УДК 623.441/443 **doi: 10.26906/SUNZ.2023.2.135**

А. О. Рибальченко

Національний технічний університет “Харківський політехнічний інститут”, Харків, Україна

АЛГОРИТМИ РІШЕННЯ ЗАДАЧІ ОПТИМАЛЬНОГО РОЗМІЩЕННЯ ДАНИХ В БІЛІНГОВИХ OLTP-СИСТЕМАХ НА ОСНОВІ РЕАЛІЗАЦІЇ РАНГОВОГО ПІДХОДУ

Анотація. У статті приведено результати розробки наближених та точних алгоритмів рішення задачі оптимального розміщення даних у білінгових OLTP-системах на основі реалізації рангового підходу. Даний тип задач відноситься до класу цілочисельного лінійного програмування (ЦЛП) з булевими змінними (БЗ). Домінуюче місце у методах рішення таких задач у даний час займають комбінаторні методи та еволюційні алгоритми. Практичне застосування даних методів ускладнено при рішенні задач великої розмірності. Для усунення даної проблеми пропонується використовувати ідею рангового підходу Наведено аналіз підходів до оптимального розміщення даних у білінгових OLTP-системах, модель рангового підходу, а також наближені та точні алгоритми. **Об'єктом дослідження** є алгоритми функціонування і розміщення інформаційних ресурсів у хмарному середовищі, концепція хмарних обчислень та багаторівневих інформаційних систем. **Предметом дослідження** є принципи розміщення даних, що зберігаються у розподілених базах даних (РБД) та циркулюючих у хмарній мережі, а також специфіка процесів обслуговування абонентів у сучасних реалізаціях OLTP-систем**. Метою наукової роботи** є розробка наближених та точних алгоритмів оптимізації розміщення фрагментів РБД по вузлах мережі хмарної структури, які дозволять збільшити продуктивність інформаційної системи за рахунок раціонального розподілу даних. **Висновки.** Запропоновано стратегії відсікання безперспективних шляхів у множинах, що призводять до наближених і точних рішень задачі ЦЛП з БЗ та побудовано ефективні точні і наближені алгоритми. Показано, що важливою перевагою розроблених алгоритмів на основі рангового підходу є той факт, що збільшення числа обмежень практично не впливає на погрішність рішень алгоритмів, тоді як для методів рішення задач дискретної оптимізації, що засновані на ідеях методу гілок та кордонів, зростання числа обмежень до декількох сотень приводить фактично до неможливості їхнього практичного застосування.

Ключові слова: ранговий підхід, цілочисельне лінійне програмування, булеві змінні, оптимізація за напрямком, наближені та точні алгоритми.

Вступ

Одним із етапів проектування розподілених баз даних (РБД) є планування фрагментації та розміщення, тобто розбиття баз даних (БД) на фрагменти та ухвалення рішення про те, де зберігатимуться ці фрагменти. Однак, проектування схем фрагментації та розміщення відношень ґрунтується на інформації про способи та методи використання РБД. Методи використання залежать від схеми фрагментації та розміщення.

Отже, задачу проектування РБД слід формулювати наступним чином: для даної конфігурації обчислювальної системи (ОС) необхідно описати схему розміщення фрагментів таким чином, щоб оптимізувати цільову функцію.

Для досягнення високої продуктивності розподілених OLTP-систем необхідні ефективні методи управління даними та грамотний вибір стратегії зберігання інформації у РБД. Особливо це стосується транзакційних систем, які побудовані на базі хмарної платформи, оскільки концепція плати у міру використання ресурсів змушує шукати компроміс між продуктивністю та витратами.

Тому, одним із актуальних завдань є оптимальне розміщення даних у розподіленому середовищі (хмарі) за такими критеріями, як загальна мінімальна вартість трафіку, який породжений функціонуванням ОС протягом одиниці часу, загальна мінімальна вартість оренди дискового простору та ін. Їх рішення дозволить досягти значного скорочення витрат та підвищення швидкості роботи системи. Даний тип задач відноситься до рішення задачі цілочисельного лінійного програмування (ЦЛП) з булевими змінними (БЗ).

Домінуюче місце у методах рішення таких задач у даний час займають комбінаторні методи. До них, у першу чергу, можна віднести методи повного перебору, гілок та кордонів, динамічного програмування, а також локальні алгоритми. Практичне застосування даних методів ускладнено при рішенні задач великої розмірності.

Спроби зменшення часу рішення задач ЦЛП з БЗ за рахунок розпаралелення зштовхуються із іншою проблемою теорії паралельних обчислень, яка полягає у тому, що з точки зору паралельних алгоритмів даний тип задач відноситься до класу сильнозв’язаних задач. Тому, погано піддається розпаралеленню.

Отже, при розробці паралельних алгоритмів для рішення задачі ЦЛП з БЗ, крім протиріччя між точністю рішення задачі та часом її рішення, виникає ще одне протиріччя – між сильною зв’язністю властивостей даної задачі та необхідністю її розпаралелення.

Таким чином, розробка алгоритмів рішення задачі оптимального розміщення даних в білінгових OLTP-системах на основі рангового підходу (РП) – є актуальною науковою задачею.

Аналіз публікацій за темою дослідження. У роботі [1] сформульований набір математичних постановок задач оптимізації розміщення даних на різних етапах роботи тренажерно-моделюючих комплексів (ТМК) та розроблені моделі розміщення

даних на основі хмарних обчислень і інформаційних кластерів.

Крім того, автором запропоновано алгоритми оптимізації розміщення модельного світу ТМК на основі комбінування підходів початкової стратегії розподілу і еволюційних методів генетичних алгоритмів (ГА). Однак, результати дослідження застосовні тільки до ТМК різного призначення, оскільки запропоновані моделі засновані на специфічних особливостях побудови та функціонування подібних систем.

На даний час, усе більшої значущості набуває процес проектування РБД. У більшості джерел у якості окремого етапу проектування РБД виділяють фрагментацію і розміщення БД (розбиття БД на фрагменти і прийняття рішення про те, де вони будуть зберігатися) [2]. Однак, завдання побудови саме оптимальної структури РБД часто залишається поза увагою, що значно впливає на продуктивність інформаційної системи у цілому.

У роботі [3] коротко викладаються методи оптимального розміщення фрагментів РБД по вузлах обчислювальної мережі. У якості цільових функцій в даній задачі можуть виступати швидкість виконання запитів у системі і загальна вартість трафіку.

У роботі [4] розглядаються підходи до оптимізації структури сховища даних у вузлах інфокомунікаційної мережі хмарного середовища.

Таким чином, проблема оптимального розміщення фрагментів БД описується математично у вигляді цільової функції однієї змінної, що є бінарною матрицею, яка характеризує розташування конкретного фрагмента на конкретному вузлі.

Стратегія виконання запитів представляється у вигляді матриць інтенсивностей запитів з певного вузла до певного фрагменту, матриць довжин запитів, а також матриць довжин відповідей на запити. Пошук глобального оптимуму цільової функції проводиться за допомогою ГА, що використовує автоматичну ідентифікацію параметрів.

Застосування отриманих результатів до будь-розподіленої інформаційної системи підвищить продуктивність її роботи за рахунок зменшення часу виконання запитів та обробки оновлень шляхом оптимального розподілу даних по вузлах комп'ютерної мережі.

Однак, у розглянутих працях не розкрито питання щодо розробки ефективних наближених та точних алгоритмів рішення задачі оптимізації структури РБД на основі рангового підходу (РП).

Метою статті є розробка ефективних наближених та точних алгоритмів рішення задачі оптимізації структури РБД на основі РП.

Основна частина

1. Аналіз підходів до оптимального розміщення даних у білінгових OLTP-системах. Завдання оптимального розміщення даних у білінгових OLTP-системах, можуть бути вирішені комбінаторними методами, а саме – методами гілок та кордонів, методами відсікання, до яких належать алгоритми Гоморі та Балаша. Проте, такі завдання є NP-повними, тобто зі зростанням розмірності завдань їхня обчислювальна складність зростає експоненційно. Тому, у багатьох дослідженнях зазначається, що для вирішення подібних завдань кластеризації та компонування найбільш успішно застосовуються ГА, які належать до класу еволюційних алгоритмів (ЕА).

Еволюційні алгоритми оптимізації – це клас методів, які засновані на імітації природного відбору, генетичних мутацій та кросоверів, що використовуються для вирішення оптимізаційних задач.

Сутність ЕА полягає у тому, що вони працюють з популяцією можливих рішень, а не з окремими точками простору параметрів, як це робиться, наприклад, у градієнтних методах. Починаючи з деякої початкової популяції рішень, ЕА виконують ітерації, під час яких здійснюється відбір кращих особин (індивідів), на основі яких створюються нові потомки шляхом зміни генетичного матеріалу (кросоверів та мутацій), тобто проводиться еволюція популяції. Даний процес повторюється доти, доки не буде досягнута задовільна точність або не буде вичерпана видима прогресивність.

Основна перевага ЕА полягає у тому, що вони можуть ефективно вирішувати задачі оптимізації з великою кількістю варіантів вхідних параметрів або складною необхідністю задати точну форму функції витрат. Також, ЕА можуть працювати у обмеженому просторі пошуку, де інші методи можуть втратити свою ефективність.

До недоліків ЕА можна віднести наступні:

– евристична природа алгоритму не гарантує отримання абсолютно раціонального рішення;

– еволюція може зупинитися на непродуктивній гілці (цей недолік можна запобігти за допомогою використання паралельних обчислень);

– невисока ефективність ЕА на останніх стадіях пошуку локального оптимуму.

На основі викладених переваг та недоліків ЕА можна зробити висновок про те, що даний клас обчислювальних методів найбільше підходить для вирішення задач багатовимірної оптимізації, для яких немає адекватних нееволюційних методів рішення та складних завдань комбінаторної оптимізації великої розмірності.

Розглядаються наступні алгоритми вирішення представленої задачі: жадібний алгоритм (ЖА), алгоритм імітації відпалу та кілька модифікацій ГА.

Жадібний алгоритм – є простим алгоритмом оптимізації, який використовується для знаходження локальних оптимальних рішень у задачах оптимізації. Сутність ЖА полягає у тому, що на кожному кроці алгоритму вибирається найкращий з доступних варіантів, що призводить до знаходження локально оптимального рішення.

У більшості випадків ЖА не гарантує знаходження глобально оптимального рішення, а лише дозволяє отримати локально оптимальне рішення. Це пов'язано з тим, що на кожному кроці алгоритму виконується локально оптимальний вибір, не зважаючи на можливі варіанти розвитку подальших кроків алгоритму.

Однак, ЖА має свої переваги. Він є простим у реалізації і може працювати дуже швидко навіть для великих об'ємів даних. Крім того, ЖА може використовуватись як евристика для більш складних алгоритмів оптимізації, які враховують глобальну оптимальність.

Метод імітації відпалу (МІВ, англ. simulated annealing) – це стохастичний алгоритм оптимізації, який базується на аналогії з процесом охолодження металів. Основна ідея полягає у тому, що метал, який нагрівається, потрапляє у стан з високою енергією, що дозволяє металу рухатися й приймати нові форми. Після цього, метал поступово охолоджується, його енергія зменшується, що призводить до зниження рухомості атомів та, у кінці кінців, метал застигає у новій структурі з меншою енергією.

Таким же чином, МІВ починається зі стану системи з високою енергією (наприклад, початкового розв'язку) та, поступово, зменшує енергію системи шляхом зниження температури. У цей час можуть відбуватися випадкові зміни стану системи (наприклад, за допомогою переміщень, додавання або видалення елементів тощо), які можуть збільшувати енергію системи, або зменшувати її, залежно від того, як вони впливають на функцію метрики.

У МІВ, на відміну від градієнтних методів, зміна стану системи може бути не тільки у напрямку зменшення значення функції метрики, а й у зворотному напрямку, залежно від температури та інших параметрів.

Метод імітації відпалу має кілька переваг порівняно з іншими методами оптимізації:

– здатність уникнути застрягання у локальному оптимумі: МІВ здатний перескакувати через локальні мінімуми і шукати глобальний мінімум. Це досягається завдяки використанню стохастичного процесу;

– можливість роботи з функціями складної форми: МІВ здатний працювати з функціями складної форми, які мають багато локальних мінімумів та максимумів. Це досягається завдяки тому, що метод використовує випадкові зміщення;

– швидкість роботи: МІВ зазвичай швидший за генетичні алгоритми та інші методи оптимізації, оскільки він не вимагає створення та збереження повної популяції;

– не вимагає похідних: МІВ не вимагає вирахування похідних функції, що робить його корисним для оптимізації функцій, які важко або неможливо диференціювати;

– здатність до паралельної обробки: МІВ здатний до паралельної обробки, що дозволяє використовувати багатоядерні процесори та комп'ютерні кластери для прискорення процесу оптимізації.

До недоліків методу імітації відпалу можна віднести наступні:

– залежність від параметрів: МІВ має декілька параметрів, таких як температура, розмір шагу тощо, і правильний вибір цих параметрів може сильно впливати на ефективність методу;

– можливість застрягання у локальному мінімумі: МІВ може застрягнути у локальному мінімумі, як і більшість методів оптимізації. Хоча існують різні стратегії для уникнення цього, такі як додавання випадкових складових. Однак, це все ще може бути проблемою;

– обчислювальні витрати: МІВ може бути досить обчислювально витратним, особливо для складних оптимізаційних задач. Це може стати проблемою при використанні методу для великих даних або у реальному часі;

– не гарантує глобальний оптимум: хоча МІВ зазвичай знаходить гарні рішення, він не гарантує знаходження глобального оптимуму, особливо у випадку складних оптимізаційних задач з багатьма локальними мінімумами.

Таким чином, МІВ – є потужним та ефективним методом оптимізації, який може бути використаний для рішення різноманітних задач оптимізації.

На даний час набули широкого поширення адаптивні ГА. Це пов'язано з тим, що класичний ГА може бути успішно застосований далеко не до усіх завдань, а для отримання оптимального рішення у конкретній області часто потрібна зміна будь-яких параметрів алгоритму, застосування нових підходів до конструювання хромосоми та подання фітнес-функції. У зв'язку з цим, доцільним є використання адаптивних алгоритмів – тобто алгоритмів, які здатні змінювати свої параметри у процесі роботи.

Під час дослідження ГА для вирішення задачі оптимізації у розподілених системах використовувалась низка підходів та модифікацій класичного ГА. В класичному ГА, який є одним з основних видів ЕА, кожен індивід має свою фітнес-функцію, яка визначає його пристосованість до середовища. Індивіди з більш високою фітнес-функцією мають більші шанси на відбір для репродукції та наступних поколінь. У ГА з відносною фітнес-функцією, також відомому як пропорційний відбір, пристосованість індивіда визначається не його абсолютним значенням фітнес-функції, а відносною величиною у порівнянні з іншими індивідами у популяції. Індивіди з вищим значенням фітнесу у порівнянні з рештою популяції отримують більшу ймовірність на відбір для наступного покоління.

Отже, в основі обох видів ГА лежить ідея відбору найкращих рішень за допомогою певної фітнес-функції. Відмінність полягає у тому, як саме вимірюється пристосованість індивіда та як це відображається на процесі відбору та репродукції.

У стандартному ГА мутація проводиться шляхом зміни випадково обраного біту у геномі (тобто у рядку бінарних чисел, які представляють рішення). Така мутація може не вплинути на конкретну змінну, яка повинна бути змінена, а також може вплинути на змінну, яка не потребує зміни. Це може призвести до втрати цінної інформації про оптимальність рішення.

У ГА з роздільною мутацією кожна змінна мутується окремо. Тому, можна досягти більш точної оптимізації кожної змінної. Даний метод дозволяє збільшити швидкість збіжності алгоритму, оскільки окрема мутація не впливає на усі змінні одразу, а лише на ті, які потребують зміни.

Однак, необхідно мати на увазі, що роздільна мутація може привести до того, що різні змінні не взаємодіють між собою у оптимальний спосіб. Тому доцільно використовувати її у поєднанні з іншими методами оптимізації, наприклад, з мутацією зміни точки перетину або зі зміною розміру популяції.

ГА Уітлі (Whitley) – є одним з варіантів стандартного ГА і включає у себе додаткову операцію, що називається "локальною оптимізацією". Така операція дозволяє зменшити кількість ітерацій ГА, необхідних для знаходження оптимального рішення.

Основна суть ГА Уітлі полягає у тому, що на кожній ітерації алгоритму застосовується додаткова операція, яка полягає у виборі декількох найкращих особин з попередньої популяції і подальшому запуску на них локального пошуку. Локальний пошук може бути будь-яким ефективним алгоритмом пошуку оптимуму, наприклад, методом найшвидшого спуску, методом Ньютона або методом золотого перетину. Після локального пошуку отримані розв'язки вносяться у нову популяцію разом з найкращими розв'язками з попередньої ітерації. Потім, на новій популяції виконуються стандартні операції ГА, такі як селекція, кросинговер та мутація, і так далі до знаходження оптимального розв'язку.

Головною перевагою ГА Уітлі є те, що він дозволяє збільшити швидкість збіжності алгоритму за рахунок використання локального пошуку. Проте, його недоліком може бути те, що локальний пошук може застрягти у локальному оптимумі, що може призвести до погіршення результатів алгоритму.

ГА Ешельмана (англ. Eshelman's genetic algorithm) – це вид ГА, який був запропонований Кеннетом Ешельманом у 1991 році. Його основна ідея полягає у тому, щоб використовувати "турнірну" селекцію, у якій кожен індивідуум конкурує з іншими за право на наступне покоління, а не зважується його абсолютна пристосованість.

Головна особливість ГА Ешельмана – це використання методу "без заміщення" при проведенні турнірів, що означає, що кожен індивідуум може бути відібраний лише один раз для участі у турнірі, що унеможливлює відбір дублікатів.

Другою особливістю ГА Ешельмана є використання оператора "головного гена", який гарантує збереження кращих індивідуумів з попереднього покоління у наступному без яких-небудь змін, що дозволяє уникнути можливої втрати корисної інформації та зменшення різноманітності у популяції.

Таким чином, ГА Ешельмана – є ефективним методом оптимізації, який дозволяє досягти більшої різноманітності у популяції та забезпечити надійну збереженість кращих рішень від покоління до покоління.

2. Модель рангового підходу до рішення задачі оптимального розміщення даних у білінгових OLTP-системах. Оптимізація структури РБД заснована на моделі, яка має такий вигляд [5-11]:

$$f(\vec{x}) = \sum_{j=1}^{n} c_j \cdot x_j \Rightarrow \max, \qquad (1)$$

$$L = \sum_{j=1}^{n} c_j x_j \to \min, \qquad (2)$$

$$\sum_{j=1}^{n} a_{ij} \cdot x_j \le b_i, \qquad (3)$$

$$\sum_{j=1}^{n} a_{ij} x_j \ge 1, \quad i = \overline{1,m}; \qquad (4)$$

$$x_j \in \{0,1\}; c_j \ge 0; i = (\overline{1,m}); \; j = \left(\overline{1,n}\right). \qquad (5)$$

Ідея РП основана на представленні *n*-мірного одиничного куба у виді графа $G\Delta$, геометричний зміст якого полягає у наступному [5-11]. Геометрично вершина *k* графа $G\Delta$ рангу *r* – це множина векторів (x_1, x_2, ... , x_k, ... , x_n), у яких $x_k=1$, а на позиціях від 1 до *k* знаходиться *r* одиниць. Ребру, що входить у вершину *k* графу $G\Delta$, відповідає одиничний вектор (0, 0, ... , 0, 1, 0, ... , 0) *n* – мірного одиничного куба B_n з одиницею у *k*-й позиції. Тоді, шляху μ_{sj}^{r} рангу *r* у графі $G\Delta$ відповідає вектор $\vec{x}$, який дорівнює сумі одиничних векторів ребер, по яким він досяг вершину *j* рангу *r*, починаючи з вершини *s*. Множину шляхів у графі $G\Delta$ до вершин *j*, розташованим на ярусах від вершини *s*, можна представити наступним чином:

$$m_s^r(j) = m_{sj}^{r=1} \cup m_{sj}^{r=2} \cup ... \cup m_{sj}^{r=n}, \quad j = \overline{(1,n)}, \qquad (6)$$

де $m_s^r(j)$ – множина шляхів у графі $G\Delta$ від вершини *s* до вершин *j*, розташованих на *r*-х ярусах графа $G\Delta$, (ранг шляху $\mu_{sj}^{r} \in m_{sj}^{r}$ визначається числом ребер, що утворюють цей шлях).

Варто мати на увазі те, що множині шляхів $m_{sj}^{r=k}$ у графі $G\Delta$ відповідає множині векторів, що містять *k* одиниць. Отже, $\left|m_{sj}^{r}\right| = C_n^{r=k}$, тобто кожному шляху у множині $m_s^r(j)$ відповідає деякий вектор $(x_1, x_2, ..., x_n)$. Використавши (6) можливо записати наступне:

$$\left|m_s^r(j)\right| = C_n^{r=1} + C_n^{r=2} + ... + C_n^{r=n} = 2^n - 1. \qquad (7)$$

Таким чином, граф $G\Delta$ є упорядкований по рангах еквівалент *n*-мірного одиничного куба B_n, у якому шляхи $\mu_{sj}^{r} \in m_{sj}^{r}$ відповідають вершинам B_n. Для рішення задач у [1-4] запропонована система каліброваних векторів, що дозволяє звести рішення цих задач до визначення екстремальних шляхів у графі $G\Delta$. Сформульовано нтаке:

– принцип оптимізації за напрямком, обумовлений співвідношенням:

$$\forall\left(\mu_{sj}^{r} \in m_{sj}^{r}\right)\left[\mu_{sp}^{r=r+1} = L_w\left\{\mu_{sj}^{r} \cup (j,p)\right\}\right]; \qquad (8)$$
$$p = (\overline{r+1,n}); \quad j = (\overline{r,n}),$$

– принцип виділення коридору у підмножинах:

$$m_{sp}^{r=r+1} = \left\{ L_W \left(\forall (\mu_{sj}^r \cup (j, p)) \right) \right\}, \quad (9)$$

де *{Lw}* – правила відсікання шляхів μ_{sj}^r у множинах m_{sj}^r; $\mu_{sj}^r \cup (j, p)$ – шлях з вершини *s* графу $G\Delta$ шлях у вершину *p*, що проходить через проміжну вершину j та задовольняє правилам *{Lw}*, тобто шлях μ_{sp}^r отримано за рахунок приєднання до шляху μ_{sj}^r ребра (*j*, *p*), якщо таке з'єднання не суперечить правилам *{Lw}*.

3. Розробка наближених алгоритмів оптимального розміщення даних у білінгових OLTP-системах. Для рішення задач типу (1-5), пропонується використовувати узагальнену процедуру A_0, яка дозволяє визначити локальні екстремуми у *W*-областях графу $G\Delta$ щораз і, потім, виділяти глобальний екстремум з *n(n+1)/2* локальних, що отримані на основі принципу оптимізації за напрямком з використанням правил відсікання *{Lw}* шляхів у множинах, що вводяться.

Узагальнена процедура А0

КРОК 1. З вершини *s* будуються множини шляхів $m_{sj}^{r=1}$, $j = \overline{(1, n)}$, що задовольняють властивості *ν*. Виділяються шляхи $\mu_{sj}^{*r=1}$, що визначають локальні екстремуми областей Ω_j.

КРОК 2. Формуються множини шляхів $m_{sp}^{r=r+1}$ $p = \overline{(r+1, n)}$ наступного рангу, що задовольняють властивості *ν*, на базі множини шляхів m_{sj}^r попереднього рангу відповідно до співвідношення (8). В утворених множинах $m_{sp}^{r=r+1}$ здійснюється відсівання шляхів відповідно до обраного правила відсікання $\{L_w\}$ та виділяються шляхи $\mu_{sp}^{*r=r+1}$, що визначають локальні екстремуми областей Ω_p.

КРОК 3. Перевіряємо, чи усі множини $m_{sp}^{r=r+1}$ наступного рангу порожні. За умови, якщо це так, то переходимо до кроку 4, якщо ні, то перевіряємо r = (n - 1). У випадку виконання рівності переходимо до кроку 4, інакше збільшуємо *r* на 1 та виконуємо крок 2.

КРОК 4. Виділяємо серед множин локальних екстремумів Ω_j $j = \left(\overline{1, n^2/2}\right)$ глобальний і процедура A_0 закінчує роботу.

Узагальнена процедура A_0 дозволяє визначити локальні екстремуми у Ω-областях графа $D\Delta$ щораз на кроці 2 і потім на кроці 4 виділити глобальний екстремум з $n^2/2$ локальних, які отримуються на основі принципу оптимізації за напрямком (8) з використанням правил відсікання, що вводять, *{Lw}* шляхів у m_{sj}^r множинах.

При побудові наближених алгоритмів із множини правил *{Lw}*, розглянутих у [1-6], будемо розглядати таке:

$$\text{L1:} \quad \mu_{sp}^{r=r+1} = \max_{\{c_j\}} \{ \mu_{sj}^r \cup (j, p) \}, \quad p = \overline{r+1, n}; \quad j = \overline{r, n}; \quad j \neq p; \quad (10)$$

$$\text{L2:} \quad \mu_{sp}^{r=r+1} = \min_{\{\alpha_j\}} \left\{ \mu_{sj}^r \cup (j, p) \right\}, \quad p = \overline{r+1, n}, \quad j = \overline{r, n}, \quad j \neq p; \quad (11)$$

$$\text{L3:} \quad \begin{cases} \mu_{sp}^{r=r+1} = \max\limits_{c_j} \left\{ \mu_{sj}^r \cup (j, p) \right\}; \\ \mu_{sp}^{r=r+1} = \min\limits_{\alpha_j} \left\{ \mu_{sj}^r \cup (j, p) \right\}. \end{cases} \quad (12)$$

З множин m_{sj}^r будемо виключати як не перспективні шляху μ_{sp}^r, які задовольняють нерівності:

$$d_c(\mu_{sp}^r) + \gamma_p < \max_{\{c_j\}} \left\{ d_c(\mu_{sp}^{*r}) \right\}, \quad (13)$$

де $$\gamma_p = c_{p+1} + c_{p+2} + \ldots \ldots + c_n \quad (14)$$

(для вершини $j \div n$ вага γ_j – приймається рівною нулю); $d_s(\mu_{sp}^r)$ – довжина шляху до вершини *p* рангу *r* по вагах функціоналу.

При цьому, покроковий опис наближених алгоритмів, позначимо їх MAX та MIN, буде мати наступний вигляд

Алгоритм MAX

Крок 1. Виконання сортування значень функціоналу й обмежень вигляду:

$$\alpha_1 \le \alpha_2 \le \ldots \le \alpha_n; \quad (15)$$

$$c_1 \ge c_2 \ge \ldots \ge c_n; \quad (16)$$

$$\frac{c_1}{\alpha_1} \ge \frac{c_2}{\alpha_2} \ge \ldots \ge \frac{c_n}{\alpha_n}. \quad (17)$$

Крок 2. З вершини *s* будуються множини шляхів $\{\mu_{sj}^{r=1}\}$, $j = \overline{(1, n)}$, першого рангу r, що задовольняють властивості γ, і визначаються в множинах $m_{sj}^{r=1}$ шляхи максимальної довжини $\{\mu_{sj}^{*r}\}$ по вагах функціоналу *{cj}*. Для кожної вершини j визначається вага γ_j відповідно до (14). Для вершини *j=n* вагу γ_j приймаємо рівною 0.

Крок 3. Виключаються $\{\mu_{sp}^r\}$, $p = \overline{(r, n)}$ у множинах m_{sj}^r поточного рангу r, довжини яких $d_c(\mu_{sp}^r)$ задовольняють нерівності (13).

Крок 4. Формуються множини шляхів $m_{sj}^{r=r+1}$, $j = \overline{(r+1, n)}$ такого рангу, що задовольняють властивості *γ*, на базі множин шляхів попереднього рангу m_{sp}^r на основі рекурентного співвідношення (10). У множинах $m_{sj}^{r=r+1}$, що утворилися,

виділяємо самі довгі $\{\mu_{sj}^{*r=r+1}\}$ шляхи. За умови, якщо виявиться декілька шляхів максимальної довжини, то серед них вибирається шлях із меншим значенням довжини по вагах обмежень.

Крок 5. Перевіряється, чи усі множини шляхів такого *(r+1)*-го рангу порожні. За умови, якщо це так, то переходимо до кроку 5, якщо ні, то перевіряємо r=(n-1). У випадку виконання рівняння переходимо до кроку 5, інакше збільшуємо *r* на 1 та виконуємо крок 2.

Крок 6. Виділяємо у $\left\{\mu_{sj}^{*r}, \mu_{sp}^{*r=2}, ...\mu_{sw}^{*r=n}\right\}$ шлях максимальної довжини, кінець роботи.

Алгоритм MIN

Крок 1. Виконання сортування значень функціоналу й обмежень вигляду (15-17).

Крок 2. З вершини *s* будуються множини шляхів $\{\mu_{sj}^{r=1}\}$, $j=\overline{(1,n)}$, першого рангу *r*, що задовольняють властивості γ та визначаються у множинах $m_{sj}^{r=1}$ шляхи мінімальної довжини $\{\mu_{sj}^{*r}\}$ по вагах обмеження {aj}. Для кожної вершини j визначається вага γ_j відповідно до (14). Для вершини j=n вагу γ_j приймаємо рівною 0.

Крок 3. Виключаються $\{\mu_{sp}^{r}\}$, $p=\overline{(r,n)}$ в множинах m_{sj}^{r} поточного рангу r, довжини яких $d_c(\mu_{sp}^{r})$ задовольняють нерівності (13).

Крок 4. Формуються множини шляхів $m_{sj}^{r=r+1}$, $j=\overline{(r+1,n)}$ такого рангу, що задовольняють властивості γ, на базі множин шляхів попереднього рангу m_{sp}^{r} на основі рекурентного співвідношення (11). У множинах $m_{sj}^{r=r+1}$, що утворилися, виділяємо шляхи мінімальної $\{\mu_{sj}^{*r=r+1}\}$ довжини. За умови, якщо виявиться декілька шляхів мінімальної довжини по вагах обмежень, то серед них вибирається шлях із більшим значенням довжини по вагах функціоналу.

Крок 5. Перевіряється, чи усі множини шляхів такого *(r+1)*-го рангу порожні. За умови, якщо це так, то переходимо до кроку 5, якщо ні, то перевіряємо *r=(n-1).* У випадку виконання рівняння переходимо до кроку 5, інакше збільшуємо r на 1 і виконуємо крок 2.

Крок 6. Виділяємо у $\left\{\mu_{sj}^{*r}, \mu_{sp}^{*r=2}, ...\mu_{sw}^{*r=n}\right\}$ шлях мінімальної довжини по вагах обмежень і алгоритм закінчує роботу.

Крім цього становить інтерес модифікація цих алгоритмів, яку позначимо алгоритмом MAX-MIN, у якому на кроці 3 виділяється щоразу два шляхи відповідно до співвідношення (12).

4. Розробка точних алгоритмів оптимального розміщення даних у білінгових OLTP-системах

Побудуємо точний алгоритм рішення одновимірної задачі ЦЛП з БЗ.

КРОК 1. З вершини *s* будується множина шляхів $m_{sj}^{r=1}$, $j=\overline{(1,n)}$ першого рангу r, що задовольняє властивості та визначаються у множині $m_{sj}^{r=1}$ шляху максимальної довжини μ_{sj}^{*r} за вагою функціонала c_j. Для кожної вершини j формуються калібрувальні вектори $\vec{y}_j$ (18) та $\vec{z}_j$ (19):

$$y_{jk}=a_{jk}+y_{j(k-1)};\quad k=\overline{(1,n-j)};\quad y_{10}=0;\quad y_{n0}=b_1;\ j=\overline{(1,n-1)}. \tag{18}$$

$$\hat{z}_{jk}^{в}=c_{j+k}+\hat{z}_{j(k-1)}^{в};\quad k=\overline{(1,n-j)};\ \hat{z}_{j0}^{в}=0;\ \hat{z}_{n0}^{в}=0;\ j=\overline{(1,n-1)}. \tag{19}$$

КРОК 2. Виключаються шляхи у множині m_{sj}^{r} поточного рангу r, довжини якої $d_c\left(\mu_{sp}^{r}\right)$ задовольняють нерівності (14).

КРОК 3. Для кожного шляху $\left\{\mu_{sj}^{r}\right\}$, $j=\overline{(r,n)}$ поточного рангу r визначається за правилом K_1 значення $\hat{r}_в$ [11]. Виключаються шляхи, довжини яких задовольняють нерівності:

$$d_c\left(\mu_{sp}^{r}\right)+\hat{z}_{p_{\hat{r}_в}}^{в}\left(\mu_{sp}^{r}\right)<\max_{c_j}\left\{d_c\left(\mu_{sp}^{*r}\right)\right\}. \tag{20}$$

КРОК 4. Формується $m_{sp}^{r=r+1}$, $p=\overline{(1,n)}$ наступного рангу, що задовольняє властивості, на базі множини шляхів m_{sj}^{r} попереднього рангу на основі принципу оптимізації за напрямком з виділенням коридору по стратегії L_6 та виключенням векторів усередині коридору відповідно до стратегії L_8. Шлях у множині $m_{sp}^{r=r+1}$ може бути сформований, якщо він задовольняє властивості. За умови, якщо властивість ν не виконується, то шлях виключається з подальшого аналізу. У освічених множинах $m_{sp}^{r=r+1}$ виділяємо щонайдовші шляхи $\left\{\mu_{sp}^{*r=r+1}\right\}$.

КРОК 5. Перевіряється, чи уся множина шляхів (r+1)-го рангу порожня. За умови, якщо це так, то переходимо до кроку 6, якщо ні, перевіряємо r = (n – 1). У разі виконання рівності переходимо до кроку 6, інакше збільшуємо r на 1 і виконуємо крок 2.

КРОК 6. Виділяємо у множині шлях максимальної довжини і алгоритм закінчує роботу.

Висновки

Таким чином, запропоновано стратегії відсікання *{Lw}* безперспективних шляхів у множинах, що призводять до наближених і точних рішень задачі ЦЛП з БЗ. Побудовано ефективні точні та наближені алгоритми рішення задач ЦЛП з БЗ. Як показали результати експериментального дослідження, важливою перевагою розроблених алгоритмів на основі РП є той факт, що збільшення числа обмежень прак-

тично не впливає на погрішність рішень алгоритмів, тоді як для методів рішення задач дискретної оптимізації, що засновані на ідеях методу гілок та кордонів, зростання числа обмежень до декількох сотень приводить фактично до неможливості їхнього практичного застосування [12].

СПИСОК ЛІТЕРАТУРИ

1. Янюшкин, В.В. Математические модели оптимизации распределенных информационных систем тренажерно-моделирующих комплексов : автореф. дис. …канд. техн. наук : 05.13.18 / Янюшкин Вадим Вадимович. – Новочеркасск, 2010. – 19 с.
2. Многоатрибутивное формирование оптимальных по составу высоконадежных сложных систем / И.В. Ковалев [и др.]. – Красноярск: Краснояр. гос. аграр.ун-т., 2009. – 166 с.
3. Жуков, В.С. Исследование методов оптимального размещения базы данных по узлам вычислительной сети / В.С. Жуков // В мире научных открытий. – 2010. – № 4 (10). – С. 75-76.
4. Третяк В.Ф., Пашнєва А.А. Оптимізація структури сховища даних у вузлах інфокомунікаційної мережі хмарного середовища // Системи управління, навігації та зв'язку. – 2017. – №. 4 (44). – С. 122-128.
5. Голубничий Д.Ю. Інформаційна технологія відсікання неперспективних варіантів в алгоритмах рішення задачі цілочисельного лінійного програмування з булевими змінними на основі рангового підходу // Theoretical foundations in research in Engineering: collective monograph / Д.Ю. Голубничий, О.В. Коломійцев, В.Ф. Третяк, А.О. Рибальченко [та ін.]; International Science Group. – Boston, 2022. – С. 96-133.
6. Голубничий Д.Ю. Архітектура системи обміну медичними даними пацієнтів з лікарями на основі IOTA / Д.Ю. Голубничий, О.В. Коломійцев, В.Ф. Третяк, Я.О. Ключка, А.О. Рибальченко // Системи управління, навігації та зв'язку. – Полтава: Полтавський національний технічний університет ім. Кондратюка, 2022. – Вип. 1(67). – С. 56-61.
7. Коломійцев О.В. Метод рішення задачі оптимізації маршрутів для спеціалізованих машин логістичного забезпечення в автоматизованій інформаційній системі складського обліку на основі рангового підходу / О.В. Коломійцев, В.В. Старцев, В.Ф. Третяк, А.І. Нікорчук, О.І. Шаповалов, З.З. Закіров, Е.М. Полтавський, П.В. Черненко, О.А. Крамар, А.О. Рибальченко // InterConf. – Прага: Author-publishers miscellaneous, 2022. – Вип. 27(133), – С. 417-434.
8. Третяк В.Ф. Математична модель процесу виконання MDX-запитів на основі рангового підходу до рішення задачі цілочисельного лінійного програмування з булевими змінними / В.Ф. Третяк, Д.М. Запара, С.В. Новіченко, О.В. Коломійцев, А.М. Савельєв, В.І. Кривчун, М.М. Охрамович, Н.М. Шамрай, А.О. Рибальченко, О.А. Крамар // Modern Directions and Movements in Science: I міжн. НПК., 06-08 жовтня 2022 р. – Люксембург, 2022. – С. 281-292.
9. Коломійцев О.В. Задачі дискретної оптимізації та їх постановка / О.В. Коломійцев, .С.В. Осієвський, В.Ф. Третяк, З.З. Закіров, А.О. Романюк Є.М. Логвиненко, А.О. Лисиця // Scientific trends and trends in the context of globalization: II МНПК., 19-20 вересня 2021 р. – Рим, 2021. – С. 285-302. – DOI: https://doi.org/10.51582/interconf.19-20.09.2021.033.
10. Третяк В.Ф. Аналіз сучасних систем управління базами даних / В.Ф. Третяк, В., О.В. Коломійцев, Д.І. Євстрат, С.В. Ворошилов, В.М., В., Логвиненко, А.О. Лисиця, В.О. Місюра // Scientific goals and purposes in XXI century: II міжн. НПК, 07-08 жовтня 2021 р. – Сіетл, 2021. – С. 453-465. – DOI: https://doi.org/10.51582/interconf.7-8.10.2021.050.
11. Голубничий Д.Ю. Інформаційна технологія відсікання неперспективних варіантів в алгоритмах рішення задачі цілочисельного лінійного програмування з булевими змінними на основі рангового підходу // Theoretical foundations in research in Engineering: collective monograph / Д.Ю. Голубничий, О.В. Коломійцев, В.Ф. Третяк, А.О. Рибальченко [та ін.]; International Science Group. – Boston, 2022. – С. 96-133.
12. Technical and agricultural sciences in modern realities: problems, prospects and solutions: collective monograph / Hladyshev D., Brodskyi M., Lisnykh L., Rybalchenko A. – etc. – International Science Group. – Boston : Primedia eLaunch, 2023. 461 p. Available at : DOI – 10.46299/ISG.2023.MONO.TECH.2.

Received (Надійшла) 23.03.2023
Acceptedforpublication (Прийнята до друку) 31.05.2023

Algorithms for solving the problem of optimal placement of data in billing OLTP systems based on the implementation of the ranked approach

Alina Rybalchenko

Abstract. The article presents the results of the development of approximate and exact algorithms for solving the problem of optimal data placement in billing OLTP systems based on the implementation of the rank approach. This type of problem belongs to the class of integer linear programming (ILP) with Boolean variables. Combinatorial methods and evolutionary algorithms currently occupy a dominant place in the methods of solving such problems. The practical application of these methods is complicated when solving large-scale problems. To eliminate this problem, it is suggested to use the idea of the rank approach. The analysis of approaches to the optimal placement of data in billing OLTP systems, the model of the rank approach, as well as approximate and exact algorithms are given. **The object** of research is the algorithms of functioning and placement of information resources in the cloud environment, the concept of cloud computing and multi-level information systems. **The subject** of the study is the principles of data placement stored in distributed databases and circulating in the cloud network, as well as the specifics of subscriber service processes in modern implementations of OLTP systems. **The purpose** of the research work is to develop approximate and accurate algorithms for optimizing the placement of RDB fragments on network nodes of the cloud structure, which will allow to increase the productivity of the information system due to the rational distribution of data. **Conclusions.** Strategies for cutting off unpromising paths in sets are proposed, leading to approximate and exact solutions of the problem of integer linear programming with Boolean variables, and efficient exact and approximate algorithms are constructed. It is shown that an important advantage of the developed algorithms based on the rank approach is the fact that an increase in the number of constraints practically does not affect the error of algorithm solutions, while for methods of solving discrete optimization problems based on the ideas of the branch-and-bound method, an increase in the number of constraints to several hundreds actually leads to the impossibility of their practical application.

Keywords: rank approach, integer linear programming, boolean variables, optimization by direction, approximate and exact algorithms.