ДОДАТОК А

Перелік джерел посилання за науковими напрямами керівника та науковців кафедри програмної інженерії

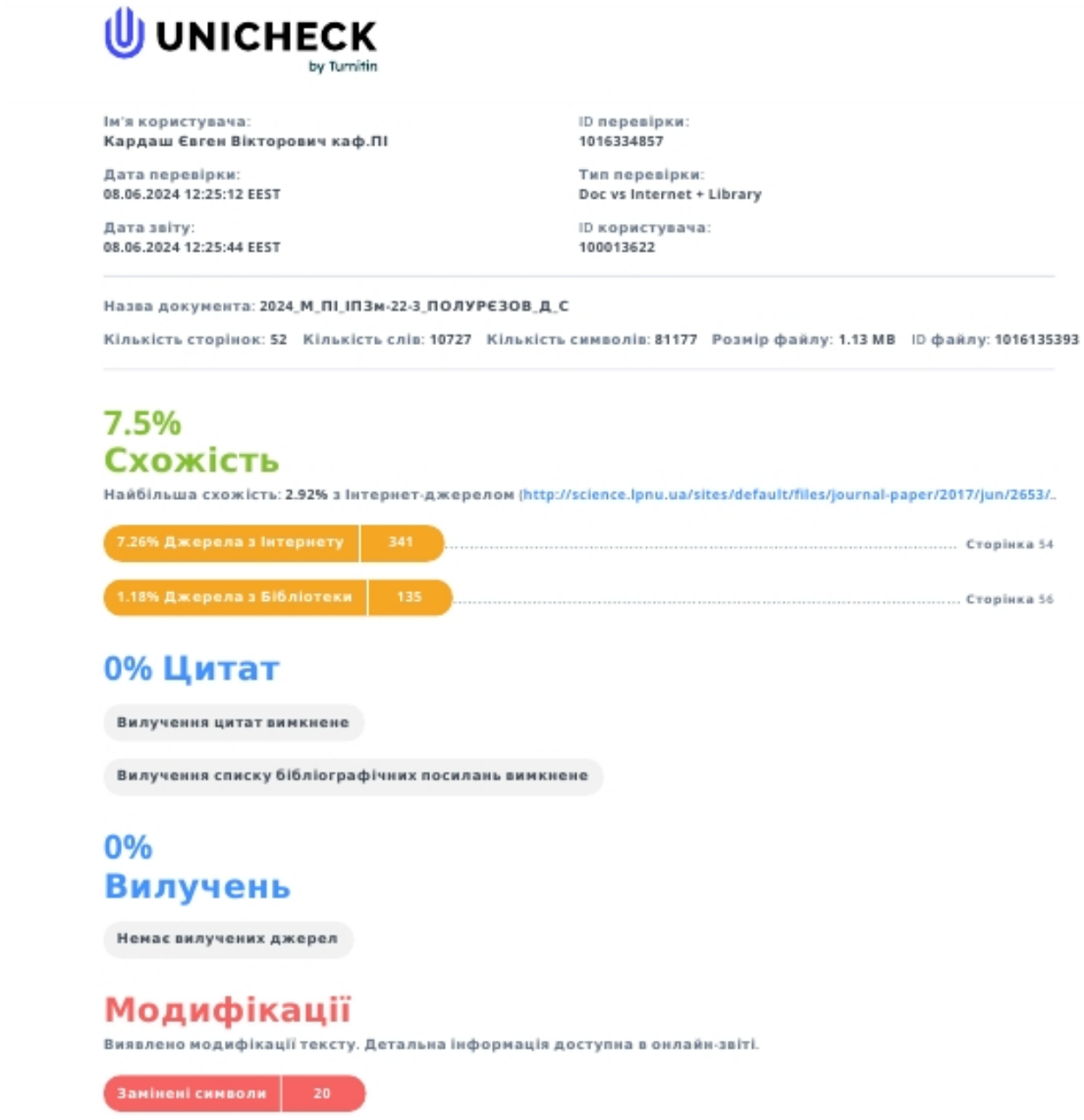
1. Kirill Smelyakov, Olha Klochko, Zoia Dudar. Building Quantile Regression Models for Predicting Traffic Flow // Proceedings of the 7th International Conference on Computational Linguistics and Intelligent Systems (COLINS), Volume I: Main Conference, 2023. In CEUR Workshop Proceedings, Vol-3387, 2023, pp. 117-132. URL: https://ceur-ws.org/Vol-3387/ (дата звернення: 16.02.2024).

2. Natalija Sharonova, Glib Tereshchenko. Application of Big Data Methods in E-Learning Systems // 5th International Conference on Computational Linguistics and Intelligent Systems (COLINS-2021), Kharkiv, Ukraine, April 22-23, 2021. – CEUR Workshop Proceedings 2870, Volume I, PP. 1302-1311. URL: http://ceur-ws.org/Vol-2870/ (дата звернення: 16.02.2024).

12. A. Khrystova, N. Kravets. Using lambda architecture for big data analysis / URL: https://scholar.google.ru/citations?view_op=view_citation&hl=ru&user=wnxpkmsAAAAJ&sortby=pubdate&citation_for_view=wnxpkmsAAAAJ:YOwf2qJgpHMC

ДОДАТОК Б

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



UNICHECK
by Turnitin

Ім'я користувача:
Кардаш Євген Вікторович каф.ПІ

ID перевірки:
1016334857

Дата перевірки:
08.06.2024 12:25:12 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
08.06.2024 12:25:44 EEST

ID користувача:
100013622

Назва документа: 2024_М_ПІ_ІПЗм-22-3_ПОЛУРЄЗОВ_Д_С

Кількість сторінок: 52 Кількість слів: 10727 Кількість символів: 81177 Розмір файлу: 1.13 MB ID файлу: 1016135393

7.5%
Схожість

Найбільша схожість: 2.92% з Інтернет-джерелом (http://science.lpnu.ua/sites/default/files/journal-paper/2017/jun/2653/..

7.26% Джерела з Інтернету 341 Сторінка 54

1.18% Джерела з Бібліотеки 135 Сторінка 56

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 20

ДОДАТОК В

Слайди презентації

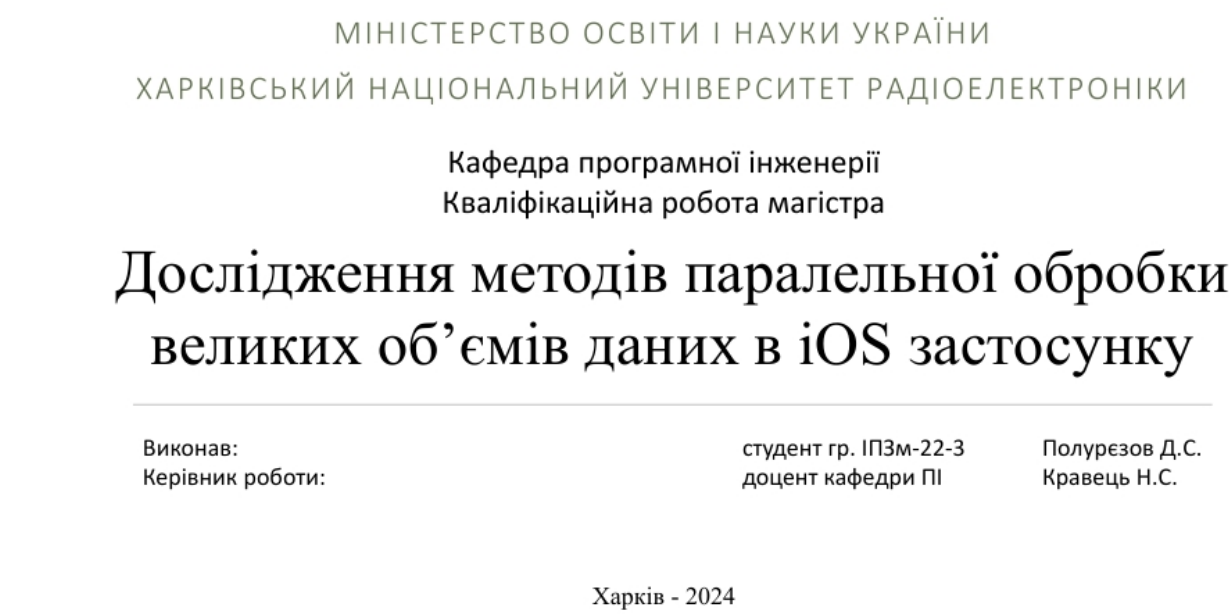


Рисунок В.1 – Слайд презентації 1

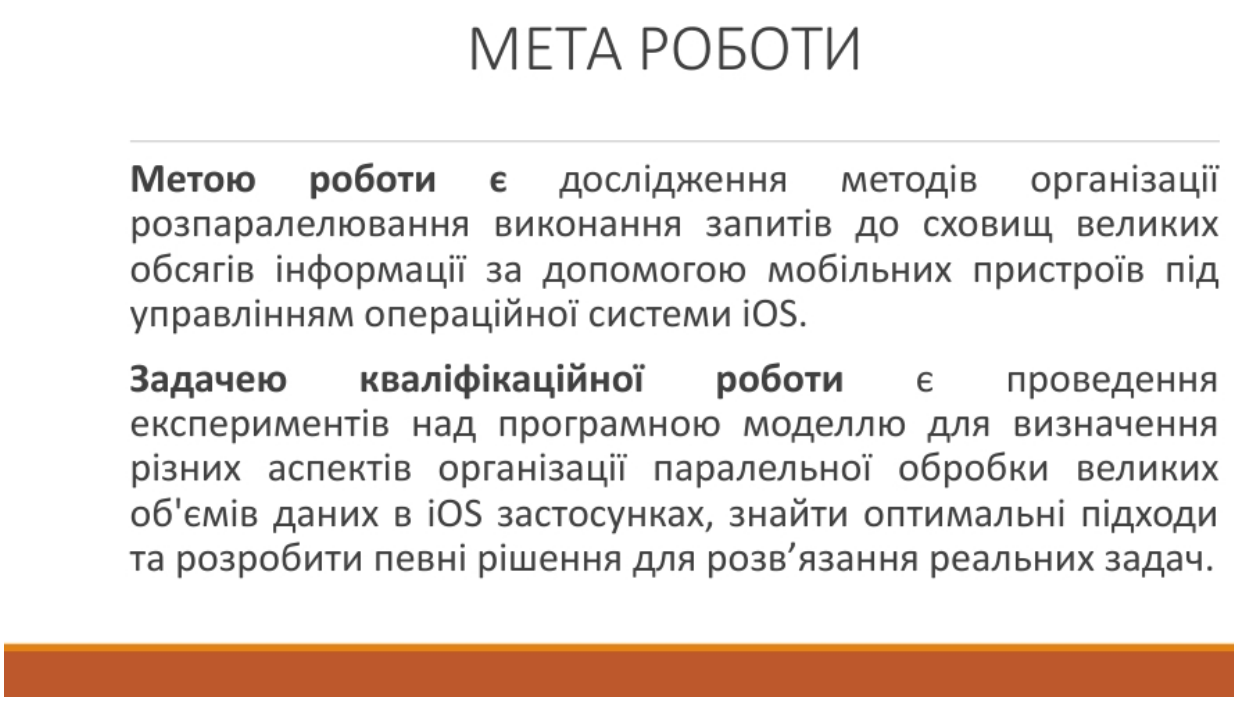


Рисунок В.2 – Слайд презентації 2

ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

Необхідно визначити шляхи підвищення продуктивності iOS-додатків за рахунок використання методів паралельної обробки запитів до BigData.

Для досягнення мети в роботі було виконано:

– проведено аналіз загальних методів організації паралельної обробки даних;

– проаналізовано підходи до організації паралелізації обчислювальних процесів на PMD Apple, що можуть бути використані для обробки великих обсягів неструктурованої інформації;

– побудовано математичну модель формування та обробки запитів до BigData;

– визначено предметну галузь та побудовано модель програмного додатку для використання на персональних мобільних пристроїв під управлінням iOS;

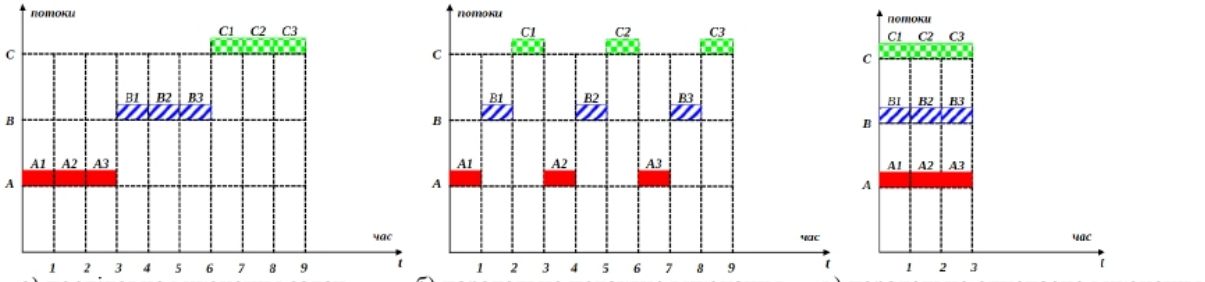
– виконано програмну реалізацію системи та проведено дослідження щодо порівняння ефективності запропонованих методів паралелізації при обробці великих обсягів даних;

– зроблено висновки щодо отриманих результатів проведених досліджень.

Рисунок В.3 – Слайд презентації 3

ОРГАНІЗАЦІЯ ПАРАЛЕЛЬНОЇ ОБРОБКИ

Під паралельними обчисленнями розуміють одночасне виконання комп'ютерної програми за допомогою декількох паралельних процесорів за допомогою потоків, що взаємодіють між собою.



а) послідовне виконання задач б) паралельне потактне виконання в) паралельне одночасне виконання

Сучасні персональні мобільні пристрої (PMD) – це енергоефективні комп'ютери з багатоядерними центральними процесорами (CPU). Найбільш перспективним варіантом застосування методів паралелізації на платформі PMD під управлінням iOS є їх використання для обробки не самих великих та надвеликих обсягів неструктурованої інформації, а запитів на їх обробку з боку сховищ Big Data

Рисунок В.4 – Слайд презентації 4

МОДЕЛІ СХОВИЩ BIG DATA

Поняття BigData з’явилось на початку XXI-го сторіччя як альтернатива традиційних систем управління БД.

Архітектурі моделі BigData:
- Business Intelligence;
- Streaming-модель;
- Лямбда-архітектура;
- Каппа-архітектура;
- Unifield-архітектура.

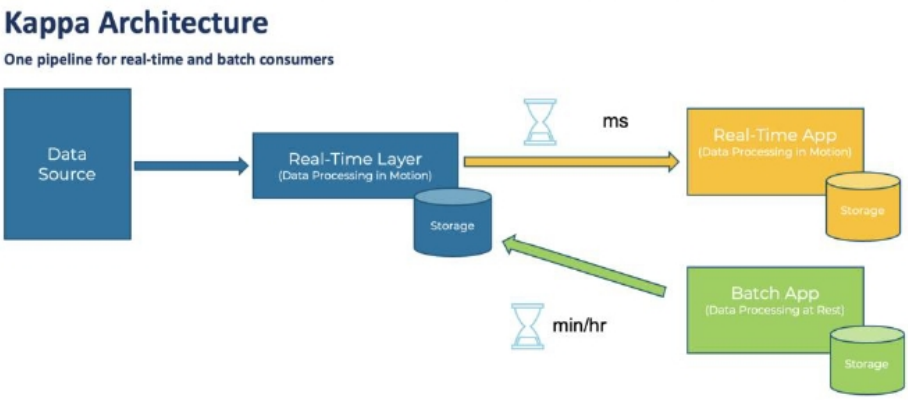


Рисунок В.5 – Слайд презентації 5

ОБРОБКА BIG DATA

Архітектура мобільного додатку, орієнтовного на паралельну обробку, суттєвим чином залежить від архітектури BigData та принципів організації доступу до даних. Тому подальші дослідження будуть спрямовані на визначенні певного архітектурного підходу при побудові програмних додатків, які будуть забезпечувати доступ до BigData за рахунок організації паралельного спрямування запитів до сховищ неструктурованої інформації Google BigTable за допомогою персональних мобільних пристроїв під управлінням операційної системи iOS

Рисунок В.6 – Слайд презентації 6

ПРЕДМЕТНА ОБЛАСТЬ

Дослідження будуть проводитись за допомогою програмного додатку, спроектованого в рамках роботи над проектом Nibble: Your Bite of Knowledge.

Це освітній продукт із короткими інтерактивними уроками з різних тем: від математики до мистецтва [13], які використовують метод надання рекомендацій для вибору та формування змісту цих уроків. Однією з ключових особливостей програмної системи Nibble є динамічна секція «For you», як за допомогою надання рекомендацій у предметній області дозволяє зробити аналітику уроків більш варіативною, підвищити відповідність конкретним потребам користувачів та підняти індивідуалізацію за рахунок запровадження власного підходу до вивчення матеріалу. Саме формування рекомендацій за рахунок виконання запитів до зовнішніх сховищ інформації було взято за основу для проведення досліджень.

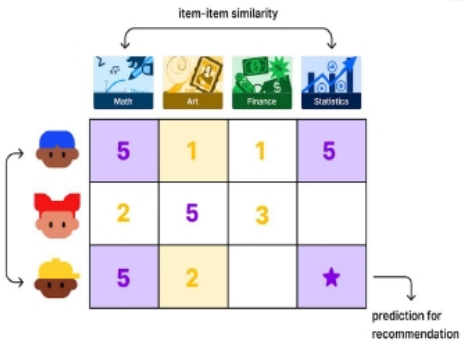


Рисунок В.7 – Слайд презентації 7

МАТЕМАТИЧНА МОДЕЛЬ ОРГАНІЗАЦІЇ ОБРОБКИ ДАНИХ

Кількість інформації можна представити виразом наступного виду:

$$I(e,f) = (1 + \log_2(n_{e,f})) * \log_2\left(\frac{|E|}{|e(f)|}\right) \quad (1)$$

Нормалізація значення важливості інформації:

$$V(e,f) = \frac{(1+\log_2(n_{e,f}))*\log_2\left(\frac{|E|}{|e(f)|}\right)}{\sqrt{\sum\left((1+\log_2(n_{e,f}))*\log_2\left(\frac{|E|}{|e(f)|}\right)\right)^2}}. \quad (2)$$

Відстань між сутностями e_1 та e_2:

$$d(e_1,e_2) = \sum_{f\epsilon F}|V(e_1,f) - V(e,f_2)|. \quad (3)$$

Відстань між сутностями можна вважати медіаною для формування кількості паралельних потоків, призначених для обробки Big Data, а максимальна кількість потоків залежить від кількості потоків, що підтримується CPU відповідної родини.

Рисунок В.8 – Слайд презентації 8

МОДЕЛІ АСОЦІАЦІЙ МІЖ СУТНОСТЯМИ ТА ХАРАКТЕРИСТИКАМИ

Основним елементом будь-якої БД слід вважати ключ. Носієм даних в моделі NoSQL БД є кортеж:

$$KV = \{\langle f, e\rangle\},$$

де f – ключ, який приймає унікальне значення для кожної пари; e – значення, яке йому відповідає.

Основу моделі збереження даних в Google BigTable складають рядки, стовпці та тимчасові мітки:

$$BigTable = \{<r, c, t>\}.$$

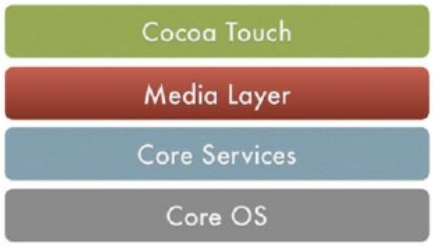
Якщо в декількох стовпцях зберігаються дані, що відносяться до одного типу, то такі стовпці, згідно з моделлю Bigtable, утворюють сімейство:

$$col\ F = \{\ c_i, c_j \mid \text{dom}\ (c_i) \in T \wedge \text{dom}\ (c_j) \in T\}.$$

Шляхом використання тимчасові мітки у якості ключа, додатки можуть виконувати в BigTable пошук, наприклад, тільки найновіших копій даних

Рисунок В.9 – Слайд презентації 9

Особливості організації обробки потоків в iOS



Рівні iOS

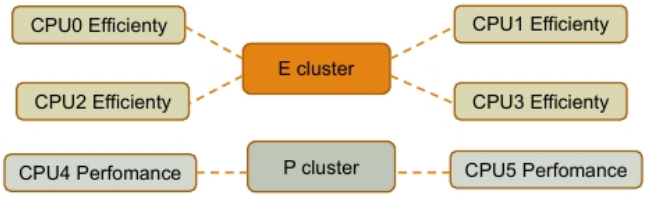
При розробці додатків для iOS термін «потік» відноситься до одиниці виконання задачі в межах процесу. Процес може мати декілька потоків, кожен з яких може виконувати код паралельно із іншими потоками у тому ж самому процесі.

Додатки для iOS мають обмежені ресурси, і створення занадто багатьох потоків може призвести до проблем продуктивності або навіть до збоїв. Тому важливо розсудливо використовувати багатопотоковість, враховуючи компроміс між паралельністю та використанням ресурсів

Рисунок В.10 – Слайд презентації 10

Архітектура процесорів Apple

Сучасні моделі багатоядерних процесорів Apple мають у своєму складі як високопродуктивні ядра (P-ядра), так і високоефективні (E-ядра).



Операційна система iOS самостійно приймає рішення щодо використання ядер процесора на основі пріоритетів, призначених потокам, більш відомих як якість обслуговування (QoS). Потоки з низьким QoS майже завжди виконуються на E-ядрах, тоді як потоки з високим QoS переважно розподіляються на P-ядра, але за потреби можуть виконуватися і на E-ядрах.

Рисунок В.11 – Слайд презентації 11

Багатопоточність в CPU Apple

iOS безпосередньо взаємодіє з процесором на рівні ядра операційної системи за рахунок використання функції Grand Central Dispatch (GCD), яка приймає блок даних і викликає його виконання декілька разів на доступних ядрах системи

Незважаючи на той факт, що iOS є багатозадачною ОС, вона не підтримує декілька одночасних процесів в межах однієї програми. Разом з тим розробникам надається у використання клас Process, призначений для створення нових дочірніх процесів, які не залежать від батьківського процесу, але містять всю інформацію, яку мав батьківський процес на момент створення дочірнього процесу. Тому головною задачею, пов'язаною з організацією паралельної обробки в iOS, є організація ефективного управління чергами.

В iOS існує один користувальницький процес, в якому можна визначити до 64 окремих потоків [17]. Для керування цими потоками Apple рекомендує використання черг розсилки: існує можливість додавати завдання до черги шляхом відправлення повідомлень до операційної системи та очікувати, доки вони не будуть виконані у певний момент часу

Рисунок В.12 – Слайд презентації 12

Вибір засобів реалізації

Розробка та програмна реалізація моделі програмної системи, яка орієнтована на організацію паралельної обробки BigData, було виконано з використанням Apple MacBook Pro 16" M1 Pro 32/512GB, операційна система macOS Monterey, середовище розробки Xcode 15.3 (15E204a), мова програмування Swift 5. Для вимірювання завантаження процесора під час виконання окремих операцій використовувалось підключення PMD до комп'ютера з використанням програми Activity Monitor.

Експерименти щодо дослідження ефективності методів паралельної обробки великих обсягів неструктурованої інформації з використанням PMD під управління iOS будуть проводитись з використанням наступного апаратного обладнання:

– Apple iPhone 14 128GB (процесор Apple A15 Bionic, iOS 17.5.1);
– Apple iPad Pro 11" (4 Gen) Wi-Fi 128GB (процесор Apple M2, iOS 17.5.1).

Рисунок В.13 – Слайд презентації 13

Вибір засобів реалізації

Середовище розробки Swift пропонує різні методи підтримки багатопотоковості, серед яких GrandCentralDispatch, OperationQueue, NSLock

GrandCentralDispatch (GCD) – це високорівневий механізм управління багатопотоковістю, який дозволяє легко та ефективно виконувати завдання паралельно. Він використовує черги для організації та виконання завдань та забезпечує автоматичне розподілення завдань між різними потоками

OperationQueue також є високорівневим інтерфейсом над GCD, який дозволяє створювати та виконувати операції. Це забезпечує додаткові можливості, такі як встановлення залежностей між операціями та управління конкурентністю

NSLock є механізмом блокування для управління доступом до ресурсів з різних потоків. Він дозволяє тільки одному потоку доступ до даних, блокуючі у конкретний момент часу всіх інших для того, щоб уникнути конфліктів доступу до даних

Рисунок В.14 – Слайд презентації 14

Модель програмної системи

Мобільний додаток було спроєктовано з використанням архітектурного підходу The Composable Architecture [21], адаптованого для операційної системи iOS

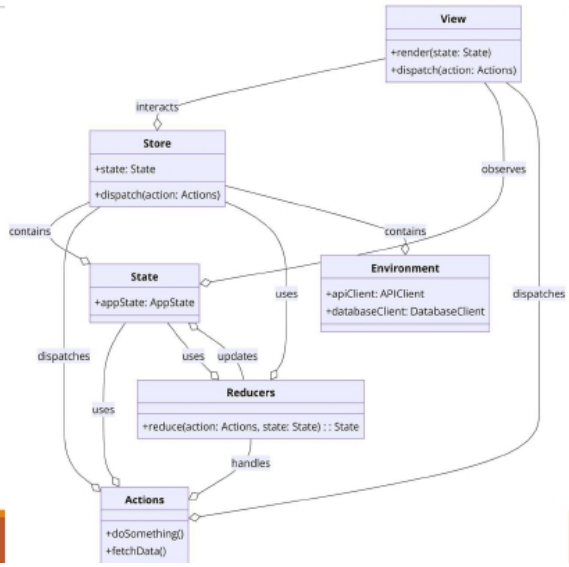


Рисунок В.15 – Слайд презентації 15

Модель програмної системи

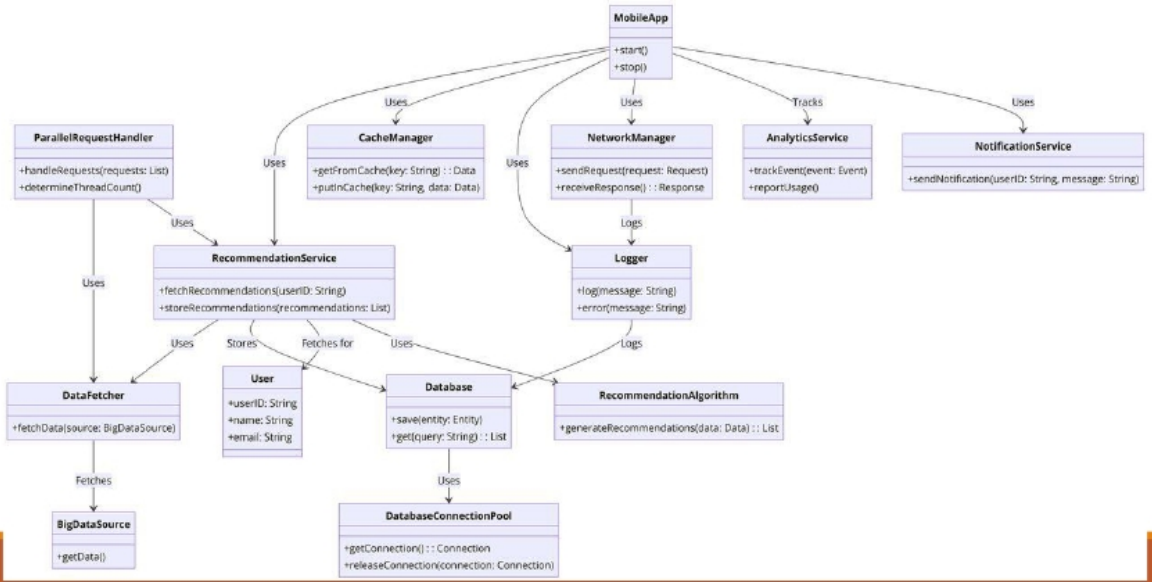


Рисунок В.16 – Слайд презентації 16

Структура локальної БД

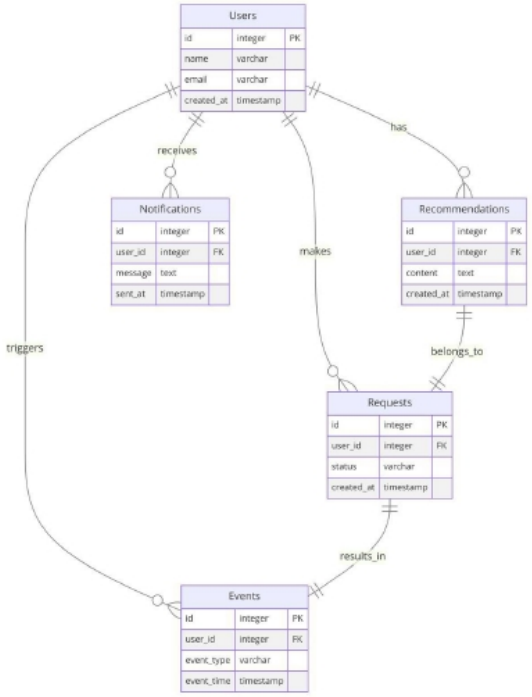


Рисунок В.17 – Слайд презентації 17

Планування експериментів

Для порівняння продуктивності організації обробки даних дослідження проводитись з використанням робочого простору, організованого з використанням Google BigTable, у відповідності до наступних сценаріїв запитів:

– пошук 10000, 100000 та 200000 випадково визначених чисел в діапазоні від 0 до 1000 серед записів у сховищі Big Data;

– вибірка зі сховища записів у кількості 10000, 100000 та 200000 записів;

– додавання 100, 500, 1000 записів до сховища;

– читання 1000, 10000, 100000 записів зі сховища.

Результати проведення експериментів будуть порівнюватись між собою за наступними параметрами:

– кількість потоків;
– навантаження процесору;
– час виконання сценарію

Рисунок В.18 – Слайд презентації 18

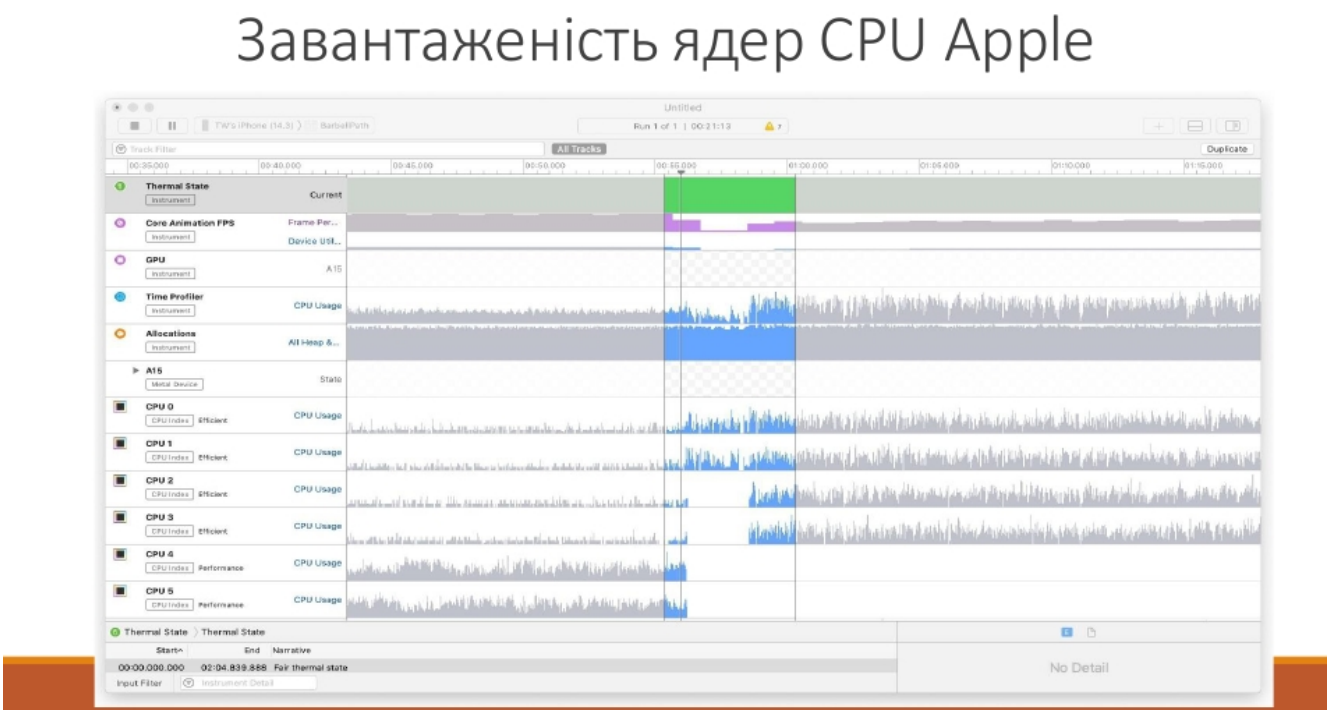


Рисунок В.19 – Слайд презентації 19

Рисунок В.20 – Слайд презентації 20

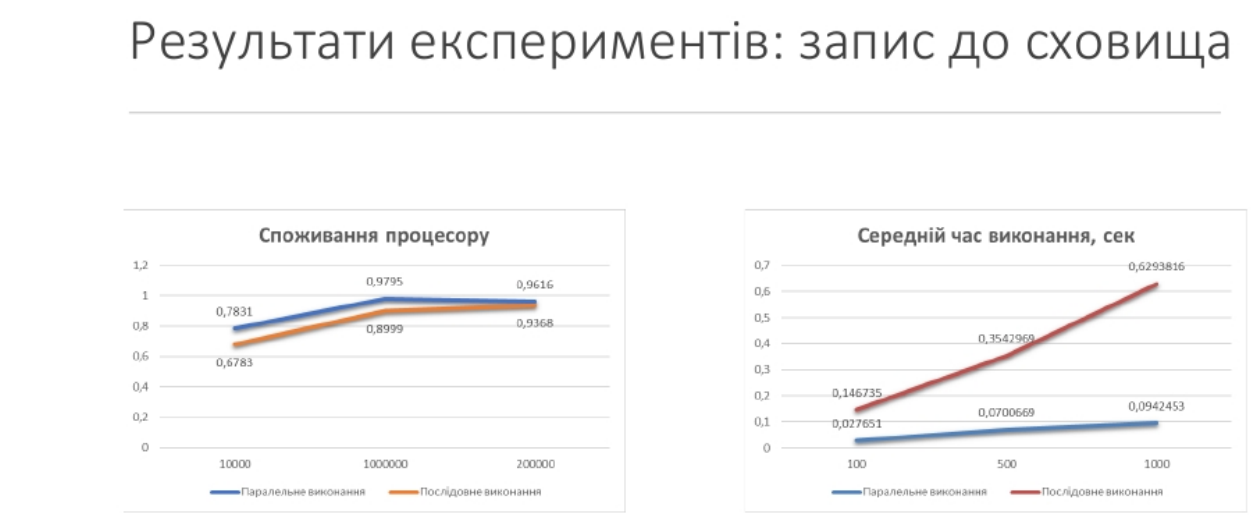


Рисунок В.21 – Слайд презентації 21

Рисунок В.22 – Слайд презентації 22

Отримані результати: висновки

1. Результати проведених експериментів свідчать про відносну ефективність використання методів розпаралелювання при розробці програмних додатків для PMD Apple у зв'язку з неможливістю організації відокремлених черг до енергоефективних ядер E та високопродуктивних ядер P.
2. Ефективність застосування GCD та OperationQueue при формуванні та обслуговуванні черг значним чином залежить від версії операційної системи та родини процесора, який було використано в певному PMD
3. Сучасні фреймворки та інструментальні засоби, що використовуються на теперішній час при розробці програмних додатків для PMD Apple iPhone під управлінням iOS, не забезпечують можливості керувати типом ядра процесора, на якому бажано запустити процес, однак достатньо ефективні при організації обробки черг і надають можливість керувати ними за рахунок зміни параметрів QoS для кожного з потоків

Рисунок В.23 – Слайд презентації 23

Висновки

Під час підготовки кваліфікаційної роботи було виконано

1. Аналіз предметної області дослідження;
2. Виконано постановку задачі;
3. Досліджено методи розпаралювання при організації обробки даних
4. Розроблено математичну модель обробки даних;
5. Спроектовано та виконано програмну реалізацію двох програмних систем для дослідження ефективності організації паралельної обробки
6. Проведено експериментальні дослідження
7. Зроблено висновки щодо ефективності застосування розпаралелювання для PMD Apple під управлінням iOS

Рисунок В.24 – Слайд презентації 24

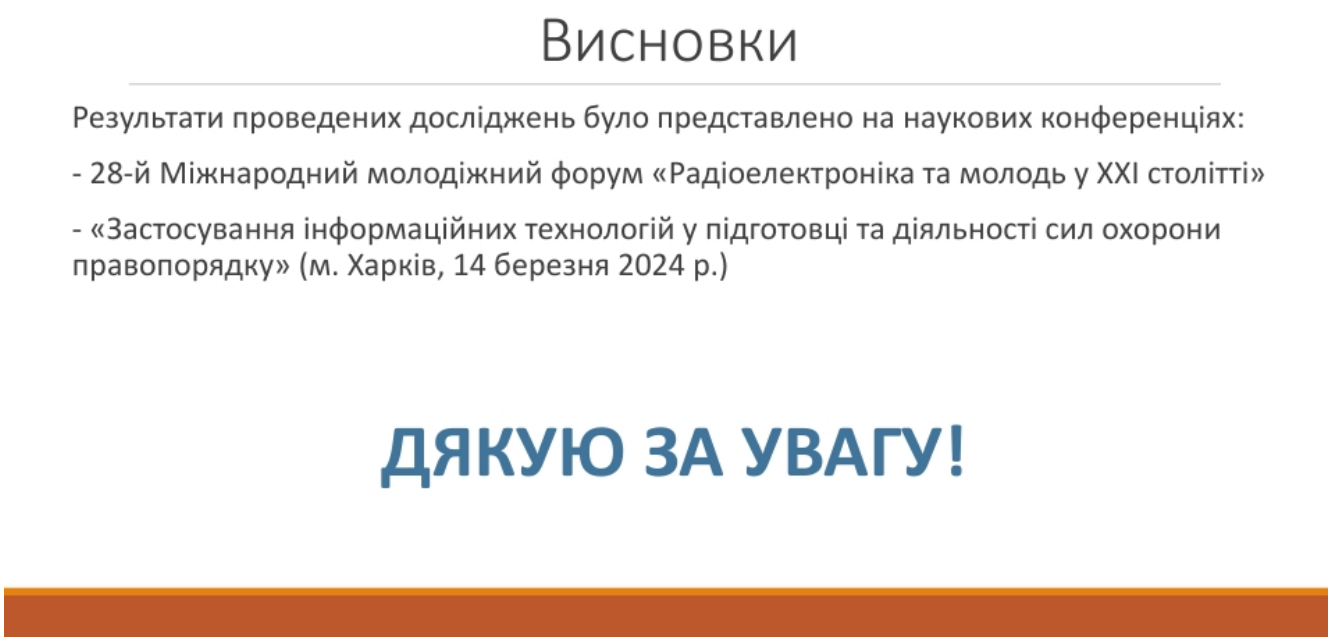


Рисунок В.25 – Слайд презентації 25

ДОДАТОК Г

Апробація результатів роботи

УДК 004.62 DOI: https://doi.org/10.30837/IYF.IIS.2024.405

ПРО ОДИН ПІДХІД ДО ОБРОБКИ ВЕЛИКИХ ОБСЯГІВ НЕСТРУКТУРОВАНИХ ДАНИХ

Полурєзов Д. С.
Науковий керівник – к.т.н., доц. Кравець Н. С.
Харківський національний університет радіоелектроніки, кафедра ПІ
м. Харків, Україна
dmytro.poluriezov@nure.ua

The issue of real-time processing of large datasets is delineated. The utilization of binary search method for handling unstructured data through parallelizing queries to remote repositories according to the number of processor cores of the mobile device is examined. This method can find its application in organizing parallel processing of substantial volumes of unstructured information on personal mobile devices operating under the iOS operating system.

Поняття Big Data з'явилось на початку XXI-го сторіччя як альтернатива традиційним системам управління базами даних. Big Data на теперішній час використовується для позначення структурованих та неструктурованих даних, що є значними за обсягом, та масштабуються горизонтально. Разом з тим слід зазначити, що великі дані – це також сукупність технологій, орієнтованих на:

– обробку дуже великих за обсягом, у порівнянні зі «стандартними» сценаріями, обсягів даних;

– опрацювання даних, що швидко надходять у дуже великих обсягах;

– операції зі структурованими та мало структурованими даними паралельно та у різних аспектах їх використання.

На теперішній час технології обробки Big Data набули поширення. Хоча проблема обробки сотень гігабайт даних вже вирішена, але дані, що були створені рік тому, набагато менш цінні, ніж дані, які було отримано протягом останньої години. Збільшення кількості потоків для паралельної обробки даних може значно підвищити продуктивність систем. Однак нескінченне збільшення кількості потоків не призведе до аналогічного збільшення продуктивності. Для оптимізації використання потоків може бути застосовано методи бінарних запитів.

Загалом будь-які дані, що можуть розглядатись як джерело великих обсягів неструктурованих даних, мають щонайменше два елементи: самі дані та їх характеристики. Формально поділимо ці об'єкти на категорії:

– сутності $\boldsymbol{e}$;

– характеристики $\boldsymbol{f}$;

– асоціації між сутностями $\boldsymbol{e}$ та характеристиками $\boldsymbol{f}$.

Наприклад:

– сутність $\boldsymbol{e}$ належить документі $\boldsymbol{f}$;

– посилання на $\boldsymbol{f}$ з'явилось у зв'язку із сутністю $\boldsymbol{e}$.

Таким чином також можна визначити:

– множину сутностей E ;

– множину характеристик F;

– для кожних e та f визначено номер асоціацій між e та f як $n_{e,f}$.

Визначимо загальну кількість сутностей через $|E|$; тоді кількість характеристик можемо визначити як потужність множини F: $|F|$. Відповідно до обраних припущень можна отримати:

– для кожної характеристики f – множину $e(f) = \{e \in E : n_{e,f} > 0\}$ для усіх асоційованих з f сутностей;

– для кожної сутності e – множину $f(ef) = \{f \in EF : n_{e,f} > 0\}$ для усіх асоційованих з e характеристик.

В тому випадку, коли присутні декілька сутностей, пов'язаних з однією характеристикою об’єкту, будемо використовувати алгоритм бінарного пошуку, який дозволяє за результатами відповіді на q бінарних запитів отримати множину з $N \cdot 2^{-q}$ елементів, яка буде містити необхідний об’єкт, а кількість запитів буде обчислюватись як $q = log_2(N)$ [1].

Такий саме спосіб можна використати і для сутностей. Існує E сутностей, що містять визначену кількість інформації: $\log_2(E)$. Якщо відомо, що будь-яка сутність асоційована з певною характеристикою (існує $e(f)$ сутностей), то кількість інформації буде визначатись як: $(|e(f)|)$.Тоді той факт, що сутність пов'язана з характеристикою f, дає змогу зменшити кількість бінарних запитів до:

$$(|e(f)|) = \left(\frac{|E|}{|e(f)|}\right).$$

Кількість асоціацій також можна визначити за допомогою бінарних запитів, які необхідно сформувати для того, щоб і надалі асоціація з необхідною сутністю була відома. Кожний бінарний запит для $n_{e,f}$. зменшує кількість цих об’єктів вдвічі, а формування q запитів зменшує цю кількість до $n_{e,f} \cdot 2^{-q}$. Асоціація буде існувати до того часу, поки кількість об’єктів буде більшою за 1. Тоді найбільша кількість запитів q , для якої ще існує асоціація, можна визначити як $N \cdot 2^{-q} = 1$, що, у свою чергу, можна визначити як $q = \log_2(n_{e,f})$. Формування будь-якого додаткового запиту буде визначатись як $1 + \log_2(n_{e,f})$.

На підставі викладеного характеристики f для сутності e можна визначати як:

$$\left(\frac{|E|}{|e(f)|}\right)$$

з фактором важливості $1 + \log_2(n_{e,f})$. Враховуючи це, результуючу кількість інформації можна представити виразом наступного виду:

$$(n_{e,f})) * \left(\frac{|E|}{|e(f)|}\right) \quad (1)$$

Формула (1) є одним з варіантів визначення для кожної сутності e важливості $I(e, f)$ у відповідності до різних характеристик f .

Виконаємо нормалізацію значення важливості:

$$V(e,f) = \frac{(1+(n_{e,f}))\times\left(\frac{|E|}{|e(f)|}\right)}{\sqrt{\sum \left(\left(1+(n_{e,f})\right)\times\left(\frac{|E|}{|e(f)|}\right)\right)^2}}. \quad (2)$$

Кожна із сутностей ***e*** має вагу ***V (e, f)***, тому мірою наближення об'єкту E_i до об'єкту E_n слід вважати відстань між відповідними векторами $V(e, f_i)$ та $V(e, f_n)$.

Для кожної ваги ***V (e, f)***, що репрезентує визначену кількість бітів, відстань між сутностями e_1 та e_2 буде обчислюватись у відповідності до:

$$d(e_1, e_2) = \sum_{f \in F} \left|V(e_1, f) - V(e, f_2)\right|. \quad (3)$$

Ця відстань залежить від кількості характеристик: якщо, наприклад, крім самих документів ми зберігаємо ще їх копії, то відстань збільшується вдвічі. Виконаємо нормалізацію відстані $d(e_1,e_2)$ в інтервалі [0,1] шляхом її ділення на максимально можливе значення цієї відстані.

В умовах невизначеності, коли значення ***A*** та ***B*** невідомі, а є лише верхні межі цих величин $\underline{a}$ та $\underline{b}$, то найбільша можлива різниця буде становити $max(\underline{a}, \underline{b})$, а саме [2]:

– якщо $\underline{a} \le \underline{b}$, то $|\underline{a} - \underline{b}| = \underline{b} - \underline{a} \le \underline{b}$ та $|\underline{a} - \underline{b}| \le$ **max** $(\underline{a}, \underline{b})$;

– якщо $\underline{b} \le \underline{a}$, то $|\underline{a} - \underline{b}| = \underline{a} - \underline{b} \le \underline{a}$ та $|\underline{a} - \underline{b}| \le$ **max** $(\underline{a}, \underline{b})$,

тобто в обох випадках виконується $|\underline{a} - \underline{b}| \le$ **max** $(\underline{a}, \underline{b})$.

Межа $max\ (\underline{a}, \underline{b})$ досягається у двох випадках:

– якщо $\underline{a} \le \underline{b}$, то при $a = 0$, $b = \underline{b}$;

– якщо $\underline{b} \le \underline{a}$, то при $a = \underline{a}$, $b = 0$.

Наведена модель буде використана для побудови та подальшого дослідження методів паралельної обробки великих обсягів неструктурованої інформації шляхом розпаралелювання та оптимізації кількості запитів до сховищ за допомогою персональних мобільних пристроїв під управлінням операційної системи iOS.

Список використаних джерел:

1. A. Khovrat, V. Kobziev, A. Nazarov and S. Yakovlev. Parallelization of the VAR Algorithm Family to Increase the Efficiency of Forecasting Market Indicators During Social Disaster. / Information Technology and Implementation (IT&I-2022), November 30 – December 02, 2022, Kyiv, Ukraine. – 12 pp. CEUR Workshop Proceedings, 2022, 3347, pp. 222–233.

2. N. Shakhovska, S. Fedushko, M. Greguš, N. Melnykova, I. Shvorob and Y. Syerov: Big data analysis in development of personalized medical system. Procedia Comput Sci. 160:229–234. 2019.

УДК 004.62

Полурєзов Д.С.

ОБРОБКА BIG DATA НА МОБІЛЬНИХ ПРИСТРОЯХ

На сьогодні в сфері розробки мобільних додатків спостерігається зростаючий інтерес до проблем обробки великих об'ємів даних за допомогою мобільних пристроїв. Вимоги користувачів до сучасних мобільних застосунків містять потребу в отриманні доступу до важливої інформації у будь-який момент часу та в будь-якому місці. Це означає, що сучасні застосунки повинні бути орієнтовані на обробку великих об'ємів даних та забезпечувати швидку і ефективну реакцію на запити користувачів. У більшості випадках користувачі очікують, що дані будуть оновлюватися в режимі реального часу, надаючи їм актуальну інформацію.

Однією з основних стратегій для покращення продуктивності та забезпечення швидкої реакції на запити користувачів є використання методів паралельної обробки даних. Це означає, що додаток може виконувати кілька операцій одночасно, розділяючи завдання на менші частини і оброблюючи їх паралельно на різних обчислювальних ядрах пристрою. Паралельна обробка дозволяє зменшити час обчислень і підвищити загальну продуктивність, що є важливим фактором для багатьох мобільних застосунків.

Поширення методів паралельної обробки великих об'ємів даних на платформі iOS серед користувачів персональних мобільних пристроїв має безпосереднє значення для різних сфер, включаючи медицину, військову справу, правоохоронну діяльність. Ці методи базуються на підвищенні продуктивності iOS-додатків за рахунок впровадження методів розпаралелювання, заснованих на GCD, NSLock та OperationQueue для формування та обробки запитів до розподілених БД.

Основним елементом будь-якої БД слід вважати ключ. Носієм даних в моделі NoSQL БД є кортеж:

$$KV = \{\langle f, e\rangle\},$$

де ***f*** – ключ, який приймає унікальне значення для кожної пари; ***e*** – значення, яке йому відповідає.

Сигнатура моделі має наступний вигляд:

$$O = \langle \pi, \sigma \rangle,$$

де $\boldsymbol{\pi}$ – операція проекції за атрибутами (ключ або значення), $\boldsymbol{\sigma}$ – селекції атрибутів (вибір значення за ключем, ключів за значеннями, наслідування ключів). Ці операції відносяться до категорії читання [1].

Одним з варіантів реалізації розподіленого зберігання великих обсягів даних є система BigTable від компанії Google [2], яка має наступні властивості:

- неповна реляційна модель даних;
- підтримка динамічного контролю над розміщенням даних.

Основу моделі збереження даних в BigTable складають рядки, стовпці та тимчасові мітки:

$$\text{BigTable} = \{< r, c, t >\}.$$

Якщо в декількох стовпцях зберігаються дані, що відносяться до одного типу, то такі стовпці, згідно з моделлю Bigtable, утворюють сімейство:

$$\text{col } F = \{ c_i, c_j \mid \text{dom}(c_i) \in T \wedge \text{dom}(c_j) \in T\}.$$

Використовувати сімейство стовпців досить зручно хоча б з того міркування, що це дозволяє стиснути однорідні дані, тим самим зменшивши їх обсяг. Саме сімейства стовпців складає одиницю доступу до даних.

Вміст інформації, що супроводжує будь-який динамічний об'єкт, що перебуває під наглядом, постійно змінюється. Щоб врахувати ці зміни, кожна з копій даних, які зберігаються в стовпці, отримують тимчасову мітка (timestamp). В BigTable в якості тимча-

сової мітки використовується 64-розрядне число, яким можна кодувати час і дату таким чином, як це потрібно клієнтським програмам. Шляхом використання тимчасові мітки додатки можуть забезпечувати в BigTable пошук, наприклад, тільки щодо найновіших копій даних [3].

Отже, для будь-якої предметної області в сервісі Google можна створити власну карту даних Bigtable, що містить задану кількість рядків і унікальний для цієї предметної області набір сімейств стовпців. Повтори даних у стовпцях для таких даних упорядковуються за значеннями тимчасових міток. Головною перевагою цього підходу є те, що таку базу неважко поділити на незалежні елементи та розподілити по множині серверів. Відсортовані за алфавітом рядки діляться на діапазони, що мають назву tablet. Оскільки рядки в кожному таблеті відсортовані за ключовим іменем, то клієнтським додаткам достатньо просто знайти потрібний таблет, а в ньому – необхідний рядок.

Для синхронізації таблетів призначений сервіс Chubby. Для кожного таблет-сервера Chubby створює спеціальний chubby-файл, за рахунок чого файл Bigtable може визначити, які із серверів є працездатними. Ще один chubby-файл містить посилання на розташування кореневого таблета (Root-tablet) з даними про розташування усіх інших. Цей файл повідомляє майстру, який з серверів якими таблетами керує.

Використання сервісу Chubby в середовищі Bigtable дозволяє забезпечити підтримку несуперечності даних у розподіленому середовищі з безліччю реплік. Подальші дослідження будуть спрямовані на визначення метрик для порівняння методів розпаралелювання та проведення експериментів з метою визначення найбільш продуктивного метода формування та обробки запитів до БД за допомогою мобільного додатку у середовищі iOS.

Список використаних джерел

1. Zhou Feng, W. Hsu, Mong Li Lee. Efficient pattern discovery for semistructured data // 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI-05). – 2005. – P. 301–309.

2. Cloud Bigtable (назва з екрану) / URL: https://cloud.google.com/bigtable (дата звернення: 16.02.2024).

3. Chang, Fay; Dean, Jeffrey; Ghemawat, Sanjay; Hsieh, Wilson C; Wallach, Deborah A; Burrows, Michael ‘Mike’; Chandra, Tushar; Fikes, Andrew; Gruber, Robert E (2006), "Bigtable: A Distributed Storage System for Structured Data", Research (PDF), Google.

УДК 007.2+ 004.942 + 004.05 +004.056.5

Пономарьов О.А., Нестеров О.М., Козубцов І.М., Ольшанський В.В, Філіпов В.В.

ПІДГОТОВКА ФАХІВЦІВ ЗВ’ЯЗКУ ТА КІБЕРБЕЗПЕКИ СЕКТОРУ БЕЗПЕКИ ТА ОБОРОНИ НА ЗАСАДАХ ЛІДЕРСЬКОЇ ПРОФЕСІЙНО-ДІЛОВОЇ ГРИ

В даний час триває пошук рішень щодо підвищення мотивації засвоєння навчального матеріалу курсантами вищих військових навчальних закладах (ВВНЗ). Впровадження інтерактивних форм навчання є одним з найважливіших напрямків вдосконалення навчального процесу у ВВНЗ. Першою інтерактивною формою навчання був навчальний комплекс системи бойової підготовки військових спеціалістів танкових військ. Для цього І. Руснак та В. Шевченко використали сучасні комп'ютерні технології, які дозволяють взаємодіяти з безліччю речей, в яких курсант може взяти певну участь. Відновлення та реалізація наукових досліджень обумовлені зміною підходу в системі бойової підготовки військових фахівців підрозділів зв'язку та кібербезпеки.

ДОДАТОК Д

Експертний висновок результатів перевірки кваліфікаційної роботи на відповідність оформлення вимогам ДСТУ 3008: 2015

Експертний висновок результатів перевірки кваліфікаційної роботи

студент (посада) | програмної інженерії (кафедра) | ІПЗм-22-3 (група)

Полурєзов Д.С.
(прізвище, ім'я, по батькові)

Зауваження

Пункт ДСТУ 3008-2015	Зміст пункту	Сторінка кваліфікаційної роботи
1	2	3
	7.1 Загальні положення	
	7.3 Нумерація сторінок звіту	
	7.4 Нумерація розділів, підрозділів, пунктів, підпунктів	
	7.5 Рисунки	
	7.6 Таблиці	
	7.7 Переліки	
	7.8 Примітки	
	7.9 Виноски	
	7.10 Формули та рівняння	
	7.11 Посилання	
	7.13 Список авторів	
	7.14 Скорочення та умовні позначки	
	7.15 Додатки	

Експерт _____ (підпис) Вадим НЕЧВОЛОД (прізвище, ініціали)

09.06.2024