsDeCe
32	864010	Maine
33	885370	Montjean

Рис. 1. Розташування атомних електростанцій у басейні р. Луари

Ділянка річки протяжністю 350 км описувалася шляхом завдання геометрії 349 поперечних перерізів русла.

Витрати води в семи притоках, що відповідають реальним витратам води для 1999 р., і дані про реальні викиди ^{3}H блоками атомних електростанцій були задані з годинною дискретністю (годинний інтервал сталості даних).

Значення витрат води, що спостерігалися в межах часового інтервалу моделювання, змінювалися від 157 до 5055 м^3/сек (пункт Angers).

Ухил дна р. Луари в межах модельованого відрізка русла змінюється в межах 1,0 - 5,8 м/км. Ця ділянка річки залежно від водності характеризується швидкостями течії від 0,3 м/сек до 2,0 м/сек.

Необхідно було також врахувати, що русло р. Луари у 18 місцях перегороджено підпірними водозливними греблями, що мають істотний вплив на гідрологію р. Луари.

Для використання моделі 350-ти кілометрову ділянку р. Луари було розбито на 33 послідовні камери. Межі камер розташовувалися в таких пунктах:

- початок і кінець модельованого відрізка річки;
- місця, де необхідно було визначити значення концентрацій;
- місця розташування підпірних гребель;
- місця впадання притоків;
- пункти розташування ядерних станцій (місця надходження забруднення).

Таке розбиття визначило довжину найкоротшої камери - 420 м, а найдовшої - 37,6 км (Рис. 1).

У моделюванні перенесення забруднення враховувалась відсутність взаємодії ^{3}H із звислими намулами.

За гідрологічну модель р. Луари для визначення поточних об'ємів камер використовувалася одновимірна стаціонарна модель [31].

У процесі визначення часу транспортування за формулою $T_R = W/Q$ враховувалося, що не весь об'єм камери V бере участь у зовнішньому водообміні. Під час виконання обчислень, виходячи з найкращого збігу модельних і виміряних піків концентрацій забруднення, було встановлено середню для всіх і-камер залежність для об'єму камери, що бере участь у водообміні, від поточної витрати води в камері

$$W_i(t) = V_i(t)(1 - k_3 EXP(-k_4 Q_i(t)/Q_{0i}))$$

тут

W_i – об'єм і-тої камери, що бере участь у водообміні;

V_i – поточний об'єм камери;

k_3, k_4 – константи;

$Q_i(t)$ – поточні витрати води в камері;
Q_{0i} – мінімальна із заданих витрата води в камері.

Для р. Луари найкращі результати можна отримати у разі $k_3 = 0{,}8794$, $k_4 = 0{,}1647$.

Розрахунок гідрології р. Луари (33 камери 350-кілометрової ділянки, для кожної години шестимісячного інтервалу моделювання) потребує 20 хвилин часу ЕОМ з процесором Intel Core i5-9600K. Розв'язання задачі перенесення ^{3}H за допомогою комп’ютерної реалізації запропонованої моделі UNDBE здійснюється за 8 секунд. Оскільки гідрологічний режим річки не залежить від транспортування ^{3}H, то розрахунок водного режиму був виконаний одноразово для всього діапазону можливих витрат води, а результати використовувалися як масив вихідних даних для багаторазових прорахунків у вирішенні задачі ідентифікації параметрів перенесення ^{3}H. Це значно прискорило визначення оптимальних значень параметрів.

На діаграмі (Рис. 2) відображено частину результатів моделювання динаміки ^{3}H у розчині (лінія) в створі Анжер у порівнянні з даними вимірювань (крапки) з 30% похибкою.

Порівняння результатів моделювання (в проекті EMRAS) показало, що запропонований метод, який використовує модель UNDBE у поєднанні з одновимірною стаціонарною гідрологічною моделлю р. Луари, забезпечує аналогічний, або навіть кращий збіг виміряних і розрахованих концентрацій, ніж методи, засновані на одновимірних і камерних ADE моделях [30].

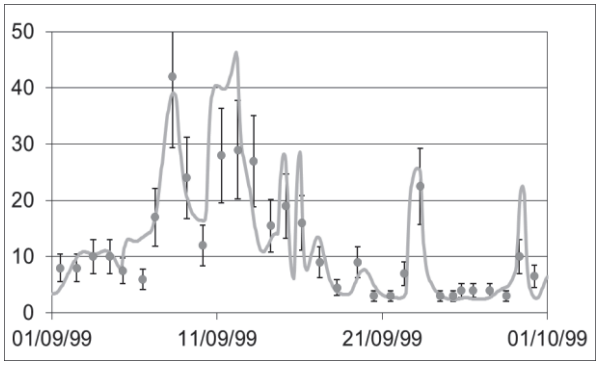


Рис. 2. Зіставлення розрахунків та вимірювань концентрацій ^{3}H у створі ANGERS (Бк/л)

Моделювання поширення ^{90}Sr

Внаслідок Чорнобильської катастрофи виникла проблема моделювання транспортування радіонуклідів каскадом шести дніпровських водосховищ. Один з основних забруднювачів – ^{90}Sr, який за високих рівнів води (під час повеней, або льодових заторів) змивається із забрудненої заплави р. Прип'ять і надходить у північну частину Київського водосховища. Оскільки всі суттєві притоки (включаючи р. Прип'ять) знаходяться в північній частині (на втоку) Київського водосховища, а місцем, де необхідно було розрахувати значення концентрацій, був пункт водозабору – м. Вишгород (виток Київського водосховища), тому водосховище моделювалося однією камерою.

Київське водосховище: довжина 110 км, обсяг 2,7 - 3,9 км^3 (залежно від рівня), чотири притоки.

В свою чергу весь каскад дніпровських водосховищ (Рис. 3) моделювався послідовністю шести камер, оскільки між водосховищами руслові ділянки дуже малі і виток одного водосховища практично є верхів’ям наступного, а головні водозабори розміщені поблизу витоку кожного водосховища.

Враховуючи наявність тільки щодобових вимірювань (добовий інтервал сталості даних), розрахунок вівся з добовою дискретністю.

Поточні об’єми водосховищ визначалися за кривими залежності об’ємів від рівнів (h) води у водосховищах

$$V(h) = n_2h^2 + n_1h + n_0,$$

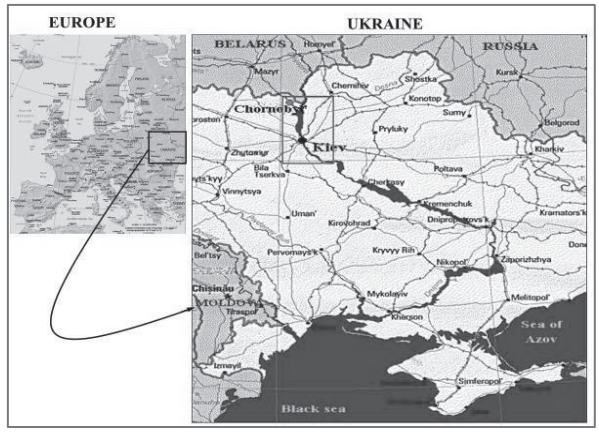


Рис. 3. Каскад дніпровських водосховищ

де n_0, n_1, n_2 – константи, а h – рівень водосховища в Балтійській системі координат [32]. Для Київського водосховища $V(h)$ має вигляд (при розмірності км^3):

$V(h) = (84.2044*h^2 - 16423*h +$
$+ 801987.8)*0.001$.

Детальну інформацію про щодобові витрати води притоків водосховищ, а також дані стосовно щодобових рівнів водосховищ було отримано у Центральній геофізичній обсерваторії імені Бориса Срезневського.

Поточні дані про концентрації ^{90}Sr у розчині по притоках водосховищ та дані про концентрації ^{90}Sr на витоку водосховищ були взяті з бази даних УкрГМІ ДСНС України та НАН України.

Розрахунки проводилися для 1991, 1994 і 1999 років, коли спостерігалося значне збільшення концентрації ^{90}Sr у річці Прип'ять під час високих повеней і льодових заторів [26]. За даними вимірювань 1994 р. було визначено гідрологічні (зокрема, залежність об'єму водосховища, що бере участь у водообміні, від рівня Київського водосховища) і токсикологічні параметри Київського водосховища та, орієнтовно (за відсутності достатньої кількості токсикологічних даних), інших водосховищ. Під час моделювання перенесення забруднення враховувалася відсутність взаємодії ^{90}Sr із звислими намулами.

На Рис. 4 наведено порівняння розрахунків концентрацій розчиненого ^{90}Sr (зелена лінія, ліва шкала графіка) і вимірювань (Різнокольорові точки представляють дані вимірювань розчиненого ^{90}Sr, виконані різними організаціями.) Точність вимірювань становить від 30 до 50% [26] у районі Вишгорода в першому кварталі 1999 року (Бк/м^3). Чорна лінія - розраховані концентрації ^{90}Sr (Бк/кг) у донних відкладеннях (їм відповідає права шкала графіка).

На Рис. 5 показано концентрації розчиненого ^{90}Sr у гирлі Прип'яті (апроксимація вимірювань) і у м. Вишгород (розрахунок).

Графіки демонструють узгодженість розрахунків з результатами вимірювань і адекватність моделі процесам, що мають місце.

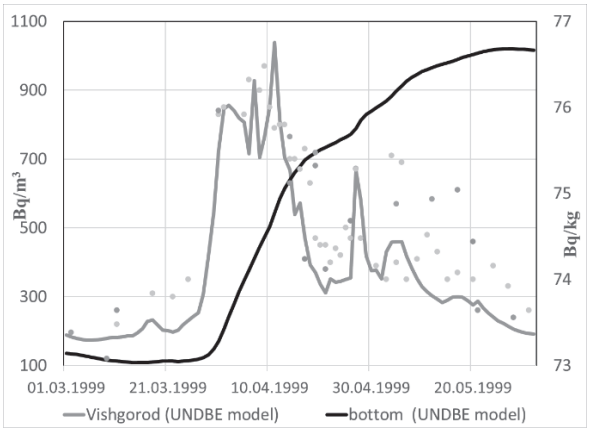


Рис. 4. Концентрація ^{90}Sr в розчині в м. Вишгород (Бк/м^3) і шарі донних відкладень (Бк/кг)

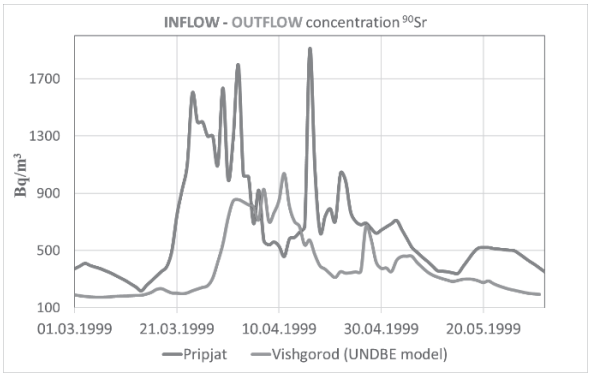


Рис. 5. Концентрація ^{90}Sr в розчині в гирлі Прип'яті і в м. Вышгород (Бк/м^3)

Зсуви пікових забруднень другого графіка відносно першого (Рис. 5) наочно ілюструють вплив параметра T_R (час транспортування забруднення від втоку до витоку) на результати моделювання.

Час розрахунку річного інтервалу для всіх шести водосховищ каскаду менше 0,1 сек.

Висновки

Модель UNDBE на відміну від моделей, що їх традиційно використовують як основу адвекційно-дифузійне рівняння, більш адекватно описує процеси перенесення радіонуклідів у проточних водоймах.

Будучи камерною моделлю, що описується системою звичайних диференціальних рівнянь із запізнілим аргументом, UNDBE, порівняно з 1-2- і 3-х мірними моделями, є простою, має значно менші вимоги до кількості початкових і граничних

даних, дуже короткий час, необхідний для здійснення розрахунків.

Малий час комп'ютерної реалізації забезпечує можливість проведення параметричної ідентифікації та оперативне використання моделі для адаптації до різних типів радіонуклідів в різноманітних водоймах, використання моделі у системах аварійного реагування.

Література

1. Y. Onishi, R. J. Seine, E. M. Amokl, C. E. Cowan, F. L. Thompson, Critical review-radionuclide transport, sediment transport, and water quality mathematical modeling; and radionuclide adsorption/desorption mechanisms. Battelle Pacific Northwest Laboratory, Richland, WA 99352, U.S. Nuclear Regulatory Commission, 1981. http://dx.doi.org/10.2172/10125120
2. J. A. Cunge, F. M. Holly, A. Verwey, Practical aspects of computational river hydraulics, London Pitman Publ, 1980. Accessed: 11.12.2023. https://pdfcoffee.com/practical-aspects-computational-river-hydraulics-pdf-free.html
3. G. I. Taylor, The dispersion of matter in turbulent flow through a pipe, in: Proceedings of the Royal Society A, 1954 223(1155) 446-468. https://doi.org/10.1098/rspa.1954.0130
4. J. W. Elder, The dispersion of marked fluid in turbulent shear flow. Journal of Fluid Mechanics, 1959, 5(04) 544-560. doi:10.1017/s0022112059000374
5. H. B. Fischer, The Mechanics of Dispersion in Natural Streams, in: Journal Hydraulics Division Proceedings, ASCE, 1967, 93(6), 187-216. https://doi.org/10.1061/JYCEAJ.0002136
6. W. Rauch, M. Henze, L. Koncsos, P. Reichert, P. Shanahan, L. Somlyódy, P. Vanrolleghem, River water quality modelling: I. state of the art, in: Water Science and Technology 1998 38(11) 237–244. doi.org/10.1016/S0273-1223(98)00660-X
7. L. Monte, P. Boyer, J. E. Brittain, L. Hakanson, S. Lepicard, J. T. Smith, Review and assessment of models for predicting the migration of radionuclides through rivers, in: Journal of Environmental 2005 79(3) 273–296. https://doi.org/10.1016/j.jenvrad.2004.08.002
8. L. Monte, Multi-model approach and evaluation of the uncertainty of model results. Rationale and applications to predict the behaviour of contaminants in the abiotic components of the fresh water environment, in: Ecological Modelling. 2009. 220(12) 1469–1480. doi.org/10.1016/j.ecolmodel.2009.03.022
9. Overview of Hydrological Dispersion Module - HDM of RODOS, RODOS-WG4-TN(99)18, 1999 Accessed: 11.12.2023. https://resy5.ites.kit.edu/RODOS/Documents/Public/HandbookV5/Volume3/RODOS_WG4_TN99_18.pdf
10. Quantification of radionuclide transfer in terrestrial and freshwater environments for radiological assessments. IAEA-TECDOC-1616, VIENNA, 2009. Accessed: 11.12.2023. https://www-pub.iaea.org/MTCD/Publications/PDF/te_1616_web.pdf
11. N. Yotsukura, H. B. Fischer, W. W. Sayer, Measurements of Mixing Characteristics of the Missouri River between Sioux City, Iowa, and Plattsmouth, Nebraska. U.S. Geological Survey Water Supply Paper 1899-G, 1970. Accessed: 11.12.2023. https://pubs.usgs.gov/wsp/1899g/report.pdf
12. T. J. Day, Longitudinal Dispersion in Natural Channels, Water Resources Research, 1975 11(6) 909-918. doi:10.1029/wr011i006p00909
13. K. E. Bencala, R. A. Walters, Simulation of solute transport in a mountain pool-and-riffle stream: a transient storage model in: Water Resources Research, 1983 19(3) 718-724. Accessed: 11.12.2023. https://www.researchgate.net/publication/252641054_Simulation_of_Solute_Transport_in_a_Mountain_Pool-and-Riffle_Stream_A_Transient_Storage_Model
14. C. Legrand-Marcq, H. Laudelout, Longitudinal dispersion in a forest stream, in: Journal of Hydrology, 1985 78(3-4) 317–324. doi:10.1016/0022-1694(85)90109-x
15. R. González-Pinzón, R. Haggerty, M. Dentz, Scaling and predicting solute transport processes in streams, in: Water Resources Research, (2013) 49(7) 4071–4088. https://doi.org/10.1002/wrcr.20280
16. J. Chabokpour, Study of pollution transport through the rivers using aggregated dead zone and hybrid cells in series models, in: International Journal of Environmental Science and Technology, 2020. doi:10.1007/s13762-020-02741-w
17. K. Richardson, P. A. Carling, The hydraulics of a straight bedrock channel: Insights from solute dispersion studies, in: Geomorphology, 2006 82 98–125. doi: 10.1016/j.geomorph.2005.09.022.
18. C. G. Narayan, C. M. Govinda, P. Ojha, Hybrid-Cells-in-Series Model for Solute Transport in a River, in: Journal of Environmental Engineering 2004 130(10) 1198-1209.

doi:10.1061/(ASCE)0733-9372(2004)130:10(1198)
19. J. L. Matthew, A. C. Luis, S. Chapra, On the relationship of transient storage and aggregated dead zone models of longitudinal solute transport in streams, in: Water Resources Research, 2000 36(1) 213-224. doi:10.1029/1999WR900265
20. K. J. Beven, P. C. Young, An aggregated mixing zone model of solute transport through porous media, in: Journal of Contaminant Hydrology, 1988 3(2–4) 129-143. https://doi.org/10.1016/0169-7722(88)90028-9
21. S. G. Wallis, P. C. Young, K. J. Beven, Experimental Investigation of the Aggregated Dead Zone Model in: Proceedings of the Institution of Civil Engineers 1989 87(1) PART 2 1-22. https://doi.org/10.1680/iicep.1989.1450
22. M. J. Lees, L. A. Camacho, P. Whitehead, Extension of the QUASAR river quality model to incorporate dead-zone mixing, in: Hydrology and Earth System Sciences, 1998 2(2-3) 353-365. https://doi.org/10.5194/hess-2-353-1998
23. P. C. Young, Identification and estimation of continuous-time hydrological models from discrete-time data, in book: Hydrology: Science and Practice for the 21st Century, Publisher: British Hydrological Society: London. 2004 406-413 Accessed: 11.12.2023. https://www.researchgate.net/publication/266446716_Identification_and_estimation_of_continuous-time_hydrological_models_from_discrete-time_data
24. T. Beer, P. C. Young, Longitudinal Dispersion in Natural Streams, in: Journal of Environmental Engineering, 1983 109(5) 1049-1067. https://doi.org/10.1061/(ASCE)0733-9372(1983)109:5(1049)
25. V. P. Sizonenko, Increasing accuracy of the box model. in: Babovic, V., Larson, L. C. (Eds.), Hydroinformatics '98—Proceedings of the Third International Conference on Hydroinformatics/Copenhagen/Denmark/24–26 August 1998, vol. 1. A. A. Balkema, Rotterdam, pp. 225–230. http://publications.pvandenhof.nl/Paperfiles/Silvis&etal_Hydroinformatics1998.pdf
26. Радиогеоэкология водных объектов зоны влияния аварии на Чернобыльской АЭС: В 2т. / Государственный комитет Украины по гидрометеороглогии, Национальная Академия Наук Украины. – К.: Чернобыльинтеринформ, 1998.
[Radiogeoecology of water bodies in the zone of influence of the Chernobyl NPP accident / State Committee of Ukraine on Hydrometeorology; National Academy of Sciences of Ukraine. Kiev, Chernobylinterinform, 1998.] [In Russian]
27. E. Hairer, S. Norsett, G. Wanner, Solving Ordinary Differential Equations I. Nonstiff Problems. Berlin Heidelberg, 1987.
28. Н. З. Шор, С. И. Стеценко, Квадратичные экстремальные задачи и не дифференцируемая оптимизация. Киев: Наукова Думка, 1989.
[N. Z. Shor, S. I. Stetsenko, Quadratic extremal problems and non-different optimization. Kiev: Naukova Dumka, 1989.] [In Russian]
29. Bielyh T., Sizonenko V., Application of the UNDBE Model in Combination with the RALG Procedure, Cybernetics and Computer Technologies. 2022, No.4(12). 82-92. DOI:10.34229/2707-451X.22.4.0]
30. Testing of Models for Predicting the Behaviour of Radionuclides, in Freshwater Systems and Coastal Areas. Report of the Aquatic Working Group of EMRAS Theme 1. IAEA-tecdoc-1678 International Atomic Energy Agency, Vienna, Austria (2012) [cited 30 Dec 2023]. http://www-ns.iaea.org/downloads/rw/projects/emras/final-reports/aquatic-tecdoc-final.pdf
31. А. В. Караушев, Речная гидравлика. Л.: Гидрометеоиздат, 1969.
[A. V. Karaushev, River hydraulics. L.: Gidrometeoizdat, 1969.] [In Russian]
32. М. С. Каганер, Гидрометеорологический режим озёр и водохранилищ СССР. Л.: Гидрометеоиздат, 1976.
[M. S. Kaganer Hydrometeorological regime of lakes and reservoirs of the USSR. L.: Gidrometeoizdat, 1976. 348 P.] [In Russian]

Одержано: 08.04.2024
Внутрішня рецензія отримана: 21.04.2024
Зовнішня рецензія отримана: 28.04.2024

Про автора:

[1]*Сизоненко Володимир Петрович*,
кандидат технічних наук,
старший науковий співробітник.
http://orcid.org/0009-0007-2274-5838.

Місце роботи автора:

[1]Інститут програмних систем
НАН України, Київ
тел. +38-097-491-99-55
E-mail: jasizyj@yahoo.com

УДК 519.168; 519.854.3 http://doi.org/10.15407/pp2024.02-03.053

В.О. Васянін

ЗАДАЧА РОЗПОДІЛУ І ОБ'ЄДНАННЯ ДИСКРЕТНИХ ПОТОКІВ КОРЕСПОНДЕНЦІЙ В ОКРЕМИХ ЗОНАХ ІЄРАРХІЧНОЇ КОМУНІКАЦІЙНОЇ МЕРЕЖІ

Стаття присвячена дослідженню підзадачі розподілу і об'єднання потоків кореспонденцій в окремих зонах магістральної мережі, яка виникає під час розв’язання загальної задачі оптимізації ієрархічної структури багатопродуктової комунікаційної мережі з дискретними потоками і параметрами. У багатопродуктовій мережі кожен вузол може обмінюватися кореспонденціями (продуктами, товарами, вантажами, повідомленнями) з іншими вузлами. Кореспонденція характеризується вузлом-джерелом, вузлом-стоком та величиною, яка для транспортних мереж задається кількістю тарно-штучних вантажів в упаковці уніфікованого розміру, а для мереж передачі даних – кількістю байт, кілобайт і т.п. У магістральній мережі всі кореспонденції транспортуються у транспортних засобах у транспортних блоках заданого розміру (ємності, обсягу), або передаються каналами зв’язку. Розмір транспортного блоку вимірюється кількістю одиниць кореспонденцій, що вміщуються в нього (наприклад, 40 тарно-штучних вантажів, 100 гігабайт). Усі магістральні вузли є сортувальними центрами, в яких кореспонденції спочатку сортуються за адресами (вузлами) призначення, а потім пакуються як збірні кореспонденції в транспортні блоки. Оскільки величина окремих кореспонденцій значно менша за розмір транспортного блоку, вони у процесі сортування можуть кілька разів і в різних вузлах об'єднуватися (упаковуватися) з кореспонденціями, що мають інші адреси призначення. У мережі виділено три рівня ієрархії – магістральний, зональний і внутрішній та чотири типи вузлів – магістральні вузли першого, другого і третього типу, що утворюють магістральний і зональний рівні мережі і вузли четвертого типу, підлеглі кожному магістральному вузлу і утворюють внутрішні рівні мережі. Типи вузлів відрізняються один від одного функціональними можливостями. Основним завданням дослідження є розробка математичної моделі і алгоритмів вирішення підзадачі оптимізації розподілу і об'єднанню (сортуванню) потоків кореспонденцій на зональних рівнях мережі. Показано, що вона може бути сформульована як задача лінійного програмування з блоковою структурою обмежень і для її вирішення може бути застосовано метод декомпозиції Данцига-Вулфа й інші методи цілочислового програмування. Для вирішення задачі на реальних мережах запропоновані наближені алгоритми, що базуються на побудові найкоротших шляхів.

Ключові слова: ієрархічні комунікаційні мережі, дискретні потоки і параметри, задачі оптимізації, комп'ютерне моделювання

V. Vasyanin

THE PROBLEM OF DISTRIBUTION AND MERGING OF DISCRETE CORRESPONDENCE FLOWS IN INDIVIDUAL ZONES OF A HIERARCHICAL COMMUNICATION NETWORK

The article is devoted to the study of the subproblem of distribution and merging of correspondence flows in separate zones of the backbone network, which arises when solving the general problem of optimizing the hierarchical structure of a multicommodity communication network with discrete flows and parameters. In a multicommodity network, each node can exchange correspondence (products, goods, cargo, messages) with other nodes. Correspondence is characterized by a source node, a drain node and a value, which for transport networks is given by the number of packaged goods in a package of a unified size, and for data transmission networks – by the number of bytes, kilobytes, etc. In the backbone network, all correspondence is transported in vehicles in transport units of a given size (capacity, volume) or transmitted via communication channels. The size of a transport block is measured by the number of units of correspondence that fit into it (for example, 40 packaged goods, 100 gigabytes). All trunk nodes are sorting centers in which correspondence is first sorted by destination addresses (nodes) and then packed as consolidated correspondence into transport blocks. Since the size of individual correspondence is much smaller than the size of the transport block, they can be combined (packed) with correspondence with other destination addresses several times and in different nodes during sorting. There are three levels of hierarchy in the network – backbone,zonal and internal

and four types of nodes – trunk nodes of the first, second and third types, forming the backbone and zonal levels of the network and nodes of the fourth type, which are subordinate to each trunk node and form internal levels of the network. Node types differ from each other in functionality.
The main task of the study is to develop a mathematical model and algorithms for solving the subproblem of optimizing the distribution and merging (sorting) of correspondence flows at the zonal levels of the network. It is shown that it can be formulated as a linear programming problem with a block structure of constraints and the Danzig-Wolf decomposition method and other methods of integer programming can be used to solve it. To solve the problem on real networks, approximate algorithms based on the construction of the shortest paths are proposed.
Keywords: hierarchical communication networks, discrete flows and parameters, optimization problems, computer modeling

Вступ

У більшості випадків існуючі і проєктовані територіально-розподілені комунікаційні мережі – транспортні, інформаційно-обчислювальні, паливно-енергетичні, поштові, телеграфні, телефонні тощо. є багаторівневими і складаються із децентралізованої розподіленої мережі (магістральної) і низових фрагментарних мереж (зональних і внутрішніх) на нижніх рівнях ієрархії. У цій роботі розглядаються багатопродуктові транспортні мережі і мережі передачі даних, для яких характерна наявність множини джерел і стоків потоків кореспонденцій (продуктів або вимог). Під кореспонденцією розуміється пара різних вузлів мережі, між якими є спрямований (адресний) дискретний потік елементів (наприклад, неділимих вантажів уніфікованого розміру, біт або символів) заданої величини. У багатопродуктовій мережі усі потоки кореспонденцій підлягають одночасній передачі з джерел у стоки. У загальному випадку на мережі може бути задана деяка множина видів (категорій) кореспонденцій, що відрізняються вагою, габаритами і іншими характеристиками, але мають загальні джерела і стоки.

У мережі виділено три рівні ієрархії – магістральний, зональний, внутрішній та чотири типи вузлів – вузли першого, другого, третього та четвертого типів. Вузли першого, другого і третього типу, що знаходяться на транспортних магістралях транспортної мережі або мережі передачі даних, а також - з'єднують їх ділянки маршрутів транспортних засобів або каналів зв'язку, становлять магістральну мережу. Усі магістральні вузли мають зони обслуговування, які утворюють зональні рівні магістральної мережі. Вузли четвертого типу знаходяться у внутрішній зоні обслуговування будь-якого магістрального вузла і разом із ним утворюють внутрішню мережу.

На рис. 1 показані фрагменти ієрархічної мережі, де i, j, k – магістральні вузли зі своїми магістральними зонами обслуговування (ЗОВ), m – вузли доставки та збору кореспонденцій у внутрішній зоні обслуговування кожного магістрального вузла (внутрішні мережі).

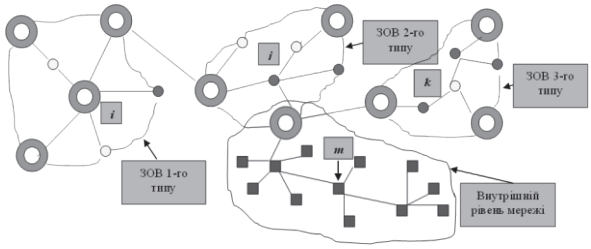


Рис. 1. Фрагменти ієрархічної мережі

Магістральні вузли різного типу різняться між собою функціональними можливостями, рівнем технічної оснащеності, кількістю обслуговуючого персоналу та ін. Деякі з них можуть сортувати потоки до всіх магістральних вузлів, інші – тільки до магістральних вузлів у зоні свого обслуговування. В окремих магістральних вузлах може бути заборонено сортування транзитних потоків кореспонденцій та обробка транзитних потоків транспортних блоків. У вузлах четвертого типу потоки не сортуються, а безпосередньо вирушають у відповідний магістральний вузол.

Структура магістральної мережі може бути відома, наприклад, спочатку задана досвідченими диспетчерами транспортної мережі, адміністраторами мережі передачі даних або визначена в резуль-

таті розв'язання задачі вибору структури мережі.

У [1] розглядається узагальнена задача упаковки та розподілу потоків кореспонденцій в ієрархічній мережі, розв'язання якої здійснюється в кілька етапів. На першому етапі розв'язується задача вибору ієрархічної структури магістральної мережі та схеми сортування кореспонденцій у вузлах мережі та пакування їх у транспортні блоки [2]. На другому етапі постає задача розподілу та маршрутизації потоків транспортних блоків зі збірними кореспонденціями, які були сформовані у процесі розв'язання першої задачі [3]. Під збірними кореспонденціями розуміються об'єднані в один транспортний блок тарно-штучні вантажі або повідомлення (потоки даних) з різними адресами призначення, які можуть не збігатися з адресою призначення транспортного блоку. Збірні кореспонденції утворюються для концентрації потоків у вузлах і ділянках маршрутів і максимального скорочення кількості транспортних блоків, необхідних для їх упаковки та транспортування або передачі у магістральній мережі.

Розв'язання задачі оптимізації структури мережі дає можливість отримати схему сортування потоків у вузлах мережі і витрати на їх сортування, а також витрати на навантаження-розвантаження вихідних і вхідних транспортних блоків.

У задачі вибору ієрархічної структури магістральної мережі виникає підзадача розподілу і об'єднання потоків кореспонденцій із вузлів другого і третього типу в зональних мережах. Згідно з принципами зонально-вузлового сортування вихідні потоки кореспонденцій з вузлів другого і третього типу розсортовуються у вузли першого типу, що лежать на межі зон їх обслуговування. Аналогічно у відповідних вузлах першого типу обробляються і потоки, що входять у вузли другого і третього типу, які надходять від вузлів з інших зон обслуговування.

У статті розглядається математична модель підзадачі розподілу і об'єднання потоків кореспонденцій із вузлів другого і третього типу в зональних мережах і алгоритми її вирішення. Показано, що структурні особливості такої підзадачі дозволяють використати для її вирішення метод декомпозиції Данцига-Вулфа [4] і отримати нижні оцінки розв'язку. Враховуючи, що реальні мережеві структури завжди функціонують в умовах невизначеності, дії випадкових чинників, дефіциту ресурсів, що динамічно змінюються з часом, а також за недостатньо точної початкової інформації, для практичного вирішення поставленої підзадачі запропоновані наближені алгоритми, що ґрунтуються на методах побудови найкоротших шляхів.

1. Постановка задачі

Нехай $G(N,P)$ – ієрархічна магістральна мережа з множиною неорієнтованих топологічних дуг P, $p=\left|P\right|$, множиною вузлів N, $n=\left|N\right|$ і $N=N_1\cup N_2\cup N_3$, де N_1,N_2,N_3 – множини вузлів першого, другого і третього типу відповідно. Під топологічною дугою розуміється фізичний відрізок лінії зв'язку, що сполучає два будь-які вузли з множини N так, що між даними вузлами на цьому відрізку немає більше жодного вузла з N. Вузли мережі відповідають пунктам відправлення, отримання і перевантаження потоків. На мережі задана цілочислова матриця потоків $A=\left\| a_{ij} \right\|_{n\times n}$, у якій потоки a_{ii} $i=\overline{1,n}$ представляють внутрішні потоки між вузлами четвертого типу в зоні обслуговування i-го вузла. Вони не підлягають розподілу по магістральній мережі, але повинні враховуватися у процесі розрахунку витрат на сортування дискретних потоків у вузлах першого, другого і третього типу (передбачається, що число вузлів четвертого типу у внутрішній зоні кожного i-го вузла першого, другого і третього типу відомо). Потоки a_{ij} з джерел i у стоки j, $i,j=\overline{1,n}$ повинні транспортуватися в транспортних блоках (контейнерах) об'єму $\omega >> a_{ij}$ і кожен потік може бути упакований у транспортний блок тільки цілком.

Перетворення потокової матриці A повинно враховувати принципи напряму потоків для вузлів другого і третього типу, згідно з якими вихідний потік із цих вузлів спрямовується (розсортовується) у вузли першого типу, що лежать на межі зон їх обслуговування. Нехай $Y = \| y_\xi \|_n$ – вектор, в якому підсумовуються додаткові об'єми обробки від перетворених потоків вузлів другого і третього типів (спочатку $Y = 0$); Z_i – зона обслуговування (ЗО), побудована на мережі G для i-го вузла; l, k – вузли першого типу в ЗО j-го і i-го вузлів, обслуговуючі відповідно $j \in N_2 \cup N_3$, $i \in N_2 \cup N_3$; S – множина пар ij, що кореспондуються. Тоді змістовна постановка задачі розподілу і упаковки потоків для вузлів другого і третього типу в зональних мережах полягає в перетворенні матриці A. Виконується наступним чином:

1. $\forall ij \in S$, $i, j \in N_1$ відповідні потоки a_{ij} не змінюються;
2. $\forall ij \in S$, $i \in N_1$, $j \in N_2 \cup N_3$, $l \neq i$ виконати:

$a_{il} \leftarrow a_{il} + a_{ij}$, $a_{lj} \leftarrow a_{lj} + a_{ij}$, $y_l \leftarrow y_l + a_{ij}$, $a_{ij} \leftarrow 0$;

3. $\forall ij \in S$, $i \in N_2 \cup N_3$, $j \in N_1$, $k \neq j$ виконати:

$a_{ik} \leftarrow a_{ik} + a_{ij}$, $a_{kj} \leftarrow a_{kj} + a_{ij}$, $y_k \leftarrow y_k + a_{ij}$, $a_{ij} \leftarrow 0$;

4. $\forall ij \in S$, $i, j \in N_2 \cup N_3$, $k \neq l$ виконати:

$a_{ik} \leftarrow a_{ik} + a_{ij}$, $a_{kl} \leftarrow a_{kl} + a_{ij}$, $a_{lj} \leftarrow a_{lj} + a_{ij}$, $y_k \leftarrow y_k + a_{ij}$, $y_l \leftarrow y_l + a_{ij}$, $a_{ij} \leftarrow 0$;

5. $\forall ij \in S$, $i, j \in N_2 \cup N_3$, $k = l$, $j \notin Z_i$ виконати п. 3;
6. $\forall ij \in S$, $i, j \in N_2 \cup N_3$, $k = l$, $j \in Z_i$ перетворення a_{ij} не виконується.

Як видно з постановки задачі для усіх кореспонденцій, потоки яких підлягають операції перетворення, необхідно вказати вузли k і l. Покажемо, що задача визначення цих вузлів може бути зведена до задачі лінійного програмування без урахування обмежень на пропускні здібності дуг.

2. Математична модель і алгоритми вирішення задачі

Транспортування або передача потоків із вузлів другого і третього типу до обслуговуючих їх вузлів першого типу і навпаки повинна виконуватися транспортними засобами магістрального рівня або магістральними каналами зв'язку, тому для таких потоків обмеження на пропускні здібності дуг будуть враховані в магістральних моделях у процесі розподілу потоків між усіма типами вузлів у мережі. Найбільш суттєвими чинниками, що визначають якість схеми розподілу потоків вузлів другого і третього типу, є витрати на їх транспортування або передачу від вузла-джерела до вузла-споживача і можливості обробки потоків цих вузлів у вузлах першого типу. Оскільки в даній задачі здійснюється тільки адресація потоків з вузлів другого і третього типу в деякі вузли першого типу (що лежать на межі зон), а фактичний розподіл потоків з урахуванням їхніх обсягів та інших обмежень здійснюється при вирішенні задачі розподілу потоків на магістральному рівні, то транспортні витрати можна вважати не залежними від обсягу потоків і прийняти лінійними від відстані. Приймемо також, що витрати на додаткову обробку (пересортовування) одиниці потоку з вузлів другого і третього типу в обслуговуючих їх вузлах першого типу однакові в усіх вузлах.

Нехай $R = \| r_{ij} \|_{n \times n}$ – топологічна матриця, де

$$r_{ij} = \begin{cases} \text{довжині дуги, якщо } p_{ij} \in P, \\ \infty, \text{ якщо дуги } p_{ij} \text{ не існує.} \end{cases}$$

Розглянемо мережу $G_D(N, P_D)$ отриману з $G(N, P)$ таким чином. Визначимо на мережі G_D використовуючи матрицю R найкоротші шляхи від усіх вузлів $i \in N_2 \cup N_3$ до вузлів першого типу, що знаходяться в ЗО i-го вузла і найкоротші шляхи між усіма вузлами першого типу. В результаті отримаємо матрицю

$D = \| d_{ij} \|_{n \times n}$ в якій елементи нерівні нескінченності представляють множину дуг P_D мережі G_D.

Твердження 1. Нехай виконується одна з умов :

$$(i, j \in N_1 \wedge d_{ij} \neq \infty) \vee ((i \in N_1, j \in N_2 \cup N_3) \wedge$$
$$\wedge d_{ij} \neq \infty) \vee ((i \in N_2 \cup N_3, j \in N_1) \wedge d_{ij} \neq \infty \quad (1)$$
$$(i \in N_1, j \in N_2 \cup N_3) \wedge d_{ij} = \infty \quad (2)$$
$$(i \in N_2 \cup N_3, j \in N_1) \wedge d_{ij} = \infty \quad (3)$$
$$(i, j \in N_2 \cup N_3) \wedge d_{ij} = \infty \wedge j \notin Z_i \quad (4)$$

тоді: 1) найкоротший шлях між будь-якими вузлами, для яких виконується умова (1), складається з однієї дуги і співпадає з найкоротшим шляхом між цими вузлами, побудованими на мережі G; 2) найкоротший шлях між вузлами, для яких виконується (2) або (3) складається з двох дуг; 3) найкоротший шлях між вузлами, що задовольняють умові (4) складається з двох або трьох дуг.

Доведення. Для доведення 1) покажемо, що довжина будь-якого шляху, побудованого на мережі G_D між вузлами $i, j \in N_1$ і що містить більше за одну дугу не менша, ніж довжина шляху, що складається з однієї дуги. Припустимо, що знайдений найкоротший шлях π_{ij} від i до j, що містить понад одну дугу. Нехай π_{ij} заданий перерахуванням вузлів $\pi_{ij} = (i, k_1, k_2, ..., k_\xi, j)$. Тоді довжина будь-якої дуги (i, k_1), $(k_1, k_2), ..., (k_\xi, j)$ у мережі G_D визначається сумою довжин певної підмножини топологічних дуг мережі G. Якщо усі топологічні дуги, що становлять відрізки $(i, k_1), ..., (k_\xi, j)$ є підмножиною множини топологічних дуг, що входять в шлях π^*_{ij}, побудований на мережі G і представлений в мережі G_D однією дугою, то довжини шляхів π_{ij} і π^*_{ij} співпадають. У разі невиконання останньої умови, довжина шляху π_{ij} не може бути менше довжини π^*_{ij}, оскільки тоді шлях π^*_{ij} не був би найкоротшим. Отримане протиріччя і той факт, що в алгоритмі побудови найкоротших шляхів, серед декількох можливих найкоротших шляхів між заданими вузлами, першим завжди буде визначений шлях, що складається з меншого числа дуг, доводить твердження 1). Доведення 2) і 3) виходить з правил формування ЗО для вузлів, побудови мережі G_D, а також з міркувань, аналогічних наведеним для доведення твердження 1).

Слід зазначити, що для випадків (2)-(4) довжини найкоротших шляхів між вузлами i і j, побудованих в мережі G і G_D, можуть не співпадати. У разі (4), коли $j \in Z_i$ мають місце внутрішньо-зональні кореспонденції, потоки яких не піддаються додатковій переробці у вузлах першого типу, тому вони не розглядаються під час формування матриці A.

Далі вважатимемо, що кожна неорієнтована дуга $p \in P_D$ замінена на дві дуги з протилежною орієнтацією і p_D означає число усіх орієнтованих дуг в G_D.

Нехай $X^i = \| x^i_j \|^T$, $j = \overline{1, p_D}$, $i = \overline{1, n}$ – вектор-стовпець, що визначає потоки кореспонденцій, які виходять з вузла i по усіх дугах мережі G_D; $A = \| a_{ij} \|_{n \times n}$ – матриця потоків, в якій збережені тільки ті елементи a_{ij}, для індексів яких справедлива одна з умов (2)-(4). Інші елементи в матриці A замінені на нулі. Позначимо через $C^i = \| c^i_j \|$, $j = \overline{1, p_D}$, $i = \overline{1, n}$ вектор-рядок витрат на транспортування одиниці потоку кореспонденцій з i по дугах мережі G_D. Побудуємо для G_D матрицю інцидентності $E = \| e_{ij} \|_{n \times p_D}$ вузли-дуги і визначимо матриці $W^\xi = \| w^\xi_{ij} \|_{n \times p_D}$, $\xi = \overline{1, n}$; вектори-стовпці $V^i = \| v^i_j \|^T$, $j = \overline{1, n}$, $i = \overline{1, n}$; вектор-стовпець пропускних здатностей вузлів $H^W = \| h_i \|^T$, $i = \overline{1, n}$. Знак «^{T}» означає транспонування. Де відповідно:

$$e_{ij} = \begin{cases} 1, \text{ якщо дуга } j \text{ спрямована до вузла } i, \\ -1, \text{ якщо дуга } j \text{ спрямована від вузла } i, \\ 0 \text{ в іншому випадку;} \end{cases}$$

$$w_{ij}^{\xi} = \begin{cases} 1, \text{ якщо } e_{ij} = 1, \\ 1, \text{ якщо } e_{ij} = -1 \text{ і } i = \xi, \\ 0 \text{ в іншому випадку;} \end{cases}$$

$$v_j^i = \begin{cases} -\sum_{\xi=1}^{n} a_{ij}, \text{ якщо } j = i, \\ a_{ij}, \text{ якщо } j = \xi. \end{cases}$$

Розв'язання задачі розподілу потоків у зональних мережах шукатимемо в класі задач лінійного програмування виду:

$$Min\ Z = C^1[p_D]X^1[p_D] + \ldots + C^n[p_D]X^n[p_D], \quad (5)$$

$$W^1[n, p_D]X^1[p_D] + \ldots + W^n[n, p_D]X^n[p_D] \le H^w[n], \quad (6)$$

$$E[n, p_D]X^1[p_D] = V^1[n], \quad \ldots \quad E[n, p_D]X^n[p_D] = V^n[n], \quad (7)$$

$$X^i[p_D] \ge 0 \text{ і цілі } i = \overline{1,n}. \quad (8)$$

Розглянемо особливості сформульованої задачі. Із Твердження 1 ясно, що для вирішення блокових задач, можна використати ефективні алгоритми побудови найкоротших шляхів за одним критерієм – мінімумом довжини шляху. Це витікає з того, що будь-який шлях для кореспонденцій з класу (2) або (3) складатиметься з двох дуг, а для кореспонденцій класу (4) – з двох або трьох дуг. Тому вибір оптимального шляху завжди робитиметься серед шляхів, що містять однакове число дуг, тобто усі шляхи еквівалентні з точки зору числа транзитних вузлів. Інша особливість задачі (5)-(8) полягає в тому, що обмеження на пропускні здатності вузлів необхідно враховувати тільки для вузлів першого типу, пов'язаних з вузлами другого і третього типів в мережі G_D хоча би однією дугою. Верхня межа числа обмежень (6), що враховуються, дорівнює величині n_1 – числу вузлів в множині N_1. Остання обставина забезпечує відносно невисоку розмірність задачі (5)-(8) і дозволяє використати для отримання нижніх оцінок її вирішення метод декомпозиції Данцига-Вулфа й інші методи цілочислового програмування [4, 5].

Підкреслемо, що у разі неврахування обмеження (6), отримане вирішення у процесі використання методу Данцига-Вулфа буде завжди цілочисловим, оскільки матриці обмежень у приведеній моделі є цілком унімодулярними, а праві частини обмежень цілочисловими векторами і вибір оптимального розв'язку здійснюється на многограннику з цілочисловими вершинами.

Оскільки реальні мережі функціонують з ресурсами, що динамічно змінюються з часом, і недостатньо точною початковою інформацією, для практичного вирішення задачі можна відмовитися від точних методів і використати алгоритми, засновані на методах побудови найкоротших шляхів. Приведемо алгоритми, що дозволяють вирішити задачу у випадках заданих і невідомих зон обслуговування вузлів.

Нехай $C^* = \| c_{ij}^* \|_{n \times n}$ – довідкова матриця найкоротших шляхів, побудована для мережі G_D, де

$$c_{ij}^* = \begin{cases} 0, \text{ якщо } (i = j) \vee \\ \vee (i, j \in N_2 \cup \mathrm{N}_3 \wedge j \in Z_i), \\ i, \text{ якщо } i \text{ і } j \\ \text{пов'язані однією дугою,} \\ k, \text{ якщо шлях від } i \text{ до} \\ j \text{ містить більше однієї дуги,} \end{cases} \quad (9)$$

де k – передостанній вузол на найкоротшому шляху від i до j; B_i^1 – множина вузлів першого типу в Z_i; $\{\alpha \mid \beta\}$ – множина α для яких вірне твердження β.

Відомо, що для побудованих найкоротших шляхів справедливе твердження $\forall i \in N\ \exists \{\pi_{ij} \mid j \in N \setminus i\}$ таке, що в $\forall j$ входить не більше за одну дугу, яка належить будь-якому найкоротшому шляху π_{ij}. Таким чином, найкоротші шляхи π_{ij} побудовані від вузла i є деревом з коренем в i, яке може бути однозначно описане рядком довідкової матриці C^*.

Алгоритм 1. Розподіл потоків в умовах заданих ЗОВ

1. Використовуючи матрицю R побудувати в мережі G найкоротші шляхи від $\forall i \in N_2 \cup N_3$ до $\forall j \in B_i^1$ і між усіма вузлами з множини N_1.

2. Використовуючи матрицю D, побудовану в п. 1, знайти найкоротші шляхи в мережі G_D. У процесі побудови шляхів сформувати довідкову матрицю C^* відповідно до (9).

3. $C \leftarrow 0$; $Y \leftarrow 0$. *** C – довідкова матриця об'єднання потоків

4. Для $\{i, j \mid (c_{ij}^* = i \vee c_{ij}^* = 0) \wedge i \neq j,\ \ i, j = \overline{1,n}\}$ виконати $c_{ij} \leftarrow i$.

5. Для $\{i, j \mid c_{ij}^* \neq i \vee c_{ij}^* \neq 0,\ \ i, j = \overline{1,n}\}$ виконати пп. 6-12.

6. $\xi \leftarrow j$; $l \leftarrow 0$; $k \leftarrow 0$.

7. Поки $c_{i\xi}^* \neq i$ виконати пп. 8-10.

8. Якщо $l = 0$, то $l \leftarrow c_{i\xi}^*$.

9. $k \leftarrow c_{i\xi}^*$.

10. $\xi \leftarrow c_{i\xi}^*$. Перейти до п. 7.

11. Якщо $k = l$, то виконати перетворення $a_{ik} \leftarrow a_{ik} + a_{ij}$, $a_{kj} \leftarrow a_{kj} + a_{ij}$, $y_k \leftarrow y_k + a_{ij}$, $a_{ij} \leftarrow 0$, інакше виконати перетворення $a_{ik} \leftarrow a_{ik} + a_{ij}$ $a_{kl} \leftarrow a_{kl} + a_{ij}$ $a_{lj} \leftarrow a_{lj} + a_{ij}$ $y_k \leftarrow y_k + a_{ij}$ $y_l \leftarrow y_l + a_{ij}$, $a_{ij} \leftarrow 0$.

12. $c_{ij} \leftarrow k$. Перейти до п. 5.

13. Стоп.

Запис в рядках 4 і 5 вираження $i, j = \overline{1,n}$ означає циклічну зміну індексів, причому другий індекс змінюється швидше за перший. Такий запис аналогічний організації внутрішнього циклу по j і зовнішнього по i, коли обидва індекси змінюються від одиниці до n з одиничним кроком.

У результаті роботи Алгоритму 1 формуються матриці A, C і вектор Y. У векторі Y підсумовуються додаткові обсяги обробки від перетворених потоків вузлів другого і третього типу (спочатку $C = 0$ і $Y = 0$). Елементи довідкової матриці об'єднання (сортування) потоків $C = \| c_{ij} \|_{n \times n}$ визначаються таким чином:

$$c_{ij} = \begin{cases} i, \text{ якщо потік } a_{ij} \text{ адресується в вузол } j, \\ k, \text{ якщо потік } a_{ij} \text{ адресується в вузол } k, \\ 0, \text{ якщо } i = j. \end{cases}$$

Матриця C використовується для запам'ятовування схеми адресації (сортування) потоків у вузлах другого і третього типів.

Для роботи Алгоритму 1 необхідно знати множини B_i^1, $i \in N_2 \cup N_3$. Раніше відмічалося, що множини B_i^1 визначаються методом експертного аналізу із залученням досвідчених фахівців або автоматизованим способом після вирішення задачі вибору структури мережі. У випадку, якщо процедура уточнення структури мережі експертами з будь-яких причин виконана не була, може виявитися, що зони обслуговування для вузлів другого і третього типу точно не відомі. Тому доцільно мати алгоритм розподілу потоків у зональних мережах, якщо заданий тільки вектор $\theta = \|\theta_i\|_n$ що визначає типи вузлів, де $\theta_i = 1$, якщо $i \in N_1$ і $\theta_i = 0$, якщо $i \in N_2 \cup N_3$.

Алгоритм 2. Розподіл потоків за невідомих ЗОВ

1. Використовуючи матрицю R побудувати на мережі G найкоротші шляхи між усіма вузлами з одночасним формуванням довідкової матриці C^* згідно (9).

2. $C \leftarrow 0$; $Y \leftarrow 0$. *** C – довідкова матриця об'єднання потоків

3. Для $\{i \mid i = \overline{1,n}\}$ виконати пп. 4-23.

4. Для $\{j \mid j = \overline{1,n}\}$ виконати пп. 5-22.

5. Якщо $i \neq j$ перейти до п. 6, інакше перейти до п. 22.

6. Якщо $\theta_i = 1 \wedge \theta_j = 1$ виконати $c_{ij} \leftarrow i$; перейти до п. 22.

7. $\xi \leftarrow j$; $l \leftarrow 0$; $k \leftarrow 0$.

8. Поки $c_{i\xi}^* \neq i$ виконати пп. 9-12.

9. Якщо $\theta_{c_{i\xi}^*} = 1$ виконати пп. 10-11, інакше перейти до п. 12.

10. Якщо $l = 0$, то $l \leftarrow c_{i\xi}^*$.

11. $k \leftarrow c^*_{i\xi}$.

12. $\xi \leftarrow c^*_{i\xi}$. Перейти до п. 8.

13. Якщо $\theta_i = 0$, то перейти до п. 14, інакше до п. 21.

14. Якщо $\theta_j = 0$, то перейти до п. 15, інакше до п. 19.

15. Якщо $l \neq 0$, то перейти до п. 16, інакше $c_{ij} \leftarrow i$ і перейти до п. 22.

16. Якщо $l \neq k$, то виконати п. 17, інакше перейти до п. 18.

17. $a_{kl} \leftarrow a_{kl} + a_{ij}$, $a_{lj} \leftarrow a_{lj} + a_{ij}$,

$y_l \leftarrow y_l + a_{ij}$,

L1: $a_{ik} \leftarrow a_{ik} + a_{ij}$ $y_k \leftarrow y_k + a_{ij}$ $c_{ij} \leftarrow k$,

L2: $a_{ij} \leftarrow 0$ перейти до п. 22.

18. Виконати

L3: $a_{il} \leftarrow a_{il} + a_{ij}$ $a_{lj} \leftarrow a_{lj} + a_{ij}$ $y_l \leftarrow y_l + a_{ij}$

$c_{ij} \leftarrow l$ перейти до L2.

19. Якщо $k = 0$, то $c_{ij} \leftarrow i$ перейти до п. 22.

20. $a_{kj} \leftarrow a_{kj} + a_{ij}$. Перейти до L1.

21. Якщо $l = 0$, то $c_{ij} \leftarrow i$, інакше перейти до L3.

22. Перейти до п. 4. *** кінець циклу по j

23. Перейти до п. 3. *** кінець циклу по i

24. Стоп.

У Алгоритмі 2 покажчики k і l визначають відповідно перший і останній вузли першого типу, що знаходяться на найкоротшому шляху між i і j. Якщо між вузлами i і j немає вузлів першого типу, або немає взагалі ніяких вузлів, значення $k = l = 0$.

В результаті роботи Алгоритмів 1 і 2 формуються: перетворена матриця потоків кореспонденцій $A = \| a_{ij} \|_{n \times n}$; елементи довідкової матриці об'єднання потоків для вузлів другого і третього типу (матриця C). У рядках і стовпцях перетвореної матриці A для вузлів другого і третього типу ненульові значення будуть присутніми тільки для вузлів першого типу, які перебувають у зоні обслуговування вузлів другого і третього типу, а в потоках між вузлами першого типу будуть враховані усі потоки вузлів другого і третього типу.

Порівнюючи трудомісткість Алгоритмів 1 і 2 з трудомісткістю вирішення задачі (5)-(8) відзначимо, що використання простих процедур, що ґрунтуються на побудові найкоротших шляхів дозволяє вказати верхню межу асимптотичної трудомісткості – $O(n^3)$, тоді як точне вирішення цілочислової задачі (5)-(8) може мати експоненціальну оцінку [6].

Висновки

Резюмуючи обговорення задачі розподілу потоків в зональних мережах, відзначимо, що вона може бути сформульована як задача лінійного програмування з блоковою структурою обмежень і для її вирішення застосуємо метод декомпозиції Данцига-Вулфа й інші методи цілочислового програмування. Для вирішення задачі на реальних мережах запропоновані наближені алгоритми, засновані на побудові найкоротших шляхів. Верхня межа часової складності наближених алгоритмів складає $O(n^3)$, тоді як точне вирішення цілочислової задачі може мати експоненціальну оцінку. Тому перевагу у вирішенні загальної задачі вибору ієрархічної структури магістральної мережі слід віддати наближеним алгоритмам з огляду на їхню простоту, невелику трудомісткість і, як показали практичні дослідження [2], прийнятну точність розв'язку.

Література

1. O.M. Trofymchuk, V.A. Vasyanin, Simulation of Packing, Distribution and Routing of Small-Size Discrete Flows in a Multicommodity Network, *Journal of Automation and Information Sciences*, 2015. **47**(7). P. 15-30.

https://doi.org/10.1615/JAutomatInfScien.v47.i7.30.

2. А.Н. Трофимчук, В.А. Васянин, Компьютерное моделирование иерархической структуры коммуникационной сети с дискретными многопродуктовыми потоками, *УСиМ*, 2016. № 2. С. 48-57.

https://doi.org/10.15407/usim.2016.02.048.
3. В.А. Васянин, Компьютерное моделирование распределения и маршрутизации дискретных многопродуктовых потоков в коммуникационной сети, *УСиМ*, 2016. № 3. С. 43-53. https://doi.org/10.15407/usim.2016.03.043.
4. G.B. Dantzig, Ph. Wolfe, Decomposition Algorithm for linear programming, *Econometrica*, 1961. Vol. 29. N. 4. P. 767-778.
5. C. Barnhart, N. Krishnan, P.H. Vange, Multicommodity Flow Problems, In Encyclopedia of Optimization: Second Edition, C.A. Floudas and P.M. Pardalos (Eds.). Springer, New York, 2009. P. 2354-2362.
6. М. Гэри, Д. Джонсон, Вычислительные машины и труднорешаемые задачи, М.: Мир, 1982. 416 с.

Одержано: 10.04.2024
Внутрішня рецензія отримана: 15.04.2024
Зовнішня рецензія отримана: 15.04.2024

Про автора:

Васянін Володимир Олександрович,
Доктор технічних наук,
Старший науковий співробітник
https://orcid.org/0000-0003-4046-5243

Місце роботи автора:

Завідувач відділу прикладної інформатики
Інституту телекомунікацій і глобального
інформаційного простору НАН України,
м. Київ,
E-mail: archukr@meta.ua
Сайт: https://itgip.org/

UDC 619.6 http://doi.org/10.15407/pp2024.02-03.062

M.R. Petryk, A.Yu. Doroshenko, D.M. Mykhalyk, O.A. Yatsenko

COMPUTER SIMULATION OF DIFFUSION TRANSPORT PROCESSES IN MULTILAYER NANOFILMS

The difficulties associated with studying diffusion in multilayer films require the progress of contemporary modeling methods and software platforms to precisely represent phenomena, taking into account transitions between adjacent layers. In addition to the indispensable role of advanced modeling techniques and software in solving the problems of studying diffusion in multilayer films, it is extremely important to admit the key contribution of sophisticated computational approaches. In this paper, the authors attempt to merge intricate mathematical models with optimal software development methodologies to address the challenge of simulating diffusion transport processes in multilayer nanofilms computationally. Based on the experimental findings and employing the suggested model, identification was conducted utilizing the theory of state control for multicomponent systems. With the help of methods of optimal control of the state of multicomponent transport systems, the analytical solution of the model and the data of experimental observations, the distributions of diffusion coefficients for the considered components of nanofilms (samples of aluminum, molybdenum, silicon) were reproduced. Numerical simulation results were compared with experimental observations. The profiles obtained from the modeling closely match the corresponding experimental profiles, especially as the duration of multilayer formation converges to the final stages of completing the protective nanofilm multilayer formation. The maximum observed deviation does not exceed 2–3%, confirming the reliability of the mathematical model and demonstrating the practical value of the results provided. A software framework is developed for the automation of the specified calculations with the possibility of extension to other subject areas with similar tasks of identifying the key factors of the process and further numerical modeling of the time-space characteristics using the obtained results.
Key words: coefficients identification, diffusion, mathematical model, multilayer nanofilm, numerical simulation, oxide film.

М.Р. Петрик, А.Ю. Дорошенко, Д.М. Михалик, О.А. Яценко

КОМП'ЮТЕРНЕ МОДЕЛЮВАННЯ ПРОЦЕСІВ ДИФУЗІЙНОГО ТРАНСПОРТУ В БАГАТОШАРОВИХ НАНОПЛІВКАХ

Проблеми, пов'язані з дослідженням дифузії в багатошарових плівках, вимагають удосконалення сучасних методів моделювання та програмного забезпечення для точного зображення явищ, враховуючи переходи між сусідніми шарами. На додаток до незамінної ролі передових методів моделювання та програмного забезпечення у розв'язанні проблем вивчення дифузії в багатошарових плівках, надзвичайно важливо визнати ключовий внесок складних обчислювальних підходів. У цій роботі автори зробили спробу поєднати складні математичні моделі з передовим досвідом розробки програмного забезпечення для розв'язання задачі комп'ютерного моделювання процесів дифузійного транспорту в багатошарових наноплівках. За результатами експериментальних даних та з використанням запропонованої моделі проведено ідентифікацію з використанням теорії контролю стану багатокомпонентних систем. За допомогою методів оптимального керування станом багатокомпонентних транспортних систем, аналітичного рішення моделі та даних експериментальних спостережень відтворено розподіли коефіцієнтів дифузії для розглянутих складових компонентів наноплівок (зразків алюмінію, молібдену, кремнію). Результати чисельного моделювання порівнюються з експериментальними спостереженнями, що відображають вміст зразка. Профілі, отримані в результаті моделювання, тісно збігаються з відповідними експериментальними профілями, особливо коли тривалість багатошарового утворення наближається до остаточного періоду завершення утворення захисної наноплівки. Максимальне спостережене відхилення не перевищує 2–3 %, що підтверджує надійність математичної моделі та вказує на практичну корисність отриманих результатів. Розроблено програмний інструментарій для автоматизації зазначених розрахунків з можливістю поширення на інші предметні області зі схожими завданнями ідентифікації ключових факторів процесу. Створено базу для подальшого чисельного моделювання часо-просторових характеристик з використанням отриманих результатів.
Ключові слова: ідентифікація коефіцієнтів, дифузія, математична модель, багатошарова наноплівка, чисельне моделювання, оксидна плівка.

Introduction

The challenges associated with investigating diffusion in multilayer films necessitate the advancement of modern modeling techniques and software frameworks to accurately depict phenomena while considering transitions between neighboring layers [1, 2]. One of the most effective approaches to thoroughly address these issues is the widely recognized integral transformation. These methods are utilized to derive solutions for diverse boundary value problems in mathematical physics pertaining to homogeneous structures, encompassing diffusion scenarios across various environments and enabling their mathematical representation.

In addition to the indispensable role of advanced modeling techniques and software frameworks in addressing the challenges of studying diffusion within multilayer films, it is crucial to recognize the pivotal contribution of sophisticated computational approaches. By utilizing cutting-edge methodologies in computer programming and software architecture, the capabilities of mathematical approaches, such as integral transformations, can be elevated to unprecedented levels of precision and efficiency. This synergy between mathematical modeling and computational innovation not only enhances our ability to accurately represent diffusion phenomena within multilayer structures but also opens new avenues for exploration and analysis.

Furthermore, the integration of modern modeling techniques with state-of-the-art computational tools facilitates a more nuanced understanding of diffusion processes across diverse material compositions and environmental conditions. By leveraging the power of numerical simulations and data-driven analyses, researchers can extract valuable knowledge on complex diffusion mechanisms that were previously inaccessible. This multidisciplinary approach enables scientists and engineers not only to solve fundamental questions in materials science and engineering but also to devise innovative strategies for optimizing the performance and functionality of multilayer films in various technological applications.

Problem formulation. In the proposed research, authors attempt to combine complex mathematical models with the best practices of software engineering to solve the problem of computer simulation of diffusion transport processes in multilayer nanofilms. The primary objectives are:

– modeling diffusion processes: to integrate complex mathematical models with contemporary software development methodologies, in domain of the diffusion within multilayer films;

– parameter identification: to employ the theory of state control for multicomponent systems to identify diffusion coefficients. It includes using methods of optimal control to analyze experimental data and accurately reproduce the distributions of diffusion coefficients for the nanofilm components;

– numerical simulation and validation: to conduct numerical simulations and compare the results with experimental observations. The target is to achieve a high degree of correspondence between the modeled and experimental profiles, particularly as the duration of multilayer formation approaches completion;

– development of a software framework: to create a software framework that automates the specified calculations and can be extended to other subject areas with similar tasks. This framework should facilitate the identification of key process factors and enable further numerical modeling of time-space characteristics based on the obtained results.

By addressing these objectives, this research seeks to overcome the limitations of existing methods and provide a robust tool for studying and optimizing the diffusion processes in multilayer films. The ultimate aim is to enhance the efficiency of experimental studies and contribute to the development of new nanomaterials with improved properties

1. Mathematical model

We propose a physical problem and corresponding mathematical model for the diffusion mass transfer process in multilayer films, based on the following multilayer medium comprised of n double layers composed

of two distinct media with differing properties (Fig. 1).

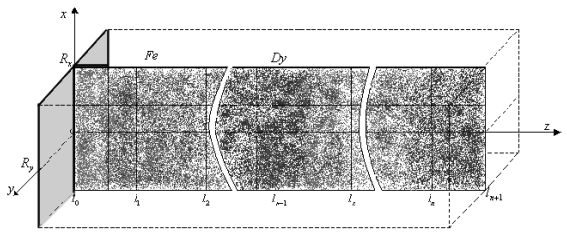


Fig.1. Schema of multilayer nanofilm

According to this representation, at each interface within the formed multicomposite, there is a mutual diffusion of components between two adjacent layers of the medium. The mechanisms governing such mutual transfer are determined by the variable gradients and rates of concentration change at the interface boundaries between layers. By incorporating changes in concentrations and their gradients over time into the boundary and interface conditions, it becomes feasible to model the mechanisms of this additional mutual transport alongside the fundamental transport equations.

When formulating a mathematical model for diffusion transfer within oxide films, a multilayer structure is taken into account. Assuming that the diffusion of atoms of constituent components of oxide films (aluminum, molybdenum, silicon) primarily drives system mixing, concentration profiles for such a multilayer system can be derived from Fick's equations, incorporating boundary conditions at the outer layers and contact conditions between successive layers. This approach provides a mathematical model describing the diffusion transfer process within a planar multilayered medium

$$\frac{\partial}{\partial z} C_k(t,x,z) + \gamma_k^2 C_k = D_0 \frac{\partial^2 C_k}{\partial x^2} + D_{z_k} \frac{\partial^2 C_k}{\partial z^2} \quad (1)$$

at domain

$$t > 0,\ x \in (0,\ R),$$
$$z : z \in \bigcup_{k=1}^{n+1} (l_{k-1},\ l_k);\ l_0 \geq 0;\ l_{n+1} = \infty,$$
$$j = \overline{1,\ 2}\ k = \overline{1,\ n},$$

where C_k is diffusion distribution; D_k is diffusion coefficient; γ_k^2 is mass distribution coefficient;

The corresponding initial and boundary conditions for the model:

$$C_k(t,\ x,\ z)\big|_{t=0} = C_{0_k}(x,\ z) \equiv C_{0_k}(z),$$
$$\left[\alpha_{11}^0 \frac{d}{dz} + \beta_{11}^0\right] C_1(t,x,z)\Big|_{z=l_0} = C_{l_0}(t,x); \quad (2)$$
$$\frac{\partial C_{n+1}}{\partial z}\Big|_{z=\infty} = 0;$$

$$\left[\begin{array}{l}\left[\alpha_{j1}^k \frac{\partial}{\partial z} + \beta_{j1}^k\right] C_k - \\ -\left[\alpha_{j2}^k \frac{\partial}{\partial z} + \beta_{j2}^k\right] C_{k+1}\end{array}\right]_{z=l_k} = 0,$$

And boundary conditions for variable x

$$\frac{\partial C_k}{\partial x}\Big|_{x=0} = 0; \quad C_k\big|_{x=R} = C_{1_k}(t,z), \quad (3)$$

where α_{ij}^k, β_{ij}^k; $k = \overline{0,n}$; $i, j = \overline{1,2}$, are coefficients determining boundary conditions and contact conditions; x, y, z are spatial coordinates.

The exact solution of the problem described by equations (1)–(3) is directly written out by applying integral Fourier transforms [3].

$$C_k(t,x,z) =$$
$$\int_0^t \int_0^R W_{l_{0,k}}(t-\tau;\ x,\ \varsigma;\ z)\ C_{l_0}(\tau,\ \varsigma) d\varsigma d\tau +$$
$$+ \sum_{k_1=1}^{n+1} \int_0^t \int_0^r \int_{l_{k_1-1}}^{l_{k_1}} H_{k,k_1}(t-\tau;\ x,\ \varsigma;\ z,\ \xi) \cdot$$
$$\cdot C_{0_k}(\varsigma,\ \xi) \cdot \delta_+(\tau) \sigma_{k_1} d\varsigma d\xi d\tau +$$
$$+ \sum_{k_1=1}^{n+1} \int_0^t \int_{l_{k_1-1}}^{l_{k_1}} W_{R_k,k_1}(t-\tau;\ x;\ z,\ \xi) \cdot$$
$$\cdot C_{1_{k_1}}(\tau,\ \xi) \sigma_{k_1} d\xi d\tau.$$

Here are Green's functions:

$$W_{l_{0,k}}(t, x; z, \xi) =$$

$$= \frac{2}{R} \sum_{m=0}^{\infty} W_{l_0,k}^{m}(t-\tau, z)(-1)^m \frac{\cos \eta_m \xi}{\eta_m};$$

$$W_{R_{k,k_1}}(t; x, \varsigma; z, \xi) =$$

$$= \frac{2}{R} \sum_{m=0}^{\infty} e^{-D_0 \eta_m^2 t} D_0 (-1)^{m+1} \eta_m \varepsilon_{k,k_1}^{m}(t; z, \xi)(-1)^m \cdot$$

$$\cdot \cos \eta_m x.$$

Cauchy's function

$$H_{k,k_1}(t; x, \varsigma; z, \xi) =$$

$$= \frac{2}{R} \sum_{m=0}^{\infty} \varepsilon_{k,k_1}^{m}(t, z, \xi) \cdot \cos \eta_m \xi \cdot \cos \eta_m x.$$

2. Experimental data and coefficients identification

According to the results of experimental data and using the proposed model, identification was carried out using the theory of state control of multicomponent systems (results obtained in [4]) (Fig. 2).

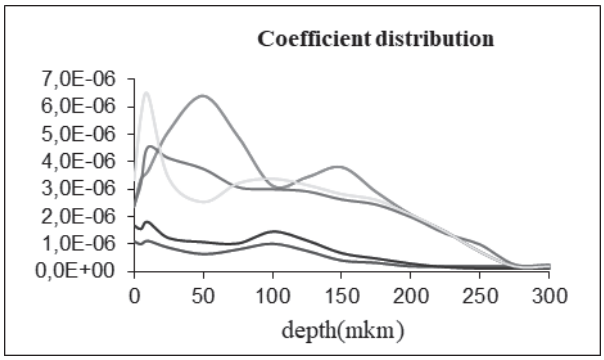


Fig. 2. Coefficients distribution at first sample

The distributions of diffusion coefficients are reproduced using methods of optimal control of the state of multicomponent transport systems, analytical solution of the model, and data of experimental observations, for the considered constituent components of nanofilms (samples of aluminum, molybdenum, silicon) (Fig. 3).

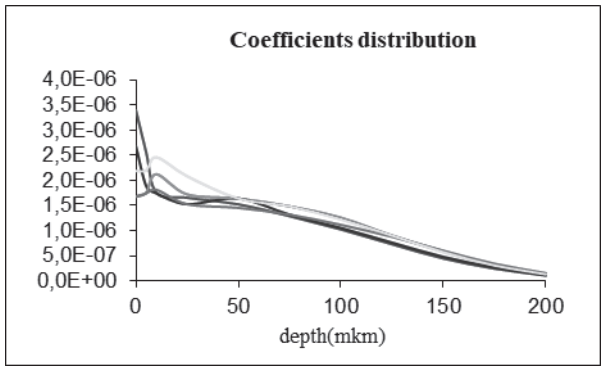


Fig. 3. Coefficients distribution at the second sample

The diffusion coefficients identified in such a way, correspond to real experimental data and are used as input parameters of the obtained mathematical solution of the model for modeling and analysis of concentration distributions of the main components of nanofilms (aluminum, molybdenum, silicon, etc.).

3. Numerical simulation

Numerical simulation results are compared with experimental observations depicting the sample content. These concentration distributions are generated for varying time intervals during the formation of the multilayer. The designated time frame corresponds to the experimental duration of three weeks. The process of forming the technological multilayer through molecular diffusion of the specified components is segmented into 5 periods, encompassing the inception of the protective multilayer from the initial period to the concluding period (Fig. 4).

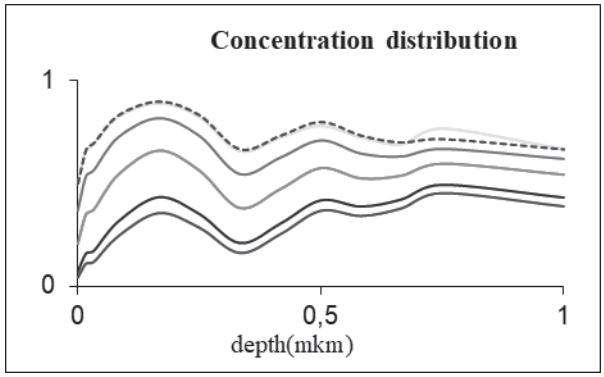


Fig. 4. Simulated concentrations for case 5

Continuous lines represent simulated data for different time durations. The dashed

line is experimentally measured data at the final duration.

As depicted in the figures above, the profiles derived from modeling closely align with the corresponding experimental profiles, particularly as the duration of multilayer formation approaches the completion period of the protective nanofilm multilayer formation. The maximum observed deviation does not surpass 2–3 %, affirming the reliability of the mathematical model and indicating the practical utility of the provided results (Fig. 5).

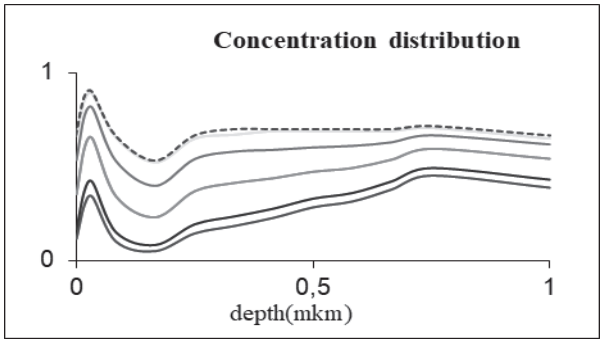


Fig. 5. Simulated concentrations

4. Software framework

The above results require complex resource-intensive calculations [5]. For this purpose, we developed a framework that automates the specified calculations with the possibility of extension to other subject areas with similar tasks of identifying the key factors of the process and further numerical modeling of the time-space characteristics using the obtained results.

The software framework in focus is implemented in Java programming language. JVM-based implementation allows us to rid of target-specific builds and follow the write-once-run-everywhere idea. At the same time, the JIT-compilation feature of JVM optimizes code execution by converting bytecode into native machine code at runtime. It allows to have runtime optimizations that are not possible with ahead-of-time compilation and are beneficial for CPU-intensive applications.

The framework design follows an object-oriented approach, and the code base is divided into major modules, such as Models, DataProcessing, Identification, and GUI. The input either can be read from the file or provided in UI. Calculated results also can be visualized or saved to the file. All components are connected with abstract interfaces, which gives the ability to easily replace one model implementation with another implementation (separation of interface and implementation).

One example is the implementation of diffusion coefficient identification, which defines the internal kinetic parameters of the process. To ascertain the coefficient distribution, we employ the gradient methods, the mathematical underpinnings of which lay in the context of parametric identification challenges in multicomponent distributed systems [6]. Tailoring our approach to the nanofilms domain, we find the method of minimum errors particularly apt. So, we picked the implementation of this method in simulations among others in the software framework. The coefficients identification algorithm is based on a gradient-identification procedure wherein, to ascertain the next approximation of the diffusion coefficient within the intraparticle space, we adhere to the following protocol (Fig. 6).

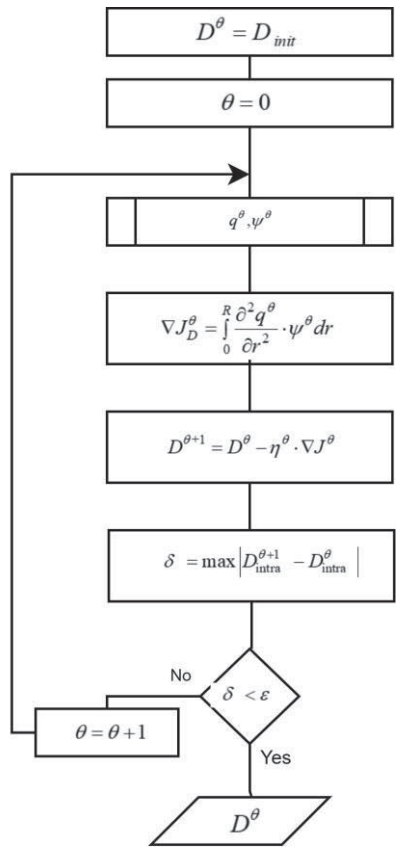


Fig. 6. Procedure for internal parameters identification

The determined distributions of diffusion coefficients within the interparticle space enable the modeling of concentration fields and integral mass distributions within the catalytic nanoporous layer with a high level of precision. As evidenced by the concentration distributions presented (Fig. 2 and Fig. 3), the profiles derived from the model closely match those obtained experimentally (Fig. 4 and Fig. 5), illustrating a noteworthy degree of consistency between the two sets of data. This alignment is further emphasized by the complete coincidence observed between the model and experimental graphs depicting the integral mass during benzene adsorption.

Conclusion

The difficulties associated with studying diffusion in multilayer films require the progress of contemporary modeling methods and software platforms to precisely represent phenomena, taking into account transitions between adjacent layers. Alongside the essential function of cutting-edge modeling techniques and software frameworks in addressing the difficulties of examining diffusion within multilayer films, it's vital to acknowledge the significant input of sophisticated computational methods.

In this paper, we endeavor to merge intricate mathematical models with optimal software development methodologies to address the challenge of simulating diffusion transport processes in multilayer nanofilms computationally. Based on the experimental findings and employing the suggested model, identification is conducted utilizing the theory of state control for multicomponent systems. Using the methods of optimal control of the state of multicomponent transport systems, the analytical solution of the model, and the data of experimental observations, the distributions of diffusion coefficients for the considered components of nanofilms (samples of aluminum, molybdenum, silicon) were reproduced. Numerical simulation results were compared with experimental observations. The profiles obtained from the modeling closely match the corresponding experimental profiles, especially as the duration of multilayer formation approaches the final stages of completing the protective nanofilm multilayer formation. The maximum observed deviation does not exceed 2–3%, confirming the reliability of the mathematical model and demonstrating the practical value of the results provided.

A software framework is developed for the automation of the specified calculations with the possibility of extension to other subject areas with similar tasks of identifying the key factors of the process and further numerical modeling of the time-space characteristics using the obtained results.

Література

1. J. Kärger, F. Grinberg, P. Heitjans, Diffusion fundamentals, Leipzig, Leipziger Unviersite, 2005.
2. M.R. Petryk, A. Khimich, M.M. Petryk, J. Fraissard, Experimental and computer simulation studies of dehydration on microporous adsorbent of natural gas used as motor fuel, *Fuel*, 2019. Vol. 239. P. 1324–1330.
3. М. Р. Петрик, П. М. Василюк, Д. М. Михалик, Н. В. Бабій, О. Ю. Петрик, Моделі процесів дифузійного переносу і методи оцінювання параметрів в багато композитних наноплівках. Тернопіль, Вид-во ТНТУ імені Івана Пулюя, 2015.
4. В. С. Дейнека, М. Р. Петрик, Идентификация параметров неоднородных задач диффузии в наномультикомпозитах с использованием градиентных методов, *Компьютерная математика*, 2012. № 1. С. 41–51.
5. M. Petryk, A. Doroshenko, D. Mykhalyk, P. Ivanenko, O. Yatsenko, Automated parallelization of software for identifying parameters of intraparticle diffusion and adsorption in heterogeneous nanoporous media. In: Shkarlet, S., et al. Mathematical Modeling and Simulation of Systems. MODS 2022. Lecture Notes in Networks and Systems. 2023. Vol. 667. P. 33–47. https://link.springer.com/chapter/10.1007/978-3-031-30251-0_3
6. И. В. Сергиенко, В. С. Дейнека, Системный анализ многокомпонентных рас-

пределенных систем, Киев, Наукова думка, 2009.

References

1. J. Kärger, F. Grinberg, P. Heitjans, Diffusion fundamentals, Leipzig, Leipziger Unviersite, 2005.
2. M.R. Petryk, A. Khimich, M.M. Petryk, J. Fraissard, Experimental and computer simulation studies of dehydration on microporous adsorbent of natural gas used as motor fuel, in: Fuel 239 (2019) 1324–1330.
3. M. Petryk, P. Vasylyuk, D.Mykhalyk, N. Babiy, O. Petryk, Models of diffusive transport processes and methods of parameter estimation in multi-composite nanofilms, Ternopil, TNTU, 2015. doi: 10.32626/2308-5916.2015-12.141-160. [in Ukrainian]
4. V. S. Deineka, M. R. Petryk, Identification of parameters of inhomogeneous diffusion problems in nanomulticomposites using gradient methods, in: Computer mathematics 1 (2012) 41–51. [in Russian]
5. M. Petryk, A. Doroshenko, D. Mykhalyk, P. Ivanenko, O. Yatsenko, Automated parallelization of software for identifying parameters of intraparticle diffusion and adsorption in heterogeneous nanoporous media, in: Shkarlet, S., et al. Mathematical Modeling and Simulation of Systems. MODS 2022. Lecture Notes in Networks and Systems, 2023, Vol. 667. P. 33–47. https://link.springer.com/chapter/10.1007/978-3-031-30251-0_3
6. V. Sergienko, V. S. Deineka, System analysis of multicomponent distributed systems, Kyiv, Naukova Dumka, 2009. [in Russian]

Одержано: 10.04.2024
Внутрішня рецензія отримана: 20.04.2024
Зовнішня рецензія отримана: 27.04.2024

Про авторів:

[1]*Петрик Михайло Романович*,
доктор фізико-математичних наук,
професор.
http://orcid.org/0000-0001-6612-7213.

[2]*Дорошенко Анатолій Юхимович*,
доктор фізико-математичних наук,
професор, провідний науковий співробітник.
http://orcid.org/0000-0002-8435-1451.

[1]*Михалик Дмитро Михайлович*,
кандидат технічних наук, доцент.
https://orcid.org/0000-0001-9032-695X.

[2]*Яценко Олена Анатоліївна*,
кандидат фізико-математичних наук,
старший науковий співробітник.
http://orcid.org/0000-0002-4700-6704.

Місце роботи авторів:

[1]Тернопільський національний технічний університет імені Івана Пулюя,
Тел. (+38) (044) 526-20-08
E-mail: softeng@tntu.edu.ua
Сайт: https://tntu.edu.ua

[2]Інститут програмних систем
НАН України,
тел. +38-044-526-60-33
E-mail: a-y-doroshenko@ukr.net,
oayat@ukr.net
Сайт: https://iss.nas.gov.ua

УДК 681.3 http://doi.org/10.15407/pp2024.02-03.069

О.І. Лисенко, В.Л. Шевченко, О.М. Тачиніна, С.О. Пономаренко, О.Г. Гуйда

СТРУКТУРА АЛГОРИТМУ МОДЕЛЮВАННЯ ОПТИМАЛЬНОГО РУХУ СКЛАДЕНОЇ ДИНАМІЧНОЇ СИСТЕМИ

Робота присвячена розробці структури алгоритму моделювання оптимального руху складеної динамічної систем (СДС) по розгалуженій траєкторії. Складеними називають системи, що складаються з окремих підсистем, траєкторії польоту яких відрізняються та отримали назву розгалужених. Розгалужені траєкторії мають складаються із ділянок траєкторії, перші з яких будуть спільними для всієї СДС, а інші гілки траєкторії будуть відрізнятись, оскільки кожна підсистема рухається до своєї цілі по своїй ділянці траєкторії. Запропонований алгоритм дозволяє в реальному часі оптимізувати такі траєкторії та здійснювати оперативну корекцію траєкторій СДС при виникненні непередбачуваних факторів впливу. Відомо, що ефективність функціонування СДС між структурними перетвореннями залежить від координат взаємного розташування і швидкості кожної підсистеми та вибору оптимальних моментів часу для структурних перетворень. Принципово важливим є оперативність визначення цих параметрів в процесі польоту. Знайдено необхідні умови оптимальності траєкторії руху СДС, які є універсальними для задач з будь-яким кінцевим числом гілок траєкторії. Реалізація запропонованих умов дозволить зменшити кількість обчислювальних процедур при розрахунках керування в умовах невизначеності початкових умов. Ці умови є методологічною основою для розроблення обчислювальних алгоритмів моделювання оптимальних траєкторій руху СДС. Необхідні умови оптимальності мають чіткий фізичний зміст і є технологічними та зручними для використання. Результати досліджень, що представлені в статті, є важливими і актуальними для побудови законів траєкторного керування існуючими і перспективними СДС.

Ключові слова: оптимальне керування, складена динамічна системи, розгалужені траєкторії, математичне моделювання, алгоритмічне забезпечення.

O. I. Lysenko, V. L. Shevchenko, O. M. Tachynina, S. O. Ponomarenko, O. H. Guida

STRUCTURE OF THE ALGORITHM FOR MODELING OPTIMAL MOVEMENT OF A COMPOUND DYNAMIC SYSTEM

The work is devoted to the development of the structure of the algorithm for modeling the optimal movement of complex dynamic systems (SDS) along a branched trajectory. Complex systems are called systems consisting of separate subsystems, the flight trajectories of which differ and are called branched. Branched trajectories should consist of trajectory segments, the first of which will be common to the entire SDS, and the other trajectory branches will be different, as each subsystem moves to its goal along its own trajectory segment. The proposed algorithm makes it possible to optimize such trajectories in real time and to carry out operational correction of SDS trajectories in the event of the occurrence of unpredictable influencing factors. It is known that the effectiveness of the SDS functioning between structural transformations depends on the coordinates of the mutual location and speed of each subsystem and the choice of optimal moments of time for structural transformations. The efficiency of determining these parameters during the flight is fundamentally important. The necessary conditions for the optimality of the trajectory of the SDS movement are found, which are universal for problems with any finite number of trajectory branches. The implementation of the proposed conditions will allow to reduce the number of computational procedures in the control calculations in conditions of uncertainty of the initial conditions. These conditions are the methodological basis for the development of computational algorithms for modeling the optimal trajectories of the SDS movement. The necessary optimality conditions have a clear physical meaning and are technological and user-friendly. The results of the research presented in the article are important and relevant for the construction of the laws of trajectory control of existing and prospective SDS.

Keywords: optimal control, complex dynamic systems, branched trajectories, mathematical modeling, algorithmic support.

Вступ

Актуальність теми. Сучасні досягнення у створенні складних механічних об'єктів, систем зв'язку і передачі даних та високопродуктивних бортових обчислювачів відкривають шлях до проєктування складних технічних систем нового покоління, здатних вирішувати єдину технічну задачу без механічного зв'язку та лише на основі інформаційного обміну між окремими підсистемами таких об'єктів.

Прикладами таких систем є складені динамічні системи (СДС). До них відносяться динамічні системи, що складаються із окремих підсистем (сукупності об'єктів), які взаємодіють між собою у польоті, а синтез керування рухом кожної підсистеми відбувається скоординовано. При цьому підсистеми можуть функціонувати спільно або окремо. Їхнє розділення відбувається за окремими командами, що подаються у строго визначеному просторовому положенні кожної підсистеми і в задані менти часу.

Прикладами сучасних СДС є багаторазові авіаційно-космічні системи (АКС) типу «повітряний старт» та групи безпілотних літальних апаратів (БПЛА), які утворюють «літаючі сенсорні мережі», або рої (роботизовані сенсорні мережі) на основі бездротових телекомунікаційних систем.

У науковій літературі прийнято називати траєкторії складених динамічних систем розгалуженими, оскільки вони складаються із початкової ділянки спільного руху всієї системи та ділянок індивідуального руху окремих підсистем СДС окремими гілками траєкторії.

Встановлено [1, 2], що ефективність функціонування СДС між структурними перетвореннями залежить від координат взаємного розташування і швидкості кожної підсистеми та вибору оптимальних моментів часу для структурних перетворень. Принципово важливим є оперативність визначення цих параметрів у польоті.

Тому задача оперативної побудови оптимальної розгалуженої траєкторії руху СДС у польоті є ключовою та визнається у світі актуальною як з наукової, так і практичної точок зору [1, 2, 5].

Аналіз стану питання.

Для розв'язку задачі оптимального керування СДС на розгалужених траєкторіях використовується математична теорія імпульсних диференціальних рівнянь із розривною правою частиною [4]. Поняття «розривної системи» є узагальнюючим і охоплює значний клас динамічних об'єктів: із імпульсним впливом, із розривами, багатоступеневих, із релейним керуванням, із проміжними умовами, складених та ін. Математичні моделі розривних систем в основному описують диференціальними рівняннями з розривними (кусково-неперервними) правими частинами. Теорія розривних динамічних систем та методи пошуку оптимальних рішень для таких систем розроблені такими дослідниками, як Л. С. Понтрягін, В. Г. Болтянский, Р. В. Гамкрелідзе, М. М. Красовський. В. А. Троїцький, В. І. Уткін. Для конкретних різновидів розривних систем теоретичні і прикладні результати отримані в роботах В. А. Боднера, Л. Т. Ащепкова, Б. Ф. Кротова, Брайсона Хою Ши, А. М. Самойленка, Н. А. Перестюка, А. А. Асланяна, О. І. Лисенка та інших авторів.

Особливістю теоретичних результатів цих авторів є те, що вони в термінах розривних систем сформулювали постановки задач оптимального керування і запропонували оптимальні рішення для конкретних динамічних системам із наявністю головного елемента (головної підсистеми) СДС. У цих роботах для створення математичних моделей розривних систем використовувався метод довизначення, або метод лінійних часових перетворень. У такій постановці задача керування СДС формулювалася як задача керування розривною системою із виділеним пріоритетним елементом, відносно якого відбувалось застосування теорії розривних систем для окремих складових. Наслідком такої постановки задачі було збільшення розмірів вектора стану і вектора керування розривної системи. Їхня кількість збільшувалась пропорційно кількості гілок траєкторії СДС. А постановка задачі пошуку оптимальної розгалуженої траєкторії для всієї

СДС (цілісна, узагальнена постановка задачі) не розглядалася.

На сьогоднішній день теоретичні рішення, отримані попередніми дослідниками, не доведені до прикладного застосування, а проєктні рішення для синтезу траєкторій руху СДС у реальному масштабі часу відсутні. Це пояснюється складністю самих математичних моделей і методів їхнього численного розв’язку, оскільки використання абстрактно-формального опису задач оптимізації розгалужених траєкторій СДС призводить до збільшення розмірності вектора стану та розмірності вектора керування розривною системою. І ця розмірність збільшується пропорційно кількості гілок траєкторії, що призводить до неможливості практичної реалізації алгоритму оперативної оптимізації розгалужених траєкторій у бортовому комп’ютері.

Також не були проведені прикладні дослідження, які б ґрунтувались на адекватному фізичному уявленні про характер – «схему» руху СДС гілками траєкторії і пояснили б механізм побудови оптимальних розгалужених траєкторій за довільною схемою з можливістю організації обчислювальних процедур.

Як результат, на практиці вже реально існують СДС типу «повітряний старт» та «літаючі сенсорні мережі», але для керування ними застосовуються методи, що не дозволяють повністю реалізувати увесь потенціал і технічні можливості діючих СДС. Вирішення цієї проблеми вимагає побудови достатніх умов оптимальності розгалужених траєкторій руху СДС. Причому ці вимоги мають бути сформульовані у зручній для реалізації в реальному масштабі часу формі (для оперативного синтезу траєкторій).

У зв’язку з цим вирішення проблеми оперативного синтезу оптимальних розгалужених траєкторій руху та оперативної оптимізації процесу керування рухом СДС на існуючих бортових обчислювальних засобах є нагальною науково-технічною потребою. А дослідження, що виконані в даній статті, є актуальними для систем траєкторного керування сучасними та перспективними СДС [3, 5, 6].

Викладення основного матеріалу.

СДС являє собою сукупність динамічних підсистем, які в процесі руху можуть об'єднуватися в групи для спільного руху, розділятися з метою самостійного маневрування, здійснювати взаємний вплив на динаміку руху [1, 3].

Розглянемо приклад руху гіпотетичної СДС за схемою, зображеною на рис. 1.

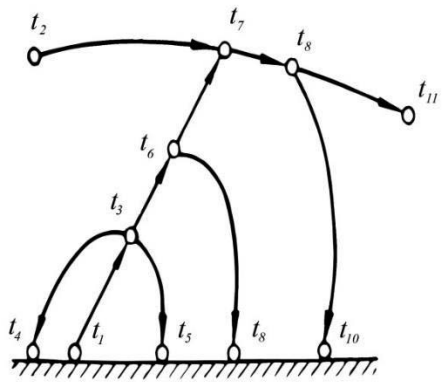


Рис. 1. Схема розгалуженої траєкторії

В момент часу t_1 чотири підсистеми в єдиному блоці починають рух, в процесі якого в момент часу t_3 відбуваються відділення від вихідного блоку двох допоміжних підсистем, які закінчують свій рух у моменти часу t_4 і t_5. У момент часу t_6 відокремлюється третя допоміжна підсистема, закінчується рух у момент часу t_8. У момент часу t_7 відбувається з'єднання четвертої підсистеми з підсистемою, що почала рух в момент часу t_2. Після закінчення спільного руху на момент часу t_9 підсистеми розстиковуються і здійснюють самостійне маневрування, що закінчується в моменти часу t_{10} і t_{11}. Траєкторія аналізованої СДС належить до класу розгалужених. Задамо критерій ефективності з урахуванням характеру руху підсистем вздовж усіх гілок траєкторії. Необхідно змоделювати оптимальну траєкторію руху СДС відповідно до заданого критерію. У вирішенні таких завдань можливі розгалужені траєкторії різної складності. Наразі сформульовані умови оптимальності тільки часткових схем розгалужених траєкторій.

Сформулюємо в термінах теорії оптимального керування достатні умови існування оптимального керування СДС довільної схеми. Розглянемо найпростіші роз-

галужені траєкторії, часові діаграми яких представлені на рис.2.

Рівняння, що описують рух СДС за траєкторією з розділенням (рис. 2, а), мають вигляд [3, 5]:

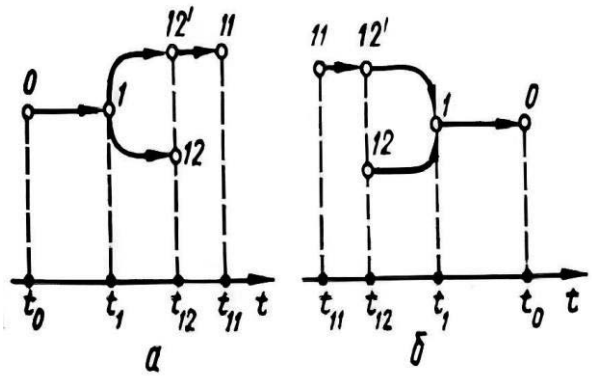


Рис. 2. Часова діаграма найпростішої розгалуженої траєкторії (а – з розділенням; 2 – з групуванням)

$$ {}_1\dot{x} = {}_1f\left({}_1x, {}_1u\right), \quad t \in [t_0, t_1]; $$

$$ {}_{11}\dot{x} = \begin{cases} {}^{12}_{11}f\left({}_{11}x, {}_{11}u; {}_{12}x, {}_{12}u\right), & t \in [t_1, t_{12}]; \\ {}_1f\left({}_{11}x, {}_{11}u\right), & t \in [t_{12}, t_{11}]; \end{cases} \quad (1) $$

$$ {}_{11}\dot{x} = {}_{12}f\left({}_{12}x, {}_{12}u; {}_{11}x, {}_{11}u\right), \quad t \in [t_0, t_{12}]; \quad (2) $$

де ${}_1x(t) \in R^n$, ${}_1u(t) \in R^{m_l}$, ${}_lu(t) \in \Omega_l$ $(l = 1, 11, 12)$. векторний критерій якості функціонування СДС запишемо в адаптивній формі:

$$ I = S({}_1x(t_0), t_0; {}_1x(t_1), t_1; {}_{11}x(t_{12}), {}_{12}x(t_{12}), t_{12}; {}_{11}x(t_{11}), t_{11}) + I_1 + I_{11} + I_{12}; $$

$$ I_1 = \int_0^1 \Phi_1\left({}_1x, {}_1u\right) dt, $$

$$ I_{12} = \int_1^{12} \Phi_{12}\left({}_{12}x, {}_{12}u; {}_{11}x, {}_{11}u\right) dt; $$

$$ I_{11} = \int_1^{12} \Phi_{11}^{12}\left({}_{12}x, {}_{12}u; {}_{11}x, {}_{11}u\right) dt + \int_{12}^{11} \Phi_{11}\left({}_{11}x, {}_{11}u\right) dt. $$

Критерії оптимальності відповідає формі Більця, згідно з якою функція S(·) за фізичним змістом відображає вимоги до параметрів руху окремих підсистем. Такими параметрами є значення координат на початку і в кінці кожної гілки траєкторії, а також значення самих моментів часу структурних перетворень СДС. Інтегральні члени критерію ставлять вимоги до характеру руху підсистеми вздовж відповідних гілок траєкторії. Взаємний вплив підсистеми в інтервалі часу $[t_1, t_{12}]\cdot$ відображено у рівнянні (1) і (2) та в часткових інтегральних умовах I_{11}, I_{12}. Рівняння, що описують критерій і динаміку руху підсистем, які групуються (рис. 2, б), мають той же вигляд, як і для системи з поділом, і відрізняються лише тим, що $t_{11} < t_{12} < t_1 < t_0$.

Опускаючи доказ [1], сформуємо теорему з урахуванням двох систем, зображених на рис. 2. Рівняння ${}_1u(t)\, t \in [t_0, t_1]$, ${}_{11}u(t) t \in [t_1, t_{11}]$, ${}_{12}u(t) t \in [t_1, t_{12}]$, вектори фазових координат ${}_1x(t_0)$, ${}_1x(t_1)$, ${}_{11}x(t_{12})$, ${}_{12}x(t_{12})$, ${}_{11}x(t_{11})$ і момент часу t_0, t_1, t_{11}, t_{12} слід вибирати такими, щоб функціонал I приймав найменше можливе значення.

Теорема. Нехай ${}_1x(t)$, ${}_1u(t)$ $t \in [t_0, t_1]$; ${}_{11}u(t)$, ${}_{12}x(t)$, ${}_{12}u(t)$ $t \in [t_1, t_{12}]$; ${}_{11}x(t_{11})$, ${}_{11}u(t)$ $t \in [t_{12}, t_{11}]$ - допустимі процеси.

Для оптимальності процесів потрібне існування рішень ${}_1\lambda(t)$ $t \in [t_0, t_1]$, ${}^{12}_{11}\lambda(t)$, ${}_{12}\lambda(t)$ $t \in [t_1, t_{12}]$, ${}_{11}\lambda(t) t \in [t_{12}, t_{11}]$, зв'язаних векторних рівнянь.

$$ {}_1\dot{\lambda} + \partial H_1 / \partial {}_1x = 0, $$

$$ {}^{12}_{11}\dot{\lambda} + \partial H^{12}_{11} / \partial {}_{11}x + \partial H_{12} / \partial {}_{11}x = 0, $$

$$ {}_{12}\dot{\lambda} + \partial H^{12}_{11} / \partial {}_{12}x + \partial H_{12} / \partial {}_{12}x = 0, $$

$$ {}_{11}\dot{\lambda} + \partial H_{11} / \partial {}_{11}x = 0 $$

таких, що справедливі умови:

трансверсальності для пов'язаних функцій і гамільтоніанів

$$ \partial S / \partial {}_1x(t_0)\big|_\Lambda - (-1)^\beta {}_1\lambda(\hat{t}_0) = 0, $$

$$ \partial S / \partial t_0\big|_\Lambda + (-1)^\beta H_1\big|_\Lambda = 0, $$

$$ \partial S / \partial {}_{1i}x(t_i)\big|_\Lambda - (-1)^\beta {}_{1i}\lambda(\hat{t}_{1i}) = 0 \quad (i = 1, 2), $$

$$ \partial S / \partial t_{11}\big|_\Lambda - (-1)^\beta H_{11}\big|_\Lambda = 0; $$

стрибка для сполучених функцій та гамільтоніанів

$$\partial S / \partial_1 x(t_1)\big|_\Lambda + (-1)^{\beta}\left[{}_1\lambda(\hat{t}_1) - {}^{12}_{11}\lambda(\hat{t}_1) - {}_{12}\lambda(\hat{t}_1)\right] = 0,$$

$$\partial S / \partial t_1\big|_\Lambda - (-1)^{\beta}\left(H_1\big|_\Lambda - H^{12}_{11}\big|_\Lambda - H_{12}\big|_\Lambda\right) = 0,$$

$$\partial S / \partial_{11} x(t_{12})\big|_\Lambda + (-1)^{\beta}\left[{}^{12}_{11}\lambda(\hat{t}_{12}) - {}_{11}\lambda(\hat{t}_{12})\right] = 0,$$

$$\partial S / \partial t_{12}\big|_\Lambda - (-1)^{\beta}\left(H^{12}_{11}\big|_\Lambda - H_{12}\big|_\Lambda - H_{11}\big|_\Lambda\right) = 0;$$

Мінімум гамільтоніанів у момент часу $t \in [t_h, t_1]$ для рівняння ${}_l u(t) \in \Omega_l$

$$H_1\big|_\Lambda = \min H_l\big|_{\Lambda.{}^l u(t)} \ (l = 1, h = 0; l = 11, h = 12);$$

Мінімум лінійної комбінації гамільтоніанів у моменти часу $t \in [t_1, t_{12}]$ для управління ${}_l u(t) \in \Omega_l$ (l=11,12)

$$H^{12}_{11}\big|_\Lambda + H_{12}\big|_\Lambda = \min\left(H^{12}_{11}\big|_{\Lambda.{}^{11}u(t)\,{}_{12}u(t)} + H_{12}\big|_{\Lambda.{}^{11}u(t)\,{}_{12}u(t)}\right)$$

Тут знак Λ означає оптимальність змінних та параметрів; символ $|_{\Lambda.\xi}$ означає, що вираз визначається за умови оптимальних величинах змінних та параметрів, крім ξ; параметр β приймає значення 1 або 2 відповідно: 1 - для схеми з поділом; 2 – для схеми з групуванням (рис. 2);

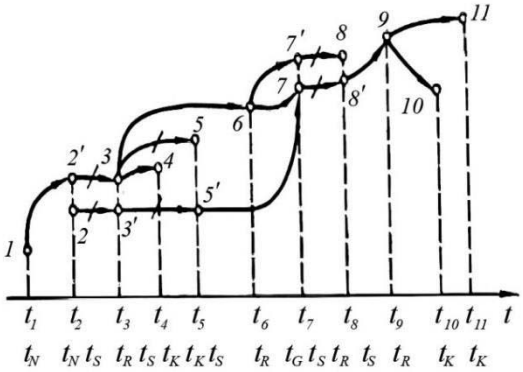


Рис. 3. Часова діаграма розгалуженої траєкторії

$$H_l(\cdot) = \Phi_l(\cdot) + {}_l\lambda^T\, {}_l f(\cdot) \quad (l = 1, 11, 12),$$

$$H^{12}_{11}(\cdot) = \Phi^{12}_{11} + {}^{12}_{11}\lambda^T\, {}^{12}_{11} f(\cdot).$$

Зазначимо, що для механічних систем умова стрибка по n-й фазовій координаті, що позначає масу, має вигляд [4-6]

$$\xi_{11} \ge 0, \ \xi_{12} \ge 0, \ \xi_{11} + \xi_{12} = 1.$$

Доказ теореми можна провести за методом, викладеним у роботі [1] якщо розглядати СДС як систему зі змінною структурою та розміром вектора стану.

Відповідно до викладеної теореми, розглядаючи складну розгалужену траєкторію як сукупність простих, сформулюємо структуру алгоритму моделювання оптимального руху СДС.

Для оптимальності розгалуженої траєкторії довільної схеми необхідне існування в інтервалах часу між моментами t_N (початок руху), t_R (розділення), t_G (групування), t_K (кінця руху підсистеми) розв'язання пов'язаних векторних рівнянь

$$\dot{{}_L\lambda} + \partial H_l / \partial_L x + \sum_q^M \partial H_q / \partial_L x = 0, \quad (3)$$

де L – індекс ділянки розгалуженої траєкторії;

M, q— відповідно число підсистем, динамічні властивості яких залежать від фазових координат L-ї ділянки, і індекси ділянок розгалуженої траєкторії, якими переміщуються ці підсистеми. Для таких рівнянь справедливі умови:

1) трансверсальності в моменти $\hat{t}_N = \hat{\tau}_1$ і $\hat{t}_K = \hat{\tau}_2$

$$\partial S / \partial_L x(\tau_i)\big|_\Lambda - (-1)^i\, {}_L\lambda(\hat{\tau}_i) = 0 \qquad (i = 1, 2), \quad (4)$$

$$\partial S / \partial \tau_i\big|_\Lambda + (-1)^i H_L\big|_\Lambda + \sum_\nu^P \left(H_\nu\big|_{\Lambda, \hat{\tau}_i - 0} - H_{\nu'}\big|_{\Lambda, \hat{\tau}_i + 0}\right) = 0, \quad (5)$$

де P - кількість підсистем, динамічні властивості яких змінюються в момент початку або закінчення руху підсистеми, що переміщається по L-тій ділянці траєкторії;

ν - індекси ділянок розгалуженої траєкторії, якими переміщуються ці підсистеми;

2) стрибка в моменти $\hat{t}_R = \hat{\tau}_1$ і $\hat{t}_G = \hat{\tau}_2$ пов'язані з поділом підсистеми, що переміщається по L-тій ділянці, на r підсистем, або групуванням r-підсистем в підсистему, яка переміщується по L-тій ділянці розгалуженої траєкторії,

$$\partial S / \partial_L x_i(\tau_i)\big|_\Lambda - (-1)^i\, {}_L\lambda_j(\hat{\tau}_i) - (-1)^i \times \sum_{q'}^R {}_{q'}\lambda_j(\hat{\tau}_i) = 0$$

$$(j = 1, n-1; i = 1, 2),$$

$$\partial S / \partial_L x_n(\tau_i)\big|_\Lambda + (-1)^i\, {}_L\lambda_n(\hat{\tau}_i) - (-1)^i \times \sum_{q'}^R \xi_{q'\,q'}\lambda_n(\hat{\tau}_i) = 0, \quad (6)$$

$$\xi_{q'} \geq 0,\ \sum_{q'}^{R} \xi_{q'} = 1\ \ (i = 1,2);$$

$$\partial S / \partial \tau_i \big|_{\Lambda} - (-1)^{\ i} H_L \big|_{\Lambda} + (-1)^i \sum_{q'}^{r} H_{q'} \big|_{\Lambda} +$$

$$+ \sum_{\nu'}^{P'} \left(H_{\nu'} \big|_{\Lambda, \hat{\tau}_i - 0} - H_{\nu'} \big|_{\Lambda, \hat{\tau}_i + 0} \right) = 0,$$

де q індекси ділянок розгалуженої траєкторії, по яких переміщаються підсистеми після поділу або перед групуванням;

P'— кількість підсистем, що не беруть участь у поділі або угрупованні, але змінюють динамічні властивості в моменти часу t_R і t_G;

ν' - індекси ділянок розгалуженої траєкторії, якими переміщуються зазначені підсистеми. Фазова координата ${}_L x_n(t)$, яка описує зміну маси.

Умова стрибка на μ-ій ділянці розгалуженої траєкторії в момент часу $\hat{t}_S$, який співпадає з моментом часу структурних перетворень у СДС, що не належать до μ-тої ділянки, але впливають на нього, має вигляд

$$\partial S / \partial_{\mu} x(t_S) \big|_{\Lambda} - {}_{\mu}\lambda(\hat{t}_S - 0) + {}_{\mu}\lambda(\hat{t}_S + 0) = 0; (8)$$

3) мінімуму лінійної комбінації гамільтоніанів у моменти часу між $\hat{t}_N$, $\hat{t}_R$, $\hat{t}_G$, $\hat{t}_K$

$$\sum_{q''}^{L} \mathrm{H}_{q''} \big|_{\mathrm{L}} = \min_{q'' \mathrm{u} \hat{I} W_{q''}} \sum_{q''}^{L} \mathrm{H}_{q''} \big|_{\mathrm{L}, q'' \mathrm{u}}, \qquad (9)$$

де Λ — кількість підсистем, які мають взаємовпливове керування у вказаних інтервалах часу;

q'' — індекси ділянок розгалуженої траєкторії, за якими ці підсистеми переміщуються.

Сформульоване правило є методологічною основою для синтезу обчислювальних алгоритмів, які дозволяють моделювати оптимальні траєкторії руху різноманітних СДС. Програма моделювання оптимальних розгалужених траєкторій може стати частиною математичного забезпечення системи автоматизованого проєктування перспективних СДС.

Приклад складання структури алгоритму моделювання оптимального руху складової динамічної системи за гілкою траєкторії.

За заданою схемою розгалуженої траєкторії (рис. 1) складемо її часову діаграму (рис. 3), на якій в послідовному порядку розташовані моменти часу структурних перетворень на схемі руху СДС із зазначенням їхньої належності до відповідного моментам часу: t_N, t_R, t_G, t_K. Перекресленою стрілкою відмічені ділянки траєкторії, пересуваючись уздовж яких підсистеми взаємодіють.

Адитивний критерій оптимальності визначається термінальною частиною $S(\cdot)$, яка залежить від координат підсистем в моменти часу t_i $(i = \overline{1,11})$ та цих моментів часу, а також сумою часткових інтегральних критеріїв $I_i = \int_{t_a}^{t_b} \Phi_i(\cdot) dt$ $(i = \overline{1,15})$; $a \neq b$, $a = \overline{1,11}$; $b = \overline{1,11}$), записаних для кожної ділянки гілки траєкторії (див. рис. 3). який закладений між розташованими на ній сусідніми точками. Рух підсистем гілками траєкторії описується рівняннями $\dot{x} = f(\cdot)$, де $f(\cdot)$ — функція, що залежить від керувань і координат підсистеми, а також від керувань і координат взаємодіючої підсистеми. (На ділянках з перекресленими стрілками). Згідно з правилом для оптимальності траєкторії (див. рис. 3) необхідно вирішити 15 сполучених векторних рівнянь типу (3), складених за даними табл. 1, що задовольняють 39 умов типу (4)-(9), які задаються табл. 2 і 3. Як індекс гілки або її ділянки використовується послідовність цифр, відповідних початку і кінцю гілки або її ділянки (див. рис. 3).

Таблиця 1

Умова мінімізації (3)

L	M	q
1, 2′	0	—
2′, 3	1	2, 3′
2′, 3’	1	2′, 3
3, 4	0	—
3, 6	0	—
3′, 5′	1	3, 5
3, 5	1	3′, 5′
5′, 7	0	—
6, 7	0	—
6, 7′	0	—

7′, 8	1	7, 8′
7, 8’	1	7′, 8
8′, 9	0	—
9, 10	0	—
9, 11	0	—

Таблиця 2

Умова мінімізації (9)

Інтервал	Λ	q''
$[t_1, t_2]$	1	1, 2′
$[t_2, t_3]$	2	2′, 3; 2, 3′
$[t_3, t_4]$	1	3, 4
$[t_3, t_5]$	2	3, 5; 3′, 5′
$[t_3, t_6]$	1	3, 6
$[t_5, t_7]$	1	5′, 7
$[t_6, t_7]$	1	6, 7
$[t_6, t_7]$	1	6, 7′
$[t_7, t_8]$	2	7′, 8; 7, 8′
$[t_8, t_9]$	1	8′, 9
$[t_9, t_{10}]$	1	9, 10
$[t_9, t_{11}]$	1	9, 11

Для завершення розв'язання задачі моделювання оптимальної розгалуженої траєкторії необхідно доповнити перераховані диференціальні рівняння і алгебраїчні диференціальні рівняння руху підсистем гілками траєкторії. Дані таблиці 1-3 є вихідною інформацією, яка дозволяє перейти до застосування стандартних підпрограм вирішення звичайних диференціальних рівнянь і алгебраїчних рівнянь і тим самим практично завершити рішення задачі моделювання оптимальної траєкторії СДС.

Зазначимо, що послідовність моментів часу $t_1 < t_2 < ... < t_{11}$ у задачі з вільним часом задається, виходячи з фізичних міркувань, і є наближеною. Якщо в результаті розв'язання задачі вона порушується (зміна послідовності розгалужень траєкторії допустимо фізичним змістом завдання), необхідно повторити розрахунки для нової уточненої послідовності моментів часу.

Таблиця 3

Умова трансверсальності стрибка в момент часу t_i

Рівняння	t_1	t_2	t_3	t_4	t_5
(4)	L=1, 2′; i=1	L=2, 3′; i=1	—	L=3, 4; i=2	L=3, 5; i=2
(5)	L=1, 2′; i=1; P=0	L=2, 3′; i=1; P=1; v=1, 3	—	L=3, 4; i=2; P=0	L=3, 5; i=2; P=1; v=3′, 7
(6)	—	—	L=2, 3′; i=1; r=3; q′=3,6; 3,4; 3, 5	—	—
(7)	—	—	L=2′, 3; i=1; r=3; q′=3, 6; 3, 5; 3, 4; P′=1; v′=2, 5′	—	—
(8)	—	μ=1, 3	μ=2, 5′	—	μ=3′, 7

t_6	t_7	t_8	t_9	t_{10}	t_{11}
—	—	L=7′, 8; i=2	—	L=9, 10; i=2	L=9, 11; i=2
—	—	L=7′, 8; P=1 v=7, 9	—	L=9, 10; i=2	L=9, 11; i=2
L=3, 6; i=1; r=2; q′=6, 7′; 6, 7	L=7, 8′; i=2; r=2; q′=6, 7; 5′, 7	—	L=8′, 9; i=1; r=2; q`=9, 10; 9, 11	—	—
L=3, 6; i=1; r=2; q′=6, 7′; 6, 7	L=7, 8′; i=2; r=2; q′=6, 7; 5′, 7; P′=1; v=6, 8	—	L=8′, 9; i=1; r=2; q′=9, 10; 9, 11; P′=0	—	—
—	μ=6, 8	μ=7, 9	—	—	—

На закінчення зазначимо, що завдання оптимізації розгалуженої траєкторії з різними формами обмежень за допомогою відомих методів [3] може бути приведене до виду, розглянутого в даній статті.

Висновки

1. Розроблена структура алгоритму моделювання оптимального руху складеної динамічної систем розгалуженою траєкторією. Алгоритм дозволяє в реальному часі оптимізувати розгалужені траєкторії руху для реалізації цільового призначення СДС та виконувати оперативну корекцію траєкторій руху підсистем при виникненні непередбачуваних на попередньому етапі критичних факторів впливу.

2. Розроблені достатні умови оптимальності є універсальними для планування траєкторій із будь-яким обмеженим числом гілок траєкторії та різноманітними математичними моделями складених систем. Це дозволяє суттєво знизити обчислювальні затрати під час розрахунків оптимального керування в умовах невизначеності початкових умов для структурних перетворень СДС і зшивання траєкторій.

3. Необхідні умови оптимальності мають чіткий фізичний зміст і є технологічними та зручними для використання.

4. Результати досліджень є важливими і актуальними для побудови законів траєкторного керування існуючими і перспективними СДС.

Література

1. Теорія оптимальних розгалужених траєкторій: монографія / О. І. Лисенко, О. М. Тачиніна, С. О. Пономаренко, О. Г. Гуйда. Київ: КПІ ім. Ігоря Сікорського, 2023. 260 с. URI: https://ela.kpi.ua/handle/123456789/52094.
2. Tachinina O., Lysenko O. Methods for the Synthesis of Optimal Control of Deterministic Compound Dynamical Systems With Branch. Handbook of Research on Artificial Intelligence Applications in the Aviation and Aerospace Industries. 2020. P. 323–351. URL: https://doi.org/10.4018/978-1-7998-1415-3.ch014
3. Using Krotov's Functions for the Prompt Synthesis Trajectory of Intelligent Info-communication Robot / O. Tachinina et al. Studies in Systems, Decision and Control. Cham, 2023. P. 255–283. URL: https://doi.org/10.1007/978-3-031-43579-9_6
4. Імпульсні диференціальні рівняння з багатозначною та розривною правою частиною: монографія / Н.А. Перестюк, В.А. Плотніков, А.М. Самойленка, Н.В. Скрипник. - К., 2007. - 427 с.
5. Alekseeva I. V., Lysenko O. I., Tachinina O. M. Necessary optimality conditions of control of stochastic compound dynamic system in case of full in-formation about state vector. Mathematical machines and systems. 2020. Vol. 4. P. 136–147. URL: https://doi.org/10.34121/1028-9763-2020-4-136-147
6. Lysenko O. I., Tachinina O. M., Alekseeva I. V. ALGORITHM OF OPTIMAL CONTROL OF UAV GROUP. Electronics and Control Systems. 2018. Vol. 2, no. 56. URL: https://doi.org/10.18372/1990-5548.56.12945
7. GIUNTI M., MAZZOLA C. DYNAMICAL SYSTEMS ON MONOIDS: TOWARD A GENERAL THEORY OF DETERMINISTIC SYSTEMS AND MOTION. Methods, Models, Simulations And Approaches Towards A General Theory Of Change. 2012. P. 173–185. URL: https://doi.org/10.1142/9789814383332_0012 .

References

1. Lysenko O. I., Tachynina O. M., Ponomarenko S. O. & Guida O. G. (2023) Theory of optimal branched trajectories: monograph. Kyiv: KPI Igor Sikorskyi, 2023. 260 p. URI: https://ela.kpi.ua/handle/123456789/52094 [in Ukrainian]
2. Tachinina, O. & Lysenko, O., (2020). Methods for the Synthesis of Optimal Control of Deterministic Compound Dynamical Systems With Branch. У: Handbook of Research on Artificial Intelligence Applications in the Aviation and Aerospace Industries. IGI Global. p. 323–351. URI: doi.org/10.4018/978-1-7998-1415-3.ch014
3. Tachinina, O., Lysenko, O., Romanchenko, I., Novikov, V. & Sushyn, I., (2023). Using Krotov's Functions for the Prompt Synthesis Trajectory of Intelligent Info-communication Robot. У: Studies in Systems, Decision and Control. Cham: Springer Nature Switzerland. p. 255–283. URI: doi.org/10.1007/978-3-031-43579-9_6
4. Impulse differential equations with multivalued and discontinuous right-hand side:

Monograph / N.A. Perestyuk, V.A. Plotnikov, A.M. Samoilenko, N.V. Violinist - K., 2007. - 427 p.
5. Alekseeva, I. V., Lysenko, O. I. & Tachinina, O. M., (2020). Necessary optimality conditions of control of stochastic compound dynamic system in case of full in-formation about state vector. Mathematical machines and systems. 4, 136–147. URI: doi.org/10.34121/1028-9763-2020-4-136-147
6. Lysenko, O. I., Tachinina, O. M. & Alekseeva, I. V., (2018). Algorithm of Optimal Control of UAV Group. Electronics and Control Systems. 2(56). URI: doi.org/10.18372/1990-5548.56.12945
7. GiuntI, M. & Mazzola, C., (2012). Dynamical Systems on Monoids: Toward a General Theory of Deterministic Systems and Motion: Methods, Models, Simulations And Approaches Towards A General Theory of Change.World Scientific. p. 173-185. URI: doi.org/10.1142/9789814383332_0012

Одержано: 10.04.2024
Внутрішня рецензія отримана: 16.04.2024
Зовнішня рецензія отримана: 25.04.2024

Про авторів:

[1]Лисенко Олександр Іванович,
Доктор технічних наук, професор.
https://orcid.org/0000-0002-7276-9279.

[2]Шевченко Віктор Леонідович,
доктор технічних наук, професор.
https://orcid.org/0000-0002-9457-7454.

[3]Тачиніна Олена Миколаївна,
Доктор технічних наук, професор.
https://orcid.org/0000-0001-7081-0576.

[1]Пономаренко Сергій Олексійович,
Кандидат технічних наук,
старший науковий співробітник.
https://orcid.org/0000-0001-5512-3778

[4]Гуйда Олександр Григорович,
кандидат наук з державного управління, доцент.
https://orcid.org/0000-0002-2019-2615

Місце роботи авторів:

[1]Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»,
Тел.: +38-044-204-94-94
E-mail: mail@kpi.ua
https://kpi.ua/

[2]Інститут програмних систем
НАН України,
тел. +38-044-522-62-42
E-mail: iss@isofts.kiev.ua
https://iss.nas.gov.ua/

[3]Національний авіаційний університет
Тел: +38-044-406-79-01
E-mail: post@nau.edu.ua
https://nau.edu.ua/

[4]Таврійський національний університет імені В.І.Вернадського,
Тел.: +38-044-529-05-16
E-mail: crimea.tnu@gmail.com
http://tnu.edu.ua/

УДК 518:517.944 519.6 http://doi.org/10.15407/pp2024.02-03.078

О.С. Дученко, А.Н. Нестеренко, О.В. Попов

МОДЕЛЮВАННЯ ПРОЦЕСІВ ТЕПЛОПЕРЕНОСУ НА КОМП'ЮТЕРАХ ГІБРИДНОЇ АРХІТЕКТУРИ

Робота присвячена розробленню методів і алгоритмів комп'ютерного моделювання процесів теплопереносу, які відіграють суттєву роль в галузі механіки суцільних середовищ. Знаходження температурних полів у процесі зварювання для визначення кінетики температурних напружень і деформацій сприяє підвищенню технолого-експлуатаційних показників зварних конструкцій. У роботі запропоновано методику визначення температурних полів під час зварювання тонколистових елементів конструкцій. В результаті моделювання процесу теплопереносу, враховуючи нерівномірність нагрівання зварюваних елементів у процесі зварювання, отримано нестаціонарну теплову задачу. Для чисельного розв'язання цієї задачі методом скінченних елементів отримано дискретну задачу – систему звичайних диференціальних рівнянь першого порядку, тобто задачу Коші. В роботі запропоновано високопродуктивний комп'ютерний алгоритм методу Рунге-Кутта четвертого порядку для розв'язування задачі Коші. Алгоритм розроблено для сучасних суперкомп'ютерів, побудованих на нових багатоядерних процесорах. Запропонований алгоритм базується на багаторівневій моделі паралельних обчислень. На верхньому рівні паралелізму обчислення проводяться у вигляді паралельних процесів, між якими розподіляється пам'ять, забезпечується синхронізація обчислень та обмін інформацією. Розпаралелювання на цьому рівні доцільно здійснювати засобами MPI. На другому рівні паралелізму обчислення проводяться за допомогою директив OpenMP або програмних модулів бібліотеки Intel MKL з використанням спільної пам'яті. Третій рівень паралелізму передбачає обробку даних векторними арифметико-логічними пристроями, включення яких до програми відбувається автоматично за допомогою компілятора. Розроблений алгоритм апробовано на розв'язанні задачі визначення поля температур, яка виникає під час дослідження життєвого циклу зварних конструкцій. Такі розрахунки використовуються, зокрема, в задачі визначення кінетики температурних деформацій та напружень у процесі зварювання тонких пластин. Наведено деякі результати тестування запропонованого методу та розробленого алгоритму на суперкомп'ютері Інституту кібернетик імені В.М. Глушкова НАН України СКІТ.

Ключові слова: рівняння теплопровідності, температурне поле, задачі Коші для систем звичайних диференціальних рівнянь.

O.S. Duchenko, A.N. Nesterenko, O.V. Popov

MODELING OF HEAT TRANSFER PROCESSES ON HYBRID ARCHITECTURE COMPUTERS

The paper is devoted to the development of methods and algorithms for computer simulation of heat transfer processes, which play a significant role in the field of continuum mechanics. Finding the temperature fields during welding to determine the kinetics of temperature stresses and deformations helps to increase the technological and operational indicators of welded structures. This paper proposes a method of determining temperature fields during welding of thin-sheet elements of structures. This paper proposes a method of determining temperature fields during welding of thin-sheet elements of structures. A non-stationary thermal problem was obtained as a result of the simulation of the heat transfer process, taking into account the nonuniformity of the heating of the welded elements during the welding process. A discrete problem was obtained for the numerical solution of this problem by the finite element method – a first-order system of ordinary differential equations, that is, the Cauchy problem. This paper proposes a high-performance fourth-order Runge-Kutta computer algorithm for solving the Cauchy problem. The algorithm is developed for modern supercomputers built on new multi-core processors. The proposed algorithm is based on a multilevel model of parallel computing. At the upper level of parallelism, calculations performs in form of parallel processes, memory is distributed between them, also ensures synchronization of calculations and information exchanges. Parallelization at this level is expedient to be carried out by means of MPI. At the second level of parallelism, calculations performs using OpenMP directives or Intel MKL library software modules using shared memory. The third level of parallelism involves data processing with vector arithmetic and logic devices, which are included in the program automatically with the help of the compiler. The developed algorithm was tested on solving the problem of determining the temperature field, which arises during the

study of the life cycle of welded structures. Such calculations are used, in particular, in the problem of determining the kinetics of temperature deformations and stresses during welding of thin plates. Some results of testing the proposed method and the developed algorithm on the SKIT supercomputer of the V.M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine are presented.
Key words: thermal conductivity equation, temperature field, Cauchy problems for systems of ordinary differential equations.

Вступ

Комп'ютерне моделювання процесів теплопереносу відіграє суттєву роль в галузі механіки суцільних середовищ. Так термодеформаційні процеси та перетворення в металах визначають технологічну міцність зварного шва та зони термічного впливу. Тому знаходження температурних полів під час зварювання для визначення кінетики температурних напружень і деформацій сприяє підвищенню технологоексплуатаційних показників зварних конструкцій. В результаті моделювання процесу теплопереносу, враховуючи нерівномірність нагрівання зварюваних елементів в процесі зварювання, отримано нестаціонарну теплову задачу.

Для чисельного розв'язання цієї нестаціонарної задачі методом скінченних елементів отримано дискретну задачу – задачу Коші для системи звичайних диференціальних рівнянь першого порядку. Як правило такі розрахункові задачі мають дуже велику кількість рівнянь – від десятків тисяч до мільйонів. Тому для комп'ютерного моделювання процесів теплопереносу доцільно використовувати високопродуктивні обчислювальні системи. В [1] запропоновано методи та паралельні алгоритми розв'язування задач Коші на комп'ютерах MIMD-архітектури. Сучасні суперкомп'ютери, побудовані на нових багатоядерних процесорах, дають можливість підвищення продуктивності за рахунок використання багаторівневої моделі паралельних обчислень.

Верхній рівень багаторівневої моделі паралельних обчислень – паралелізм рівня обчислювальних вузлів (process level parallelism – PLP), який дає можливість проводити обчислення у вигляді паралельних процесів, передбачає використання розподіленої між паралельними процесами пам'яті, забезпечує синхронізацію обчислень та обмін інформацією між процесами. Другий рівень – паралелізм рівня потоків (thread level parallelism – TLP) передбачає використання потоків, які працюють із загальною пам'яттю. Наступний рівень – паралелізм обробки даних векторними арифметико-логічними пристроями (data level parallelism – DLP) дає можливість автоматичного включення паралелізму до програми за допомогою компілятора.

Таким чином для отримання ефективних алгоритмів необхідно:

1. розділити задачу на підзадачі так, щоб оптимізувати комунікаційні витрати;
2. вибрати архітектуру паралельного комп'ютера;
3. врахувати структуру пам'яті процесорних пристроїв;
4. розподілити дані та обчислення між процесами, забезпечивши рівномірне завантаження обчислювальних вузлів комп'ютера.

Вихідна задача. Розглянемо задачу знаходження нестаціонарного поля температур для тонкої пластини.

У загальному вигляді температурне поле описується нестаціонарним нелінійним рівнянням теплопровідності. Для тонкої пластини можна знехтувати нерівномірністю поля температур по товщині пластини. Тоді тривимірне рівняння теплопровідності зводиться до двовимірного:

$$c\frac{\partial T}{\partial t}=\frac{\partial}{\partial x}\left(\lambda\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(\lambda\frac{\partial T}{\partial y}\right)+\mathrm{w}, \qquad (1)$$

де c – питома теплоємність (в $Дж/см^3$), λ – теплопровідність матеріалу (в $Дж/(см\cdot с)$), w – об'ємна потужність джерела тепла (в $Дж/(см^3\cdot с)$). Теплофізичні характеристики (c і λ) багатьох матеріалів у широкій області температур відомі з довідників.

Оскільки процес плавлення та затвердіння в багатьох випадках відбувається в інтервалі температур від T_S (темпера-

тура твердої фази) до T_L (температура рідкої фази) і виділення прихованої теплоти плавлення здійснюється шляхом збільшення питомої теплоємності в області фазового перетворення, то питома теплоємність визначається наступним чином:

$$c = \begin{cases} c(T), & T \le T_S, \\ c(T_S) + \dfrac{q_{ск}}{T_L - T_S}, & T_S < T < T_L, \\ c(T_L), & T > T_L, \end{cases}$$

де $q_{ск}$ – прихована теплота плавлення (в Дж/(К·см3)).

В околі місцезнаходження джерела тепла утворюється рідка ванна розплаву, де процес переносу тепла відбувається як за рахунок теплопровідності, так і за рахунок перемішування. Для врахування впливу перемішування в цій ділянці використовують ефективний коефіцієнт теплопровідності:

$$\lambda = \begin{cases} \lambda(T), & T \le T_S, \\ n\lambda(T_S), & T > T_S \quad (n = 3 \div 5). \end{cases}$$

За граничні умови приймається закон конвективного теплообміну тіла з навколишнім середовищем:

$$-\lambda \cdot \frac{\partial T}{\partial n} = \alpha \cdot (T - T_0), \qquad (2)$$

де α – коефіцієнт конвективного теплообміну, T_0 – температура навколишнього середовища, похідна температури береться по зовнішній нормалі до контуру пластини.

Як початкова умова приймається рівність температури тіла і навколишнього середовища в початковий момент часу t_0:

$$T(t_0) = T_0. \qquad (3)$$

Таким чином, з урахуванням конвективного потоку тепла з поверхні рівняння теплопровідності (1) має вигляд:

$$c\frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(\lambda\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\lambda\frac{\partial T}{\partial y}\right) - \frac{2\alpha(\mathrm{T} - \mathrm{T}_0)}{\delta} + \mathrm{w} \qquad (4)$$

де δ – товщина пластини.

Будемо вважати, що об'ємна потужність джерела за товщиною розподіляється рівномірно. Якщо початок координат розмістити у точці розташування цього джерела, то об'ємна потужність джерела обчислюється за формулою:

$$w = q_0 e^{-\left(K_x x^2 + K_y y^2\right)},$$

де K_x, K_y − коефіцієнти зосередженості теплового потоку, а $q_0 = Q\dfrac{\sqrt{K_x K_y}}{\pi\delta}$ (Q – ефективна потужність джерела).

Початково-крайова задача (2)-(4) може бути сформульована для функції $u \in U_0(\Omega)\times[t_0, t_f]$ у варіаційній формі:

$$\forall v \in U_0 \quad d(u', v) + a(u, v) = l(w, v), \quad u(t_0) = 0, \qquad (5)$$

де $u = T - T_0$, u' – перша похідна функції u за часом t, $d(z,v) = \iint\limits_\Omega czv dx$, $a(u,v) =$

$$= \iint\limits_\Omega \left[\lambda\left(\frac{\partial v}{\partial x_1}\frac{\partial u}{\partial x_1} + \frac{\partial v}{\partial x_2}\frac{\partial u}{\partial x_2}\right) + \frac{2\alpha}{\delta}vu\right]dx +$$

$$+ \int\limits_\Gamma \alpha v u d\gamma, \quad l(w,v) = \iint\limits_\Omega vw dx$$ – лінійні по функції v функціонали.

Дискретна задача. Для розв'язання варіаційної задачі (5) можна використати метод скінчених елементів (МСЕ). Якщо $U_0^h \subset U_0$ – скінченновимірний підпростір МСЕ, і $v^h(x) \in U_0^h$, то дискретна варіаційна нестаціонарна задача має вигляд: $\forall v^h \in U_0^h$

$$d\left((u^h)', v^h\right) + a(u^h, v^h) = l(w, v^h), \quad u^h(t_0) = 0. \qquad (6)$$

Запишемо $u^h(x,t) = \sum\limits_{i=1}^{n} U_i(t)\varphi_i(x)$ і, $v^h(x) = \sum\limits_{i=1}^{n} V_i\varphi_i(x)$, де $\{\varphi_i\}_{i=1}^n$ – множина базисних функцій підпростору U_0^h. Тоді $d((u^h)', v^h) = V^T MU'$, $a(u^h, v^h) = V^T KU$, $l(w, v^h) = V^T W$ і замість задачі (6) отримуємо задачу Коші для системи звичайних диференціальних рівнянь (СЗДР)

$$M\frac{dU}{dt}+KU-Y=0,\quad U(t_0)=0 \qquad (7)$$

або

$$\frac{dU}{dt}=-M^{-1}KU+M^{-1}W,\quad U(t_0)=0\,. \quad (8)$$

Тут матриця M – зазвичай діагональна, а матриця K – стрічкова.

Постановка задачі Коші для СЗДР. Задачу Коші для СЗДР на інтервалі $[t_0, T]$ розглядатимемо у вигляді:

$$\frac{dy}{dt}=f(t,y), \qquad (9)$$

$$y(t_0)=y_0\ , \qquad (10)$$

де $y=(y_1,y_2,\ldots,y_n)^T$ – шуканий вектор розв'язку.

В багатьох практичних задачах (9) можна записати у матричній формі (8) з діагональною матрицею M і матрицею K розрідженої структури (причому часто стрічкової).

Для розв'язання зазначеної СЗДР використаємо метод Рунге-Кутта 4-го порядку, який є найбільш розповсюдженим методом чисельного розв'язування задач з початковими умовами для СЗДР та реалізується за формулами:

$$y^{(i+1)}=y^{(i)}+(k_1+2k_2+2k_3+k_4)/6, \qquad (11)$$

де

$$\begin{aligned}
k_1&=h_i f(t_i,y^{(i)}),\\
k_2&=h_i f(t_i+h_i/2,y^{(i)}+0{,}5k_1),\\
k_3&=h_i f(t_i+h_i/2,y^{(i)}+0{,}5k_2),\\
k_4&=h_i f(t_i+h_i,y^{(i)}+k_3),
\end{aligned} \qquad (12)$$

$i=0,1,2,\ldots$ верхній індекс – номер точки, нижній індекс – номер компоненти вектора.

Отже, для знаходження розв'язку СЗДР (9)-(10) методом Рунге-Кутта 4-го порядку з автоматичним вибором кроку інтегрування для досягнення заданої точності розв'язку можна виділити наступні підзадачі [2]:

1) обчислення вектор-функції $f(t,y)$ для знаходження векторів k_1, k_2, k_3, k_4 за формулами (12);

2) обчислення вектора розв'язку $y=(y_1,y_2,\ldots,y_n)^T$ у наступній точці інтегрування за формулою (11);

3) обчислення похибки отриманого розв'язку, перевірка умови досягнення заданої точності та корекція кроку інтегрування, якщо задана точність не досягнута;

4) обчислення константи Ліпшиця.

Паралельний алгоритм методу Рунге-Кутта 4-го порядку точності. Реалізація методу Рунге-Кутта 4-го порядку для розв'язування СЗДР з використанням багаторівневого паралелізму передбачає автоматичний розподіл вихідних даних та компонент векторів правих частин і розв'язків у локальній пам'яті ядер кожного процесора, а також автоматичний вибір довжини машинного слова та кроку інтегрування для отримання заданої точності розв'язку.

Для збалансованого завантаження обчисленнями процесорів використовується одновимірний блочний розподіл даних та обчислень. Водночас розмір блоків обчислюється за формулою:

$$s_q=\begin{cases}\lceil n/p\rceil+1, & q<n-p\lceil n/p\rceil\\ \lceil n/p\rceil, & q\ge n-p\lceil n/p\rceil\end{cases},$$

де n – порядок СЗДР, p – кількість процесів, $q=0,1,\ldots,p-1$ – логічний номер процесу, значення $\lceil a\rceil$ дорівнює цілій частині числа a. За цією схемою розподіляються компоненти векторів y та $f(t,y)$. Така схема дозволяє виконувати автоматичний розподіл обчислень компонент вектор-функції СЗДР між p процесорами [3,4].

Ще однією перевагою такої схеми розподілу є те, що в багатьох випадках обміни даними здійснюються лише між процесорами, які містять сусідні блоки.

Запропонований паралельний алгоритм методу Рунге-Кутта 4-го порядку реалізується наступним чином.

1. На рівні паралелізму обчислювальних вузлів із розподіленою пам'яттю (I-й рівень), використовуючи p MPI-процесів, реалізуються підзадачі 1–3 на кожному кроці інтегрування алгоритму, а підзадача 4 – по завершенні процесу інтегрування. Для підзадач 1–3 виконуються обміни даними, використовуючи стандартні функції MPI.

2. Кожна з підзадач 1–3, як правило, зводиться до виконання однорідних операцій над великим обсягом даних (переважно це множення розрідженої матриці на вектор та додавання векторів). Тому тут доцільно використати спільну пам'ять, реалізувавши рівень паралелізму потоків (II-й рівень) в середовищі OpenMP. Тут для виконання кожної з підзадач 1–3 виконується розпаралелення між деякою кількістю потоків (threads). Слід зауважити, що обчислення значень компонент вектор-функції $f(t,y)$ (підзадача 1), як правило, через достатньо високий порядок векторів потребує значно більшої кількості арифметичних операцій – лише у такому випадку застосування другого рівня паралелізму має сенс. З іншого боку, якщо для збереження даних і виконання обчислень достатньо одного вузла із спільною пам'яттю, то доцільно перший рівень не використовувати.

3. На рівні паралелізму обробки даних (III-й рівень) розпаралелення виконується за рахунок використання опцій компілятора.

Слід зазначити, що кількість арифметичних операцій, необхідних для обчислення компонент правої частини СЗДР (вектор-функції $f(t,y)$) істотно впливає на ефективність і прискорення обчислень. Тому поряд із розпаралеленням обчислення правої частини СЗДР для підвищення ефективності алгоритму і забезпечення необхідного рівня похибок доцільно для цих операцій використати змішану розрядність. Тобто залежно від властивостей оператора задачі здійснити обчислення або на одинарній розрядності (*float*), або на подвійній (*double*), або на ще вищій розрядності.

В таблиці 1 представлено час розв'язання теплової задачі на суперкомп'ютері СКІТ Інституту кібернетики імені В.М. Глушкова НАН України. СЗДР порядку $n = 28\,471$ (n_x=400; n_y=70; $n=n_x \times n_y$) розв'язувалась на інтервалі [0,1]. У другій колонці таблиці наведено часи розв'язування задачі з використанням лише першого рівня паралелізму; у третій – першого та другого рівнів; у четвертій – усіх трьох рівнів паралелізму.

Результати, наведені у таблиці, свідчать, що розроблений паралельний алгоритм дозволяє значно зменшити час розв'язування задачі (у 7,67 раз у порівнянні з однопроцесорним варіантом). Крім того, зростання часу розв'язування задачі на дванадцяти процесах свідчить, що оптимальною архітектура комп'ютера для розв'язування СЗДР порядку $n = 28\,471$ є вісім процесів.

Таблиця 1

Часи розв'язування теплової задачі

Кількість MPI-процесів	Час (сек)		
	I рівень	I і II рівні	Три рівні
1	1,84	1,5	0,77
2	0,96	0,78	0,42
4	0,48	0,43	0,29
8	0,32	0,32	0,22
12	0,29	0,31	0,24

Висновки

Використовуючи багаторівневу модель паралельних обчислень з урахуванням особливостей архітектури комп'ютера та структури даних задачі розроблено ефективний алгоритм розв'язування СЗДР, який програмно реалізовано на паралельних комп'ютерах гібридної архітектури. Водночас час розв'язання задач суттєво скорочується, що дає можливість розв'язувати задачі високих порядків у реальному часі, які висуває сучасне життя перед наукою.

Література

1. А.Н. Химич, И.Н. Молчанов, А.В. Попов, Т.В. Чистякова, М.Ф. Яковлев, Параллельные алгоритмы решения задач вычислительной математики, Наукова думка, Киев, 2008.
2. М.Ф. Яковлев, В.М. Бруснікін, Т.О. Герасимова, Розв'язування задач з початковими умовами для систем звичайних диференціальних рівнянь на багатоядерному комп'ютері з графічними прискорювачами Інпарком, *Математичні машини і системи*, 2015. №2. С. 20–27.
3. М.Ф. Яковлев, Т.О. Герасимова, А.Н. Нестеренко, Особливості розв'язування систем

нелінійних та диференціальних рівнянь на паралельних комп'ютерах, *Питання оптимізації обчислень* (*ПОО – XXXV*). *Праці міжнародного симпозіуму*. – Київ: Інститут кібернетики ім. В.М. Глушкова НАН України, 2009. № 2. С. 435–439.
4. А.Н. Химич, М.Ф. Яковлев, Т.А. Герасимова, Некоторые вопросы решения систем обыкновенных дифференциальных уравнений на MIMD-компьютерах, *Кибернетика и системный аналіз*, 2007. № 2. С. 175-182. https://doi.org/10.1007/s10559-007-0049-3

References

1. A.N. Khimich, I.N. Molchanov, A V. Popov, T.V. Chistyakova, and M.F. Yakovlev, Parallel Algorithms for the Solving of Computational Mathematics Problems. [in Russian], Naukova Dumka, Kyiv, 2008.
2. M. Yakovlev, V. Brusnikin, T. Gerasymova, The solving of initial-value problems for systems of ordinary differential equations on multi-core computer with graphic accelerators Inparcom, in: Mathematical machines and systems, 2015, pp. 20–27.
3. M. Yakovlev, T. Gerasymova, A. Nesterenko, Characteristic features of the solving both of non-linear systems and systems of ordinary differential equations on parallel computers, in: Proceedings of the international symposium "Optimization problems of computations" (OPC – XXXV). Kyiv: V.M. Glushkov Institute of cybernetics of NAS of Ukraine, 2009, Kyiv: Vol. 2. pp. 435-439.
4. A. Khimich, M. Yakovlev, T. Gerasymova, Some questions related to the solving of systems of ordinary differential equations on MIMD computers, in Cybernetics and system analysis. 2007, (2). pp. 175-182. https://doi.org/10.1007/s10559-007-0049-3

Одержано: 10.04.2024
Внутрішня рецензія отримана: 21.04.2024
Зовнішня рецензія отримана: 29.04.2024

Про авторів:

[1]*Дученко Олександр Сергійович*,
молодший науковий співробітник.
http://orcid.org/0009-0009-4157-1743.

[2]*Нестеренко Алла Никифорівна*,
молодший науковий співробітник.
http://orcid.org/0000-0001-6174-1812.

[3]*Попов Олександр Володимирович*,
доктор фізико-математичних наук,
старший науковий співробітник,
провідний науковий співробітник,
http://orcid.org/0000-0002-1217-2534.
alex50popov@gmail.com
тел. (067)-234-12-08

Місце роботи авторів:

[1]Інститут кібернетики ім. В.М. Глушкова НАН України
Тел. (+38) (044) 526-20-08
E-mail: incyb@incyb.kiev.ua,
https:\\www.incyb.kiev.ua

[2]Інститут кібернетики ім. В.М. Глушкова НАН України,
Тел. (+38) (044) 526-20-08
E-mail: incyb@incyb.kiev.ua,
https:\\www.incyb.kiev.ua

[3]Інститут кібернетики ім. В.М. Глушкова НАН України,
Тел. (+38) (044) 526-20-08
E-mail: incyb@incyb.kiev.ua,
https:\\www.incyb.kiev.ua

УДК 004.4:004.942 http://doi.org/10.15407/pp2024.02-03.084

Ю.І. Сивицький, В.Л. Шевченко

КОМП'ЮТЕРНА МОДЕЛЬ ТРАНСФОРМАЦІЇ ОРГАНІЗАЦІЇ

Стаття присвячена актуальному питанню створення комп'ютерної системи підтримки управлінських рішень щодо оптимізації процесів трансформації організацій при адаптації під умови виконання нових проєктів. Метою статті є підвищення ефективності діяльності великих організаційних структур за рахунок створення комп'ютерних моделей, які з одного боку мають достатній рівень адекватності, а з іншого - мають наочну інтерпретацію основних вхідних параметрів, що дозволяє легко їх визначати на основі емпіричних даних. За основу системи підтримки управлінських рішень розглядається комп'ютерна модель у вигляді імітаційної моделі. В роботі виконаний аналіз існуючих досліджень, обґрунтована актуальність задачі. Виконаний аналіз існуючих експоненціальних та лінійних моделей. Обґрунтована адекватність логістичних моделей. Розглянута диференціальна форма логістичних моделей. Звичайне диференціальне рівняння логістичної моделі розв'язане з метою отримання інтегральної форми логістичного рівняння. Введені параметри комп'ютерної моделі, які легко визначати чисельно на основі емпіричних даних. Створена математична модель корисного ефекту організації в умовах трансформації, яка встановлює залежність корисного ефекту від вхідного ресурсу (часу). Модель створена у вигляді комбінації декількох логістичних залежностей, кожна з яких відповідає за зростання або за зменшення корисного ефекту. В моделі враховані залежності зростання корисного ефекту для основної та нової технологій, зниження корисного ефекту внаслідок морального старіння технології та поступове зниження корисного ефекту внаслідок директивного вимкнення старої технології. Структура моделі дозволяє її масштабування на більш складні сценарії розвитку. Введене поняття ступеня нечутливості корисного ефекту до невеликих величин вхідних ресурсів на початкових етапах розвитку організації. Досліджена залежність вихідного результату від ступеня нечутливості. Модель реалізована алгоритмічною мовою MatLab.

Ключові слова: Комп'ютерна модель, імітаційна модель, логістична залежність, корисний ефект, ресурс, підтримка управлінських рішень, автоматизація, оптимізація

Yu. Syvytskyi, V. Shevchenko

COMPUTER MODEL OF ORGANIZATION TRANSFORMATION

The article is devoted to the topical issue of creating a computer system to support management decisions regarding the optimization of the transformation processes of organizations when adapting to the conditions of the implementation of new projects. The purpose of the article is to increase the efficiency of large organizational structures by creating computer models that, on the one hand, have a sufficient level of adequacy, and on the other hand, have a visual interpretation of the main input parameters, which allows them to be easily determined on the basis of empirical data. A computer model in the form of a simulation model is considered as the basis of the management decision support system. In the work, the analysis of existing studies is performed, the relevance of the problem is substantiated. The analysis of existing exponential and linear models was performed. Reasoned adequacy of logistic models. The differential form of logistic models is considered. The ordinary differential equation of the logistic model is solved in order to obtain the integral form of the logistic equation. The parameters of the computer model are introduced, which are easy to determine numerically on the basis of empirical data. A mathematical model of the beneficial effect of the organization in the conditions of transformation was created, which establishes the dependence of the beneficial effect on the input resource (time). The model is created as a combination of several logistic dependencies, each of which is responsible for increasing or decreasing the beneficial effect. The model takes into account the dependences of the growth of the useful effect for the main and new technologies, the decrease of the useful effect as a result of the moral obsolescence of the technology, and the gradual decrease of the useful effect due to the directive shutdown of the old technology. The structure of the model allows its scaling to more complex development scenarios. The concept of the degree of insensitivity of the useful effect to small amounts of input resources at the initial stages of the organization's development is introduced. The dependence of the initial result on the degree of insensitivity was studied. The model was implemented using the MatLab algorithmic language.

Keywords: Computer model, simulation model, logistic dependence, useful effect, resource, management decision support, automation, optimization

Вступ

Впровадження автоматизованих систем управління (АСУ) підприємствами (ERP, Enterprise Resource Planning) починається з ретельного вивчення і корегування бізнес-процесів підприємства [1]. Також загальновідомо, що в кожному класі задач (проєктів) різні організаційні структури підприємства мають різну ефективність. Тому перехід на нові класи задач (проєктів) може потребувати зміни організаційної структури підприємства. Малі проєкти зазвичай виконуються в рамках уже існуючих оргструктур. Але для виконання великих проєктів, виходячи з масштабів фінансування та можливих збитків у випадку невдачі, часто створюють нові організації або трансформують організації, що вже існували. Мета – отримання організаційної структури найбільш ефективної в рамках конкретного великого проєкту. Рішення щодо шляхів трансформації в останньому випадку приймаються на основі досвіду, відчуття, натхнення тощо. Чим більше масштаб організації та масштаб проєкту, тим складнішою буде ситуація. Потрібне дуже глибоке обґрунтування рішень, щоб переконувати осіб, від яких залежить фінансування проєктів трансформації та розвитку організацій. Водночас втрати великих проєктів у випадку невдачі сягають космічних величин. Наявність жорсткої конкуренції із зовнішніми організаціями та серед інсайдерів організації так само питання не полегшують. Тож питання оптимізації та обґрунтування рішень щодо напрямків та шляхів трансформації організаційних структур при відкритті нових проєктів є **актуальною задачею**. Відповідно, **актуальною задачею** є створення комп'ютерних моделей трансформації організацій, які би входили до складу програмних систем підтримки управлінських рішень.

1. Аналіз існуючих досліджень

Створення комп'ютерної моделі потрібно розпочинати із ретельного вивчення об'єкту моделювання, тобто основних видів організаційних структур, процесів їх розвитку та трансформації.

У роботах [2, 3] проаналізовані основні види організаційних структур: ієрархічна, матрична, змішана, проєктна тощо. Недолік: відсутні чисельні оцінки і навіть підходи до створення чисельних характеристик розглянутих видів організаційних структур підприємств.

Загальні методичні підходи щодо вибору організаційної структури проєкту, детальний аналіз характеристик і варіантів використання можливих організаційних структур наведені в [4]. В [5], крім того, надана структурована послідовність факторів, на які слід звертати увагу при виборі організаційної структури проєкту. Але, як і раніше, в цих роботах відсутні чисельні оцінки. Виходом із ситуації має бути створення моделей, які враховують залежність корисного ефекту проєкту в залежності від виду структури проєкту.

Загальні підходи до побудови комп'ютерних моделей динамічних процесів розглянуті в [6]. Відповідні чисельні методи в [7, 8, 9]. Недолік: загальні підходи не враховують специфіку моделювання організаційних структур, а також ресурси, що витрачає організаційна структура.

В роботі [1] запропонований підхід для аналізу ефективності різних організаційних структур в динаміці розвитку у часі. Недолік: були розглянуті абстрактні структури, які не корелюють зі стандартними видами структур: ієрархічна, матрична, змішана тощо [2]. До того ж не врахована можливість різкої зміни структури проєкту (організації) в часі.

Оптимізаційне моделювання різкої зміни структури динамічної системи розглядалося у [10]. Недолік: як об'єкти змін структури розглядалися динамічні системи на прикладі літальних апаратів, динаміка яких за своєю природою суттєво відрізняються від динаміки проєктів та організаційних структур.

Таким чином виявлено протиріччя між потребою у методичному апараті програмних систем підтримки ухвалення рішень щодо трансформації організаційних структур і відсутністю єдиного підходу,

який би гарантував отримання найкращого рішення щодо трансформації. Основним інструментом підтримки таких рішень є комп'ютерне моделювання.

Мета статті: підвищити ефективність діяльності великих організаційних структур за рахунок створення комп'ютерних моделей. Ці моделі мають мати достатній рівень адекватності процесам, що моделюються. З іншого - моделі мають мати наочну інтерпретацію основних вхідних параметрів, що дозволило б легко їх визначати на основі емпіричних даних.

2. Логістичні моделі розвитку

Закономірності розвитку організації (бізнесу, технології тощо) за етапами життєвого циклу зазвичай описують, як залежність корисного ефекту організації від вхідних ресурсів, що були витрачені на його створення.

В умовах відсутності обмежень технологій, що використовує організація, залежність корисного ефекту від витрачених ресурсів найчастіше має характер експоненти в зоні зростання [1] (наприклад, зростання суми грошей, покладених на депозит або закон Мура – зростання потужності обчислювальної техніки). Якщо технологія має обмеження розвитку: за рахунок обмежень самої технології, обмежень масштабування виробництва, обмежень щодо супутніх вхідних ресурсів (наприклад, кадрових), нормативних обмежень тощо, то закономірність розвитку найчастіше також має характер експоненти. Але тепер вже в зоні насичення, тобто наближення до асимптоти, до якої процес розвитку наближується знизу. В цілому частина життєвого циклу розвитку складається з етапу експоненційного зростання і етапу експоненційного входження в зону насичення. Перехід між цими двома експонентами має майже лінійний характер. Щоб підкреслити цю властивість, інколи між експонентами додають окрему ділянку лінійного розвитку, коли корисний ефект на виході строго пропорційний кількості вхідних ресурсів.

За вхідні ресурси можуть бути матеріали, фінанси, персонал, інтелектуальна власність, імідж тощо. Тобто вхідним ресурсом може бути будь-що, що можна перетворити на корисний ефект.

Якщо не обмежуватися лише одним етапом життєвого циклу, то дослідники зазвичай намагаються використовувати більш узагальнені закономірності, які охоплюють одночасно і етап експоненційного зростання, і етап експоненційного насичення. В такому випадку використовують S-подібні залежності, найбільш адекватною серед яких вважається логістична [1] (рис.1). Ця залежність лежить між двома асимптотами та має властивість центральної симетрії.

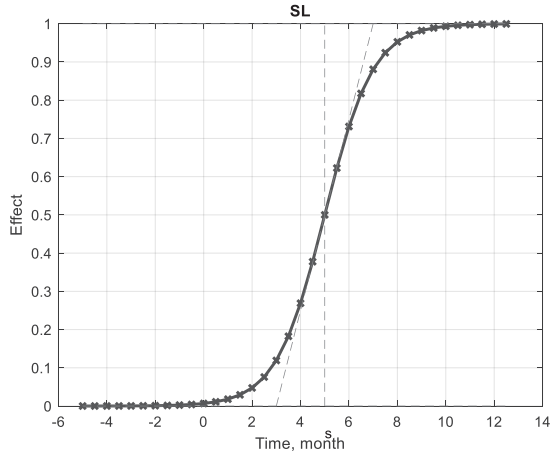


Рис. 1. Логістична залежність корисного ефекту бізнесу (організації) залежно від витрат ресурсу часу

Логістична залежність в диференціальній формі

$$\frac{dy}{dt} = m\,(y - Y_{min})(Y_{max} - y)$$

має чітку фізичну інтерпретацію. Ліва частина рівняння $\frac{dy}{dt}$ відповідає швидкості зростання корисного ефекту y. За вільну змінну обирається вхідний ресурс. У нашому випадку – це час t.

В рівнянні присутні дві асимптоти Y_{min} та Y_{max}, які розташовуються паралельно осі абсцис. Логістична залежність зростає від нижньої асимптоти Y_{min} до верхньої асимптоти Y_{max}. Швидкість зростання y пропорційна добутку відстаней y від нижньої $(y - Y_{min})$ та верхньої $(Y_{max} - y)$ асимптот. Масштаб швидкості та, відповідно, кут нахилу логістичної кривої в точці симетрії визначається коефіцієнтом m.

Якщо коефіцієнти m, Y_{min}, Y_{max} не змінюються відносно вільної змінної (часу), то звичайне диференціальне рівняння, яке задає логістичну залежність, має аналітичний розв'язок. Для цього розділимо змінні, тобто всі елементи з вільною змінною t розмістимо праворуч від знаку рівності, а всі елементи зі змінною корисного ефекту y розмістимо ліворуч.

$$\frac{dy}{(y - Y_{min})(Y_{max} - y)} = m\, dt.$$

Перетворимо вираз так, щоб можна було застосовувати табличні інтеграли

$$\frac{d(y - Y_{min})}{y - Y_{min}} - \frac{d(Y_{max} - y)}{Y_{max} - y} = m\,(Y_{max} - Y_{min})\, dt.$$

Інтегруємо

$$\ln(y - Y_{min}) - \ln(Y_{max} - y) = m\,(Y_{max} - Y_{min})\, t + c.$$

Перетворюємо до більш зручного вигляду

$$\ln\frac{(y - Y_{min})}{(Y_{max} - y)} = m\,(Y_{max} - Y_{min})\, t + c.$$

Знаходимо експоненту від обох частин та записуємо вираз для корисного ефекту

$$y = \frac{Y_{max} + e^{-(m\,(Y_{max} - Y_{min})\, t + c)}}{1 + e^{-(m\,(Y_{max} - Y_{min})\, t + c)}} = Y_{min} + \frac{Y_{max} - Y_{min}}{1 + e^{-(m\,(Y_{max} - Y_{min})\, t + c)}}.$$

Записуємо постійну інтегрування через вже відомі константи

$$\mathrm{c} = -(m\,(Y_{max} - Y_{min})\,\Delta t).$$

Тут Δt зсув точки симетрії вздовж осі абсцис. Після підстановки отримуємо вираз для логістичної залежності в інтегральній формі

$$y = Y_{min} + \frac{Y_{max} - Y_{min}}{1 + e^{-(m\,(Y_{max} - Y_{min})(\,t - \Delta t))}}.$$

Надалі будемо застосовувати узагальнене позначення, для частини виразу

$$SL_0(t - \Delta t) = \frac{1}{1 + e^{-(m\,(Y_{max} - Y_{min})(\,t - \Delta t))}},$$

яка відповідає, так званій SL-функції. SL-функцію можна вважати нормованою логістичною залежністю, що зростає між асимптотами від 0 до 1 з кутом нахилу 45 градусів в точці симетрії.

Звернемо увагу на те, що логістична залежність в інтегральній формі також містить багато фізично зрозумілих складових. Але деякі з них не зовсім зручно знаходити на підставі емпіричних даних. Введемо додаткові позначення:

a - верхняя асимптота,

d - нижня асимптота,

s – абсцисса точки симетрії,

T – постійна логістичної залежності.

Перші три коефіцієнти лише позначені більш лаконічно. Але введений додатковий коефіцієнт T, суттєво спрощує врахування кута нахилу логістичної залежності в точці симетрії. Коефіцієнт T дорівнює довжинам відрізків, що перпендикуляр до осі абсцис в точці симетрії та дотична до логістичної кривої в точці симетрії відрізають на обох асимптотах. Зрозуміло, що за умови центральної симетрії логістичної кривої, довжини цих відрізків однакові (рис.1).

Така формалізація інтегральної форми запису логістичної залежності суттєво спрощує знаходження чисельних значень коефіцієнтів для комп'ютерної моделі розвитку організації на основі логістичних залежностей.

3. Комп'ютерна модель динаміки розвитку організації в умовах трансформації

Далі використаємо логістичні залежності для побудови загальної моделі розвитку організації на всіх етапах життєвого циклу, як на етапах зростання корисного ефекту організації

$$y = d + (a - d) \cdot SL_0(t - \mathrm{s}),$$

так і на етапах падіння корисного ефекту організації

$$y = d - (a - d) \cdot SL_0(t - \mathrm{s}).$$

В другому випадку логістична залежність побудована з від'ємною величиною амплітуди $(a - d)$.

В розгорнутому вигляді логістична залежність корисного ефекту E від часу t

має вигляд

$$E = SL(t) = d + \frac{a-d}{1+e^{-\frac{2}{T}(t-s)}}.$$

Результуючий корисний ефект від діяльності організації на будь-якій ділянці життєвого циклу (в довільний час t життєвого циклу) знаходимо як суму різних логістичних складових із позитивними та від'ємними знаками.

$SL_{\Sigma}(t) = SL_{+}(t) + SL_{-}(t) + SL_{1-}(t) + SL_{2}(t)$, де

$SL_{+}(t)$ - залежність зростання базової технології;

$SL_{-}(t)$ - залежність падіння базової технології внаслідок амортизації та морального старіння;

$SL_{1-}(t)$ - залежність падіння базової технології внаслідок керівного рішення щодо її зупинення для заміни на більш прогресивну;

$SL_{2}(t)$ - залежність зростання нової технології.

В даному випадку аргументом усіх залежностей є час t. Хоча в інших випадках аргументом логістичної залежності корисного ефекту може бути будь-який інший вхідний ресурс або комбінація різних ресурсів (матеріальних, фінансових, людських тощо).

Загальна картина життєвого циклу у вигляді залежності корисного ефекту від вхідного ресурсу містить етапи зростання корисного ефекту, етапи падіння, етапи повторного зростання після трансформацій, спрямованих на усунення ефектів морального старіння технологій, які використовувала організація. Моральне старіння технологій може бути пов'язане з появою нових прогресивніших технологій, з діями конкурентів, з ефектом звикання споживачів до певних технологій і відповідно - втратою інтересу тощо.

Комп'ютерне моделювання життєвого циклу організації було виконано для різних початкових умов і для різних параметрів залежностей розвитку. Основний сценарій моделювання включав зростання корисного ефекту організації після початку впровадження певних інноваційних технологій, етап насичення (вихід технології на максимум свого можливого розвитку), падіння корисного ефекту внаслідок морального старіння, трансформації щодо зміни технології на більш прогресивну, повторний етап насичення, повторне падіння корисного ефекту. Складніші сценарії розвитку в даній моделі також можуть бути реалізовані аналогічним чином. Математична модель була реалізована у вигляді імітаційної моделі алгоритмічною мовою MatLab. Приклад результатів моделювання наведений на рис.2.

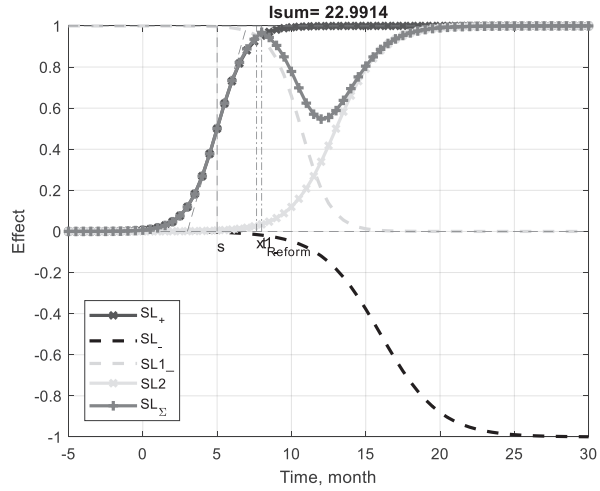


Рис.2. Складові та результуюча залежності корисного ефекту бізнесу залежно від витрат ресурсу часу

Як бачимо, на першому етапі зростання ефекту майже не відрізняється від звичайної логістичної кривої, оскільки основна складова розвитку базової технології $SL_{+}(t)$ має найбільші значення. Але вже на цьому етапі від'ємна логістична залежність морального старіння $SL_{-}(t)$ технології працює, хоча її вплив поки не дуже помітний. У певний період часу це призводить до локального зниження результуючого ефекту. Це зниження могло тривати майже до нуля, але в певний момент починається трансформація організації, яка має на увазі також зміну технологій на більш прогресивні. Тобто починає працювати нова технологія $SL_{2}(t)$, але водночас директивним чином стара технологія $SL_{+}(t)$ вимикається. На програмному рівні це реалізується додаванням такої ж величини, але з від'ємним знаком $SL_{1-}(t) = -SL_{+}(t)$. В результаті вплив старої технології виключається

$$SL_{+}(t) + SL_{1-}(t) = 0.$$

Глибина падіння корисного ефекту внаслідок морального застарівання техно-

логії залежить від часу початку трансформації організації t_{Reform} (переходу на нову технологію). За допомогою моделювання можна підібрати момент часу $t_{Reform} \in [t_0, t_1]$, який забезпечить найменше падіння корисного ефекту

$$\max_{t_{Reform}} \min_{t} SL_{\Sigma}(t)$$

або найбільший сумарний корисний ефект за період роботи $[t_0, t_1]$ в умовах трансформації.

$$\max_{t_{Reform}} \left\{ \int_{t_0}^{t_1} SL_{\Sigma}(t)\, dt \right\}.$$

4. Дослідження нечутливості щодо малих вхідних ресурсів

Як відомо, початкові періоди росту на перших етапах життєвого циклу мають дуже повільний темп. Водночас реакція системи на невеличкі величини вхідних ресурсів майже така, як і на нульові. Наприклад, якщо відбувається інвестиція в розвиток виробництва програмного забезпечення на рівні 100 або 1000 доларів, то вихідний ефект буде такий же, як і для 0 доларів. Для забезпечитення хоч якого помітного корисного ефекту на рівні організації початкова інвестиція має бути на рівні кількох тисяч або десятків тисяч доларів. Для кожного виду технології ця мінімальна інвестиція буде різна. Аналогічно витрати часу на рівні декількох годин або днів, наймовірніше теж не принесуть помітного корисного ефекту. Тобто існує свого роду нечутливість системи до невеликих витрат вхідних ресурсів (фінансових, матеріальних, кадрових, часових). Для дослідження впливу нечутливості до малих вхідних значень ресурсів введемо поняття нечутливості, яке буде вимірюватися у відсотках від максимально можливого корисного ефекту. Тобто насправді ми будемо аналізувати не значення вхідного ресурсу, а значення корисного ефекту $SL_{\Sigma}(t)$ до якого він мав би привести. На першому етапі початкового зростання замість $SL_{\Sigma}(t)$ можна використовувати $SL_{+}(t)$.

Наприклад, нечутливість системи визначається на рівні Insensitivity = 0.1, а максимально можливе значення корисного ефекту дорівнює $E_{max} = a$.

Якщо для певного вхідного ресурсу часу корисний ефект дорівнюватиме $SL_{\Sigma}(t) = 0.05\, E_{max}$, то вихідне значення корисного ефекту приймається рівним $SL_{\Sigma}(t) = 0$.

Якщо для певного вхідного ресурсу часу корисний ефект дорівнюватиме $SL_{\Sigma}(t) = 0.2\, E_{max}$, то вихідне значення корисного ефекту приймається рівним $SL_{\Sigma}(t) = 0.2\, E_{max}$.

Дослідження залежності інтегрального корисного ефекту

$$\int_{t_0}^{t_1} SL_{\Sigma}(t)\, dt$$

від величини нечутливості до малих вхідних ресурсів Sensitivity за весь період трансформацій $[t_0, t_1]$ показало (рис.3), що для нечутливості Insensitivity = $[0, 0.5]$ вихідний корисний ефект майже не міняється (змінюється не більше ніж на 4%). Для постановок задач розвитку на стратегічному рівні похибки припускаються на рівні 10-20% без суттєвої втрати якості управлінських рішень. Тому зміну корисного ефекту на рівні 4% можна вважати рівною нулю.

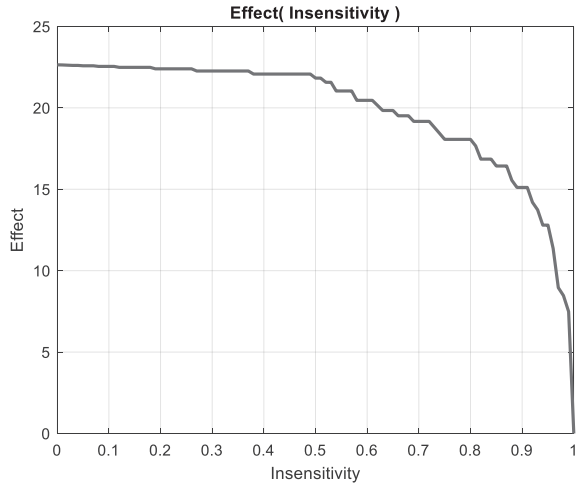


Рис.3. Залежність інтегрального ефекту за період часу від рівня нечутливості системної мети розвитку щодо малих рівнів вихідного корисного ефекту

В діапазоні значень Insensitivity = $[0.5, 0.9]$ зміна корисного ефекту дуже помітна і майже лінійно зменшується на 30%.

Останні 70% зміни корисного ефекту системи відбуваються в діапазоні Insensitivity = $[0.9, 1]$.

Більшість реальних задач відповідають діапазону Insensitivity = [0, 0.5]. Дуже рідко розвиток проєктів може відбуватися в діапазоні Insensitivity = [0.5, 0.9]. І майже ніколи не відбувається в діапазоні Insensitivity = [0.9, 1].

Звернемо увагу на те, що результати моделювання, наведені на рис. 2, відповідають показнику нечутливості Insensitivity = 0.

Висновки

В роботі досліджені основні підходи щодо створення комп'ютерної моделі розвитку організації в умовах трансформації її структури (зміни технологій). Модель призначена для пошуку оптимальних рішень щодо початку часу трансформації.

Було з'ясовано, що для прийнятої постановки задачі найбільш адекватними є комп'ютерні моделі на основі логістичних залежностей. Такі моделі дозволяють легко знаходити чисельні значення параметрів на основі емпіричних даних.

Модель в складі програмної системи підтримки управлінських рішень дозволяє прогнозувати наслідки різних управлінських рішень з метою обрання найкращого.

В роботі використані критерії максимізації найнижчого рівня корисного ефекту організації за весь час прогнозування. А також інтегральний критерій сумарного корисного ефекту, що отримає організація у випадку реалізації конкретного сценарію трансформації.

Комп'ютерна модель реалізована алгоритмічною мовою MatLab.

Напрями подальших досліджень: вдосконалення моделі шляхом розширення переліку факторів, що беруться до уваги під час моделювання.

Література

1. Шевченко В.Л. Оптимізаційне моделювання в стратегічному плануванні.– К.: ЦВСД НУОУ, 2011. -283с.
2. A Guide to the Project Management Body of Knowledge PMBOK, 6-th edition, Project Management Institute, Inc., 2017. (на англ.мові)
3. A Guide to the Project Management Body of Knowledge PMBOK, 7-th edition, Project Management Institute, Inc., 2021. (на англ.мові)
4. Ноздріна Л.В. Управління проектами : підручник [для студентів вищих навчльних закладів] / Л.В. Ноздріна, В.І. Ящук, О.І. Полотай ; М-во освіти і науки України, Укоопспілка ; Львів. комерц. акад. – Київ : Центр учб. л-ри, 2010. – 430, [1] с. : іл., табл. – ISBN 978-966-611-01-0030-4
5. Микитюк П.П. Управління проектами. Навч.посібн. [дл.студ.вищ.навч. закл.] / П.П.Микитюк – Тернопіль, 2014. – 270с.
6. Шевченко В.Л. Імітаційне моделювання. Математичне моделювання процесів: Навч.посібник. / [В.І. Мірненко, В.Л.Шевченко, Д.С.Берестов, Р.М.Федоренко, А.В.Шевченко]; за ред. В.Л.Шевченка. – К.: КНУ ім.Т.Шевченка, 2020.– 164 с.
7. Numerical methods. Study material. Core Course. B Sc Mathematics. VI Semester. University of calicut. School of distance education. Sri.Nandakumar. 2011. – 223p.
8. Шевченко В.Л. Обчислювальні методи: Навч.посібник. / [В.Л.Шевченко, Д.С.Берестов, М.В.Ткаченко, Р.М.Федоренко] ; за ред. В.Л.Шевченка. – К.: КНУ ім.Т.Шевченка, 2019. – 132 с.
9. Introduction to Numerical Methods and MatLab Programing for Engineers. Todd Young and Martin J. Mohlenkamp. Ohio University. 2015.
10. Теорія оптимальних розгалужених траєкторій / О.І. Лисенко, О.М. Тачініна, С.О. Пономаренко, О.Г. Гуйда – К.: КПІ ім. Ігоря Сікорського., 7 БЦ, 2023. – 260 с. ISBN 978-617-549-163-8.

References

1. Shevchenko V.L. Optimization modeling in strategic planning.– K.: TsVSD NUOU, 2011. -283p.
2. A Guide to the Project Management Body of Knowledge PMBOK, 6-th edi-

tion, Project Management Institute, Inc., 2017. (на англ.мові)
3. A Guide to the Project Management Body of Knowledge PMBOK, 7-th edition, Project Management Institute, Inc., 2021. (на англ.мові)
4. Nozdrina L.V. Project management: a textbook [for students of higher educational institutions] / L.V. Nozdrina, V.I. Yashchuk, O.I. Patch it up; Ministry of Education and Science of Ukraine, Ukoopspilka; Lviv. commerce Acad. - Kyiv: Center for Education. l-ry, 2010. – 430, [1] p. : ill., tab. – ISBN 978-966-611-01-0030-4
5. Mykytyuk P.P. Project management. Training manual [for higher education closed.] / P.P. Mykytyuk – Ternopil, 2014. – 270 p.
6. Shevchenko V.L. Imitation modeling. Mathematical modeling of processes: Study guide. / [V.I. Mirnenko, V.L. Shevchenko, D.S. Berestov, R.M. Fedorenko, A.V. Shevchenko]; under the editorship V.L. Shevchenko. – K.: KNU named after T. Shevchenko, 2020. – 164 p.
7. Numerical methods. Study material. Core Course. B Sc Mathematics. VI Semester. University of calicut. School of distance education. Sri.Nandakumar. 2011. – 223p.
8. Shevchenko V.L. Computing methods: Study guide. / [V.L. Shevchenko, D.S. Berestov, M.V. Tkachenko, R.M. Fedorenko] ; under the editorship V.L. Shevchenko. - K.: KNU named after T. Shevchenko, 2019. - 132 p.
9. Introduction to Numerical Methods and MatLab Programing for Engineers. Todd Young and Martin J. Mohlenkamp. Ohio University. 2015.
10. Theory of optimal branched trajectories / O.I. Lysenko, O.M. Tachinina, S.O. Ponomarenko, O.H. Guida - K.: KPI named after Igor Sikorsky, 7 BC, 2023. - 260 p. ISBN 978-617-549-163-8.

Одержано: 12.02.2024
Внутрішня рецензія отримана: 19.02.2024
Зовнішня рецензія отримана: 08.03.2024

Про авторів:

[1]*Сивицький Юрій Ігорович*,
аспірант.
http://orcid.org/0009-0008-9947-6653

[1]*Шевченко Віктор Леонідович*,
д.т.н., професор.
http://orcid.org/0000-0002-9457-7454.

Місце роботи авторів:

[1]Інститут програмних систем
НАН України,
тел. +38-044-522-62-42
E-mail: gii2014@ukr.net
Сайт: www.iss.nas.gov.ua