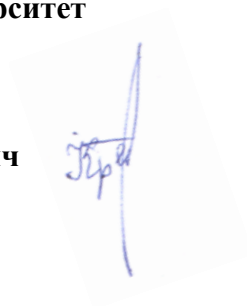
Національний авіаційний університет

КРАВЕЦЬ Іван Михайлович



УДК 004.658.3 (042.3)

РОЗПОДІЛЕННЯ НАВАНТАЖЕННЯ В БАЗАХ ДАНИХ ВЕЛИКОГО ОБ'ЄМУ МЕТОДОМ ГОРИЗОНТАЛЬНОЇ ФРАГМЕНТАЦІЇ

05.13.05 – комп'ютерні системи та компоненти

АВТОРЕФЕРАТ
дисертації на здобуття наукового ступеня кандидата технічних наук

Київ – 2010

Дисертацією є рукопис.

Робота виконана в Національному авіаційному університеті Міністерства освіти і науки України.

Науковий керівник: доктор технічних наук, професор,
заслужений винахідник України
Жуков Ігор Анатолійович,
Національний авіаційний університет,
завідувач кафедри комп'ютерних систем та мереж.

Офіційні опоненти: доктор технічних наук,
старший науковий співробітник
Огір Олександр Степанович,
Інститут проблем моделювання в енергетиці
ім. Г.Є. Пухова НАН України,
головний науковий співробітник відділу
математичного і комп'ютерного моделювання;

кандидат технічних наук, доцент
Марковський Олександр Петрович,
Національний технічний університет України
"Київський політехнічний інститут",
доцент кафедри обчислювальної техніки.

Захист відбудеться "__" _____ 2010 р. о __ годині на засіданні спеціалізованої вченої ради Д 26.062.07 Національного авіаційного університету за адресою:
03680, м. Київ, просп. Космонавта Комарова, 1.

З дисертацією можна ознайомитись у науково-технічній бібліотеці Національного авіаційного університету за адресою:
03680, м. Київ, просп. Космонавта Комарова, 1.

Автореферат розісланий "__" _____ 2010 р.

Вчений секретар
спеціалізованої вченої ради  О. П. Мартинова

ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ

Актуальність теми. Система електронних інформаційних комунікацій, що формується на очах нинішнього покоління, кардинально змінює ситуацію в сфері збору, зберігання та обробки даних. Технології глобальних комп'ютерних мереж, що стрімко розвиваються, формують в інформаційній області нову систему відносин, яка відображає реалії технічного рівня сучасного людства. Інтенсивність змін в значній мірі диктується тим величезним значенням, яке набуває інформація в постіндустріальному суспільстві, в якому інформація стає головним товаром, ресурсом та інструментом одночасно.

Більшість організацій почали міркувати над тим, як автоматизувати свій документообіг, чи взагалі відмовитися від того виду інформації, що зберігається у паперовому вигляді і потребує для цього великі приміщення для її зберігання. Одним із шляхів вирішення цієї проблеми є створення централізованої бази даних (БД). Але розвиток технологій систем управління БД збігся за часом важливих подій в комп'ютерній мережі та розподіленими обчислювальними технологіями. Кінцевим результатом стала поява розподілених систем управління БД (СУБД). Ці системи є домінуючим інструментом управління даними великого об’єму.

Час реакції таких систем визначається не тільки продуктивністю комп’ютерів, що входять до їх складу, а й розміром даних, які вони оброблять. На сьогодні основну увагу приділяють створенню кластерних систем для розподілення навантаження на БД великого об’єму і зменшення часу реакції. Це в якійсь мірі вирішує проблематику навантаження, але тимчасово. Оскільки, об’єм даних постійно росте, а нагромадження великої кількості серверів потребує додаткових витрат: сучасне серверне обладнання коштує великі гроші, які не кожна організація може витратити для таких цілей; кожен новий сервер із СУБД потребує додаткового догляду та адміністрування. При використанні реплікації даних, виникають різні обмеження на чергу вхідних запитів. Це частково може відобразитися на швидкодії комп’ютерної системи; створення кластерної системи в режимі дзеркалювання – це чіткий приклад того, як використовуються нераціонально апаратні ресурси, так як відбувається дублювання даних по всім серверам.

Аналіз літературних джерел показав, що відомі методи для розподілення навантаження для СУБД (та і *WEB*-серверів в цілому) ефективні для завдань невеликої розмірності, носять загальний або приватний характер, не враховують характерні для комп’ютерних мереж особливості або складні для реалізації. Це свідчить про те, що задача зменшення навантаження на БД великого об’єму є актуальною.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота виконувалась у рамках науково-дослідних робіт Національного авіаційного університету (НАУ):

– НДР «Створити автоматизовану інформаційно-аналітичну систему Мінтрансзв'язку» (номер державної реєстрації №0108U000488) у період за 2007-2008р.р.

– НДР № 668-ДБ-10 «Методи та засоби підвищення ефективності розподілених обчислювальних систем на базі тензорних нейронних мереж» (номер державної реєстрації №0110U000221) у період за 2010 р.

Мета і завдання дослідження. Метою досліджень є розробка методів та засобів для підвищення ефективності розподілення навантаження в БД великого об'єму.

Мета зумовила необхідність розв'язання наступних задач:

– аналіз існуючих рішень для розподілення навантаження в БД великого об'єму та для оптимізації графів з навантаженими вершинами;

– забезпечення логічної повноти даних, які фізично знаходяться на різних вузлах, а також, визначення правил ефективної фрагментації;

– розробка евристичного методу для горизонтальної фрагментації даних великого об'єму у розподілених БД (РБД);

– розробка системи, що буде формувати базу знань, як сукупність відомостей про частоту звертань до кортежів між РБД;

– розробка програмного модуля, що реалізує базову об'єктно-орієнтовану модель, яка ґрунтується на генетичному алгоритмі (ГА).

Об'єктом дослідження є процес збору, зберігання та розподілення даних в комп'ютерній мережі.

Предметом дослідження є методи та засоби для розподілення навантаження в БД великого об'єму.

Методи дослідження. Для оптимізації графів з навантаженими вершинами використовуються методи теорії графів та евристичний метод, які зводяться до знаходження найкоротшого маршруту; розробка методу горизонтальної фрагментації та оптимізація багатоекстремальних функцій базуються на математичному моделюванні, теоріях множин, ймовірностей та на методі вивчення ГА шляхом його формалізації. В основі розробки програмного модуля лежить об'єктно-орієнтоване програмування та методи теорії алгоритмів.

Наукова новизна одержаних результатів. В процесі розв'язання поставлених задач автором одержані такі нові результати:

– вперше розроблено метод динамічної горизонтальної фрагментації даних великого об'єму, який відрізняється від відомих використанням частоти звертань до кортежів даних, що дає змогу зменшити кількість збиткових фрагментів;

– розроблено новий евристичний метод для пошуку найкоротшого маршруту у графі з навантаженими вершинами та дугами, який відрізняється від існуючих тим, що не потребує обчислення всіх можливих маршрутів. Даний метод дозволяє розраховувати маршрути у графах великої розмірності без суттєвої втрати швидкодії;

– удосконалено спосіб оптимізації багатоекстремальних функцій за допомогою ГА, який на відміну від відомих, дозволяє зменшити час знаходження екстремуму;

– вперше розроблено метод декомпозиції *SQL*-запиту для визначення найбільш ефективного маршруту вибору даних із різних РБД.

Практичне значення одержаних результатів полягає в тому, що розроблено об'єктно-орієнтований модуль «*iSmartRoute*», що ґрунтується на евристичній методології. Його використання для оптимізації графів великої розмірності з навантаженими вершинами та дугами є швидшим у 520 раз за існуючі алгоритми з квадратичною чи кубічною складностями.

Поєднання методу декомпозиції *SQL*-запиту та методу для горизонтальної фрагментації даних дає змогу формувати базу знань, що представляє собою частоту звертань до кортежів з різних РБД. На основі такої статистичної інформації можна виконати горизонтальну фрагментацію БД великого об'єму із ефективним використанням дискового простору по декількох РБД з мінімальним числом збиткових фрагментів.

Застосування способу для оптимізації багатоекстремальних функцій дозволяє знаходити екстремум за мінімальний час, включаючи складні функції. Це досягається завдяки розробленим операторам кросинговера та мутації, покращеним характеристикам, що використовуються для ГА на основі якого й відбувається оптимізація.

Розроблені методи були використані в інформаційно-аналітичній системі (ІАС) «Мінтрансзв'язку» України та автоматизованій системі управління навчальним процесом в Інституті комп'ютерних технологій НАУ. Це дало змогу підвищити ефективність діяльності Інституту за рахунок надання оперативної і вчасної інформаційної підтримки в основних областях діяльності Інституту, таких як: кадрова політика, наукова та навчально-методична діяльність.

Отримані результати дисертаційної роботи реалізовані і впроваджені в начальному процесі на кафедрі комп'ютерних систем та мереж НАУ в дисциплінах: «Інтелектуальні комп'ютерні системи», «Комп'ютерні мережі» та «Мережеорієнтовані комп'ютерні технології» та в автоматизованій системі управління навчальним процесом в Інституті комп'ютерних технологій НАУ, що підтверджено відповідними актами про впровадження.

Особистий внесок здобувача. Основний зміст дисертаційної роботи та її результати повністю відображені в опублікованих наукових роботах автора та отримані здобувачем самостійно. Всі теоретичні і практичні результати, які складають основний зміст дисертаційної роботи опубліковано в 12 наукових працях. В роботах, написаних та опублікованих у співавторстві, здобувачеві належить: [1-2] – розробка автоматизованої системи управління навчальним процесом в Інституті комп'ютерних технологій НАУ; [3] – аналіз існуючих рішень для розподілення навантаження в БД великого об'єму (так і *WEB*-серверів в цілому); [4-5] – постановка задачі для організації розподілення навантаження БД в інформаційно-аналітичній

системі; [6] – аналіз ефективності використання постійних з'єднань для розподілення навантаження на *WEB*-серверах; [7] – проведення і аналіз результатів експериментальних досліджень при застосуванні горизонтальної фрагментації БД інформаційно-аналітичної системи; [9] – розробка інформаційно-аналітичної системи «Мінтрансзв'язку» України; [10] – метод для горизонтальної фрагментації та розподілу даних в інформаційно-аналітичній системі; [11] – метод декомпозиції *SQL*-запиту у РБД.

Апробація результатів дисертації. Результати досліджень за темою дисертації доповідались та обговорювались на міжнародних науково-технічних конференціях:

– «Современные информационные и электронные технологии» (18-22 травня 2009р., м. Одеса, Україна);

– «*International scientific technical conference*» (22-25 квітня 2009р., ХАІ, м. Кіровоград , Україна);

– Міжнародна науково-технічна конференція «Комп'ютерні системи та мережні технології» (10-12 червня 2009р., НАУ, м.Київ, Україна);

– «*Advanced computer systems and networks design and application: proceedings of the 4-th International conference*» (9-11 листопада 2009р., ЛПІ, м. Львів, Україна);

– Міжнародна науково-технічна конференції «Комп'ютерні системи та мережні технології» (15-17 червня 2010р., НАУ, м. Київ, Україна).

Публікації. Основні результати дисертаційної роботи опубліковані в 12 наукових працях, серед яких: 8 – у наукових фахових виданнях України, 4 – матеріалах конференцій.

Структура та обсяг дисертації. Дисертаційна робота, що викладена на 172 сторінках друкованого тексту, складається із вступу, чотирьох розділів і висновків, викладених на 130 сторінках основного тексту, списку використаної літератури із 127 найменувань. Дисертаційна робота містить 58 рисунків, 20 таблиць, 4 додатки та 2 акти про впровадження.

ОСНОВНИЙ ЗМІСТ РОБОТИ

У вступі обґрунтовано актуальність теми, сформульовано мету і завдання дослідження, відзначено отримані під час роботи нові наукові результати, їхня практична цінність, реалізація, апробація та структура дисертації.

У першому розділі проаналізовано способи для розподілення навантаження в базах даних великого об'єму. Розглянуто відомі методи балансування навантаження для *Web*-серверів. Приведено огляд існуючих методів та засобів для розподілення інформації в СУБД, зокрема, опрацювання розподілених запитів, межоперабельність в контексті розподіленої БД, тиражування даних. Виконано постановку задачі дисертаційного дослі-

дження, направленого на організацію розподілення навантаження для БД великого об'єму методом горизонтальної фрагментації.

У другому розділі проведено формалізацію ГА та моделі для популяції кінцевого розміру. Це дало змогу розробити новий евристичний метод «*iSmartRoute*» для оптимізації графів великої розмірності. Також, удосконалено спосіб оптимізації багатоекстремальних функцій, результати якого підтвердили практичну і наукову цінність еволюційних алгоритмів.

Оптимізація графів великої розмірності (сотні вершин та дуг і більше) – ключова проблема сьогодення. При знаходженні найкоротшого маршруту між двома вершинами існує безліч класичних алгоритмів (Дейкстри, Беллмана-Форда, Флойда-Уоршелла та інші). Недоліком даних алгоритмів є те, що вони мають квадратичну чи кубічну складності. Кількість шляхів експоненціально зростає в залежності від розміру графа, тому при значній насиченості (кількості ребер / дуг) жоден з них не буде ефективним по часі.

Нехай дано зважений граф $G=(V, E, w)$ (рис. 1), де V – множина вершин $\{A,B,C,D,E,F\}$; E – множина дуг; w – дійсна вагова функція w: $E \to R$; $c_1, c_2, \ldots, c_6$ – навантаження на вершинах; $d_1, d_2, \ldots, d_7$ – навантаження на дугах.

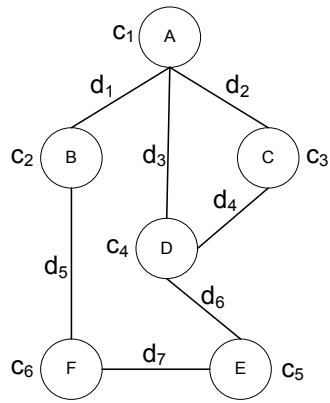


Рис. 1. Зважений граф з навантаженими вершинами та дугами

Вага шляху $p = (v_0, v_1, ..., v_n)$ – це сума ваг вершин та дуг, що входять у цей шлях:

$$\omega(p) = \sum_{i=1}^{n} \omega(v_{i-1}, v_i).$$

Вага найкоротшого шляху з u в v дорівнює, за визначенням,

$$\delta(u,v) = \begin{cases} \min\{\omega(p): \quad \overset{p}{u \to v}\}, & \text{якщо існує шлях із } u \text{ до } v, \\ \infty & \text{інакше.} \end{cases}$$

Тоді, найкоротший шлях з u в v – це будь-який шлях p з u в v, для якого $\omega(p) = \delta(u,v)$.

Для знаходження найкоротшого маршруту у графі з навантаженими вершинами та дугами використовується евристичний підхід та ГА.

Основу розробленого модуля «*iSmartRoute*» для пошуку найкоротшого маршруту на графі з навантаженими вершинами складають класи: *GeneticSearch*, *iSmartRoutePopulation* та *iSmartRouteIndividual*.

Представлення даних. Навантаження на вершинах представляється у вигляді асоціативного масиву, де ключ – це індекс вершини, а його значення – це вага вершини, ω. А навантаження на дугах – у вигляді матриці суміжності зі скінченною кількістю вершин n (пронумерованих числами від 1 до n), в якій значення елементу a_{ij} рівне вазі, ω, дуги з i-ї.

Функція пристосованості (фітнес-функція). Функція пристосованості дає кількісну оцінку оптимальності рішення (тобто, хромосоми) в ГА таким чином, що саме ця хромосома може бути вибрана по відношенню до інших хромосом.

Оптимальним хромосомам, або принаймні хромосомам, які є більш оптимальними, дозволено породжувати і змішувати свої набори даних на основі різних методів, створюючи нове покоління, яке, маючи надію, буде ще кращим.

$$F(G) = \sum_{i=1}^{n} \sum_{j=1}^{m} c_i d_{j,}, \qquad (1)$$

де n – кількість вершин; m – кількість дуг; c_i – навантаження на i-й вершині; d_j – навантаження на j-й дузі.

Фітнес-функція (1) повертає повну вагу маршруту, який закодований у хромосомі. Якщо порядок генів (вершин) у хромосомі відповідає не дійсному маршруту (що можливо при першому етапі ініціалізації випадкової популяції), тоді функція пристосованості поверне «-1».

Функція репродуктивності (кросинговер) використовується для зміни генетичного коду хромосоми від одного покоління до іншого з метою породження нащадка, який і буде знайденим рішенням в процесі схрещування.

Нехай дано 2 батьківські хромосоми: «Хромосома 1» та «Хромосома 2» (рис. 2), що представляють собою графи з двома різними маршрутами. Необхідно побудувати матрицю суміжності для обох графів, основуючись на вершинах і їх сусідах.

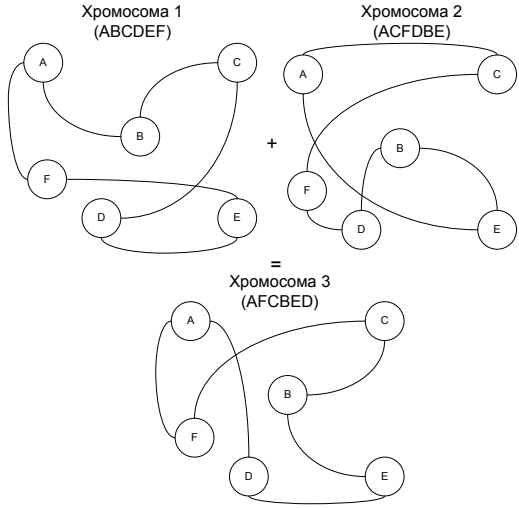


Рис. 2. Графічне зображення процесу репродуктивності (кросинговера) при схрещування двох особин

Для «Хромосоми 1» маршрут дорівнює *ABCDEF,* а для «Хромосоми 2» – *ACFDBE* відповідно. Тоді матриці суміжностей будуть наступними:

$\leftarrow A \leftrightarrow B \leftrightarrow C \leftrightarrow D \leftrightarrow E \leftrightarrow F \rightarrow$	$\leftarrow A \leftrightarrow C \leftrightarrow F \leftrightarrow D \leftrightarrow B \leftrightarrow E \rightarrow$
A: B F	A: C E
B: A C	B: D E
C: B D	C: A F
D: C E	D: F B
E: D F	E: A B
F: A E	F: C D

Далі шляхом об'єднання цих двох матриць та ігноруючи дублікатами, отримується:

A: B F $\cup$ C E = B F C E
B: A C $\cup$ D E = A C D E
C: B D $\cup$ A F = B D A F
D: C E $\cup$ F B = C E F B
E: D F $\cup$ A B = D F A B
F: A E $\cup$ C D = A E C D

Наступним кроком створюється нова «Хромосома 3» (нащадок) :

1. *R*:=[*A*]. Початкову вершину вибирається випадковим чином із {*A, A*}. Видаляються всі входження *A* у сусідніх списках. *A*={*B, F,C,E*}, шукається вершина з найменшою кількістю сусідів для *B, F, C* чи *E*. Оскільки їх кількість однакова, то випадково вибирається *F*.

1. *R*:=[*A,F*]. Найменшим набором для *E*, *C* та *D* є *C*={*B,D*} та *E*={*D,B*}. Випадково вибирається *C*.

2. R:=[A,F,C]. B={E}, D={E,B}. Вибирається B.

3. R:=[A,F,C,B]. Найменшим набором для D та E є D={E} та E={D}. Випадково вибирається E.

4. R:=[A,F,C,B,E]. У E залишилась одна вершина D.

5. R:=[A,F,C,B,E,D]. Довжина набору R еквівалентна довжині батьківської хромосоми. Кінець.

Функція мутації. Мутація застосовується для зміни генетичного коду хромосоми. Імовірність мутації зазвичай досить мала (частіше всього $0 < p_m < 0.1$), адже ставиться у відповідність із реальним світом живих організмів, де вона відбувається дуже рідко. Рівень мутації визначається *iSmartRoutePopulation::mutation_rate* в основі розробленого модуля.

Припустимо, що дано хромосому з таким набором генів {*A, C, F, D, B, E*}, тоді $L=6$ – її довжина (кількість генів у ній). g_1, g_2 – номери генів, що будуть мутувати. Вони можуть набувати таких значень: $\{g_1 \in N, 0 < g_1 <= L\}$ та $\{g_2 \in N, 0 < g_2 <= L, g_1 \neg g_2\}$ відповідно.

Нехай $g_1 = 3$, а $g_2 = 6$ (номери вибираються випадковим чином), тоді в результаті мутації буде хромосома з генами {*A, C, E, D, B, F*}, що зображена на рис. 3.

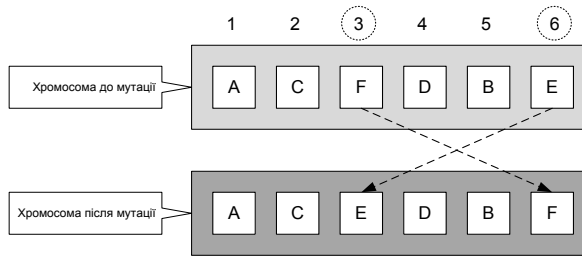


Рис. 3. Графічне зображення мутації хромосоми з генами {*A, C, F, D, B, E*}

Ефективність роботи розробленого евристичного методу «*iSmartRoute*» для знаходження найкоротшого маршруту у графі з навантаженими вершинами та дугами було порівняно із класичним алгоритмом Дейкстри. Випадковим чином генерувались графи із кількістю вершин n, де $100 < n < 1000$ (табл. 1). Вершини графа – це алфавітний набір із латинських літер {*A, B, ..., Z, AA, AB, ..., AZ, ..., ZZ*}. Дуги між вершинами створювались випадковим чином так, щоб мінімальна кількість маршрутів між початковою і кінцевою вершинами була не менше «кількість вершин / 2».

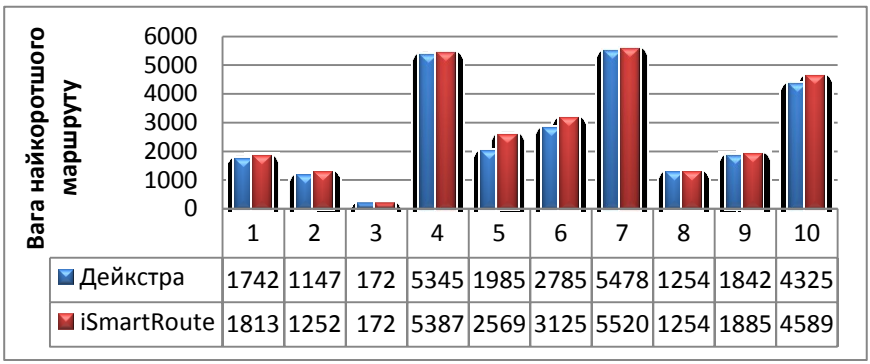
Таблиця 1

Кількість вершин для 10-ти випадкових графів

№ графа	1	2	3	4	5	6	7	8	9	10
К-ть вершин, n	374	678	113	754	562	621	936	324	478	879

На евристичний метод «*iSmartRoute*» було накладено ряд обмежень із-за того, що алгоритм Дейкстри не працює із дугами, що мають від'ємну

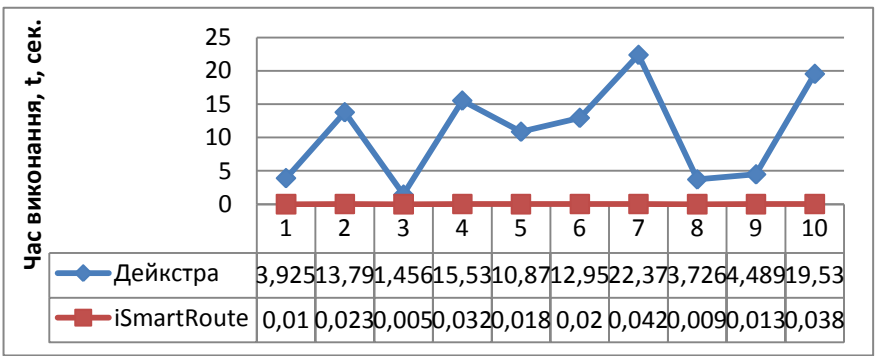
вагу та не підтримує роботи з одночасним навантаженням на вершинах та дугах. Інші алгоритми не брались до уваги, так як вони мають аналогічну квадратичну чи кубічну складності при аналізі графа з насиченою кількістю дуг. «*iSmartRoute*» виконувався із 10% мутацією та розміром популяції в 25 особин. Максимальна кількість поколінь дорівнювала 100.



	1	2	3	4	5	6	7	8	9	10
Дейкстра	1742	1147	172	5345	1985	2785	5478	1254	1842	4325
iSmartRoute	1813	1252	172	5387	2569	3125	5520	1254	1885	4589

Рис. 4. Перелік найкоротших маршрутів при роботі алгоритму Дейкстри та евристичного методу «*iSmartRoute*» для 10-ти графів

Результат знайдених найкоротших маршрутів для 10-ти графів після роботи обох алгоритмів показано на рис. 4. Сумарна вага найкоротших маршрутів для алгоритму Дейкстри – 26075, а для евристичного методу «*iSmartRoute*» – 27566, що на 5,72% більше. Тільки у двох випадках (при графі №3 та №8) евристичний метод «*iSmartRoute*» знайшов найкоротший маршрут, у всіх інших випадках, він був наближений до значень, що були отримані алгоритмом Дейкстри. Час, що був витрачений на пошук найкоротших маршрутів зображено на рис. 5.



	1	2	3	4	5	6	7	8	9	10
Дейкстра	3,925	13,79	1,456	15,53	10,87	12,95	22,37	3,726	4,489	19,53
iSmartRoute	0,01	0,023	0,005	0,032	0,018	0,02	0,042	0,009	0,013	0,038

Рис. 5. Часові характеристики при пошуку найкоротшого маршруту алгоритмом Дейкстри та евристичним методом «*iSmartRoute*»

Сумарний затрачений час на пошук найкоротшого маршруту для алгоритмів був таким: алгоритм Дейкстри – 108,6221 сек., та евристичний

метод «*iSmartRoute*» – 0,209 сек. відповідно (рис. 5). Це свідчить про те, що алгоритм Дейкстри був у 520 раз (99,80%) повільніше. Суттєво час його роботи збільшувався при збільшенні вершин для графа, що видно на рис. 6.

Рис. 6. Залежність часу від кількості вершин у графі при пошуку найкоротшого маршруту

Слід зауважити, що збільшення вершин на графі не суттєво впливає на роботу евристичного методу «*iSmartRoute*». Це зумовлено лише об'ємом даних які необхідно ініціалізувати та збільшенням розміру хромосоми індивідууму. Кількість поколінь і принцип тренування системи залишаються не змінними.

У третьому розділі розроблено метод декомпозиції *SQL*-запиту у РБД, що формує сукупність відомостей про частоту звертань до кортежів; проведено експериментальне дослідження використання постійних з'єднань з БД в традиційній кластерній системі та у сегментованій; розроблено метод горизонтальної фрагментації БД великого об'єму.

Нехай відомий час відкриття таблиці для зчитування, $\tau_{\text{вк}}$, і час для її закриття, $\tau_{\text{зк}}$. Знаючи час зчитування блоку даних, $\tau_{\text{б}}$, можна визначити час зчитування певного блоку, V_i, через $\frac{V_i+\tau_{\text{б}}}{V_{\text{б}}}$.

Тоді, час виконання запиту можна представити у вигляді:

$$\tau = \sum_{i=1}^{n}\left(\tau_{\text{вк}_i} + \tau_{\text{зк}_i} + \frac{V_i + \tau_{\text{б}}}{V_{\text{б}}}\right) + \tau_{\text{зд}},$$

де n – кількість таблиць у запиті; $\tau_{вк,\ i}$ – час відкриття i-ї таблиці; $\tau_{зк,\ i}$ – час закриття i-ї таблиці; $\tau_{б}$ – час зчитування блоку даних; $\tau_{зд}$ – загальний час виконання операцій з'єднання; V_i – об'єм і-ї таблиці; $V_{б}$ – об'єм блока.

Для вибору найкоротшого маршруту з'єднання таблиць у РБД необхідно представити *SQL*-запит у вигляді орграфа *G:=(V,A)*, де *V* – це множи-

на вершин, що представляє таблиці БД, та A – множина пар різних вершин (дуг), основою яких є умова приєднання двох таблиць.

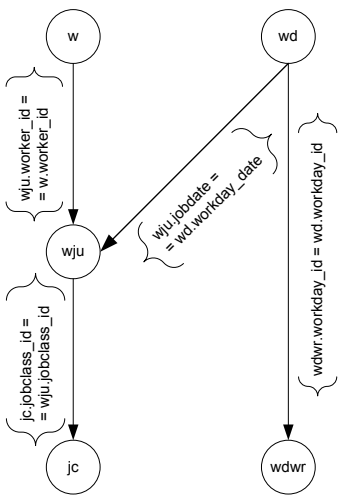
Наступним кроком необхідно побудувати зв'язний мультиграф, де кожній його вершині зіставляється навантаження c_i (час доступу і зчитування таблиці), а дузі – навантаження d_j (час на приєднання інцидентних їй таблиць, включаючи загальний час підключення до іншої РБД). Таким чином, для вибору оптимального маршруту необхідно вирішити задачу оптимізації на графі з навантаженими вершинами і дугами.

Задача оптимізації на графі полягає у виборі найменше навантаженого підграфа при умові, що результуючий підграф являється зв'язним. Для знаходження найкоротшого маршруту використовується розроблений евристичний метод «*iSmartRoute*» з фітнес-функцією:

$$f(G) = \left[\sum_{i=1}^{n} c_i + \sum_{j=1}^{m} d_j\right] * x_i \ \rightarrow min$$

де $c_i = \tau_{\text{вк}_i} + \tau_{\text{зк}_i} + \frac{V_i + \tau_{\text{б}}}{V_{\text{б}}}$ – навантаження на i-у вершину; n – кількість вершин; m – кількість дуг; d_j – навантаження на j-у дугу; x_i=1, якщо i-а вершина(таблиця) підлягає фрагментації, 0 – в іншому випадку.

На основі отриманого мінімального маршруту проводиться об'єднання таблиць із різних РБД та формування сукупності відомостей про частоту звертань до кортежів між РБД. Для аналізу ефективності оброблення запиту в РБД було взято *SQL*-запит, в основу якого входило приєднання п'яти таблиць із трьох РБД і на основі нього побудовано граф, виконавши перехід від таблиць до вершин (рис. 7):




```
SELECT STRAIGHT_JOIN
w.worker_firstname,w.worker_lastname,wd.workd
ay_date,wju.jobauit_start,wju.jobauit_end,jc.jobcl
ass_name,wdwr.workday_wagerate
FROM (workers w, workdays wd)
LEFT OUTER JOIN worker_jobaudit wju ON
(wju.worker_id = w.worker_id AND wju.jobdate =
wd.workday_date)
LEFT OUTER JOIN jobclasses jc ON
(jc.jobclass_id = wju.jobclass_id)
LEFT OUTER JOIN workday_wagerates wdwr
ON (wdwr.workday_id = wd.workday_id)
WHERE w.worker_spared = 0 AND wd.workdate
> DATE_SUB ( CURDATE(), INTERVAL 31
DAY)
ORDER BY wd.workday_date.
```

Рис. 7. Результуючий орграф на основі *SQL*-запиту до СУБД

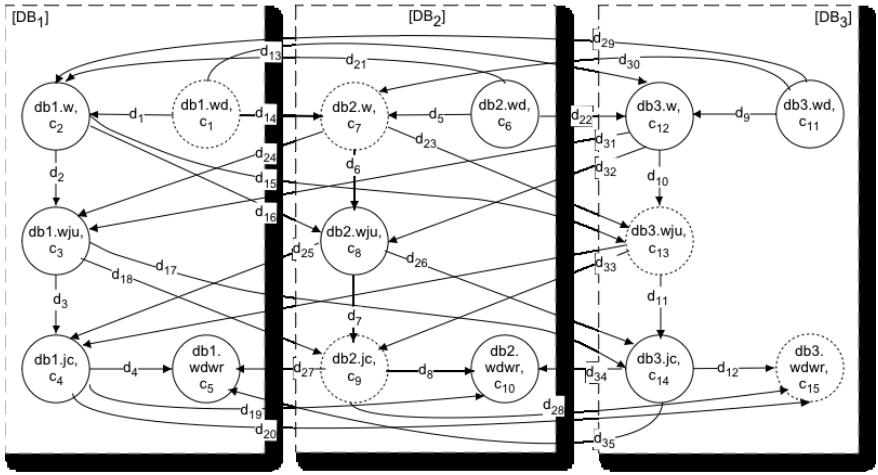


Рис. 8. Мультиграф з навантаженими вершинами та дугами

Наступним кроком побудовано зв'язний мультиграф з навантаженими вершинами та дугами (рис. 8). Він являв собою абстрактне представлення орграфа (рис. 7) на основі трьох РБД. На ньому схематично зображено: DB_1-DB_3 – розподілена БД по N=3 серверам; c_1-c_{15} – навантаження на вершинах; d_1-d_{35} – навантаження на дугах; c_1-c_7-c_{13}-c_9-c_{15} – початковий маршрут із попереднім станом фрагментованих даних; c_1-c_7-c_8-c_9-c_{10} – оптимізований маршрут.

Кінцевий список альтернативних маршрутів відображено у табл. 2, їх кількість дорівнює дев'ять і всі вони проходять рівно через п'ять вершин, оскільки запит (рис. 7) містить приєднання 5-ти таблиць.

Таблиця 2

Альтернативні маршрути

№	Маршрут	Час виконання, мсек
1	1,7,3,9,5	530
2	1,7,3,9,10	432
3	1,7,3,9,15	548
4	1,7,8,9,5	383
5	**1,7,8,9,10**	**285**
6	1,7,8,9,15	401
7	1,7,13,9,5	617
8	1,7,13,9,10	519
9	1,7,13,9,15	635

Маршрут №5 є оптимізований і має мінімальний час для виконання запиту. Початковий маршрут №9 – має 625мсек, що більше ніж у 2 рази.

Маючи інформацію про частоту звертань до кортежів даних наступних кроком було розроблення методу горизонтальної фрагментації в БД великого об'єму.

Нехай $R[A_1,A_2 \dots A_n]$ – відношення, де A_i, $i=1 \dots n$ – атрибути. Горизонтальний фрагмент можна отримати, наклавши обмеження: $R_i=\sigma_{cond}(R)$, де $cond_i$ є умовою. Таким чином, відновити початкове відношення можна об'єднавши наступні фрагменти: $R = R_1 \cup R_2 \cup \dots \cup R_k$.. Вертикальний фрагмент можна отримати добутком:

$$R_i = \prod_{\{A_{x_1}, A_{x_2}, \dots, A_{x_p}\}} (R),$$

де A_{xi}, $i=1,\dots, p$ – атрибути. Початкове відношення може бути відновлено об'єднанням фрагментів у такий спосіб: $R = R_1 \otimes R_2 \dots R_l$

Розглянемо БД великого об'єму (10^{12} байт і більше), що розподілена між 6-ма серверами «*A*», «*B*», «*C*», «*D*», «*E*» та «*F*», і яку можна представити у вигляді графа, де вузли являють собою набір вершин, а грані – показують безпосереднє з'єднання між вузлами. Кожна грань має відповідні витрати. Необхідно розподілити m кортежів по n вузлах графа при заданому значенні витрат між вершинами. Також заданою є статистика по частоті звертань до кортежів у графі (ця частота визначається розробленим методом декомпозиції *SQL*-запитів у РБД. Завдання розподілу кортежів зводиться до оптимізаційного завдання, метою якого є мінімізація витрат граней між різними вузлами РБД. В основі даного завдання лежить ГА та удосконалений спосіб оптимізації багатоекстремальних функцій. У запропонованому алгоритмі використовується популяція фіксованого розміру.

Наступним кроком необхідно розподілити m кортежів по вузлах, які позначено як $t_1, t_2,\dots, t_m$. Обмежень на максимальне або мінімальне число кортежів, збережених на одному вузлі, не встановлюється. Потенційне рішення завдання (хромосома) являє собою рядок фіксованої довжини $\{x_1, x_2,\dots,x_m\}$, де гени x_i, x_i $\{1, 2,\dots, n\}$, позначають на якому вузлі розташований i-й кортеж. Потенційне рішення оцінюється на підставі реального значення фітнес-функції F, $F: X \to R$, де X визначає область потенційного рішення. Функція пристосування хромосоми враховує витрати граней між вершинами (вузлами) і статистику щодо частоти звертань до кортежів у графі:

$$F(\vec{x}) = \sum_{i=1}^{n} \sum_{j=1}^{m} f_{ij} c_i W_j \ \to min, \qquad (2)$$

де f_{ij},($i\{1, 2, \dots, n\}$, $j\{1, 2, \dots, m\}$) являє собою частоту звернення до кортежу j з вузла i графа; c_iW_j, $i\{1, 2, \dots, n\}$, j $\{1,2,\dots,m\}$ являє собою витрати граней між вузлом i та вузлом, що містить кортеж j, що позначений як Wj. Фітнес-функція (2) має бути мінімізованою.

Для ГА використовується розрядний відбір, одно точковий оператор кросинговера та слабкий оператор мутації (10%). Алгоритм закінчує свою

роботу після певного числа генерацій, які не покращують краще значення популяції. Краще рішення, отримане в процесі пошуку, розглядається як рішення завдання.

У четвертому розділі проведено організацію розподілення навантаження в БД інформаційно-аналітичної системи методом горизонтальної фрагментації та описано її основні характеристики, включаючи її функціонування та захист інформації.

База даних інформаційно-аналітичної системи створена як сукупність взаємопов'язаних заходів, які призначені для розширення функціональних можливостей окремих інформаційно-аналітичних систем структурних підрозділів Держзв'язку України. Вона містить статистичну та аналітичну інформацію та забезпечує її систематизацію відповідно до видів економічної діяльності, номенклатури товарів та послуг в сфері інформатизації, надає можливість побудови аналітичних звітів про реалізацію завдань щодо формування і розвитку інформаційного суспільства в Україні.

Для тестування ІАС було розроблено два різних комплекси: перший («К1») – це один *Web*-сервер на якому знаходився програмний комплекс і не розподілена БД ІАС, а другий («К2») – це *Web*-сервер з програмним комплексом та трьома РБД , де дані були розподілені методом горизонтальної фрагментації.

Після завершення 1000 запитів отримано статистичні дані (табл. 3). Із них видно, що для «К1» лише 987 запитів завершились успішно, 13 – були помилковими. Це свідчить про перенавантаження на ньому. В цей час сервер взагалі не приймав *Web*-запитів.

Таблиця 3

Статистичні дані при тестуванні комплексів «К1» та «К2»

Параметр	«К1»	«К2»
К-ть одночасних запитів	100	100
Загальний час виконання тесту, сек.	15.714629	0.364127
К-ть запитів на сервер за один сеанс	1000	1000
К-ть помилкових відповідей від сервера	13	0
К-ть успішних відповідей від сервера	987	1000
Середня к-ть запитів за 1 сек.	63.63	2746.29
Середній затрачений час на один запит, мс.	1571.463	36.413
Середній затрачений час при всіх одночасних, мс.	15.715	0.364

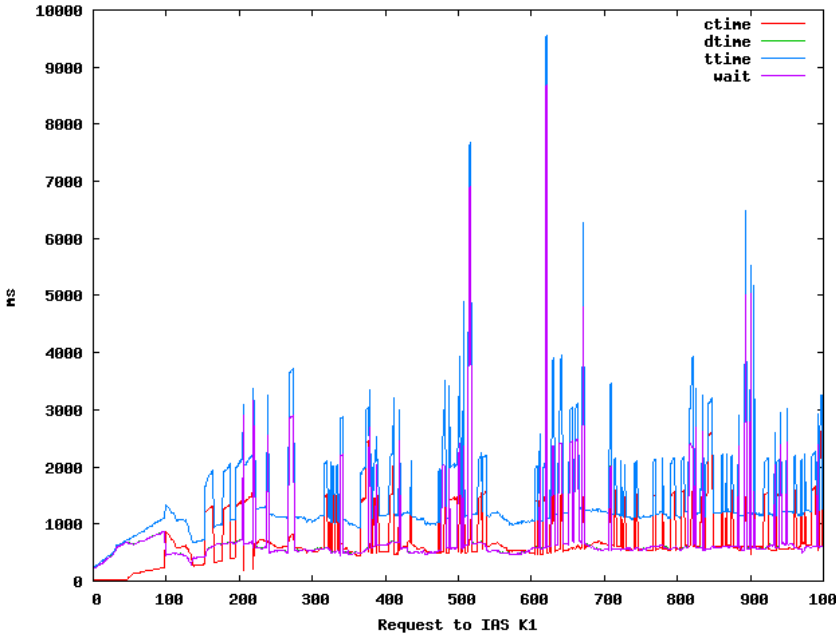


Рис. 9. Результати оброблення запитів комплексом К1 ІАС

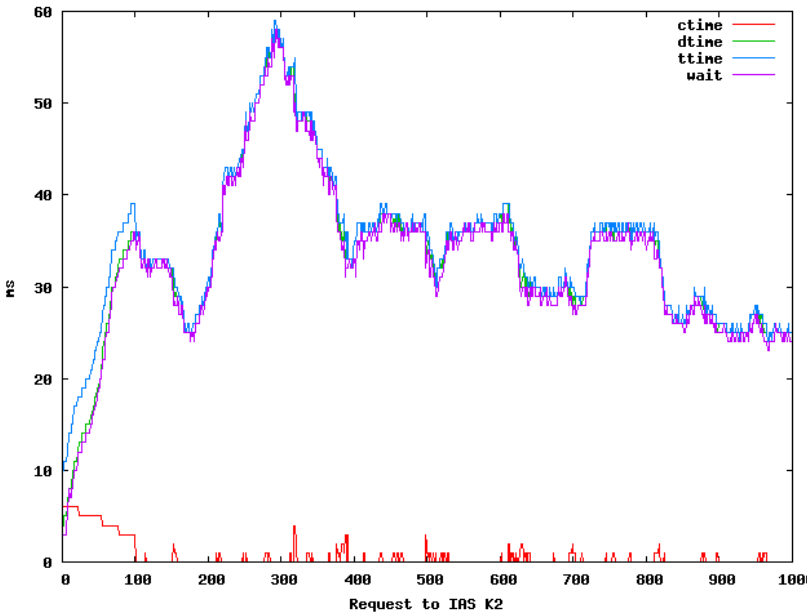


Рис. 10. Результати оброблення запитів комплексом К2 ІАС

Впровадження в ІАС методу горизонтальної фрагментації даних великого об'єму, дало змогу зменшити навантаження на БД, а також розподілити дані по тим РБД, які використовуються найчастіше. Комплекс «К2» виявився не тільки стабільнішим,а я й швидшим майже у 44 рази за «К1», що дало йому змогу виконати на 2683 запитів більше за одну секунду. Також слід відзначити пікові значення на обох графіках (рис. 9 і рис. 10). У «К1» спостерігались у більшості випадків одинакові показники виконаних запитів, а ось у «К2» – з подальшою кількістю запитів час виконання зменшувався. Це було досягнуто завдяки кешуванню і збереженню результатів в окрему БД.

ВИСНОВКИ

Мета та завдання дисертаційної роботи зумовили розв'язати науково-технічну проблему із організацією розподілення навантаження для БД великого об'єму. В ній подано нове вирішення актуальної проблеми шляхом горизонтальної фрагментації даних по кільком РБД. Дана дисертаційна робота присвячена застосуванню ГА для вирішення задач оптимізації графів з навантаженими вершинами та дугами та оптимізації багатоекстремальних функцій.

В результаті виконання дисертаційної роботи отримані наступні основні наукові і практичні результати і зроблено наступні висновки:

1. Вперше розроблено новий евристичний метод «*iSmartRoute*» для пошуку найкоротшого маршруту у графі з навантаженими вершинами та дугами, який відрізняється від існуючих тим, що для визначення найкоротшого маршруту не потребується обчислення всіх можливих маршрутів. Даний метод дозволяє розраховувати маршрути у графах великої розмірності без суттєвої втрати швидкодії;

2. Експериментальні дослідження між розробленим евристичним методом «*iSmartRoute*» та відомим алгоритмом Дейкстри, що має квадратичну складність, показали наступне: сумарне загальне відхилення «*iSmartRoute*» при знаходженні найкоротших маршрутів для 10-ти графів великої розмірності було рівним 5,72%, тоді як затрачений час на роботу алгоритму Дейкстри був на 99,80% більшим. Пошук найкоротшого маршруту алгоритмом Дейкстри був у 520 раз повільнішим.

3. Вперше розроблено новий метод декомпозиції *SQL*-запиту у РБД між різними розподіленими БД. Також, даний метод формує сукупність відомостей про частоту звертань до кортежів між РБД.

4. Вперше розроблено новий метод для горизонтальної фрагментації даних в БД великого об'єму, який у сукупності із методом декомпозиції *SQL*-запиту і його статистичними даними, дає змогу: проводити динамічну фрагментацію на основі частоти звертань до кортежів у реальному часі; зменшити час виконання *SQL*-запитів; ефективно використовувати дисковий простір серверів БД та зменшити навантаження на них, шляхом

ефективного розподілу даних з мінімальним числом збиткових фрагментів по кільком РБД.

5. Проведено експериментальне дослідження використання постійних з'єднань з БД в традиційній кластерній системі та у сегментованій. Отримані результати показали, що при використанні постійних з'єднань *Web*-сервер за 1 сек. може отримати у 2,32 рази більше запитів.

6. Удосконалено спосіб оптимізації багатоекстремальних функцій за допомогою ГА, який на відміну від відомих, дозволяє знаходити екстремум функції без обчислення всіх її можливих значень за мінімальний час.

7. Проведено експериментальне дослідження ефективності використання удосконаленого способу оптимізації складних багатоекстремальних функцій: «Сферичної моделі», «Долини Розенброка» та функції Растрігіна. Отримані результати показали, що із 50-ти тестувань «Сферичної моделі» та функції Растрігіна було знайдено локальний мінімум при 76% та 96% відповідно. А ось для «Долини Розенброка» глобального мінімуму не було знайдено, хоча до самої долини було наближено.

8. Розроблені методи та програмний модуль «*iSmartRoute*» були апробовані на практиці і довели свою ефективність, зокрема, в автоматизованій системі управління навчальним процесом в Інституті комп'ютерних технологій НАУ та інформаційно-аналітичній системі «Мінтрансзв'язку» України, що уже на початковому етапі дало змогу пришвидшити її роботу у 44 рази.

СПИСОК ОПУБЛІКОВАНИХ РОБІТ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

1. Жуков І. А. Адміністративно-начальна інформаційна система Інституту комп'ютерних технологій / І.А. Жуков, І.А. Клименко, І.М. Кравець // Проблеми інформатизації та управління: зб. наук. пр. – 2007. – Вип. 1(19). – С. 56–57.

2. Жуков І. А. Інформаційне забезпечення адміністративної та навчальної діяльності Інституту комп'ютерних технологій / [І.А. Жуков, І.А. Клименко, І.М. Кравець та ін.] // Вісник Національного технічного університету України «Київський політехнічний інститут», Інформатика, управління та обчислення: зб. наук. пр. – 2007. – Вип. 46. – С. 245–257.

3. Жуков І. А. Методи балансування навантаження для Web-серверів / І.А. Жуков, І.М. Кравець // Проблеми інформатизації та управління: зб. наук. пр. – 2007. – Вип. 3(21). – С. 46–54.

4. Жуков І. А. Розподілення навантаження баз даних в інформаційно-аналітичній системі / І.А. Жуков, І.М. Кравець // Проблеми інформатизації та управління: зб. наук. пр. – 2007. – Вип. 4(22). – С. 56–61.

5. Жуков І. А. Організація розподілення навантаження баз даних в інформаційно-аналітичній системі / І.А. Жуков, І.М. Кравець // Науковий вісник Чернівецького університету. Фізика. Електроніка.: Тематичний ви-

пуск «Комп'ютерні системи та компоненти»: зб. наук. пр. – 2008. – Вип. 426. – Ч.ІІ. – С. 44–50.

6. Жуков І. А. Постійні з'єднання з базами даних, як один із методів розподілення навантаження на WEB-серверах / І.А. Жуков, І.М. Кравець // Проблеми інформатизації та управління: зб. наук. пр. – 2008. – Вип. 2(24). – С. 5–13.

7. Zhukov I. A. Organization of distribution load database in the analysis and information system / I.A. Zhukov, I.M. Kravets // Radioelectronic and computer systems. – 2009. – Volume 5. – P. 25–30.

8. Кравець І. М. Оптимізація багатоекстремальних функцій за допомогою генетичного алгоритму / І.М. Кравець // Проблеми інформатизації та управління: зб. наук. пр. – 2010. – Вип. 2(30). – С. 56–61.

9. Жуков І. А. Програмно-апаратні засоби інформаційно-аналітичної системи / І.А Жуков, І.М. Кравець // Современные информационные и электронные технологии (СИЭТ-2009): междунар. науч.-практич. конф., 18-22 мая 2009 г.: тезисы докл. – Одесса: ОНПУ, 2009. – С. 115.

10. Жуков І. А. Еволюційний алгоритм фрагментації та розподілу даних в інформаційно-аналітичній системі / І.А. Жуков, І.М. Кравець // Комп'ютерні системи та мережні технології (CSNT-2009): ІІ міжнар. наук.-техн. конф., 10-12 червня 2009 р.: тези допов. – К.: Вид-во Нац. авіац. ун-ту «НАУ-друк», 2009. – С. 40.

11. Zhukov I.A. An algorithm of fragmentation optimization in distributed database / I.A. Zhukov, I.M. Kravets // Advanced Computer Systems and Application: 4-th International Conference (ACSN-2009), November 9-11, 2009. – Lviv, 2009. – P. 72–75.

12. Кравець І. М. Інтелектуальне розподілення навантаження для баз даних великого об'єму / І.М. Кравець // Комп'ютерні системи та мережні технології (CSNT-2010): ІІІ міжнар. наук.-техн. конф., 15-17 червня 2010 р.: тези допов. – К.: Вид-во Нац. авіац. ун-ту «НАУ-друк», 2009. – С. 54.

АНОТАЦІЯ

Кравець І.М. Розподілення навантаження в базах даних великого об'єму методом горизонтальної фрагментації. – Рукопис.

Дисертація на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.05 – комп'ютерні системи та компоненти. – Національний авіаційний університет Міністерства освіти і науки України, Київ, 2010.

Дисертація присвячена розробці загальної методології, методики та алгоритмів для організації розподілення навантаження в БД великого об'єму. Розроблено новий евристичний метод «*iSmartRoute*» для оптимізації графів великої розмірності. Також, удосконалено спосіб оптимізації багатоекстремальних функцій результати якого підтвердили практичну і

наукову цінність еволюційних алгоритмів. Розроблено метод декомпозиції *SQL*-запиту у РБД, що формує сукупність відомостей про частоту звертань до кортежів. Проведено експериментальне дослідження використання постійних з'єднань з БД в традиційній кластерній системі та у сегментованій. Розроблено метод горизонтальної фрагментації БД великого об'єму. Проведено організацію розподілення навантаження в БД інформаційно-аналітичної системи методом горизонтальної фрагментації та описано її основні характеристики, включаючи її функціонування та захист інформації

Ключові слова: СУБД, розподілена база даних, евристична методологія, генетичний алгоритм, графи великої розмірності, оптимізація багатоекстремальних функцій.

АННОТАЦИЯ

Кравец И.М. Распределения нагрузки в базах данных большого объема методом горизонтальной фрагментации. – Рукопись.

Диссертация на соискание ученой степени кандидата технических наук по специальности 05.13.05 – компьютерные системы и компоненты. – Национальный авиационный университет Министерства просвещения и науки Украины, Киев, 2010.

Диссертация посвящена разработке общей методологии, методики и алгоритмов для организации распределения нагрузки в базах данных большого объема.

Выполнен анализ существующих способов распределения нагрузки в базах данных большого объема. Рассмотрены известные методы балансировки нагрузки для *Web*-серверов. Приведены обзор существующих методов и средств для распределения информации в СУБД, в частности, обработка распределенных запросов, межоперабельность в контексте распределенной базы данных, тиражирование данных.

Разработан новый эвристический метод «*iSmartRoute*» для поиска кратчайшего маршрута в графе с нагруженными вершинами и дугами, который отличается от существующих тем, что для определения кратчайшего маршрута не требуется вычисления всех возможных маршрутов. Данный метод позволяет обрабатывать графы большой размерности без существенной потери производительности. Усовершенствован способ оптимизации многоэкстремальных функций результаты которого подтвердили практическую и научную ценность эволюционных алгоритмов. Разработан метод декомпозиции *SQL*-запроса в распределенной базе данных, который формирует совокупность сведений о частоте обращений к кортежам. Проведено экспериментальное исследование использования постоянных соединений с базой данных в традиционной кластерной системе и в сегментированной.

Разработан новый метод для горизонтальной фрагментации данных в базах данных большого объема, который в совокупности с методом декомпозиции *SQL*-запроса и его статистическим данным, позволяет: проводить динамическую фрагментацию на основе частоты обращений к кортежей в реальном времени; уменьшить время выполнения *SQL*-запросов; эффективно использовать дисковое пространство серверов баз данных и уменьшить нагрузку на них, путем эффективного распределения данных с минимальным числом избыточных фрагментов по нескольким распределенным базам данных. Проведено организацию распределения нагрузки в базе данных информационно-аналитической системы методом горизонтальной фрагментации и описаны ее основные характеристики, включая ее функционирования и защиту информации.

Ключевые слова: СУБД, распределенная база данных, эвристическая методология, генетический алгоритм, графы большой размерности, оптимизация многоэкстремальных функций.

ABSTRACT

Kravets I.M. Distribution of load balancing in large volume databases using horizontal fragmentation. – Manuscript.

PhD (Engineering) thesis, speciality (according to Ukrainian nomenclature of specialities) 05.13.05 . Computer systems and components. National Aviation University Ministry of education and science of Ukraine, Kyiv, 2010.

The thesis describes developing methodologies, techniques and algorithms for distribution of load in a large volume database. A new heuristic method iSmartRoute optimization for large-scale graphs is proposed. Improved way of optimization for functions with multiple extremums, the results of which confirmed the practical and scientific value of evolutionary algorithms. Developed a method of decomposing SQL-query in a distributed database to generate aggregate data about the frequency of traffic to tuples. An experimental study using permanent connections to the database in the traditional cluster system and segmented has been performed. Developed a method of horizontal fragmentation of large volume databases. Performed an organization of load balancing in database data-processing system using horizontal fragmentation and describes its main characteristics, including its performance and data protection.

Keywords: database, distributed database, heuristic methodology, genetic algorithm, large-scale graphs, optimization of multi-extreme functions.