Хмельницький національний університет
Факультет програмування та
комп'ютерних і телекомунікаційних систем
Кафедра комп'ютерної інженерії та системного програмування

ДИПЛОМНА РОБОТА МАГІСТРА

Галузь знань _____12 – Інформаційні технології_____

Спеціальність _____123 –Комп'ютерна інженерія _____

на тему «Розподілена динамічна система кластеризації для видобутку просторових даних.»

ДРКІСПр. 013042.17.01.01 ПЗ

Виконав: студент 2 курсу, група КІ2м-19-1 _____ Товстуха Е.В.
Підпис Ініціали, прізвище

Керівник кандидат техн. наук _____ Бобровнікова К.Ю.
Науковий ступінь, вчене звання Підпис Ініціали, прізвище

До захисту допускаю:
Зав. кафедри КІСП, д.т.н., проф.
Т.О. Говорущенко
25 05 2021 р.

Хмельницький, 2021

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ПРОГРАМУВАННЯ ТА КОМП'ЮТЕРНИХ І ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМ

Кафедра КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА СИСТЕМНОГО ПРОГРАМУВАННЯ

Освітній рівень МАГІСТР

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 123 КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

Освітня програма ОСВІТНЬО-НАУКОВА ПРОГРАМА «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ»

ЗАТВЕРДЖУЮ

Зав. кафедри Т.О.Говорущенко

" 07 " 09 2020 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ)

Товстусі Едуарду Волдимировичу
Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Розподілена динамічна система кластеризації для видобутку просторових даних

Керівник проекту (роботи) Бобровнікова К.Ю., к.т.н.
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 15.01.2021 р. № 7

2. Строк подання студентом проекту (роботи) на кафедру 25.05.2021 р.

3. Вихідні дані до проекту (роботи) Завдання на дипломне проектування

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз відомих методів кластеризації для видобутку просторових даних

Модель процесу кластеризації для видобутку просторових даних

Метод кластеризації для видобутку просторових даних

Впровадження та оцінка ефективності розподіленої динамічної системи для видобутку просторових даних

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

6. Консультанти розділів дипломного проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Лисенко С.М., професор кафедри КІСП		
Антиплагіат	Нічепорук А.О., доцент кафедри КІСП		

7. Дата видачі завдання «_07_»_____09_____2020 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1	Вибір напряму дослідження та узгодження тематики ДРМ з керівником	07.09.2020	виконано
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	05.10.2020	виконано
3	Робота над розділом 1 – аналіз відомих моделей, методів за темою; постановка задачі	09.11.2020	виконано
4	Робота над розділом 2 – розробка моделей для вирішення поставленої задачі	05.01.2021	виконано
5	Робота над науковою статтею	25.01.2021	виконано
6	Робота над розділом 3 – розробка методів для вирішення поставленої задачі	15.02.2021	виконано
7	Робота над розділом 4 – проектування та розробка ПЗ для вирішення поставленої задачі, експериментальна частина	05.04.2021	виконано
8	Оформлення пояснювальної записки згідно вимог	15.04.2021	виконано
9	Попередній захист ДРМ	05.05.2021	виконано
10	Захист ДРМ на засіданні ЕК	Травень 2021	

Студент _____ __Товстуха Е.В._____
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ __Бобровнікова К.Ю.__
Підпис Ініціали, прізвище

РЕФЕРАТ

Тема дипломної роботи: Розподілена динамічна система кластеризації для видобутку просторових даних.

Автор роботи: Товстуха Е. В.

Керівник роботи: к.т.н. Бобровнікова К. Ю.

Пояснювальна записка: 79 с., 27 рис., 9 табл., 3 дод., 87 джерел.

ПЕРЕЛІК КЛЮЧОВИХ СЛІВ: розподілена система, динамічна кластеризація, кластер, просторові дані, видобуток просторових даних, стиснення даних.

Об’єктом дослідження є процес розподіленої динамічної кластеризації для видобутку просторових даних

Предметом дослідження є модель, метод розподіленої динамічної кластеризації для видобутку просторових даних та розподілена динамічна система кластеризації для видобутку просторових даних.

Метою дипломної роботи є підвищення ефективності розподіленої динамічної кластеризації для видобутку просторових даних шляхом розроблення розподіленої динамічної системи кластеризації для видобутку просторових даних.

Для розв’язання поставлених задач використовувалися методи теорії множин, теорії комп’ютерних мереж, теорії кластерного аналізу.

Наукова новизна отриманих результатів:

- удосконалено модель процесу розподіленої динамічної кластеризації для видобутку просторових даних, яка, на відміну від відомих моделей, використовує алгоритми стиснення та агрегації даних;
- удосконалено метод розподіленої динамічної кластеризації для видобутку просторових даних, який, на відміну від відомих методів, ґрунтується на побудованій моделі розподіленої динамічної кластеризації для видобутку просторових даних, застосовує методи стиснення даних та динамічної кластеризації та є основою розподіленої динамічної системи кластеризації для видобутку просторових даних. Застосування розробленого методу

дозволить підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами.

На основі проведених досліджень розроблено розподілену динамічну систему кластеризації для видобутку просторових даних.

Практична цінність отриманих результатів полягає у дипломної роботи полягає в розробленні розподіленої динамічної системи кластеризації для видобутку просторових даних, яка надає можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими системами.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ANN – artificial neural networks

CART - Classification and Regression Tree

SAR - Spatial Autoregressive Model

SVM – support vector machine

ПЗ - програмне забезпечення

РК – розподілена кластеризація

ВСТУП

Технології для збору просторових даних є досить складними; починаючи від супутникових зображень, давачів погоди, дистанційне зондування з високою роздільною здатністю до GPS, мобільні пристрої та опитування населення.

Просторові дані є основною частиною світових даних (близько 80% дані просторові). Для цього типу великих даних потрібні нові методи data mining та засоби візуалізації, щоб зрозуміти явища, що керують ними. Оскільки дані, які можна зібрати, дуже різноманітні, в рамках магістерської роботи зосередимось на одному типі даних, який є просторовими даними.

Актуальність роботи полягає в розробленні розподіленої динамічної системи кластеризації для видобутку просторових даних, що використовуватиме запропонований розподілений метод кластеризації на основі методів видобутку даних.

Метою дипломної роботи є підвищення ефективності розподіленої динамічної кластеризації для видобутку просторових даних шляхом розроблення розподіленої динамічної системи кластеризації для видобутку просторових даних.

Поставлена мета досягається розв’язанням таких основних задач:

– необхідно розробити метод розподіленої динамічної кластеризації для видобутку просторових даних, який ґрунтуватиметься на розробленій моделі процесу розподіленої динамічної кластеризації для видобутку просторових даних;

– метод повинен застосовувати методи стиснення даних та використовувати динамічну розподілену кластеризацію з метою підвищення ефективності кластеризації для видобутку просторових даних;

– на основі розроленого методу створити розподілену динамічну систему кластеризації для видобутку просторових даних, яка надасть можливість підвищити ефективність кластеризації для видобутку просторових даних.

Об’єктом дослідження є процес розподіленої динамічної кластеризації для видобутку просторових даних.

Предметом дослідження є модель, метод розподіленої динамічної кластеризації для видобутку просторових даних та розподілена динамічна система кластеризації для видобутку просторових даних.

Наукова новизна отриманих результатів:

1. Удосконалено модель процесу розподіленої динамічної кластеризації для видобутку просторових даних, яка, на відміну від відомих моделей, використовує алгоритми стиснення та агрегації даних.

2. Удосконалено метод розподіленої динамічної кластеризації для видобутку просторових даних, який, на відміну від відомих методів, ґрунтується на побудованій моделі розподіленої динамічної кластеризації для видобутку просторових даних, застосовує методи стиснення даних та динамічної кластеризації та є основою розподіленої динамічної системи кластеризації для видобутку просторових даних. Застосування розробленого методу дозволить підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами.

Практична цінність отриманих результатів. В результаті виконаного наукового дослідження розроблена розподілена динамічна система кластеризації для видобутку просторових даних, яка надає можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими системами.

У даній роботі викладено вимоги до методології побудови розподіленої динамічної системи кластеризації для видобутку просторових даних.

Для розв’язання поставлених задач використовуються основні положення теорії множин, теорії комп’ютерних мереж, теорії кластерного аналізу.

Розподілений метод надасть перевагу у розподілі ресурсів, що дозволить структурувати дані та пришвидшити вирішення складних задач. Накладні витрати для зв’язку між окремими компонентами розподіленої системи можуть бути дуже великою проблемою, якщо вони не контролюються. Тому необхідно зменшити такі комунікаційні витрати. Для цього буде використано розподілену обчислювальну інфраструктуру (як апаратну, так і програмну) для впровадження

запропонованої системи та вивчення її поведінки. З цією метою буде використано як хмарні обчислення, так і MapReduce. Хмарні обчислення будуть використовуватися в якості розподіленої платформи для запуску розподіленої системи. MapReduce буде використовуватись в якості програмної платформи для її реалізації в хмарі.

За темою дипломної роботи опублікована одна стаття у фаховому науковому виданні [1].

1 АНАЛІЗ ВІДОМИХ МЕТОДІВ КЛАСТЕРИЗАЦІЇ ДЛЯ ВИДОБУТКУ ПРОСТОРОВИХ ДАНИХ

1.1 Огляд відомих методів кластеризації для видобутку просторових даних

З розповсюдженням центрів обробки даних у всьому світі набутий роками потенціал хмарних сховищ величезний. Однією з важливих сфер застосування просторових даних є системи забезпечення енергоефективності та енергозбереження в таких кіберфізичних системах як розумний будинок. Огляд літератури показав, що проблема обробки та кластеризації просторових даних в таких системах є надзвичайно актуальною [1]. Більше того, хмарні обчислення мають можливості високопродуктивних обчислень[1-5]: висока продуктивність мережі, швидке зберігання, великий обсяг пам'яті, дуже високі обчислювальні можливості. Тому однією із ключових цілей нашої системи обробки даних є використання можливостей хмарних інфраструктур. Однак інструменти видобутку та аналізу все ще в своєму користуванні на початку зародження. Такі інструменти набирають популярності, і кожна велика компанія шукає інші засоби аналізу даних для вивчення їхніх історичних даних, поглядів на посилення процесів прийняття рішень та розуміння їхнього бізнесу. Видобуток даних визначається як “процес виявлення значущих нових кореляційних закономірностей та тенденції шляхом відсіювання великих обсягів даних, що зберігаються у репозиціях використання технологій розпізнавання образів [6-9], а також статистичної та математичної методу [10] . Видобуток даних, який також називають виявленням знань у базах даних(KDD) процес, як і раніше є одним з найпотужніших та найефективніших методів вилучення знань, прийняття рішень, прогнозування, розпізнавання зразків тощо. З його визначення, видобуток даних призначений для роботи на великих наборах даних. Однак,виклики великих даних вивели процес видобутку за його межі,що вже є проблемою.

Наприклад, якщо є необхідність видобутку даних з вебу за допомогою вишевказаного методу, то він неможливий для з наступних причин:

1. Веб-дані дуже динамічні і змінюються надзвичайно швидкими темпами, тому видобуток цим способом робити не варто в більшості випадків.

2. В більшості алгоритмів видобутку даних припустимо, що дані безшумні. Очевидно, це дуже вагоме припущенняа,так як реальні дані далеко не вільні від шуму, включаючи веб-дані звичайно. Це значно порушує точність результатів.

3. Інтернет дані дуже багатовимірні [11], і простір пошуку зазвичай зростає експоненційно з кількістю розмірностей. Це дуже погано впливає на продуктивність багатьох алгоритмів майнінгу.

4. Інтелектуальний аналіз даних, частіше, використовує машинне навчання та статистичні методи для аналізу даних та / або даних фази попередньої обробки. Однак більшість із цих методів не є розробленими для дуже великих наборів даних, як у Big Data [12-16]. Їх складність або експоненціальний або середній порядок многочлена. Норма для аналізу великих даних алгоритмів - це лінійна складність. Вибірка, кластеризація та розділення – це інші поширені методи, які використовуються для зменшення розміру набору даних, який буде видобуто. вони створюють інші проблеми, такі як повнота

За останні два десятиліття просторові масиви даних набули популярності завдяки значному прогрессу в техніці комп'ютерного обладнання та величезних сумм географічних даних, які були зібрані за допомогою сучасного збору данних [17-21],наприклад таких методів, як глобальні системи позиціонування (GPS), веб-просторові дані спільного використання та картографування, дистанційне зондування з високою роздільною здатністю. Інші найпоширеніші джерела даних - супутникові знімки, медична візуалізація, медичні зображення, структура білка людського тіла та всі ті дані, які можуть бути представленими у вигляді кубоїдів, багатокутника, циліндра тощо [22 – 24].

Термін "просторовий" означає всі ті набори даних, які пов'язані з простором або географічними регіонами. Просторові дані - це дані, що стосуються об’єктів, які займають простір. Просторова база даних зберігає просторові об’єкти, представлені просторовим типом даних та просторові взаємозв'язки серед таких об'єктів [23,24].

Аналіз просторових даних має глибоке коріння в традиційних полях просторового аналізу,таких як статистичний просторовий аналіз, картографія, обчислювальна геометрія. Статистичний просторовий аналіз був найпоширенішим підходом до аналізу просторових даних протягом тривалого періоду [25,26]. Однак статистичний просторовий аналіз в якості недоліка має припущення про те, що серед просторово розподілених даних існує статистична незалежність [27-30]. Це спричиняє проблеми, оскільки просторові дані насправді є взаємопов'язаними об'єктами.

Існують розроблені моделі регресії для певної міри вирішення цієї проблеми. Однак це рішення є складним, і ним може скористатись лише фахівець з достатньою кількістю знань у цій галузі та статистичних знаннях. Крім того, статистичний просторовий метод не може добре обробляти нелінійні дані, і вони це роблять погано, працюючи з неповними даними [31-35]. Статистичний просторовий аналіз[36] має високу обчислювальну складність і має великі затрати з боку ресурсів, які застосовуються в реальних задачах. З іншого боку, з появою аналізу даних дослідники винаходять різні методи виявлення знань із наборів даних. Багато повторно проведено пошукових робіт щодо досліджень аналізу даних у відповідних баз даних та транзакцій, але мало досліджень проводиться в інших типах наборів даних, таких як просторові набори даних.

На сьогоднішній день для боротьби з викликом великих даних було використано дуже мало підходів. Перший метод - використання паралелізму для спільного використання обчислень серед великої кількості процесорів [37]. Усі популярні алгоритми видобутку даних були реалізовані паралельно, і численні паралельні версії кожного алгоритму можна знайти у загальнодоступних доменах. Ці версії були розроблені для специфічних паралельних архітектур або парадигми паралельного програмування. Хоча розпаралелювання алгоритмів видобутку корисне для пришвидшення обчислень,вони припускають, що вхідні дані були попередньо оброблені, без шуму та відповідно недоступні для процесорів, що не є випадковим у більшості програм. Більше того, ці версії не стосуються

неоднорідності даних, і вони мають залежності даних, які вимагають зв’язку та синхронізації механізмів, отже роблять їх неефективними.

Другий метод – використання розподілених систем для спільного використання обчислень завдань серед великої кількості вузлів автономної обробки. Цей тип підходів може мати справу з неоднорідними даними, але вони слабо пов'язані і мають значні накладні витрати щоб координувати всі вузли обробки та загальні результати видобутку. Знову ж таки,можна знайти багато розподілених версій популярних підходів до видобутку даних в літературі. Хоча ці підходи в основному стосуються правил асоціацій алгоритмів видобутку та класифікації, існує лише кілька розподілених версій алгоритмів кластеризації для великих наборів даних.

Варто зазначити, що у більшості додатків великі данні збираються з використанням різних інструментів вводу та/або датчиків зберігання їх у різних місцях з причиною близькості або ємність для зберігання. Під близькістю мається на увазі, що дані зберігаються в межах місця, де він був зібраний та / або виготовлений. По правильному потрібно зберігати ці дані в різних місцях, де є достатньо місця для зберігання, а також для тиражування. Ще одне зауваження щодо цих додатків: що дані, зібрані та / або отримані в різних місцях, також відрізняються якістю. Різні прилади та давачі частіше виробляють або збирати різні дані. У деяких випадках дані одного місця можуть мати різну атрибути, ніж дані іншого місця. Така проблема неоднорідністі даних в даний час вирішується шляхом створення складних мета-словників та складних інструментів інтеграції, які не завжди необхідні [38-41]. Тому основна ідея запропонованої розподіленої системи кластеризації полягає в тому, щоб видобувати дані локально в кожному місці, а потім добувати знання про ці місця, щоб генерувати нові знання, від яких може отримати користь кожен.

Локальний видобуток даних має багато переваг:

1. Обсяг даних не є проблемою, оскільки кожне місце оброблятиме свою частку даних. Це значно зменшує розмір проблеми.

2. Неоднорідність даних (або різноманітність великих даних) також не є проблемою, оскільки неоднорідність буде розглядатися локально в кожному місці для підготовки даних до місцевого видобутку.

3. Місцеві дані можна видобувати за допомогою алгоритмів та процесів, які є відповідними до місцевих характеристик даних та місцевих потреб.

4. Місцеві видобуті результати можуть бути використані і місцевими користувачами.

5. Локальний аналіз даних дозволить краще зберегти їх конфіденційність, оскільки деякі дані можуть бути дуже чутливими та вразливими, якщо їх потрібно перемістити до центрального місця.

1.2 Методи контрольованого спуску (SDM)

Майнінг правил асоціації спочатку мав на меті виявити закономірності, пов'язані між елементами у великих базах даних транзакцій [42]. Правила асоціації складаються з пошуку частих закономірностей, асоціацій, кореляцій або причинних структур серед наборів елементів або об'єктів у транзакційних або реляційних базах даних, які можуть використовуватись в основному для аналізу ринкових кошиків, аналізу поведінки споживачів та маркетингових кампаній [43].

Введемо поняття частого видобутку наборів предметів. Нехай $I = \{i_1, i_2, ..., i_m\}$- набір предметів (тобто предметів, придбаних в операціях,наприклад, комп’ютер, молоко, велосипед тощо). Нехай D - набір транзакцій, де кожна транзакція T - це сукупність елементів, така що $T \subseteq I$. Нехай X - це сукупність предметів і, які кажуть, що транзакція T містить X тоді і тільки тоді, коли $X \subseteq T$. Правило асоціації у вигляді: $X \Rightarrow Y$, де $X \subseteq I$; $Y \subseteq I$ і $X \cap Y = 3$.

Правило $X \Rightarrow Y$ утримується в наборі транзакцій D із впевненістю в підтримці C. Опора визначається як частина транзакцій у D, що містять $X \cup Y$ та частка транзакцій у D, яка, якщо вона містить X, то вони також містять Y. Впевненість правила вимірює силу, а підтримка вказує на частоту правила. Часто бажано звернути увагу на ті правила, які мають досить велику підтримку .Існує

багато алгоритмів правил асоціацій. Найпопулярнішими є алгоритми Apriori та FP-Growth. Алгоритм Apriori [44-47] знаходить часті набори предметів відповідно до визначеного користувачем набору. Під час першого проходу алгоритму він будує набори з 1 елемента. Потім він генерує часті набори з 1 елемента, обрізаючи деякі дати 1-позицій, якщо їх значення підтримки нижчі за мінімальну підтримку.

Після того, як алгоритм знаходить усі часті набори з 1 елемента, він поєднує в собі 1-набір предметів один для одного, щоб побудувати кандидатів в набір 2-елементів та обрізати деякі рідкісні набори предметів з кандидатів в набори із 2 елементів для створення частого набори з 2 елементів. Цей процес повторюється, доки більше не може бути кандидатів в створенні набори [48].

Алгоритм Apriori має високу обчислювальну складність, оскільки причина цього породжує дуже велику кількість кандидатів. Кілька уточнень запропоновано алгоритмом Apriori, щоб пришвидшити процес частого видобутку наборів.

FP-зростання був одним з алгоритмів, з якими було запропоновано боротися з обмеженнями алгоритму Apriori. Алгоритм росту FP виявляє відмови від наборів предметів, не створивши всіх можливих кандидатів. Це генерує ті самі часті набори предметів без складності Apriori [49]. Алгоритм стискає великий набір транзакцій у компактний, часто в структуру дерева зразків (FP-дерево), яка сильно ущільнена, але повна для частих видобутків шаблонів. Більше того, алгоритм дозволяє уникнути дорогого сканування набору транзакцій. Він використовує методологію "розділяй і владарюй", щоб розділити завдання на менші [49]. Подібно до видобутку правил асоціації в транзакційних або реляційних баз даних, Koperski та Han поширили цю концепцію на просторові масиви даних. Правило просторової асоціації має вигляд $X \Rightarrow Y(c)$, де X і Y – множини просторових або непросторових атрибутів, а c - впевненість у правилі.

Приклад правила просторової асоціації: $\epsilon(x, школа) \Rightarrow близько\ до(x, парк)(80\%)$. Це правило говорить, що 80% шкіл знаходяться поруч із парками. Є різні види просторових атрибутів, які можуть складати правило просторової асоціації. Деякі

правила просторової асоціації можна видобувати у просторових наборах даних, враховуючи просторові властивості та атрибути[50-51]. Були проведені різні дослідницькі роботи , застосовуючи правила асоціації до видобутку просторових наборів даних.

Більшість з цих робіт були зосереджені більше на пропонуванні ефективних підходів, спрямованих на зменшення часу генерації набору кандидатів, оскільки проблема стає ще більш критична важливий при роботі з дуже великими просторовими наборами даних. Інші роботи також зосереджувались на застосуванні правил асоціації в різних мережах, які включають просторові дані. Наприклад, Appice та ін. звернувся до проблема георефернції даних перепису. Зокрема, вони розслідували розміщення просторового виміру в переписах для виявлення просторових правил асоціації або спеціальних зразків класифікацій, також відомих як контрольоване навчання, - це процес групування елементів даних у класи (категорії), які вже були попередньо визначені.

Шахрайскі додатки для тестування та кредитного ризику особливо добре підходять для цього типу аналізу. Найпопулярнішими методами класифікації є, наприклад, дерева цілей (найпопулярнішими алгоритмами є C4.5, ID3 та CART), штучні нейронні мережі (ANN), метод опорних векторів (SVM), байєсівські мережі праці, генетичні алгоритми тощо. Різниця між класифікацією та передбачення пов’язане з типом оброблених значень. Тоді як класифікація катіон передбачає категоріальні (дискретні, невпорядковані) мітки, моделі прогнозування функції безперервного значення або впорядкована вартість.

Просторові методи класифікації поширюють загальне призначення класифікацій методів розгляду не тільки атрибутів об’єктів, що підлягають класифікації, але й атрибутів сусідніх об’єктів та їх просторові відношення. В [52] запропонував підхід, заснований на алгоритмі ID3, і він використовує концепцію сусідніх графіків. Він приймає як просторові, так і непросторові властивості класифікованих об'єктів. Об'єкти вважаються сусідами, якщо вони задовольняють одному з наступного: топологічні відношення (перетинаються), метричні відношення (близькі до) прямих відносин (південь, захід тощо).

В роботі [53] запровадили візуальний метод до просторової класифікації, де поєднується традиційний алгоритм прийняття рішень з C4.5 [54] з візуалізацією карти для виявлення просторових закономірностей правил класифікації.

Алгоритм C4.5 був використаний в [55] для аналізу і прогнозувати поведінки просторового вибору. Також використовувалися методи дерева рішень Файядом, а саме класифікувати зображення зоряних об'єктів для виявлення зірок і галактик. Об'єкти в навчальному наборі даних були класифіковані астрономами, де було побудовано близько десяти навчальних наборів для алгоритму дерева рішень. Запропонована система застосовує дерева рішень (класифікатор), мінімальний набір надійних даних і множину правил. Запропонований метод стосується наборів даних зображень,призначений лише для астрофізики.

В роботі [56] використовував методи класифікації для класифікації пікселів зображення позначаючи категорії. Автори використовували класифікацію на основі пікселів та об'єктів. Ці методи для картографування районів лісових пожеж в Каліфорнії. На основі об’єкта класифікація складається з сегментування зображення на кластери подібних сусідніх пікселів, що називаються "об'єктами". Автори зупинились на малозаселених територіях. Це міські райони, де все більша кількість будинків будується в умовах пожежонебезпечних районів. Методика класифікації забезпечує переробку цифрового зображення, яка є життєздатним методом отримання землекористування та земельного покриву карти (LULC), які можуть допомогти запобігти пожежі. Цей метод є відносно економічним варіантом, порівняно з міськими зображеннями з високою просторовою роздільною здатністю. Моделі просторового прогнозування утворюють особливу групу регресійного аналізу, який надає інформацію про просторові взаємозв'язки між об'єктами даних. Просторова авторегресивна модель (SAR) є популярним прикладом просторово-регресивного методу. Його можна застосувати до будь-яких наборів даних, що містять спостереження на географічних районах або на будь-яких одиницях із просторовим поданням для прогнозування залежності між об'єктами даних.

Наприклад, в [57] використовували SAR,щоб знайти просторово-часову авторегресію між об'єктами даних для оцінки ціни продажу нерухомості з використанням даних про житло з округу Ферфакс, штат Вірджинія. Це призвело до значного поліпшення оцінки ціни на властивості в межах даної області.

Однак для того, щоб знайти залежності між точками даних, SAR потрібно перевірити всі точки даних. Ця операція передбачає маніпуляцію з $(n \times n)$ просторової вагової матриці. Складність обчислень зростає через необхідність обчислення логарифму визначника великої матриці, яка обчислюється шляхом знаходження всіх власних значень $\overline{W}$ матриці $((n \times n)$, що є симетричним еквівалентом матриці W через власні значення).

Останні дослідження спрямовані на розроблення підходів до вдосконалення коефіцієнтів питомого поглинання, щоб він міг обробляти дуже великі набори даних [57–58]. Оскільки модель SAR дає точне рішення, деякі з них дослідження виконували приблизні рішення, звертаючи увагу на неточності моделі при обробці дуже великих наборів даних, в [57], наприклад, автори запропонували використовувати два різних наближених рішення, розкладання рядів Тейлора та поліноми Чебишева для обчислення логарифму визначника. Помічено, що приблизні методи не тільки споживають дуже мало пам'яті, але вони також виконуються дуже швидко,забезпечуючи при цьому дуже точні результати [59].

1.3 Парадигма програмування MapReduce

Apache Hadoop був представлений і став найпопулярнішою моделлю паралельної та розподіленої обробки великих даних. MapReduce - це серце Apache Hadoop. Це парадигма програмування, яка дозволяє отримати масштабованість на сотні або тисячі серверів в кластері Hadoop. Це фреймворк програмування програм, який здатен обробляти великомасштабні набори даних, використовуючи паралелізм між кластерами обчислювальних вузлів. MapReduce набув популярності своєю простотою, гнучкістю, відмовостійкістю та масштабованістю незабаром після його створення. Для покращення та прискорення роботи на

Hadoop MapReduce було впроваджено багато алгоритмів інтелектуального аналізу даних. Більше того, багато дослідників почали пропонувати рішення на основі парадигми MapReduce та моделей хмарних обчислень [60-62]. Кластеризація є одним із методів видобутку даних, який прийняв MapReduce, отже, багато дослідників використовують MapReduce для кластеризації великих даних [63–68], як правило, це втілення існуючих алгоритмів кластеризації для даного додатку за допомогою Hadoop MapReduce .

Hadoop - це фреймворк з відкритим кодом, призначений для розподіленої обробки великих наборів даних у великій та динамічній мережі комп'ютерів. Цей фреймворк може обробляти дані відповідно до великого обсягу, швидкості та різноманітності. Це також сприяє розробці розподілених додатків. Hadoop обробляє апаратні збої на рівні програми. Основними особливостями Hadoop є:

1. Масштабованість: програма, яка працює на одній машині, також може працювати на 1000 машинах. Якщо потрібна більша потужність, то просто додаємо більше машин.

2. Ключовою перевагою використання Hadoop є його відмовостійкість. Коли дані надсилаються на окремий вузол, ці ж дані також реплікуються на інші вузли кластера, а це означає, що у випадку відмови існує інша копія, доступна для використання.

3. Унікальний метод зберігання Hadoop заснований на розподіленій файловій системі, яка в основному "відображає" дані скрізь, де вони розташовані в кластері. Інструменти для обробки даних часто знаходяться на тих самих серверах, де розташовані дані, що призводить до набагато швидшої обробки даних. У разі роботи з великими обсягами неструктурованих даних, Hadoop здатний ефективно обробляти терабайти даних за лічені хвилини, а петабайти за кілька годин.

4. Hadoop надає багато простих API і є дуже потужним. Він може мати справу з величезними даними (петабайти).

Hadoop складається з двох основних компонентів: MapReduce та HDFS. Перший - це частина обробки, а другий - частина даних. Кілька машин з Hadoop

створюють кластер. Скупчення може складатися з тисяч машин. Замість того, щоб обробляти дані послідовно, Hadoop розбиває файли на блоки, які можна обробляти одночасно.

HDFS (розподілена файлова система Hadoop) - це файлова система, яка працює поверх існуючої файлової системи кожного вузла. Ця файлова система найкраще працює з великими файлами і використовує поняття "блоки" для зберігання файлу. Ці блоки мають фіксований розмір (за замовчуванням 64 МБ). Поняття блоків має кілька переваг. Фіксований розмір полегшує обчислення кількості блоків, які можуть бути включені на диск. Блоки дозволяють зберігати файли, більші за ті, що можуть зберігатися на одному конкретному вузлі. Блоки також копіюються на декількох вузлах, щоб забезпечити доступ до даних. Є два конкретних вузли HDFS: NameNode і DataNode. Кластер має лише один NameNode, він відповідає за метадані файлів і за простір імен файлової системи. Це повинен бути найпотужніший вузол кластера, і він повинен мати якомога більше оперативної пам'яті, оскільки він зберігає в пам'яті всі метадані файлової системи. Якщо NameNode втрачено, то втрачаються також усі дані кластера. Щоб запобігти цьому, можна визначити реплікацію NameNode, яка називається "StandByNameNode". У кластері є багато вузлів, які називаються "Вузли даних". Ці вузли зберігають блоки даних, і коли клієнт хоче отримати деяку інформацію, йому потрібно поставити під сумнів NameNode, щоб знати, який DataNode зберігає дані. Періодично DataNodes надсилають інформацію про блоки, які вони зберігають. Вузли даних - це найнижчий шар кластера. Коли в систему надходить новий файл, на NameNode надсилається "запит на створення".

MapReduce - це парадигма програмування для додатків з інтенсивним використанням даних, це основа для легкої розробки додатків, які паралельно обробляють великі обсяги даних (багатотерабайтні набори даних) на великих кластерах (тисячі вузлів) в надійній, відмовостійкій манері. Програма MapReduce складається з двох основних компонентів: завдання відображення та завдання стиснення. Програма MapReduce приймає свої вхідні дані у формі пари (ключ, значення), які обробляються паралельно [69].

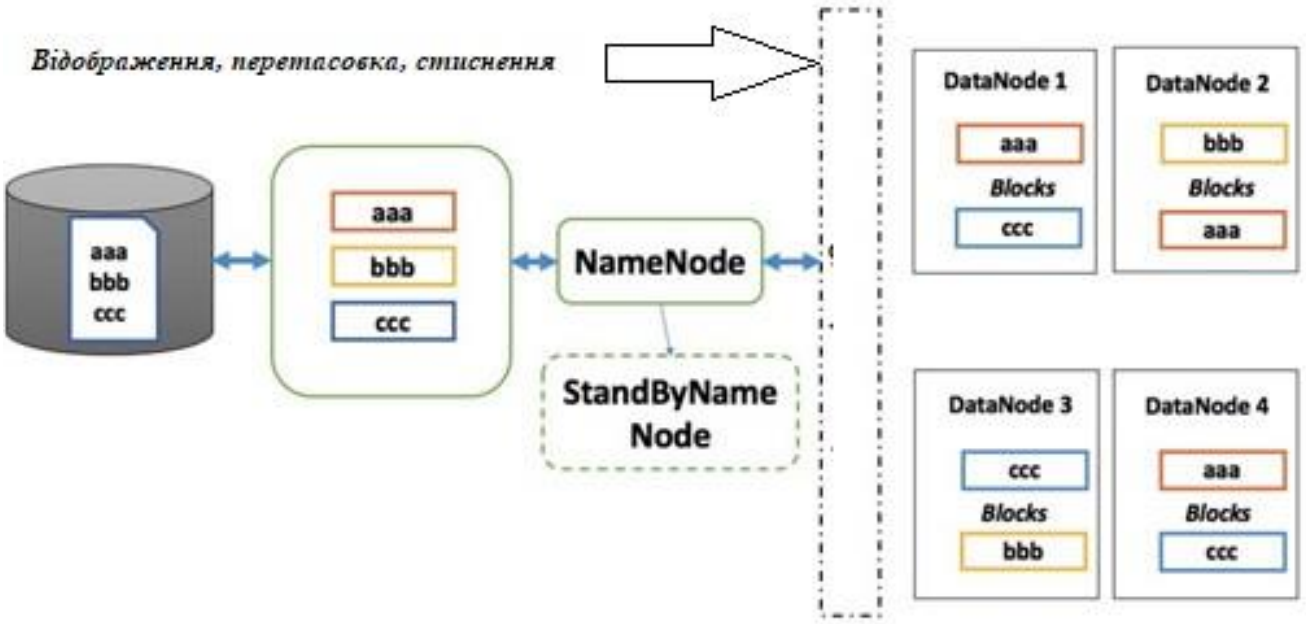


Рисунок 1.1 - Архітектура HDFS

Як показано на рисунку 1.1, завдання в MapReduce містить три фази: відображення, перетасовка і стиснення. У більшості випадків користувачеві потрібно лише написати функцію відображення та функцію стиснення. Фаза відображення, для кожної пари введення $(k\ 1, \upsilon\ 2)$, функція map генерує один або більше списків вихідних пар $(k\ 2, \upsilon\ 2)$. У фазі перетасовки вихідні пари розділяються, а потім передаються на редуктори. У фазі стиснення пари з одним і тим же ключем згруповані разом як $(k\ 2(list\ \upsilon\ 2))$. У фазі стиснення функція зменшення формує остаточний список пар вихідних даних $(k\ 3, \upsilon\ 3)$ для кожної групи. Процес MapReduce можна коротко описати нижче.

Вхідні та вихідні дані завдання зберігаються у HDFS. Фреймворк піклується про планування завдань, моніторинг їх і повторне виконання невдалих завдань. Зазвичай обчислювальні вузли та вузли зберігання однакові, тобто структура MapReduce та HDFS працюють на одному наборі вузлів. Ця конфігурація дозволяє структурі ефективно планувати завдання на вузлах, де дані вже є, що призводить до дуже високої сукупної пропускної здатності в кластері [69] (рис. 1.2).

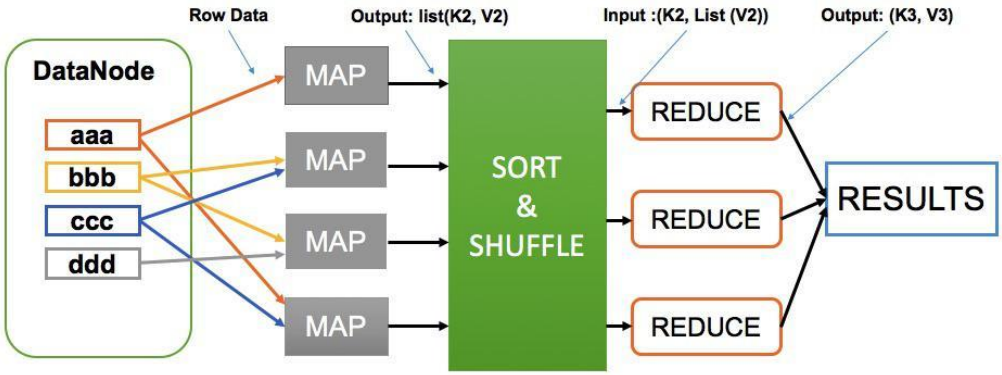


Рисунок 1.2 – Модель MapReduce

Фреймворк MapReduce складається з одного головного JobTracker та декількох підлеглих TaskTrackers. Кожен вузол, підключений до мережі, має право поводитися як підлеглий TaskTracker. Майстер отримує завдання від клієнта і планує карту і зменшує завдання на taskTracker, контролюючи їх і повторно виконуючи невдалі завдання. Користувачі виконують завдання за вказівкою ведучого (рисунок 1.3).

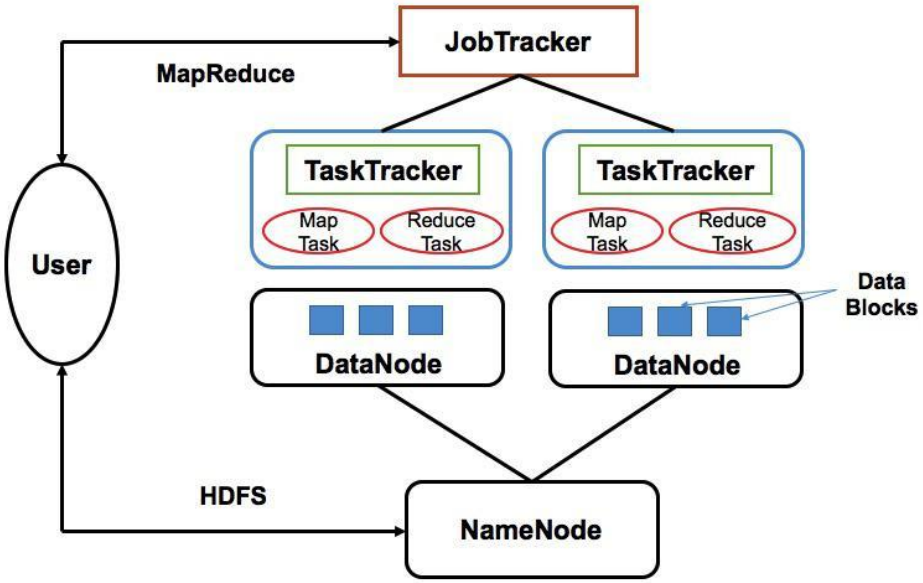


Рисунок 1.3 – Загальна структура фрейморку MapReduce

1.4 Постановка задчі

Дослідження джерел показало, що проблема кластеризації для видобутку просторових даних є надзвичайно актуальною. Отже, метою роботи є підвищення кластеризації для видобутку просторових даних шляхом розроблення методу розподіленої динамічної кластеризації для видобутку просторових даних. Для досягнення мети роботи, необхідно вирішити наступні завдання:

1) провести огляд відомих методів розподіленої динамічної кластеризації для видобутку просторових даних та визначити їх недоліки;

2) розробити метод розподіленої динамічної кластеризації для видобутку просторових даних, який би усував недоліки відомих методів та надав можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами;

3) на основі розробленого методу створити розподілену динамічну систему кластеризації для видобутку просторових даних, застосування якої надасть можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими системами.

2 МОДЕЛЬ ПРОЦЕСУ КЛАСТЕРИЗАЦІЇ ДЛЯ ВИДОБУТКУ ПРОСТОРОВИХ ДАНИХ

З метою забезпечення можливості обробки великих обсягів даних та вирішення супутніх проблем виникає необхідність у розробленні моделі процесу кластеризації для видобутку просторових даних. Процес кластеризації для видобутку просторових даних можна узагальнити для будь-якого типу даних, але головну увагу в роботі було зосереджено лише на просторових наборах даних.

2.1 Концепція моделі процесу кластеризації для видобутку просторових даних

Є багато факторів, які впливають на алгоритми видобутку великих даних. Перший фактор – це розмір вхідного набору даних, іншими словами, масштабованість. Розмір даних може впливати на продуктивність алгоритму інтелектуального аналізу даних у двох аспектах: час, необхідний для побудови та використання даної системи кластеризації, і точність результату кластеризації при збільшенні вхідних даних. Другим і третім факторами є неоднорідність та конфіденційність вхідних наборів даних. При збиранні великих обсягів наборів даних з різних джерел можна дійти висновку, що вони неоднорідні. Тобто ці дані містять різні рівні шуму та характеризуються певними атрибутами, що впливають на їх чутливість. Крім того, часто джерела належать різним організаціям, які не предають свої дані з різних причин (захист конфіденційності клієнтів, конкуренція тощо). Неоднорідні набори даних часто вимагають різних алгоритмів їх аналізу та видобутку для отримання корисних знань з них. Четвертий фактор – це вхідні дані та швидкість їх передачі. Обраний метод аналізу даних повинен бути здатним обробляти дані у міру їх отримання, без будь-якого погіршення продуктивності або затримки. Одним з найкращих способів вирішення цих проблем є аналіз даних безпосередньо з джерела даних. Це означає, що дані

будуть проаналізовані локально в усіх з усіх джерел. Цей метод має багато переваг:

1.Масштабованість: загальний набір даних уже розподілений, і можна використовувати потужність кожного джерела (компонента розподіленої системи) для обробки власних даних та прискорення процесу їх видобутку.

2. Неоднорідність: кожен компонент розподіленої системи може запустити алгоритм кластеризації, який найкраще підходить для його типу даних. Єдина вимога – представити результати в уніфікованому та послідовному вигляді, щоб загальні результати можна було використовувати та інтерпретувати на пізніх стадіях процесу прийняття рішень.

3. Конфіденційність: вона дотримується, оскільки дані аналізуються локально та лише розподілюються як результати локального видобутку. Набагато простіше контролювати конфіденційність на рівні результатів, ніж на рівні необроблених даних.

4. Швидкість: при обробці даних на місцевому рівні набагато легше мати справу з локальною швидкістю даних, ніж загальною швидкістю.

5. Ефективність: метод поєднує як паралельну, так і розподілену реалізацію для обробки даних великого обсягу.

З вищезазначеного ясно, що видобуток даних у джерелі буде вирішувати багато проблем. Наприклад, переміщення даних до центрального вузла вимагає високої пропускної здатності мережі, великої ємності зберігання та високої обробки центрального вузла. Вибрати алгоритм видобутку даних, з яким можна впоратися з шумом і великими обсягами даних непросто, і, можливо, не вдасться знайти ритм, за допомогою якого можно буде добувати дані із задовільньою якістю. Отже, видобуток даних локально має багато переваг. Ще потрібно вміти поєднувати локальні результати та створювати такі глобальні результати, що відображають тенденцію до цілих даних. Отже, модель процесу процесу кластеризації для видобутку просторових даних має грунтуватись на двох основних етапах: на першому етапі видобувається локальний набір даних, а на

другому він генерує глобальні результати (або кластери в даному випадку) з локальних кластерів (рисунок 2.1).

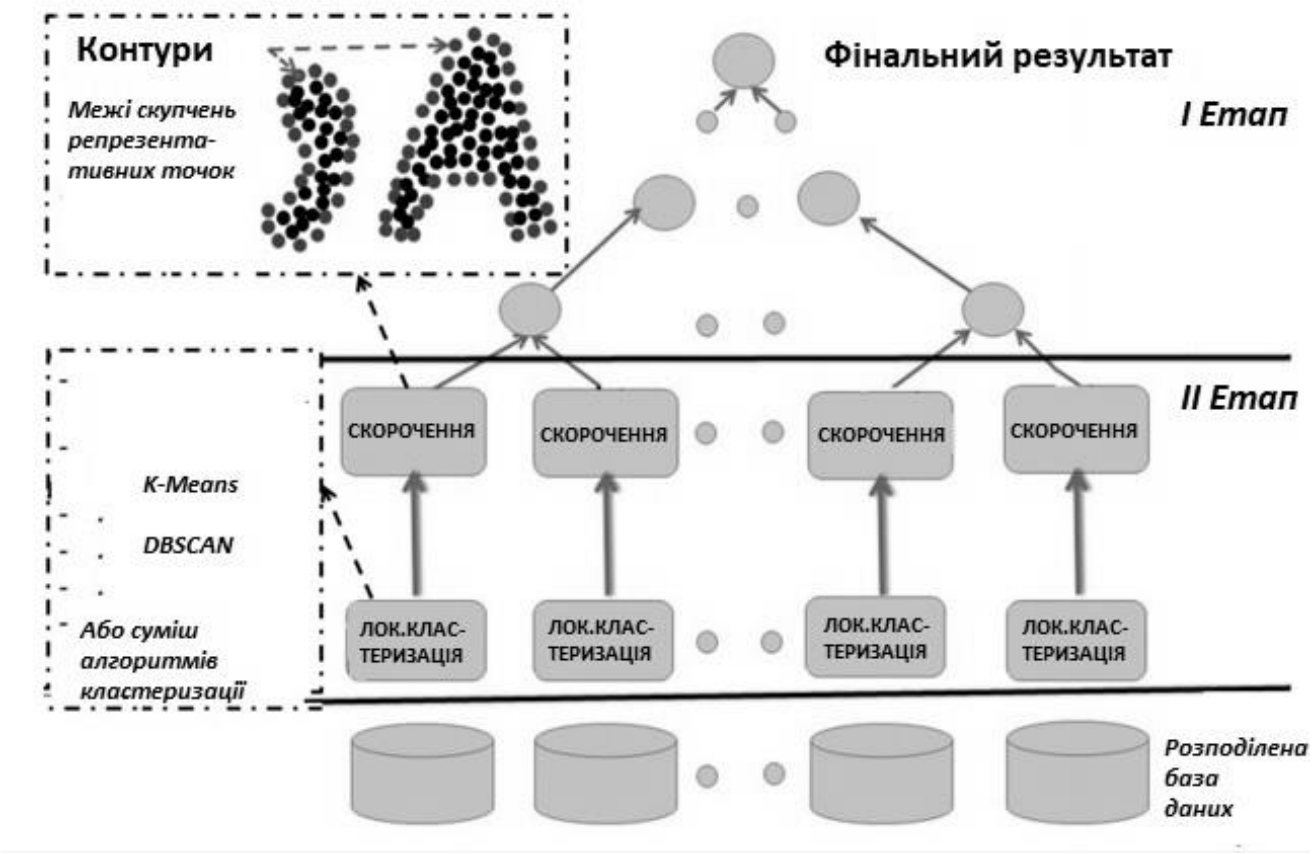


Рисунок 2.1 - Узагальнена схема процесу кластеризації для видобутку просторових даних

2.2 Модель процесу кластеризації для видобутку просторових даних

Представимо модель процесу кластеризації для видобутку просторових даних у вигляді кортежу:

$$\Psi = \langle \Omega, \Pi, O, \Theta, \varphi_1 \rangle, \tag{2.1}$$

де Ω – множина компоненів розподіленої динамічної системи,

$\Omega = \{\omega\}_{i=1}^{N\omega}$, де $N\omega$ - кількість компонентів розподіленої динамічної системи;

Π – множина алгоритмів кластеризації;

O – множина просторових даних, $O = \{o\}_{i=1}^{p}$, де p – кількість розділів просторових даних;

Θ – множина алгоритмів стиснення даних, $\Theta = \{\theta_1, \theta_2\}$, де θ_1 – алгоритм побудови α-форми, θ_2 – алгоритм побудови вектору балансування;

φ_1 – функція розподілення просторових даних між компонентами розподіленої динамічної системи з множини Ω та агрегації результатів кластеризації.

Опишемо множину алгоритмів кластеризації наступним кортежем:

$$\Pi = \langle \Xi, C, \varphi_2, \varphi_3 \rangle, \tag{2.2}$$

Ξ – множина об'єктів кластеризації;

C – множина кластерів, $C = \{c\}_{i=1}^{N_c}$, де N_c – кількість кластерів;

φ_2 – функція відстані до центру кластера;

φ_3 – функція кластеризації, $\Xi \xrightarrow{\varphi_3} C$.

Представимо функцію φ_1 наступним чином:

$$\varphi_1 = \langle s_1, s_2 \rangle, \tag{2.3}$$

де s_1 – етап розподілення просторових даних між компонентами розподіленої динамічної системи;

s_2 – етап агрегації результатів кластеризації.

Етап розподілення просторових даних s_1 між компонентами розподіленої динамічної системи передбачає, що загальні дані складаються з P розділів (або підмножин). Далі буде приведено, яким був би оптимальний P, враховуючи розмір вхідних даних. У випадку, коли цільова обчислювальна платформа розгортається, P має бути кількістю вузлів обробки в системі. Для кожного учасника P_i кластеризуємо його дані у C_i кластери. При розгортанні на

розподіленій або паралельній системі з вузлами обробки P, ця фаза є суто паралельною, як і кожен вузол обробки виконує алгоритм кластеризації на своєму розділі даних незалежно від інших. Локальним алгоритмом кластеризації може бути будь-який алгоритм кластеризації. Це може бути алгоритм, заснований на центроїді (K-means), алгоритм на основі щільності (DBSCAN) або навіть суміш алгоритмів кластеризації. Вподальшому будуть розглянуті тематичні дослідження, в яких виконується локальна кластеризація або K-means, або DBSCAN. Це потрібно для того, щоб продемонструвати ефективність PK підходу. Отримані кластери на кожному вузлі називаються локальними кластерами або локальними моделями. Локальні кластери сильно залежать від методів кластеризації, які використовуються відповідним їм вузлам обробки. Наприклад, для просторових наборів даних скупчення зазвичай диктується методом, що використовується для їх отримання. На першому етапі це не є великою проблемою, оскільки точність кластеру впливає лише на локальні результати даного вузла. Більше того, час відгуку кожного вузла обробки залежить від розміру локального набору даних, а також від складності алгоритму кластеризації, що використовується локально. Отже, якщо розмір кожного розділу не пропорційний потужності його обробного вузла, існує ризик очікування перед тим, як розпочати другий етап. Всі ці випадку будуть ще розлянуті далі.

Етап агрегації результатів кластеризації s_2 складається із збору локальних моделей від кожного вузла. Введемо таке поняття як лідери. Лідери обираються відповідно до деяких характеристик, таких якїх потужність, обчислювальна потужність, підключення тощо. Лідери несуть відповідальність для об’єднання локальних кластерів, які перекриваються. Цей етап вимагає обміну усіма локальними кластерами з керівниками. Якщо даних велеикий обсяг, ця операція дуже швидко наситить мережу. Отже, треба уникати надсилання всіх вихідних даних через мережу. Ключова ідея методу PK полягає у відправці лише представників кластеру, які складають від 1% до 2% від загального обсягу даних

Представники кластеру складаються з внутрішніх представників даних плюс граничних точок скупчення. Багато існуючих методів зменшення даних

зосереджуються лише на розмірі набору даних, тобто вони намагаються зменшити ємність сховища, не звертаючи уваги до знань, що містяться в даних [70,71].

Ще однин можливий метод кластеризації на основі щільності пропонує вибрати набір точок, що називаються репрезентативними пунктами [72] . Цей прийом достатньо ефективний, однак вибір представників все ще залишається проблемою з точки зору їх якості та розміру [73, 74]. Найкращий спосіб представити просторове скупчення - це його форма та щільність. Форма скупчення представлена його граничними точками (так звані контури) (Рисунок 4.1) . Багато алгоритмів для вилучення меж кластера можуть бути знайдені в літературі [75 – 79]. Було використано два алгоритми для вилучення контурів скупчення: алгоритм форми, запропонований в [80], в основі якого лежить тріангуляція для формування меж кластера, та алгоритм вектора рівноваги, запропонований в [81] на основі алгоритму кластеризації щільності. Обидва алгоритми ефективні для побудови неопуклої межі. Алгоритми здатні точно характеризувати форму широкого діапазону різних точкових розподілів і щільностей з розумною складність $o(N \log N)$. Глобальні моделі (зразки) формуються на другому етапі РК. Цей етап виконується розподіленим чином, але, на відміну від першого етапу, вузли обробки повинні обмінюватися своїми локальними кластерами або клестерами-посередниками. Цей етап складається з двох основних етапів, які можна повторювати поки не будуть сформовані всі глобальні кластери. Спочатку кожен керівник збирає місцеве скупчення своїх сусідів. Потім лідери об’єднують місцеві кластери. Процес злиття кластерів триватиме поки не дійдемо до кореневого вузла. Кореневий вузол буде містити глобальні кластери (рисунок 2.1). Як уже згадувалося вище, передача лідерам місцевих кластерів може згенерувати величезні накладні витрати. Тому метою є мінімізація даних зв'язку та обчислювального часу, отримуючи при цьому точні глобальні повторні імпульси. В запропонованому підході застосовано обмін лише межами (контурами) кластерів замість обміну цілими кластерами між системними вузлами. Межі кластерів представляють новий набір даних, і вони є набагато

меншими, ніж оригінальні набори даних. Отже, межі скупчень стануть локальними результатами на кожному вузлі в системі. Ці місцеві результати надсилаються лідерам за топологією дерева (рисунок 2.1). Глобальні результати будуть розташовані біля кореня дерева.

Розглянемо множину алгоритмів стиснення даних Θ. Основною відмінністю розробленого методу є те, що на етапі агрегації результатів кластеризації застосовується обмін лише частиною об'єктів кластеризації, які називаються репрезентативними точками. Ці точки (об'єкти кластеризації) відповідають межі локальних скупчень. В запропонованому підході застосовано два алгоритми вилучення граничних точок з набору даних, а саме: алгоритм α-форми та вектор рівноваги.

Розглянемо особливості алгоритму побудови α-форми θ_1 – це простий, гнучкий та ефективний алгоритм. Це простий багатокутник, що характеризує форму набору вхідних точок у площині, що називається характерною формою. Алгоритм базується на тріангуляції Делоне [82]. Вироблена алгоритмом форма управляється одним нормованим параметром ,який можна використовувати для створення широкого діапазону характерних форм, різних між опуклим корпусом в одній крайності і однозначно визначеною формою с мінімальною площею. Перш ніж будуть надані кроки алгоритму α- форми, треба дати визначення основних термінів.

Розглянемо особливості кластерів опуклої і не опуклої форми. Опукла форма або опуклий багатокутник визначається його зовнішніми точками. Малоопуклий багатокутник повинен виконувати умову, що всі його кути повинні бути менше 180 градусів. Якщо ні, то багатокутник не опуклий або також називається увігнутим багатокутником. На рисунку 2.2 показаний приклад обох опуклих і не опуклих форм набору точок даних кластерів.

Рисунок 2.2 – Опукла та не опукла форма кластерів

На рисунку 2.2 можна побачити опуклу та не опукла форми. Тріангуляція Делоне у математиці та обчислювальній геометрії для множини P точок на площині - це триангуляція $DT(P)$. Це така триангуляція, що жодна точка в P не знаходиться всередині описаного кола будь-якого трикутника в $DT(P)$. Тріангуляція Делоне максимізує мінімальний кут усіх кутів трикутників у тріангуляції. Вони як правило, уникають тріщинних трикутників. Інший спосіб визначення триангуляції Делоне - це шляхом перевірки кутів сформованих трикутників: дивлячись на рисунок 2 .3 можна зрозуміти, що якщо сума кутів α_1 і α_2 менше або дорівнює 180 градусам, тоді трикутник відповідає умові Делоне. В іншому випадку трикутник не задовольняє умову Делоне (червоний край переривання). В цьому випадку нам потрібно лише перевернути край, щоб умова була задоволена.

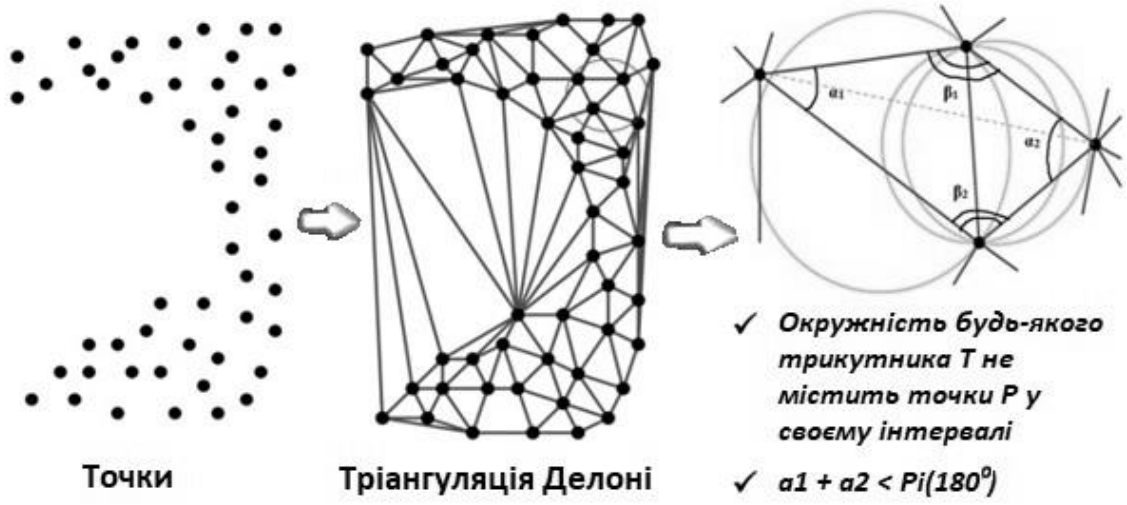


Рисунок 2.3- Тріангуляція Делоне

Кроки алгоритму побудови α-форми θ_1 подано наведено нижче та подано на рисунку 2.4.

1. Сформувати триангуляцію Делоне набором вхідних точок (P).

2. Видалити з триангуляції найдовший зовнішній край так, щоб:

2.1. Край, який потрібно видалити, був довший за параметр довжини l

2.2. Зовнішні краї результуючої тріангуляції утворюють межу простого многокутника (скупчення точок).

3. Повторюйте другий крок до тих пір, поки не буде видалено більше країв.

4. Повернути багатокутник (контур), утворений зовнішніми краями триангуляції.

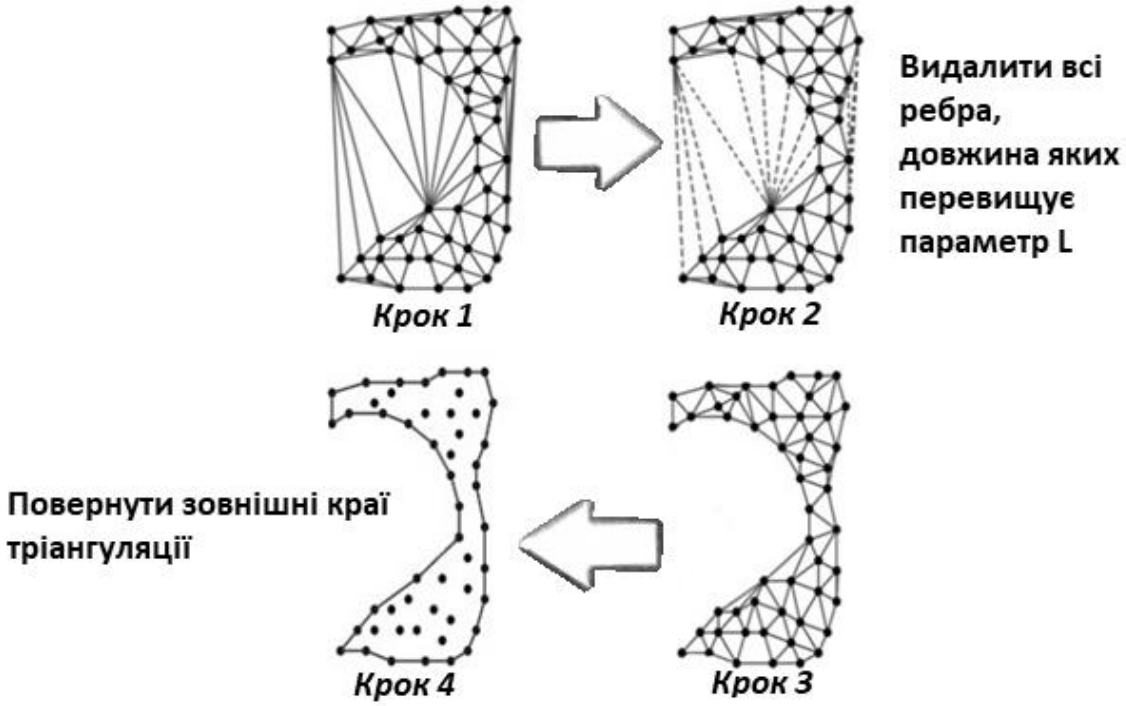


Рисунок 2.4 - Алгоритм побудови α-форми

Алгоритм α-форми є ефективним та оптимальним алгоритмом, що вимагає лише $o(n\log n)$ одиниць часу для виконання. Форма, вироблена алгоритмом, параметризується за допомогою одного параметра нормованої довжини, яким легко керувати.

Також розглянемо особливості алгоритму побудови вектору рівноваги θ_2. Вектор рівноваги - ще один алгоритм, який використовується для виявлення межі локальних скупчень об'єктів кластеризації. Цей алгоритм працює з кластеризацією на основі щільності. Тому він більше підходить для вилучення контурів із скупчень, які належать до кластеру на основі щільності. Перш ніж детально описувати цей алгоритм, спочатку надамо основні поняття цієї методу.

З огляду на кластер $C \subseteq \Re^n \equiv \{p_1, p_2 ..., p_n\}$ де $N^c(p)$ точки p в кластері C визначається як набір точок $p_i \in C$ так, що відстань між p_i і p менше або дорівнює ε, визначимо зону сусідніх об'єктів кластеризації наступним чином:

$$N^C(p) = f_{pi}2C \mid dist(p, p_i) < g \quad . \tag{2.4}$$

Для того, щоб визначити граничні точки скупчення в кластері, потрібно визначити наступні складові моделі.

Вектор переміщення - вектор переміщення від $p_i \in N^C(p)$ до точки p визначається як:

$$\vec{V} = \sum_{p_i \in N^C(p)} (p - p_i) \quad . \tag{2.5}$$

Цей вектор вказує на область найменшої щільності сусідів вектору рівноваги.

Вектор рівноваги щодо точки p визначимо наступним чином:

$$\vec{b}_p = \begin{cases} \dfrac{1}{\left\|\vec{V}_p\right\|} \vec{V}_p & \text{if } \left\|\vec{V}_p\right\| > 0 \\ \end{cases} \quad . \tag{2.6}$$

Варто зазначити, що вектор рівноваги ρ вказує на найменш щільну площу сусідніх об’єктів кластеризації довжини ρ, довжина вектора не містить жодної відповідної інформації. На рисунку 2.5 показаний приклад вектора рівноваги. Межі кластера знаходяться всередині кола, вектор рівноваги представлений чорною стрілкою.

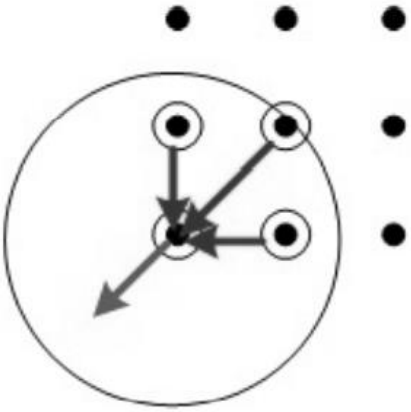


Рисунок 2.5 — Приклад вектора рівноваги

Якщо точка є граничною точкою, то точок не повинно бути у напрямку до вектора рівноваги. Ця властивість дозволяє розділити на граничні точки та внутрішні точки. Для кожної точки p він перевіряє наявність порожньої форми, яка є перетином гіперконуса нескінченної висоти, вершини, осі і апертура ρ, де ρ - заданий кут. Як показано на рисунку 2.6, перевірена область виділена сірим кольором.

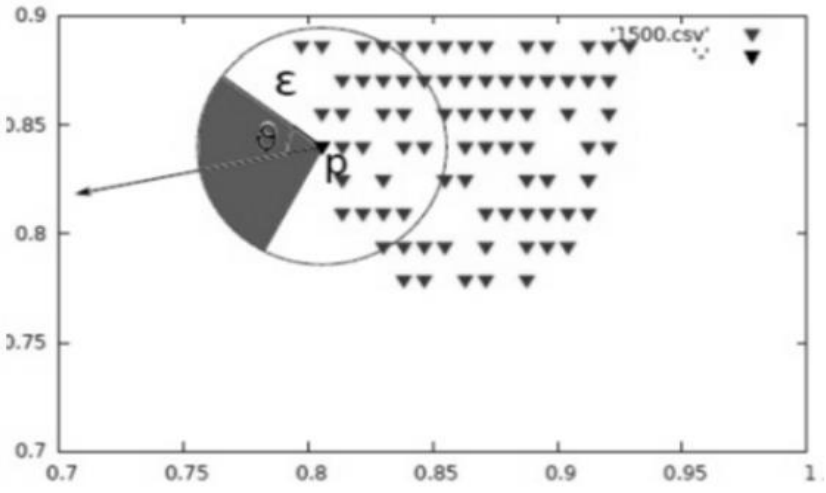


Рисунок 2.6 – Перевірка граничної точки

$$Boundary(p) = \begin{cases} true \text{ if } (\forall q \in N_{\varepsilon}^{C}(p), (q-p)\vec{b}_b < \cos(\upsilon) \\ false \text{ otherwise} \end{cases} . \quad (2.7)$$

Визначимо межу B_c кластеру C як сукупність усієї межі точок в:

$$B_c = \{ \ f \in C : \text{Boundary(p) is true} \ \}. \quad (2.8)$$

Алгоритм вибору граничних точок можна представити у вигляді алгоритму 2.1.

input : Cluster C, Set of balance vectors $\vec{b}_{p_i}$, Parameter ν.
output: Boundary points B_C of cluster C

```
1 B_C ← C;
2 for every point p ∈ B_C do
3 |   for every point q ∈ N^C(p) do
4 |   |   if (q − p) b⃗ ≥ cos(υ) then
5 |   |   |   Discard p from B_C;
6 |   |   |   Break;
7 return B_C;
```

Алгоритм 2.1 – Вибір граничних точок

Нехай C_i - набір кластерів вузла i та B_{cj} є межею скупчення $c_j \in C_i$, тоді локальна модель кластера L_i визначається:

$$L_i = \bigcup_{j=1}^{n} B_{c_j} \cup P_i, \quad (2.9)$$

де P_i - сукупність репрезентативних точок.

Висновок

В другому розділі розроблено модель розподіленої динамічної системи кластеризації для видобутку просторових даних, яка, на відміну від відомих для стиснення даних використовує алгоритм побудови α-форми та алгоритм побудови вектора рівноваги, що надає можливість зменшити обсяг даних, які передаються між компонентами розподіленої системи. Таким чином, розроблена модель враховує можливість кластеризації великих наборів даних, які можуть бути неоднорідними та розподіленими.

3 МЕТОД КЛАСТЕРИЗАЦІЇ ДЛЯ ВИДОБУТКУ ПРОСТОРОВИХ ДАНИХ

Після попереднього представлення та висвітлення характеристик розподіленої динамічної кластеризації (РК), представимо ще одну дуже важливу можливість цього підходу, а саме розробку нових алгоритмів кластеризації на основі існуючих технологій видобутку даних. Перший метод заснований на K-means, а другий - на DBSCAN. Це демонструє гнучкість запропонованого методу та його ефективність у видобутку дуже великих наборів даних. Буде приведено, що цей підхід, заснований на існуючих алгоритмах, не тільки перевершує існуючі методи, застосовані до цілих наборів даних, але він також вирішує деякі їх проблеми, такі як жорсткість їх вхідних параметрів.

Також було порівняно результати застосування запропонованого методу з іншими існуючими та відомими алгоритмами BIRCH та CURE.

Розглянемо основні кроки запропонованого методу:

1. Кожен вузол має набір даних, що представляє частину даних або загальний набір даних.

2. Кожен листовий вузол виконує локальний алгоритм кластеризації зі своїм входними параметрами.

3. Кожен вузол ділиться своїми кластерами із сусідами, щоб утворити більші кластери з використанням методу накладання даних.

4. Вузол лідера містить результати своїх груп.

5. Повторювати пункти 3 та 4, поки не будуть сформовані всі глобальні кластери.

Псевдокоди алгоритмів робои розробленого методу на етапах розподілення просторових даних між компонентами розподіленої динамічної системи та агрегації результатів кластеризації наведені алгоритмом 3.1.

```
1 Initialisation
2 Node_i ∈ N, N: The total nodes in the system.
  input : X_i: Dataset Fragment, Params_i: Input parameters for the local
          clustering: Params_i= (Eps_i, MinPts_i) for DBSCAN for example
  output: C_i: Cluster's contours of Node_i
3 foreach Node_i do
4 |   L_i = Local_Clustering(X_i,Params_i);
  |   // Node_i executes a clustering algorithm locally.
5 |   C_i = Contour(L_i);
  |   // Node_i executes a contour algorithm locally.
6 return (C_i);
```

Алгоритм 3.1 – I етап: локальна кластеризація

3.1 Розподілена динамічна кластеризація з використанням K-means

K-means - це один з основних і найбільш часто використовуваних алгоритмів кластеризації. Він також характеризується своєю простотою та хорошою обчислювальною складністю. Використовується K-means як локальний алгоритм кластеризації для кожного розділу набору даних. Іншими словами, кожен вузол обробки виконує алгоритм K-means на своїх локальних даних.Варто зауважити, що кожне виконання K- means на наборі даних вимагає заздалегідь вказаної кількості кластерів. Вважатимемо, що система має P -вузли обробки $\{\upsilon_1, \upsilon_2, ..., \upsilon_p\}$, кожен υ_i вимагає вхідного параметра K_i виконати K-means на його даних. Вся система вимагає $(K_1, K_2, ..., K_p)$ параметрів, що не є практичним і навіть неможливим, якщо нам потрібно налаштувати систему на пошук оптимальної кінцевої кластеризації. Тоді питання в тому, як встановити параметри $\{K_i\}_1^p$, такі, що остаточна кластеризація має високу якість. Це дає зрозуміти, що відповідь досить проста.

Алгоритм 3.2: PK-алгоритм: II етап - агрегація

input : D: Tree degree, C_i: Local cluster's contours generated by $Node_i$ in the phase 1

output: $C_{G_{k,level}}$: Global Cluster's contours (global results, level=0)

```
1  repeat
2      level = treeHeight;
3      Node_i joins a group G_{K,Level} of D elements;
       // Node_i joins its neighbourhood
4      Node_j=ElectLeaderNode(G_{K,Level});
       // Node_j is the leader of the group G
       // In parallel
5      foreach Node_i ∈ G_{K,Level} do
6          if (i <> j) then
7              Send (C_i, Node_j);
               // Each node sends its contours to others
                  nodes in the same group of neighbourhood
8          else
9              Recv (C ≡ ({C_l}, Node_l));
               // If the node is the leader, it will receive
                  the others node's contours in the same
                  group of neighbourhood
10             G_{K,Level}= Merge (C_i, C);
               // Merge the overlapping contours
11     level - - ;
12 until (level == 0);
13 return (C_{G_{k,0}})
```

Алгоритм 3.2 – II етап: агрегація

Згідно запропонованому методу кожен вузол динамічної розподіленої системи υ_i виконує K-means на своєму локальному наборі даних (розділі) під час етапу розподілення просторових даних між компонентами розподіленої динамічної системи за допомогою K_i і виробляє L_i - локальні кластери. В рамках першого етапа кожен υ_i обчислює контури його локального L_i скупчення. Другий етап агрегації результатів кластеризації складається з обміну контурами, розташованими в кожному вузлі υ_i з сусідніми вузлами. Кожен лідер намагається об'єднати контури своєї групи. Тому кожен лідер генерує нові контури (нові кластери). Процедура злиття триватиме до тих пір, поки не буде контурів, що перекриваються.

На цьому етапі можна зробити два наступні зауваження щодо етапа агрегації результатів кластеризації.

1. Етап агрегування може об'єднати лише два кластери, які перекриваються. Він не може розділити жоден локальний кластер, який може перекривати два різні кластери. Наприклад, припустимо, що є два абсолютно різні кластери C_a і C_b. Припустимо, що вузол v_c створив кластер C_x що перекриває обидва C_a і C_b. Навіть якщо заздалегідь відомо, що C_a і C_b є двома різними кластерами і їх не слід об’єднувати, етап агрегування РК об’єднає їх через C_x. Причиною цього є те, що C_x не є якісним кластером, і тому його слід розділити. Зазвичай це пов'язано з тим, що або кількість кластерів K_c занадто малий та / або початкові умови погані. Отже, одним із способів уникнути такої ситуації є вибір K_c досить великий для того, щоб генерувати більше малих локальних кластерів, які згодом можуть бути об'єднані з більшими під час етапу агрегації.

2. Кількість кінцевих кластерів залежить лише від локальних кластерів, що перекриваються. Це не залежить від $\{K_i\}_1^p$, $\{K_i\}_1^p$ контролює лише виробництво локальних кластерів, і це не має прямого впливу на остаточну кількість кластерів, що є надзвичайно важливим результатом.

Щоб відповісти на питання “як вибрати $\{K_i\}_1^p$ такі, щоб кінцеві кластери мали дуже хорошу якість? ", потрібно лише встановити K_i більше оптимальної кількості кластерів у наборі даних. Наприклад, якщо оптимальний $K = 10$, то $K_i > 10$. Наскільки далеко від оптимального K або наскільки це близький до нього компроміс між обчисленнями та якістю глобальних кластерів, оскільки відомо, що чим більше K великих, тим більше обчислень потрібно для виконання алгоритму.

Для більш детального роз'яснення методу буде показано в наступному розділі приклад виконання методу з використанням K-means як локальної кластеризації. Варто зауважити, що з K-means було використано як алгоритми α-форми, так і вектора рівноваги на етапі агрегування, щоб виділити контури, і також щоб вони обидва працювали добре і дали хороші результати

3.2 Приклад застосування розробленого методу з використанням K-means

Розглянемо приклад застосування розробленого методу з використанням K-means. Нехай динамічна розподілена система містить п'ять вузлів $(N = 5)$, і кожен вузол виконує алгоритм K-means з різними K_i, як показано на рисунку 5.1. Вузол$_1$ виконує K-means за допомогою $K_1 = 30$, Вузол$_2$ з $K_2 = 60$ Вузол$_3$ з $K_3 = 90$, Вузол$_4$ з $K_4 = 120$, і Вузол$_5$ з $K_5 = 150$. Отже, кожен вузол у системі генерує свої локальні кластери на основі свого вхідного параметра K. Наступний крок складається із об’єднання кластерів, що перекриваються, в межах сусідства.

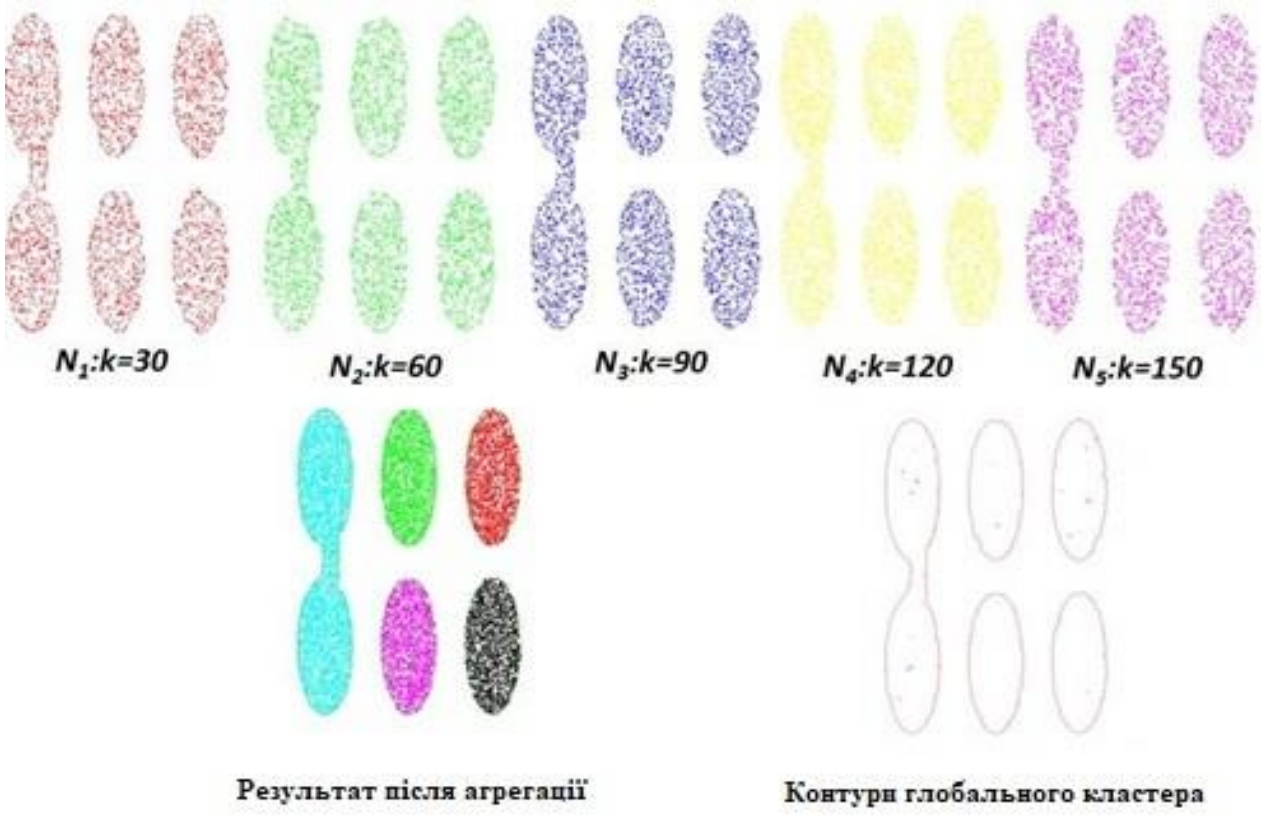


Рисунок 3.1 — Розподілений динамічний алгоритм кластеризації з використанням K-means

Як видно з рис. 3.1, хоча було почато з різних значень K локально, було створено лише п’ять глобальних кластерів (рисунок 3.1). K-means динамічно визначає кількість кластерів без апріорних знань про дані або процес оцінки кількості кластерів. Іншими словами, K-means означає, що кількість глобальних кластерів не повинна бути вказана як вхідні дані. Він обчислюється динамічно. Більше того, кожна локальна кластеризація L_i з K-means потребує K_i як параметр, який не обов'язково є точним для цієї локальної кластеризації.

Для демонстрації переваг використання алгоритму K-means в запропонованому підході порівняємо результати експериментів з алгоритмами BIRCH та CURE:

Алгоритм BIRCH виконує попередню кластеризацію, а потім використовує алгоритм ієрархічної кластеризації на основі центроїдів. Зауважимо, що часова та

просторова складність цього методу є квадратичною до кількості точок після попередньої кластеризації.

Алгоритм CURE використовує репрезентативні точки зі скороченням до середнього. Коли два кластери об'єднуються на кожному кроці, репрезентативні точки для знову об'єднаних кластерів вибираються з тих двох початкових кластерів, а не з усіх точок у об'єднаних кластерах.

Ключовим моментом запропонованого методу з використанням K-means є вибір K_i більшого за правильну кількість кластерів. Як було згадано вище, коли два кластери об'єднуються на кожному кроці алгоритму, репрезентативні точки новозлитого кластера є об'єднанням контурів двох вихідних кластерів, а не всіх точок нового кластера. Це прискорює час виконання без негативного впливу на якість сформованих кластерів. Крім того, дана методика використовує топологію дерева та структури даних кучі. Таким чином, це також зменшує обчислювальну складність алгоритму.

З метою проведення експерименту було запущено три алгоритми на одній машині, Dell-XPS (1,8 ГГц * 4 Intel Core i5 з 8 Гб пам’яті) з Ubuntu (14,04 LTS) як ОС. Було проведено експерименти з різними наборами даних. Використано три набори даних з різними формами та розмірами [!!!]. Ці набори даних узагальнені в таблиці 5.1. Кількість точок і кластерів у кожному наборі даних також наведено в таблиці 5.1.

Було запущено три алгоритми на трьох наборах даних, щоб порівняти їх із повторним аналізом до якості сформованих кластерів. Рисунок 3.2, рисунок 3.3 та Рисунок 3.4 показують кластери, знайдені трьома алгоритмами для трьох наборів даних (T_1, T_2, T_3). Було використано різні кольори, щоб показати кластери, повернені кожним алгоритмом.

Таблиця 3.1 — Набори даних, що використовуються для тестування РК-K-means

Набір даних	**Опис**	**Бали**	**Кластери**
T1	Великий овал (форма яйця)	14000	5
T2	4 маленькі кружечки і 2 зв’язаних маленьких кола	17080	5
T3	2 маленькі кола, 1 велике коло і 2 зчеплених овали	30 350	4

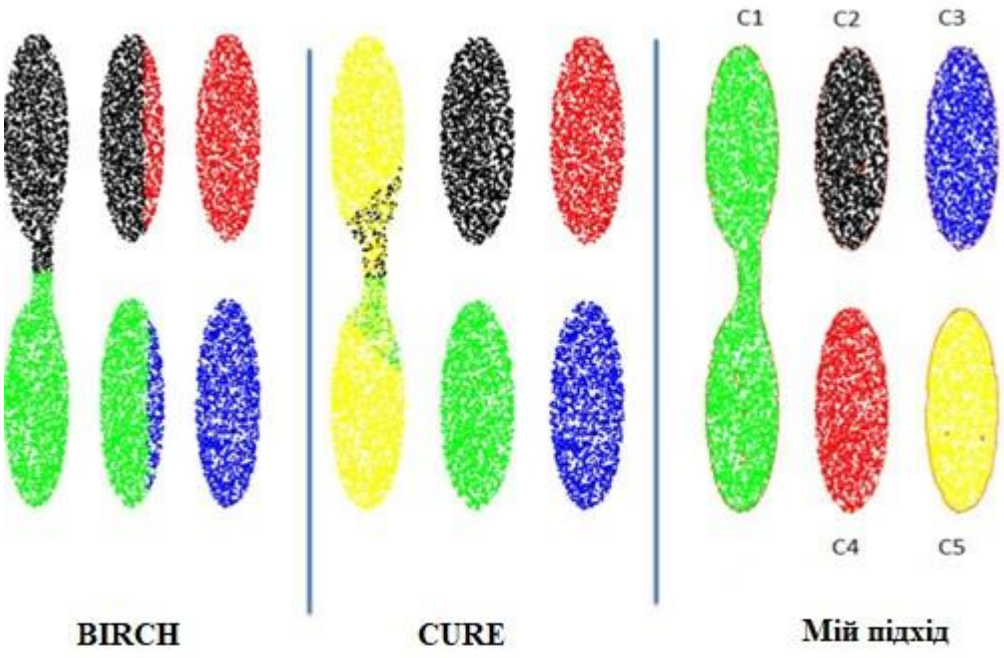


Рисунок 3.2 — Кластери, сгенеровані з використанням K-means з T_1

На рисунку 3.2 показані кластери, генеровані з T_1. Як і слід було очікувати, оскільки BIRCH використовує ієрархічний алгоритм кластеризації на основі центроїд для кластеризації попередньо кластеризованих точок, він не зміг правильно знайти всі кластери. Він розбиває більший кластер, одночасно зливаючи інші. На відміну від цього, алгоритму CURE вдається сформувати більшість кластерів, але він все ще не може виявити всі правильні кластери. Запропонований розподілений алгоритм кластеризації успішно генерує всі кластери із налаштуваннями параметрів за замовчуванням. Як показано на рисунку 3.2, після об'єднання локальних кластерів було створено п'ять кінцевих кластерів.

На рисунку 3.3 показані результати, знайдені трьома алгоритмами для набору даних T_2. Знову ж, BIRCH та CURE не змогли сформувати всі кластери, тоді як наш алгоритм успішно створив чотири правильні кластери.

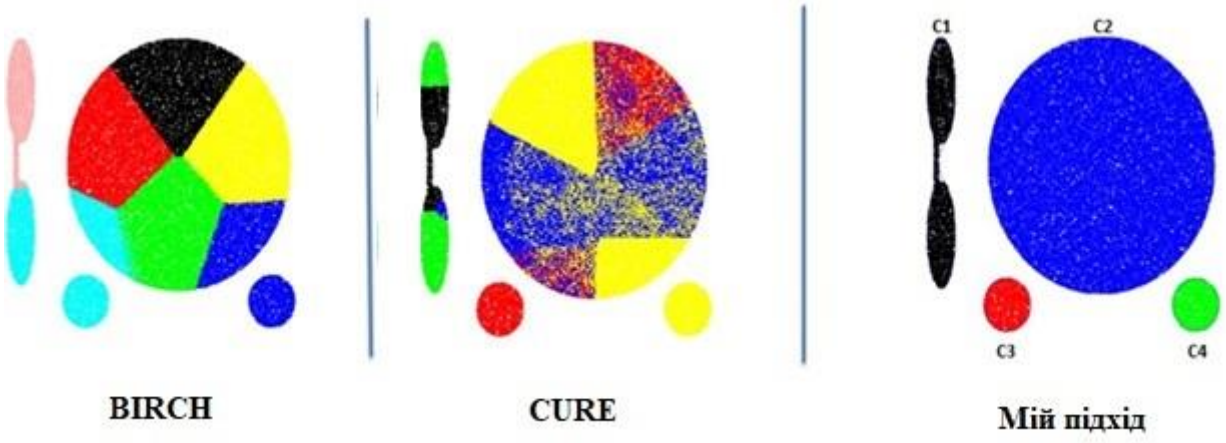


Рисунок 3.3 — Кластери, сгенеровані з використанням K-means з T_2.

Рисунок 3.4 ілюструє кластеризацію, яку було знайдено з набору даних T3. Можна побачити, BIRCH все ще не може правильно знайти всі кластери. На відміну від них, CURE знайшов п'ять скупчень, але не ідеально. Наприклад, помітно деякі червоні точки в синьому скупченні, а деякі сині точки в зеленому скупченні. Алгоритм створив п'ять кластерів правильно і досконало.

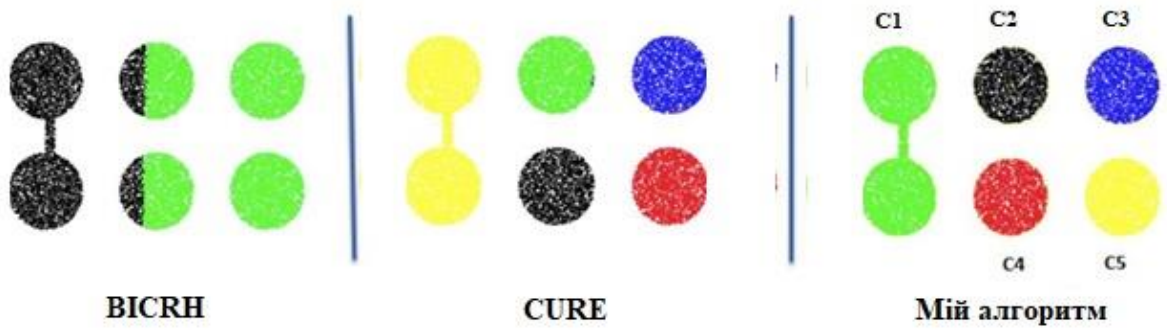


Рисунок 3.4 — Кластери, сгенеровані K-means з T_3

Як можна побачити, метод успішно створив кінцеві кластери для трьох наборів даних. Це пов'язано з тим, що:коли два кластери об'єднані, новий кластер представляється об'єднанням двох контурів двох вихідних кластерів. Це прискорює час виконання, не впливаючи на якість сформованих кластерів. Кількість глобальних кластерів динамічна.

Часова складність алгоритму K-means визначається наступним чином:

$$T_{k-means} = o(dkn)\,, \tag{3.1}$$

де d : кількість розмірів набору даних, k_i : кількість підкластерів вузла υ_i та n_i : набір даних, наданий кожному вузлу υ_i .

Таким чином, одержимо:

$$T_{PK-K-means} = o(dk_i n_i) + o(c_i \log c_i) + o(w_i \log w_i + p) \cong o(n_i) \tag{3.2}$$

Також було згенеровано загальну складність алгоритму K-means:

$$T_{DDC-K-means} = \mathcal{O}(dk_i n_i) + \mathcal{O}[\frac{n_i}{k_i} \log \frac{n_i}{k_i}] +$$
$$\mathcal{O}[(\frac{n_i}{k_i}\beta) \log (\frac{n_i}{k_i}\beta) + p] \simeq \mathcal{O}(n_i) + \mathcal{O}(n_i \log n_i)$$

$$T_{PK-K-means} = o(dk_i) + o\left[\frac{n_i}{k_i}\log\frac{n_i}{k_i}\right] +$$

$$+ o\left[(\frac{n_i}{k_i})\beta\right]\log(\frac{n_i}{k_i}\beta + p) \cong O(n_i) + o(n_i \log n_i) . \quad (3.3)$$

Буде помітно, що часова складність усього методу РК при використанні K-means як локального алгоритму кластеризації є поліномною. Це робить метод дуже ефективним з точки зору часу виконання.

Метою роботи є розробка та реалізація підходу, який може застосовувати будь-який алгоритм кластеризації на локальному рівні. Крім того, метод повинен працювати з різними і реальними наборами даних. Реальні набори даних характеризуються різними формами та розмірами, і вони можуть містити шуми та викиди.

Як показали результати експериментів, K-means працює краще, ніж і BIRCH, і CURE, і це дало точні результати. Крім того, часова складність РК-K-means лінійно зростає із збільшенням розміру даних. Однак було помічено, що форма використовувших даних має опуклу форму і не містить шумів. Тому потрібно перевірити метод за допомогою різних наборів даних, які мають не опуклі форми і містять шум. Для цього було проведено ще один експеримент, цього разу було взято ще два набори даних: T4 і T5. Зауваживши, що T5 - той самий набір даних, що і T4 для якого було прибрано шум. Зверніть увагу, що на наступних рисунках кожен колір являє собою окремий кластер.

Помітивши з рисунка 3.5. K-means не вдається знайти хороші кластери для двох наборів даних (T4 і T5), це пов'язано з тим, що алгоритм K-means прагне працювати лише з наборами даних опуклої форми, оскільки він заснований на принципі центроїду для створення кластерів. Більше того, було помічено, що результати K-means ще гірші з набором даних, що містить шум (T5). Насправді він повертає весь набір даних із шумом як остаточний кластер (рис. 5.5). Це пов’язано з тим, що K-means не справляються із шумом.

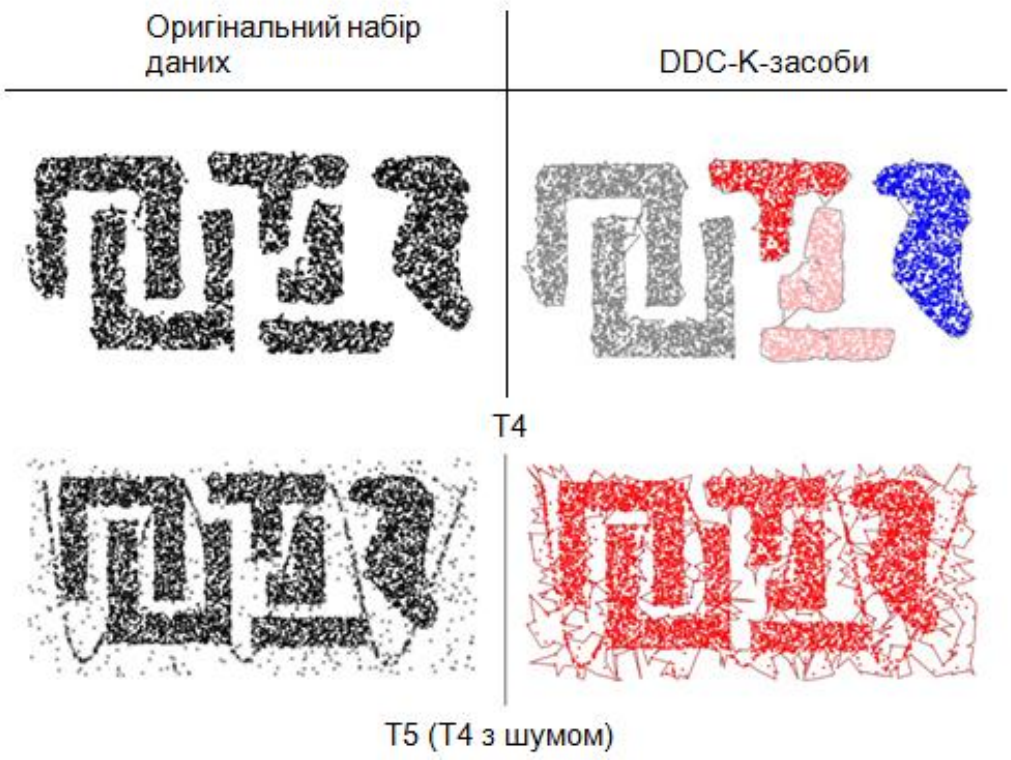


Рисунок 3.5 — K-means із неопуклою формою та зашумленими даними

3.3 Приклад застосування розробленого методу з використанням DBSCAN

Незважаючи на те, що в запропонованому підході K-means працює набагато краще, ніж деякі відомі алгоритми кластеризації (BIRCH та CURE) на просторових наборах даних, він все ще не може мати справу з усіма видами наборів даних; переважно з неопуклими формами. Крім того, K-means дуже чутливий до шуму. Тому замість K-means буде використано інший алгоритм кластеризації для просторових наборів даних, яким є DBSCAN.

Метод залишається незмінним, різниця лише на локальному рівні, де замість використання K-means для обробки локальних кластерів використано DBSCAN. Кожен вузол ni виконує DBSCAN на своєму локальному наборі даних для створення локальних скупчень. Після визначення всіх локальних скупчень буде обчислено їх контури. Ці контури будуть використовуватися як репрезентативні зразки відповідних скупчень. Глобальні кластери генеруються під час другого етапу. Цей етап складається з двох основних кроків:

1. Кожен лідер збирає локальні кластери своїх сусідів;

2. Лідери об'єднують локальні кластери, використовуючи метод накладання. Процес злиття кластерів триватиме доти, поки не дійдено до кореневого вузла. Кореневий вузол міститиме глобальні кластери. На цій фазі здійснюється лише обмін межами скупчень.

Процес злиття складається з двох етапів: злиття меж та регенерації. Злиття виконується методом, що базується на межі. Нехай L_i бути локальною моделлю, отриманою з кластерів i та B_i - множина всіх меж у L_i.

Розглянемо основні особливості застосування алгоритму DBSCAN. DBSCAN (просторова кластеризація програм із шумом на основі щільності) - це добре відомий алгоритм кластеризації на основі щільності, здатний виявляти кластери довільної форми та усувати шумні дані [43]. Коротко кажучи, алгоритм кластеризації DBSCAN має два основних параметри: радіус *Eps* та мінімальні точки *MinPts*. Для даної точки p *Eps* околиця *p* є набором усіх точок навколо *p* на відстані *Eps*. Якщо кількість точок в Eps менше, ніж *MinPts*, то всі точки в цьому наборі, разом з *p*, належать одному кластеру. У порівнянні з іншими популярними методами кластеризації, такими як K-means та BIRCH, DBSCAN має кілька ключових особливостей. По-перше, він групує дані у кластери з довільними формами. По-друге, для цього не потрібно, щоб кількість кластерів було вказано як вхідні дані. Кількість кластерів визначається характером даних та значеннями *Eps* та *MinPts*. По-третє, він не чутливий до порядку введення точок у наборі даних. Усі ці особливості дуже важливі для будь-якого алгоритму кластеризації.

DBSCAN відвідує кожну точку набору даних, можливо, кілька разів (наприклад, у якості кандидатів у різні кластери). Однак з практичних міркувань часова складність в основному регулюється кількістю викликів. DBSCAN виконує рівно один такий запит для кожної точки, і якщо використовується структура індексації, яка виконує запит сусідства в O (log n), буде одержано загальну середню складність O (n log n), якщо параметр *Eps* вибрано в значущий спосіб (тобто такий, що в середньому повертаються лише точки O (log n)). Без використання структури прискореного індексу або вироджених даних (наприклад,

усіх точок на відстані менше *Eps)*, найгірший випадок складності часу залишається O (n2). Для того, щоб уникнути повторних обчислень відстані , використовується матриця відстані розміру O ((n2 n / 2)), але для цього потрібно O (n2) пам'яті, тоді як реалізація DBSCAN без використання матриці потребує лише O (n) пам'яті.

Рисунок 3.6 ілюструє приклад застосування DBSCAN в запропонованому підході. Припустимо, що розподілена система містить п’ять вузлів (N = 5). Кожен вузол виконує алгоритм DBSCAN зі своїми локальними параметрами (*Epsi, MinPtsi*) на своєму локальному наборі даних. Як видно на рисунку 3.6, запропонований метод повернув точно потрібну кількість кластерів та їх форми. Метод нечутливий до способу розподілу вихідних даних між вузлами. Він також нечутливий до шуму та викидів. Помітно, хоча кожен вузол виконує кластеризацію за алгоритмом DBSCAN локально з різними параметрами, глобальні кінцеві кластери є правильними, навіть на зашумленому наборі даних T5 (рисунок 3.6).

Рисунок 3.6 — Приклад виконання DBSCAN

Дослідимо ефективність запропонованого методу з використанням DBSCAN та продемонструємо його ефективність у порівнянні з BIRCH, CURE та K-means. Було обрано ці алгоритми, оскільки вони належать до тієї самої категорії, що і запропонований метод, наприклад BIRCH, що належить до ієрархічної категорії кластеризації, або мають ефективний метод до оптимізації, такий як CURE. Алгоритми CURE та BIRCH однакові з тими, що визначені та використовувались у попередньому розділі .

Було запущено три алгоритми на одній машині, Dell-XPS (1,8 ГГц * 4 Intel Core i5 з 8 Гб пам'яті) з Ubuntu (14,04 LTS) як ОС.

Було проведено експерименти з різними наборами даних. Використано вісім типів наборів даних з різними формами та розмірами. Перші три набори даних (T_1, vT_2, і T_3) - це ті самі набори даних, що використовувались для оцінки K-means. Останні п'ять наборів даних (T_4, T_5, T_6, T_7 і T_8) - дуже відомі орієнтири для оцінки алгоритмів кластеризації на основі щільності. Усі вісім наборів даних зведені в таблицю 3.2. Також вказано кількість точок та кластерів у кожному наборі даних. Ці вісім наборів даних містять набір фігур або візерунків, які непросто отримати за допомогою традиційних методів.

Буде запущено чотири алгоритми на восьми наборах даних, щоб оцінити якість їх кінцевих кластерів. Було застосовано розподілену систему, яка містить п'ять вузлів, отже, показані результати є агрегуванням п'яти локальних кластеризацій. На рисунку 3.7 показано повернуті кластери кожним з чотирьох алгоритмів опуклих форм кластерів (T1, T2, і T3), а на рисунку 5.8 показані скупчення, повернуті для невипуклих форм скупчень із шумом (T_4, T_5, T_6, T_7, і T_8). Було використано різні кольори, щоб показати кластери, повернені кожним алгоритмом.

Таблиця 3.2 – Набори даних, що використовуються для перевірки методу з використанням DBSCAN

Тип	Дані	Опис	Бали	Кластери
Опуклий	T1 T1 T2 T2	Великий овал (форма яйця) Великий овал (форма яйця) 4 маленьких кружечка і 2 маленьких кружечка і 2 маленьких кружечка	14000 14000 17080 17080	5 5 5 5
	T3 T3	2 маленькі кола, 1 велике коло і 2 зчеплених овалу і 2 зчеплених овалу	30 350 30 350	4 4
Неопукла Неопукла з шумом з шумом	T4 T4	Різні форми включаючи шум включаючи шум	8000 8000	6 6
	T5	Різні форми, с деякі скупчення оточені іншими	10 000	9
	T6	Листи з шумом	8000	6
	T7 T7	Різні форми включаючи шум включаючи шум	321 321	6 6
	T8	Різні форми із шумами	8000	6

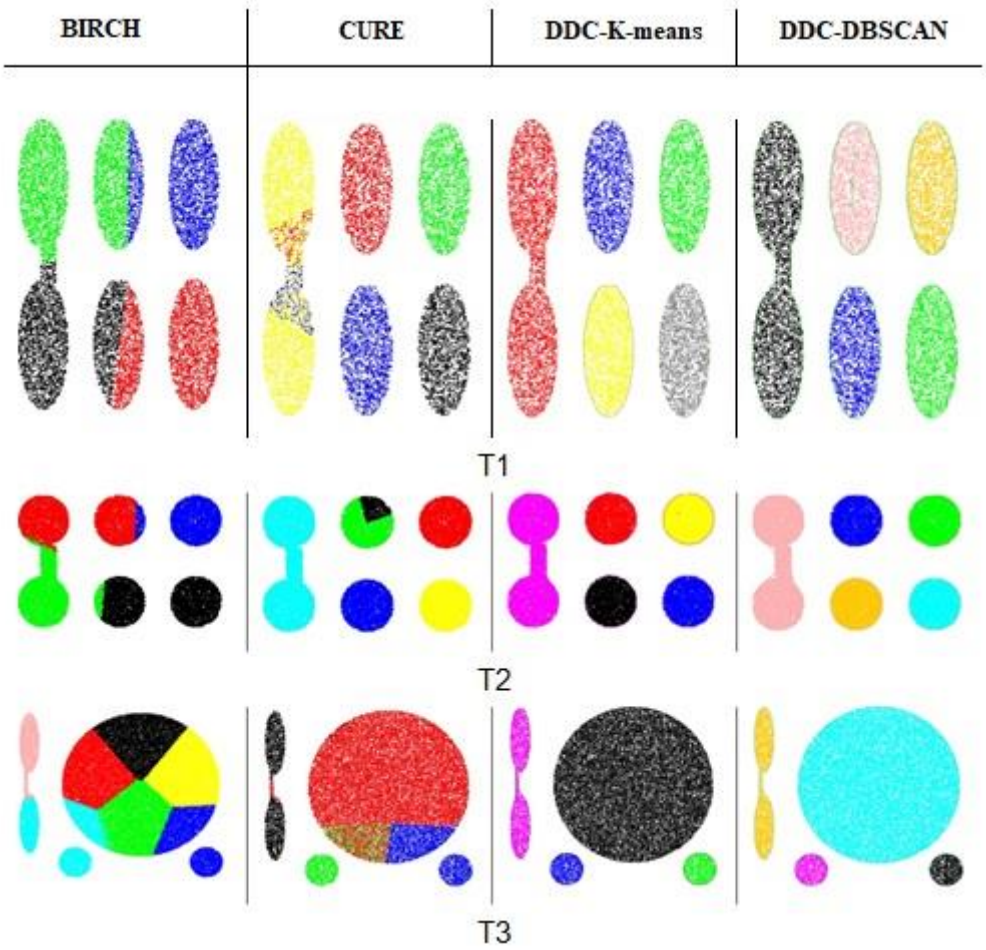


Рисунок 3.7 - Кластери, що генеруються з наборів даних опуклої форми

З результатів, показаних на рисунку 3.7, як і очікувалось, BIRCH не може правильно знайти всі кластери, він розбиває найбільший кластер, об'єднуючи інші.

На відміну від BIRCH, CURE правильно генерує більшість кінцевих кластерів, але все ще не вдається виявити всі кластери.

Тоді як алгоритми K-means та DBSCAN успішно генерують усі кластери із налаштуваннями параметрів за замовчуванням.

Рисунок 3.8 — Кластери, що генеруються з неопуклих фігур та зашумлених наборів даних

На рисунку 3.8 показані кластери, сформовані з наборів даних (T_4, T_5, T_6, T_7, і T_8). Як і слід було очікувати, BIRCH знову не зміг знайти правильні скупчення; він, як правило, краще працює з опуклими формами скупчень. Крім того, BIRCH погано кластеризує зачумлені дані. Результати CURE гірші, і він не в змозі виділити скупчення з неопуклими формами. Також було помічено, що CURE не справляється зі шумом. K-means не вдається знайти правильні кінцеві результати. Фактично він повертає весь вихідний набір даних як один остаточний кластер для кожного набору даних (T_4, T_5, T_6, T_7, і T_8). Це підтверджує, що запропонований

метод чутливий до типу алгоритму, обраного для етапу розподілення просторових даних між компонентами розподіленої динамічної системи. Оскільки другий етап стосується лише злиття локальних кластерів незалежно від того, правильні вони чи ні. Цю проблему виправляє DBSCAN, оскільки він добре підходить для неопуклих форм скупчень, а також для усунення шуму та викидів. Насправді це генерує хороші кінцеві кластери в наборах даних, які мають значну кількість шуму.

Як остаточне спостереження, ці результати доводять, що розроблений метод є дуже ефективним з точки зору точності результатів. Важливою задачею при цьому є вибір хорошого алгоритму кластеризації для етапу розподілення просторових даних між компонентами розподіленої динамічної системи. Це можна зробити, дослідивши початкові набори і відповідно вибрати алгоритм кластеризації.

Більше того, DBSCAN та K-means є динамічними, оскільки правильна кількість кластерів повертається автоматично.

Зазначу, що було проведено експерименти із застосуванням розроблено методу, які довели, що він є гнучким і здатним приймати будь-який локальний алгоритм кластеризації. В якості базових алгоритмів кластеризації було застосовано K-means та DBSCAN. Також було проведене порівняльне дослідження застосування розробленого підходу з двома відомими алгоритмами кластеризації, BIRCH та CURE. Порівняння проводилось за якістю сформованих кластерів. Було доведено, що як K-means, так і DBSCAN перевершують відомі алгоритми з точки зору якості кластеризації. Також було обчислено часову складність K-means, та DBSCAN. Було виявлено, що K-means має кращу часову складність (лінійну) порівняно з DBSCAN (квадратичну), але останній алгоритм може бути застосований на зачумлених даних, тоді як K-means дуже чутливий до шуму та до неопуклої форми даних.

3.4 Розподілена кластеризація із використанням MapReduce

Розглянемо застосування розробленого методу із використанням парадигми програмування MapReduce, використовуючи розподілену структуру JADE на середніх наборах даних, що працюють на кластері комп'ютерів. MapReduce – це модель, розроблена для роботи з дуже великими розподіленими наборами даних, що працюють на дуже великих розподілених системах, таких як інфраструктура хмарних обчислень. Реалізація цього методу з використанням цієї парадигми дозволить показати поведінку та можливості розробленого методу у роботі з дуже великими наборами даних та забезпечити прийнятний час виконання без втрати якості остаточних результатів.

Як було зазначено вище, розроблений метод кластеризації для видобутку просторових даних складається з двох основних етапів: 1) локальної моделі, де буде генеровано локальну кластеризацію з подальшим вилученням контурів з цієї локальної кластеризації; 2) глобальної моделі, де буде об'єднано перекриття локальних контурів для отримання глобальних результатів. PK-MR також має два етапи, 1) етап відображення та (2) етап відновлення. Картографи виконують алгоритм кластеризації (вибравши алгоритм DBSCAN), а потім застосовують контурний алгоритм для вилучення репрезентативних точок з локальної кластеризації. Другий етап складається з декількох редукторів для виконання злиття між контурами локальної кластеризації, попередньо сформованої на етапі відображення. Етапи картографування та зменшення виконуються паралельно. Іншим словом, картографи обчислюють всі кластери одночасно на першому рівні. Після генерації локальних кластерів картографа обчислює контури, що також виконується паралельно між усіма картографами. Редуктори роблять етап злиття, який також виконується паралельно.

PK-MR обробляє дані розподіленим способом, HDFS виділить передавати дані випадковим чином до різних вузлів (різних процесорів). На відміну від фреймворку JADE, використовуваного в попередньому розділі (глава 6), у PK-MR

не потрібно ділити дані вручну, розподіленою функцією алгоритму керує сама система Hadoop, і мій метод не має нічого спільного з даними розподіл.

Картографів повністю незалежні від редукторів, оскільки вбудовані редуктори є контурами локальних кластерів (результати картографувань). Отже, це дає свободу застосовувати різні алгоритми кластеризації, не впливаючи на фазу зменшення.

3.3.1 Фаза картографування

Початковим кроком картографа є отримання набору даних, який виділяється у його вузлі обробки шляхом опитування файлової системи. Це обробляється та зберігається у файлах HDFS. Формат даних стандартизований функцією ReadHDFSFile та функцією StandardiseData. Функція ReadHDFS-File () вимагає зазначення шаблону файлу.

Ця інформація надається за допомогою "ідентифікатора рядка", а функція StandardiseData нормалізує дані.

Основним кроком картографа є виконання алгоритму кластеризації. Кожен маппер запускає алгоритм DBSCAN зі своїми конкретними параметрами (Eps, MinPts). Після генерації локальних кластерів кожен вузол виконує контурний алгоритм на своїх локальних кластерах.

Картограф розподіляє ключі до сформованих контурів. Ключ обраний однаковим для всіх картографів (ключ = 2) для спрощення фази зменшення.

Останнім кроком етапу відображення є запис результату у файл HDFS, який зчитує редуктор на наступній фазі. Контури записуються у стандартизованому форматі, кожен рядок представляє контур і має два стовпці: перший - це ключ контуру, а другий - для координат точок.

Псевдокод для фази відображення наведено в алгоритмі 3.3.

Input : X_i:List<lineNo, Point>, Eps_i:ℝ, $MinPts_i$:ℕ

Output: List<key, List<Contour>>

1 <key, List<Contour>>

2 $L_i \leftarrow$ DBSCAN(X_i, Eps_i, $MinPts_i$); // Local clusters generated by $Node_i$

3 $C_i \leftarrow [\,]$;

// For each cluster

4 **foreach** $L_i^k \in L_i$ **do**

5 $c^k \leftarrow$ ComputeContour(L_i^k);

6 Append(C_i, c^k) // add contour to the list

7 **return** <key, C_i >;

Алгоритм 3.3 – Фаза відображення

Вхід фази зменшення - це стандартизований вихід з фази відображення, який складається з пар (ключ, контур) локальних кластерів. Основним кроком редуктора є вибір стратегії злиття. Встановивши однаковий ключ для всіх картографів на етапі відображення. Тому кожен контур кожного вузла порівнюється з усіма контурами всіх інших вузлів. Якщо два контури перекриваються, то вони будуть об’єднані, щоб сформувати новий контур. Редуктор відповідає за злиття контурів, що перекриваються, і має на меті побудувати ітеративно більші кластери до досягнення кінцевого результату.

Вважається, що контур cj перекриває інший контур ck тоді і тільки тоді, коли багатогранник (P (cj)), що отримується з балів, що складають cj перетинає багатогранник (P (ck)), що отримується з балів, що складають ck. Багатокутник - це багатокутник у N-мірному просторі.

Злиття двох контурів cj та ck дає ще один контур cjk що є об'єднанням двох багатогранників, що утворюють cj та ck.

Етап відновлення виконується паралельно і розподілено. Він реалізований за допомогою деревної структури. На кожному рівні ієрархії половина вузлів

надсилатиме свої контури своїм сусідам. Вони виконують злиття алгоритм на кластерах тощо. Hadoop опікується управлінням та контролем комунікацій. Алгоритм відновної фази наведено в алгоритмі 3.4.

Input : $\mathcal{C}$:List<key, List<Contour>>
Output: <key, List<Contour>>

```
1  C_res ← [ ];
2  foreach <key, C_i > ∈ C do
3      foreach c^j ∈ C_i do
4          merged ← False;
5          foreach c^k ∈ C_res do
6              if Overlap(c^j, c^k) then
7                  c^jk ← Merge(c^k, c^j);
8                  Remove(C_res, c^k); // remove contour from the
                       list
9                  Append(C_res, c^jk); // add contour to the list
10                 merged ← True;
11         if not(merged) then
12             Append(C_res, c^j); // add contour to the list
13 return <key, C_res >;
```

Алгоритм 3.4 – Відновлення фази

3.5 Обчислювальна складність розподіленої кластеризації

У цьому розділі обчислено теоретичну складність нашого підходу. Нехай M це кількість вузлів і n_i набору даних, що надається кожному вузлу υ_i в системі.Складність нашого методу - це сума складності його компонентів;місцевий видобуток, локальне скорочення та глобальне агрегування.

Етап 1 - Локальна кластеризація: Нехай $\Gamma(n_i)$ позначає алгоритм локальної кластеризаціїпрацює на вузлі υ_i, а $\Delta(c_i)$- час, необхідний для виконання скорочення алгоритму (контурний алгоритм). Вартість цієї фази визначається:

$$T_{phase_1} = \underset{i=1}{\overset{M}{\text{Max}}}(\Gamma(n_i) + \Delta(c_i)), \tag{3.4}$$

де c_i- точки кластера, породжені вузлом υ_i c_i може бути апроксимовано: $c_i \cong \frac{N}{Mk_i}$, де N - загальний набір даних, M - число вузлів у системі, а k_i- кількість кластерів, що генеруються вузлом υ_i. Складність алгоритму скорочення становить $o(c_i \log c_i)$ (див. Розділ 2 .4) ,що становить $o(\frac{N}{Mk_i}) \log \frac{N}{Mk_i}$. Оскільки M і k_i фіксовані, отже, в гіршому випадку,складність дорівнює $o(N \log N)$.

Етап 2 - Агрегація: Агрегація локальних кластерів залежить відієрархічне поєднання контурів локальних скупчень. У міру злиттябудь-які два скупчення засновані на перетині їх контурів, складностіцієї фази $o(\omega_i \log \omega_i + p)$ Де ω_i- загальна кількість вершинконтурів, а p - кількість точок перетину між заданими контурами (багатокутниками). ω_i можна приблизно розрахувати $\omega_i \cong \frac{N}{Mk_i}\beta$, де $\beta \in [0-1]$ - це відсоток, який представляє контурні точки. Експериментально встановлено, що β = 0,02 розміру кластера.Загальна складність: Загальна складність розробленого методу становить:

$$T_{\text{Total}} = o(\Gamma(n_i)) + o(c_i \log c_i) + o(\omega_i \log \omega_i + p) \tag{3.5}$$

$$T_{Total} = o(\Gamma(n_i)) + o\left[\frac{N}{Mk_i} \log \frac{N}{Mk_i}\right] + o\left[(\frac{N}{Mk_i}\beta) \log(\frac{N}{Mk_i}) + p\right] \cong$$

$$\cong o(\Gamma(n_i)) + o(N \log N) \tag{3.6}$$

де $o(\Gamma(n_i))$ – часова складність алгоритму локальної кластеризації.

3.6 Висновки

У третьому розділі представлено удосконалений метод розподіленої динамічної кластеризації для видобутку просторових даних, який ґрунтується на запропонованій моделі розподіленої динамічної кластеризації для видобутку просторових даних. Також було оцінено обчислювану складність розподіленої кластеризації. Проведені обчислення показали, що розроблений метод дозволяє підвищити ефективність кластеризації просторових даних.

4 ВПРОВАДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ РОЗПОДІЛЕНОЇ ДИНАМІЧНОЇ СИСТЕМИ ДЛЯ ВИДОБУТКУ ПРОСТОРОВИХ ДАНИХ

Основною ідеєю підходу РК є мінімізація обміну даними при цьому максимізуючи якість глобальних кластерів. Метод, що використовується для агрегування об'єднання просторових локальних кластерів у глобальні кластери дозволяє обмінятися близько 2% від оригінальних наборів даних [44], що є високоефективним. Структура РК генерує дуже високо якісну кластеризацію та долає недоліки існуючих програм. У цій главі вивчається результативність такого розподіленого підходу кластеризації з точки зору часу виконання, ефекту щодо часу відгуку, використовуваної моделі зв'язку, масштабованості та нарешті, його пришвидшення порівняно з послідовною версією. Зазвичай розподілені системи мають ту перевагу, що надають їм вузли за дуже низькою вартістю, однак часто їх вузли є ерогенні, означає, що вони можуть мати різну обробну потужність. В ідеальній ситуації слід призначати навантаження вузлам і також відповідно підключаючись до їх обчислювальної потужності, щоб вони могли закінчити обчислення фази більш-менш одночасно (з мінімальним часом очікування), перш ніж вони розпочнуть фазу спілкування. Однак реальність зовсім інша. Дані, зібрані на кожному вузлі, можуть мати будь-який розмір, який не залежить від його потужності.

У процесі, коли обчислення супроводжуються комунікаціями, як це має місце у підході РК, деякі вузли можуть закінчуватися. Їх обчислення (локальна кластеризація) набагато раніше за інші. Якщо ми хочемо розпочати комунікацію (обмінятися представниками місцевих кластерів), вузлам, які закінчили свої обчислень набагато раніше доведеться чекати, поки всі інші вузли не закінчать їх обчислення. Це може виявитися дуже дорогим, коли ці дії здійснюються у високопродуктивних мережах. Тому ми вивчаємо два типи комунікаційних моделей: синхронні та асинхронні. При синхронному зв'язку існує чітке розділення між ними: фаза обчислень і фаза зв'язку. Ми не починаємо комунікації, поки не закінчаться всі обчислення. Тоді як з асинхронним зв'язком,

існує перекриття між обчисленнями та катіонні фази. Вузли, які вже закінчили свої обчислення, можуть ініціювати обмін представниками своїх місцевих кластерів. У цьому дослідженні увагу було зосереджено на:

1. Типах використовуваних комунікацій;

2. Синхронних та асинхронних комунікаціях. Як зазначалося вище, це матиме більший вплив, коли зв'язок швидкий. Для дуже повільних комунікацій перевага перекриття між обчисленнями та зв'язками.

3.Пришвидшення підходу РК із використанням DBSCAN як базового алгоритму для кластеризації локальних розділів. Алгоритм DBSCAN відомий тим, що має високу складність $o(n^2)$

4. Під час масштабування підходу РК кількість вузлів у системі зростає (використовуючи як K-means, так і DBSCAN як локальну кластеризацію).

4.1 Методологія оцінки ефективності

Методологія походить від головної цілі дослідження, яка є розроблення струкртури кластеризації видобутку даних для великих наборів даних (або великих данів). Основними критеріями є:

1. Розподілена системна платформи, що складається з різнорідних вузлів обробки.

2. Розробка програмного забезпечення, а саме платформи, яка може допомогти в реалізації фреймворку

3. Мережевий протокол та характеристики для оцінки впливу комунікацій на загальну продуктивність фреймворку і, нарешті, дуже добрі орієнтири для валідації дати експериментальних результатів . Для апаратної платформи оберемо групу різнорідних машин з різними обробними можливостями, різними обсягами пам’яті та різними ОС. Оскільки є зацікавленість в оцінці продуктивності програми РКproach, використаємо відкриту платформу для розробки програмного забезпечення під назвою JADE(Java Agent DEvelopment) для реалізації та тестування алгоритму РК за допомогоюяк K-means, так і DBSCAN. Зараз є

наміри оцінити підхід на основі деяких прискорених критеріїв,масштабованість та час відгуку алгоритмів з перекриттям та без нього між фазою обчислення та фазою зв'язку.Крім того, ми порівняли РК (K-засоби та DBSCAN) з деякими існуючими та популярними алгоритмами кластеризації. Зверніть увагу, що ці алгоритми були реалізовано з використанням тієї самої платформи за однакових умов. Це дозволяє уникнути деяких накладних витрат.

4.1.1 Час відповіді РК - Випадок I

Мета тут - вивчити час виконання підходу РК з використаннямяк K- Means, так і DBSCAN, і порівнюючи їх із алгоритмом BRECH і CURE, порівнюємо чотири алгоритми та продемончтруємо вплив використання паралельної та розподіленої архітектура для вирішення обмежених можливостей централізованої системи. Було використано шість наборів даних, які є одними з тих, що використовуються для тестування якості кластеризації в попередньому розділі . DBSCAN може обчислити час виконання DB-алгоритму двома способами. Перший спосіб - це включення часу, необхідного для формування розрахунку матриці відстані. Другий спосіб припускає, що матриця відстані вже сформована. Також не забуваємо, що матриця відстані обчислюється лише один раз. Зверніть увагу, що цей підхід у цьому випадку виконується на одній машині. Зв'язок між процесами здійснюється по одній і тій же машині. Тому ми не враховуємо вартість мережі.

4.1: Час виконання (мс) BIRCH, CURE, РК-K та РК-DBSCAN з (w) і без (w / o) обчислення матриці відстані

Таблиця 4 .1 ілюструє час виконання чотирьох методів на різних наборах даних

Набір даних		Execution Time (ms)				
	SIZE	BIRCH	CURE	K-Means	DBSCAN	
					W	W/O
T1	14,000	329	145,682	291	1,046	642
T2	17,080	311	405,445	338	1,284	824
T3	30,350	337	1,238,163	502	1,904	1,223
T4	8,000	235	75,098	251	643	347
T5	10,000	252	147,844	271	846	472
T6	8,000	214	93,450	235	609	378

Зверніть увагу, що час виконання не включає час післяоброблюваний час, оскільки вони однакові для чотирьох алгоритмів.

Також таблиця 4.1 підтвердила факт, що обчислення матриці відстані в DBSCAN є дуже значним. Більше того, час виконання PK-DBSCAN набагато менший, ніж час виконання CURE через шість наборів даних. Таблиця 4. 1 ще показує, що PK-K-means мають дуже велику швидкість, що відповідає його лінійній обчислювальній складності. BIRCH також дуже швидкий, проте якість

його результатів не є хорошою, йому не вдалося знайти правильні кластери у всіх шести наборах даних .PK-DBSCAN трохи повільніший за PK-K-Means, але він повертаєтьсяякісні результати для всіх тестів набагато кращі, ніж PK-K-Means, що має досить добрі результати для опуклих форм скупчень і дуже погані результати для неопуклих форм кластерів. Загальні результати підтверджують, що кластеризація PK-DBSCAN вигідно порівнюється з усіма випробуваними алгоритмами для комбінованого показника ефективності (якість результатів та час відгуку або комбінація часу сплетіння).

Надалі визначимо ефективність підходу PK відповідно розподіленій системі через підключені машини через мережу та з урахуванням мережевих платежів. Запропонований підхід є більш детальним для розподілених систем, ніж для паралельних систем. Тому також варто проаналізувати переваги використання синхронних або асинхронних процесів, оскільки розподілені системи є асинхронними та блокуючими. Через це операції мають сильний вплив у часі спілкування [45].

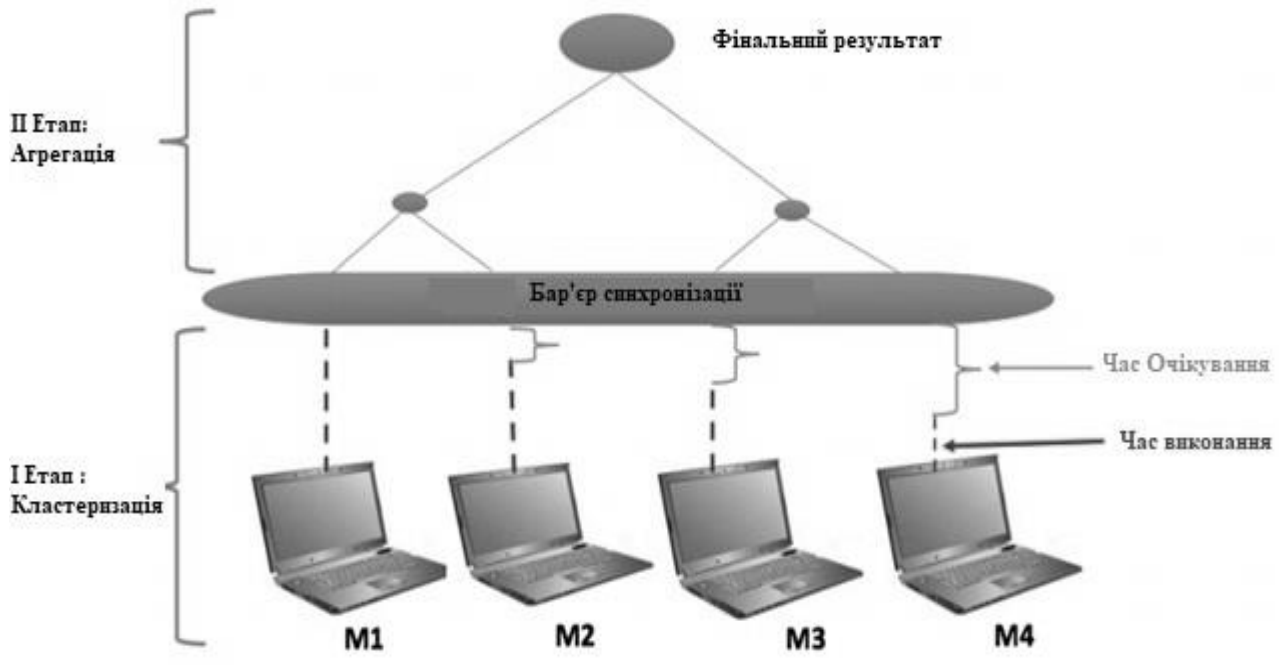


Рисунок 4.1 – Синхронні комунікації

У синхронній моделі, як показано на малюнку 4.1, хоча машини M_3 і M_4 і закінчили свої обчислення до M_1 і M_2 , вони не можуть надсилати свої результати, поки M_1 і M_2 також не закінчать. У цій моделі не тільки обчислення та комунікації не збігаються, але й машина, яка закінчила рано, витратила трохи часу, чекаючи поки інші закінчать [45] .

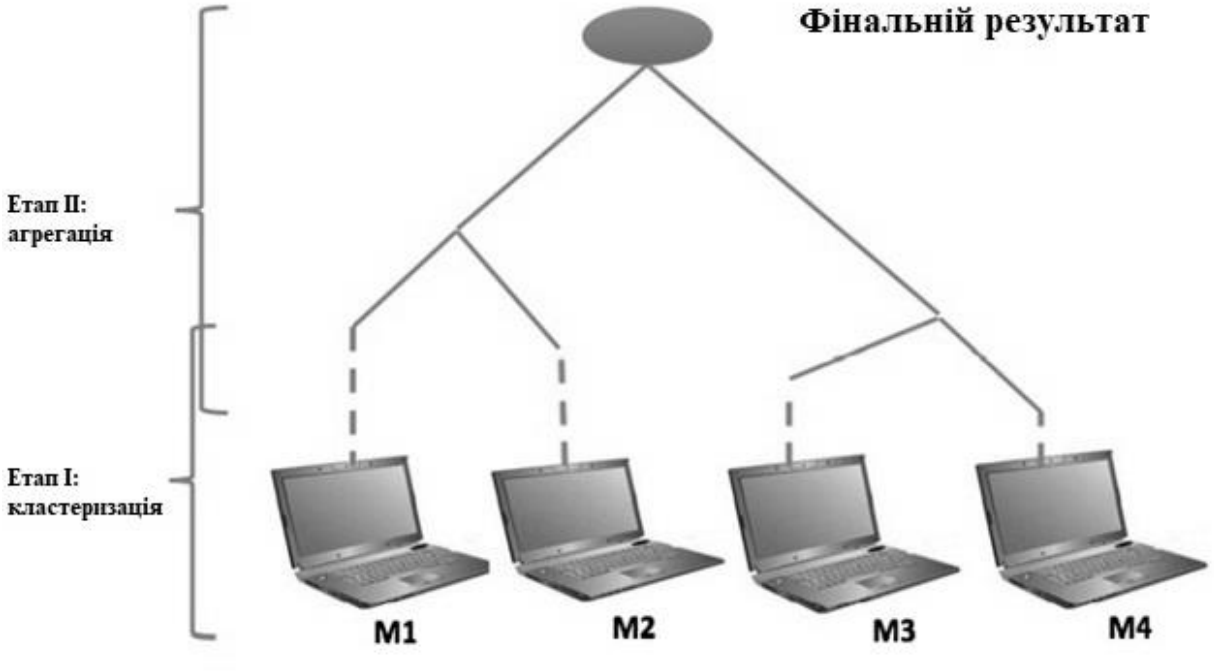


Рисунок 4.2 - Асинхронні комунікації

В асинхронній моделі машини, які закінчили достроково, можуть очікувати до наступного кроку. Машини управляють своїми комунікаціями, під час чого перша та друга фази перекриваються. Ця модель набагато більше підходить для обчислення, де вузли неоднорідні, а комунікації зазвичай повільні. Як видно на прикладі, наведеному на рисунку 4.2, M_3 і M_4 починають зливати свої результати до того, як M_1 і M_2 закінчать свої обчислення.

Прискорення РК розраховується відповідно до послідовної версії додатка. Послідовна версія складається з кластеризації всіх даних на одному компоненті.

Тому він не вимагає ні зменшення, ні агрегування. Нехай T_1 - час виконання послідовної версії і T_p час виконання РК на p вузлах. Прискорення α визначається як $\alpha = \frac{T_1}{T_p}$. Зверніть увагу, що якщо складність алгоритму кластеризації є поліномною, то діоптимального прискорення можна досягти (p), за умови, що не існує накладних витрат через комунікації та додаткову роботу.

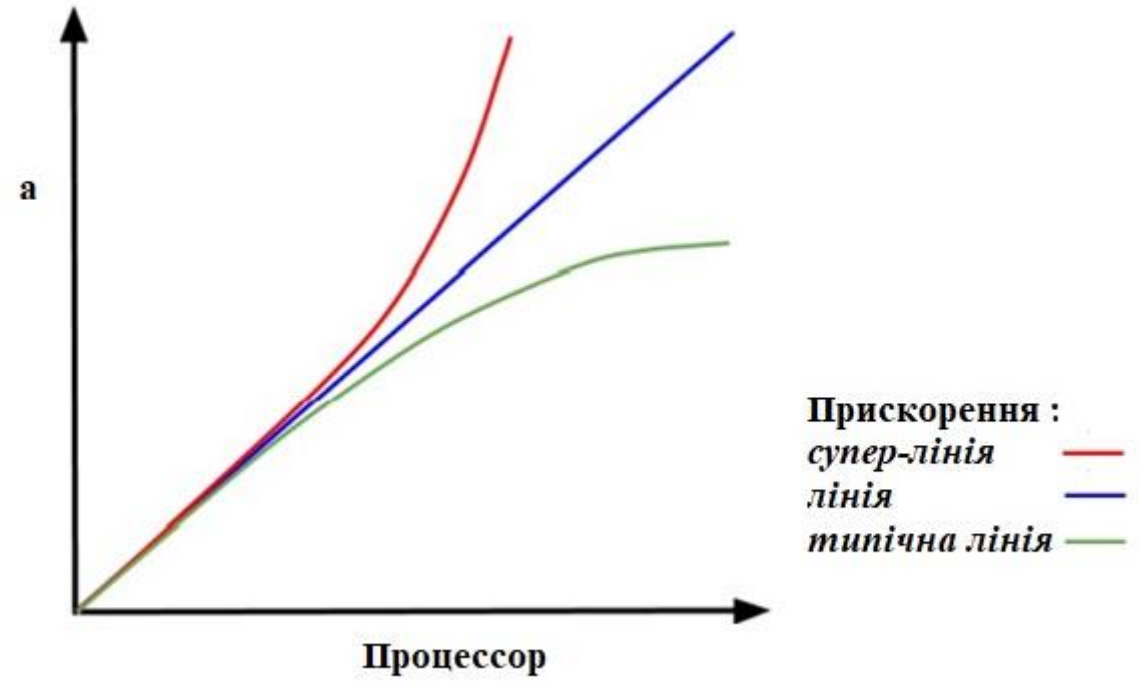


Рисунок 4.3: Прискорення

Якщо складність алгоритму кластеризації - $o(n^2)$, тоді оптимальним прискоренням може бути p^2; це називається супер прискоренням (див. рисунок 4.3).

Існує два типи експериментів із масштабованістю, що враховують масштабованість даних, в яких досліджуємо, як на час виконання впливає збільшення розміру набору даних та масштабованість відповідно до кількості вузлів у системі, де розглядається , як на час виконання впливає збільшенню у кількості вузлів у системі за допомогою набору даних фіксованого розміру. Перший можна передбачити за складністю алгоритму. Отже, через

масштабованість підходу РК треба дізнатись яка кількість вузлів у системі збільшується, що дасть уявлення про те, яка оптимальна кількість вузлів, які нам потрібні для певного розміру набору даних, щоб час відгуку становив мінімум. Для цього було вивчено два випадки. У першому випадку вивчено можливість підходу РК на одній машині (без плати за мережу). Для цього випадку було взято як K-Means, так і DBSCAN як локальну кластеризацію. Внаслідок обчислювальної складність алгоритму DBSCAN та необхідності завантаження при вкладанні набору даних у локальну пам’ять, використовуємо максимальний розмір набору даних 50 000 точок даних для перевірки масштабованості РК-DBSCAN в одній машині одночасно збільшуючи кількість процесів. Тоді як для РК-K-Means через її лінійну складність ми взяли для перевірки більший набір даних (1 000 000 точок даних). У другому випадку вивчалась масштабованість підходу РК у реальній розподіленій системі (з урахуванням мережевих платежів). Тут розглядається, як реагує час виконання до збільшення кількості вузлів у системі. У цьому випадку було взято алгоритм DBSCAN як локальний алгоритм кластеризації, і використано два набори данихз різними розмірами. Мета цього дослідження - знайти оптимальну кількість вузлів, необхідних для певного розміру набору даних. Для того, щоб збалансувати дані розподілу під час вивчення масштабованості підходу РК для обох випадків згаданих вище, було розділено дані випадковим чином і розподілено їх порівну серед вузлів у системі.

4.2 Результати проведення експериментів

Цей підхід був застосований у розподіленій обчислювальній системі. Розподілена обчислювальна система складається з різнорідних робочих столів (різних ЦП, ОС, обсягів пам'яті, навантаження тощо). Використовуємо JADE як платформу для розробки форми для реалізації підходу. JADE базується на рівному рівні (P2P) архітектури зв'язку. Обрана ця структуру для її виконання:

1. Неоднорідність: система, заснована на JADE, може бути розподілена по всіх машинах (для яких навіть не потрібно використовувати одну і ту ж ОС) [83] .

2. Масштабованість: JADE масштабується лінійно, і це хороший кандидат для розвитку розподілених додатків із великим навантаженням [83] .

3. Зручний для користувача: конфігурацією Jade можна керувати за допомогою віддаленого графічного інтерфейсу,що полегшує користування [84] .

4. Динамічний: конфігурацію Jade можна навіть змінити під час виконання переміщення агентів з однієї машини на іншу, як і коли потрібно[84] .

5. Мінімальні вимоги: JADE повністю реалізовано на Java-мові та з мінімальною вимогою до системи є версія 5 JAVA (середовище виконання або JDK) [87] .

Вузли системи (робочі столи) підключені до локальних мереж. Це дозволяє додавати стільки вузлів, скільки потрібно, залежно від експерименту.У таблиці 4.2 перелічено типи машин, що використовуються для проведення експериментів. Головною метою тут є продемонструвати ефективність РК у різнорідних розподілених обчислювальних середовищах.

Таблиця 4.2 – Характеристики використовуваних машин

Назва машини	Операційна система	Процесор	Пам'ять
Dell-XPS L421X	Ubuntu	1,8 ГГц * 4	8 ГБ
Dell-XPS L421X	(V.14.04 LTS)	Intel Core i5	8 ГБ
	(V.14.04 LTS) Ubuntu	Intel Core i5 2,00 ГГц * 4	
Dell-Inspiron-3721			4ГБ
Dell-Inspiron-3721	(V.14.04 LTS)	Intel Core i5	4ГБ
	(V.14.04 LTS) Ubuntu	Intel Core i5 1,8 ГГц * 4	
Dell-Inspiron-3521			6 ГБ
Dell-Inspiron-3521	(V.16.04 LTS)	Intel Core i5	6 ГБ
	(V.16.04 LTS) Монетний двір Linux	Intel Core i5 3,06 ГГц * 2	
iMac - Early 2010			4ГБ
iMac - Early 2010	(V.17.1 Ребекка)	3,06 ГГц * 2	4ГБ
	(V.17.1 Ребекка) Ubuntu		

Продовження таблиці 4.2 – Характеристики використовуваних машин

Назва машини	Операційна система	Процесор	Пам'ять
	(V.16.04 LTS)	Intel Core i5	
Dell-Inspiron-5559	Ubuntu (V.16.04 LTS)	2,30 ГГц * 4 Intel Core i5	8 ГБ
iMac - Early 2009	OS X El Capitan (V.10.11.6)	2,93 * 2 ГГц Intel Core Due	8 ГБ
MacBook Air	OS X El Capitan (V.10.11.3)	1,6 * 2 ГГц Intel Core i5	8 ГБ

РК тестується з використанням різних розділів різних розмірів. Виходячи з цілей експериментів, були створені різні сценарії. Ці сценарії в основному різняться способом розподілу наборів даних між вузлами передачі розподіленої платформи. Для кожного сценарію було записано час виконання локальної кластеризації (Крок 1), включаючи крок злиття контурних розрахунків, час агрегування та час простою (Крок 2). Нарешті, було вказано також загальний час виконання підходу, необхідний для завершення всіх етапів. Далі буде описано різні розглянуті сценарії.

У цьому сценарії надано кожній машині випадковий шматок набору даних. Розмір розділу, створеного для кожної машини, знаходиться в діапазоні між 500 балів та 10 000 балів. Оскільки набір даних відносно невеликий, ми вибрали вісім машин для обчислювальної платформи. У таблиці 4.3 наведено час виконання кожної машини для запуску ритму (крок 1 і крок 2) з використанням синхронної та асинхронної комунікації катіонів відповідно. Він також показує загальний час,

необхідний для завершення всіх етапів.Зверніть увагу, що у разі синхронного зв'язку злиття здійснюється за двійкове дерево. З таблиці 4.3 видно, що час, необхідний кожній машині, щоб діяти першому кроку алгоритму однаковий як для синхронного, так і для асинхронного виконання, тоді як час другого кроку інший. Також звернемо увагу, що кожна машина повертає різний час виконання всього ритму. Це тому, що машини мають різну потужність (табл. 4.2) .

Загальний час виконання алгоритму при використанні асинхронного зв'язку менший у порівнянні з використанням синхронного зв'язку.

Це пов’язано з тим, що при синхронному спілкуванні машини мають більший час очікування (до 60% часу очікування).

У цьому сценарії розподіляємо весь розмір набору даних на одній машині а іншим машинам було виділено по одній вісімці набору даних кожній. Цей сценарій вибрано, щоб показати найгірший випадок часу очікування.

У таблиці 6.4 наведено час виконання, який виконується кожною машиною з методикою РК (Крок 1 і Крок 2) з використанням синхронного та асинхронного зв'язку відповідно. Він також показує загальний час, необхідний для закінчення всіх кроків. Зверніть увагу, що у випадку синхронного зв'язку злиття відбувається під бінарним деревом.

З таблиці 4.4 можна помітити, що різниця між виконанням часів синхронного та асинхронного РК все ще залишається значним.

При синхронному зв'язку машини повинні завершити свій перший крок до того, як усі вони почнуть зливати свої результати (крок 2), тоді як для асинхронної моделі сім машин здійснюють злиття (крок 2), поки остання машина закінчує кластеризацію (крок 1).

Звернемо увагу, що машина М3 витрачає найдовший час обробки (20 100 мс), щоб закінчити крок 2 з використання асинхронного зв'язку. Причина полягає в тому, що машина М3 є найповільнішою серед машин системи. Більше того, у цьому випадку М3 повинна зачекати, поки машина М1 закінчить свій перший крок, щоб зробити злиття.

Таблиця 4.3 - Час (мс), витрачений вісьмома машинами на запуск сценарію I із використанням синхронізації синхронних та асинхронних коммунікацій

		Синхронний			Асинхронний		
Машина	Розмір набору даних	Крок 1	Крок 2	Час	Крок 1	Крок 2	Час
M1	10 000	21 270	1 104	22 374	21 270	554	21 824
M2	2500	1060	20 862	21 922	1060	2515	3,575
M3	3275	5093	16 930	22 023	5093	2017	7110
M4	5000	4,592	17 644	22 236	4,591	2620	7211
M5	1666	227	21 642	21 869	227	391	618
M6	2000	292	21 736	22 028	292	416	708
M7	5000	7520	14 665	22 185	7515	13 949	21 464
M8	1500	200	21 842	22 042	195	4 605	4800
		Загальний час виконання		22 374	Загальний час виконання		21 824

У цьому сценарії виділяємо для семи машин цілий і останній набір даних, рівний одній восьмій з набору даних. Цей сценарій обраний для демонстрування впливу складності локальної кластеризації на машини та на час очікування деяких потужних машин.У таблиці 4.5 наведено час виконання, необхідний кожній машині для запуску алгоритму (Крок 1 і Крок 2) з використанням синхронних та асинхронних комунікацій.

Таблиця 4.4 - Час (мс), витрачений вісьма машинами для запуску сценарію II із використанням синхронізації синхронних та асинхронних коммунікацій

Параметри експерименту		Синхронний			Асинхронний		
Машина	Розмір набору даних	Крок 1	Крок 2	Час	Крок 1	Крок 2	Час
M1	10 000	21 270	973	22 243	21 270	595	21 865
M2	1250	215	21 775	21 990	215	518	733
M3	1250	640	21383	22 023	640	20 100	20 740
M4	1250	304	21 730	22 034	304	497	801
M5	1250	161	22 034	22 195	161	394	555
M6	1250	171	21 856	22 027	170	286	456
M7	1250	245	21 918	22 163	245	509	754
M8	1250	185	21 854	22 039	185	858	1043
		Загальний час виконання	22 243	Загальний час виконання	21 865		

Таблиця показує загальний час, необхідний для закінчення всіх кроків. Зверніть увагу, що у разі синхронного зв'язку злиття проводиться під бінарним деревом.

Цей сценарій протилежний попередньому.

На відміну від попереднього Таблиця 4.5 показує, що різниця між часом виконанням синхронних та асинхронних версій РК є меншою.

В обох випадках машини витрачають більше часу на завершення першого кроку.

Отже, час очікування менший для синхронної моделі, ніж для асинхронної. Також можна помітити, що у М3 є найбільший час для кроку 2, яким це обумовленочас очікування, оскільки М3 потрібно почекати, поки машина М1 закінчить свій крок 1,тоді М1 об'єднує свої результати з М3 4.2.4

У цьому сценарії було враховано можливості машин і розділено набори даних відповідно до їх можливостей. Тому навантаження рівномірно розподілено між ними, і очікуємо, що вони закінчать перший етап більш скорішим темпом.

Це дозволяє скоротити час очікування машині і негайно перейти до другої фази.

Загальний час виконання синхронно та асинхронної версії повинні бути подібними.

Ця справа сприяє більш синхронному підходу.

Як і передбачалося, таблиця 4.6 показує, що немає суттєвої різниці міжміж двома часами виконання. Зверніть увагу, що невелика різниця на користьсинхронна версія пов'язана з тим, що в асинхронній моделімашинам все ще потрібно виконати алгоритм, який перевіряє, який із них закінченийспочатку і отримаємо контури для злиття.

Ефективне прискоренняМета тут - порівняти нашу паралельну кластеризацію з послідовним алго-ритм і показати прискорення РК над послідовною версією кластеризації,як зазначено у рівнянні 4.1.

Найкращий сценарій запуску послідовної версії DBSCAN

Таблиця 4.5 - Час (мс), витрачений вісмома машинами для запуску сценарію III із використанням синхронних та асинхронних коммунікацій.

Параметри експерименту		Синхронний			Асинхронний		
Машина	Розмір набору даних	Крок 1	Крок 2	Час	Крок 1	Крок 2	Час
M1	10 000	21 270	35978	57 248	21 270	905	22 175
M2	10 000	21 590	34 869	56 459	21 590	11513	33 103
M3	10 000	53 005	3008	56 013	53 005	3292	56 297
M4	10 000	32 424	24 691	57 115	32 424	6,996	39 420
M5	10 000	17 364	38 493	55857	17 364	4612	21 976
M6	10 000	15841	41 237	57 078	15841	2066	17 907
M7	10 000	38 732	18 483	57 215	38727	18 459	57186
M8	1250	185	56 915	57 100	184	16 077	16 261
		Загальний час виконання		57 248	Загальний час виконання		57186

Таблиця 4.6: -Час (мс), витрачений вісьма машинами для запуску сценарію IV із використанням синхронних та асинхронних коммунікацій.

Параметри експерименту		Синхронний			Асинхронний		
Машина	Розмір набору даних	Крок 1	Крок 2	Час	Крок 1	Крок 2	Час
M1	1500	256	1,505	1761	256	1159	1415
M2	1660	260	598	858	260	1512	1772
M3	500	252	1061	1313	252	626	878
M4	1000	253	621	874	253	608	861
M5	1500	255	1492	1747	255	600	855
M6	1400	260	605	865	260	514	774
M7	1000	259	1030	1,289	259	939	1,198
M8	1500	250	603	853	250	1500	1750
		Загальний час виконання		1761	Загальний час виконання		1772

Помітно, що найшвидша машина в системі T1 = 15841 мс. Кластеризація розділу одного і того ж набору даних на одній машині займе T d1= 258 мс.

Втрата часу РК на однакових наборах даних на восьми неоднорідних машинах збалансує навантаження T_p = 1761 мс (табл. 4.6).

Тому з Рівняння 4.1очевидно, що пришвидшення все ще є з суперлінійним прискоренням.

Далі буде приведено, скільки вузлів обробки потрібно, щоб кластеризувати набір даних розміром N.

У цьому розділі йде дослідження масштабованості при підходу РК, беручи два випадки.

У першому випадку досліджується масштабованість підходу без включення мережевого заряду, а у другому випадку масштабованість підходу в реальній розподіленій системі, включаючи мережеві платежі.

Ці два випадки докладно описані нижче.

Мета тут - визначити вплив кількості вузлів у системіна час виконання підходу РК. В одній іх машин працює система без мережевих платежів.

Використано два набори даних, які містять D_1 - 150 000 точок даних, що використовуються для тестування РК-DBSCAN, а D_2 містить 1 000 000 точок даних, що використовуються для перевірки РК-K-means.

На рисунку 4.5 показано час виконання проти кількості вузлів у системі для РК-DBSCAN на рисунку 4.4 показано час виконання щодо кількості вузлів в системі для РК-K-means.

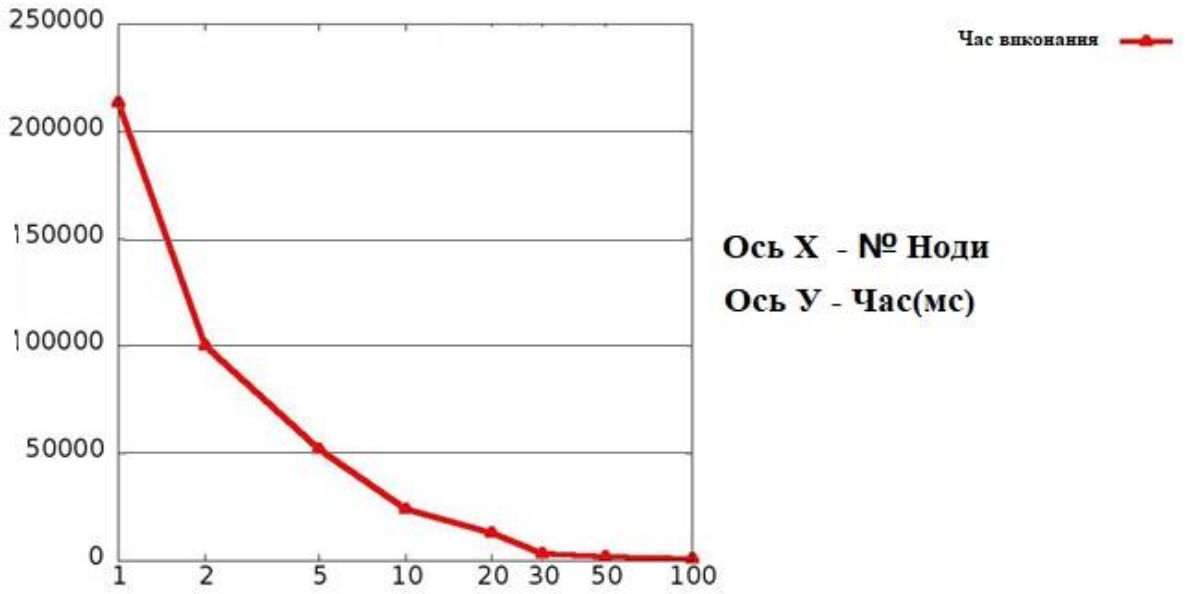


Рисунок 4.4 - РК-K-means масштабованість без мережевих платежів

Як бачимо, РК-K-means знадобилося лише кілька секунд для кластеризації 1 000 000 точок в розподіленій системі, що містить до 100 вузлів. Це зрозуміло з того, що РК-DBSCAN також зайняв кілька секунд (включаючи обчислення матриці під час для РК-DBSCAN) для кластерування 50 000 точок даних у розподіленій системі, що містить до 100 вузлів.

Такий алгоритм зайняв ще менше часу виконання, коли було задіяно час обчислення матриці.

Таким чином, алгоритм з обома K-means та алгоритми DBSCAN можуть комфортно обробляти багатовимірні дані через низьку складність.

Нарешті, можно помітити, що РК-K-means обробляє більші дані в порівнянні з РК-DBSCAN за допомогою однієї машини, яка це є через високу обчислювальну складність алгоритму DBSCAN.

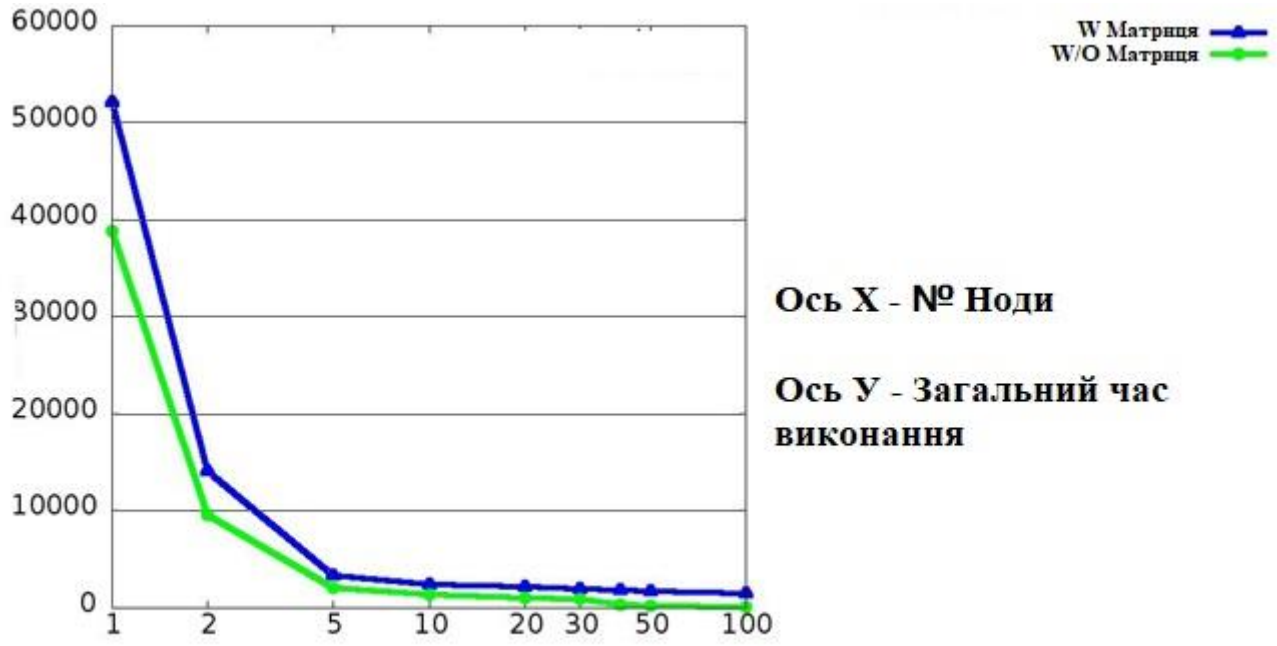


Рисунок 4.5 - Масштабованість PK-DBSCAN без мережевих платежів.

Метою тут є показати, що методика PK добре масштабується, навіть якщо включені мережеві платежі, а також можна динамічно визначати оптимальну кількість вузлів обробки, необхідних для кластеризації набору даних розміром N (два набори даних): перший набір даних D_1 містить 10000 точок даних і другий D_2 містить 30000 точок даних.

На рисунку 6.6 показано час виконання щодо кількості машин в системі, що використовує перший набір даних D_1 і рисунок 6.7 показує час виконання проти кількості машин у системі з використанням другого набору даних D_2 .

Як видно з рисунків 6.6 та 6.7, час виконання першої фаза (кластеризація та контур) постійно зменшується, оскільки кількість розповсюджена в системі розподілу.

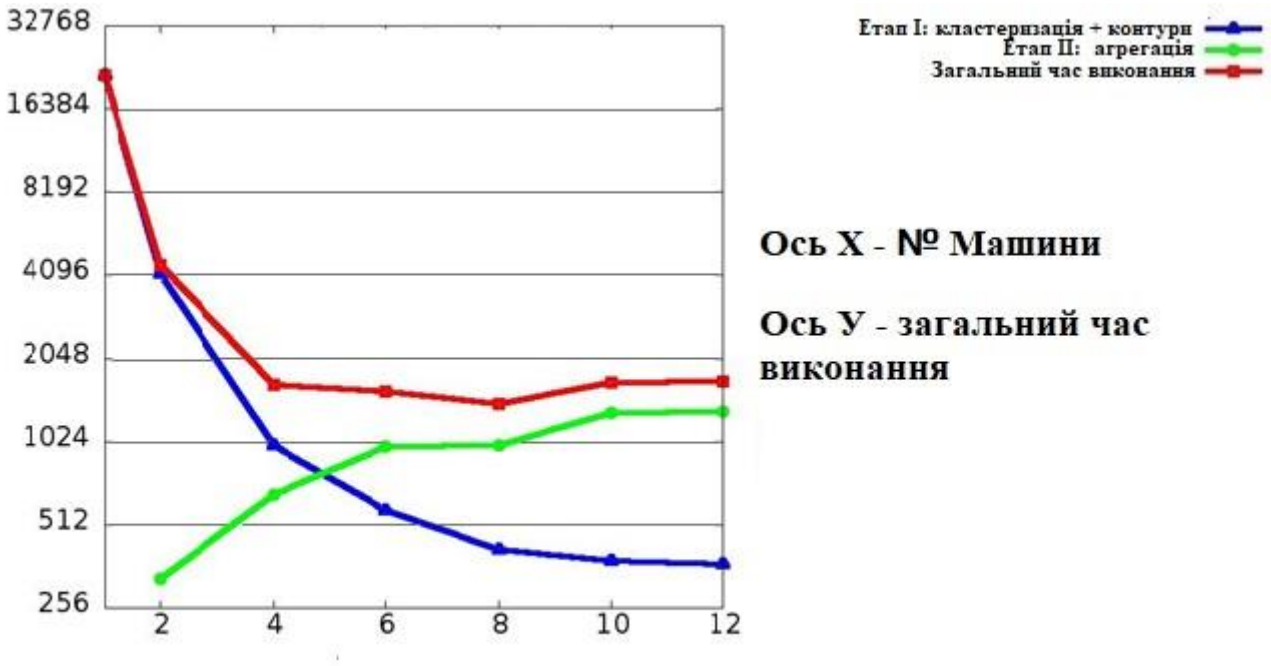


Рисунок 4.6 - Масштабованість PK-DBSCAN з використанням D 1, включаючи мережеві платежі

Однак час другого етапу (злиття) продовжує поступово збільшуватися із збільшенням кількості машин в розподіленій системи. Це тому, що кількість комунікацій (робочі витрати) на другій фазі збільшується, коли кількість машин збільшується.

Крім того, загальний час виконання алгоритму (що є сумою двох етапів) постійно зменшується, оскільки кількість технологічних процесів вузлів збільшується, поки не досягне певного рівня, де загальний час виконання починає збільшуватися (на 8 машинах для набору даних D_1 і на 16 машинах для набору даних D_2).

Оптимальна кількість вузлів обробки, необхідних для виконання РК повертається, коли накладні витрати на підхід перевищують час виконання локальної кластеризації.

Це дуже цікава характеристика (наскільки можливо визначити кількість машин, які можна виділити заздалегідь).

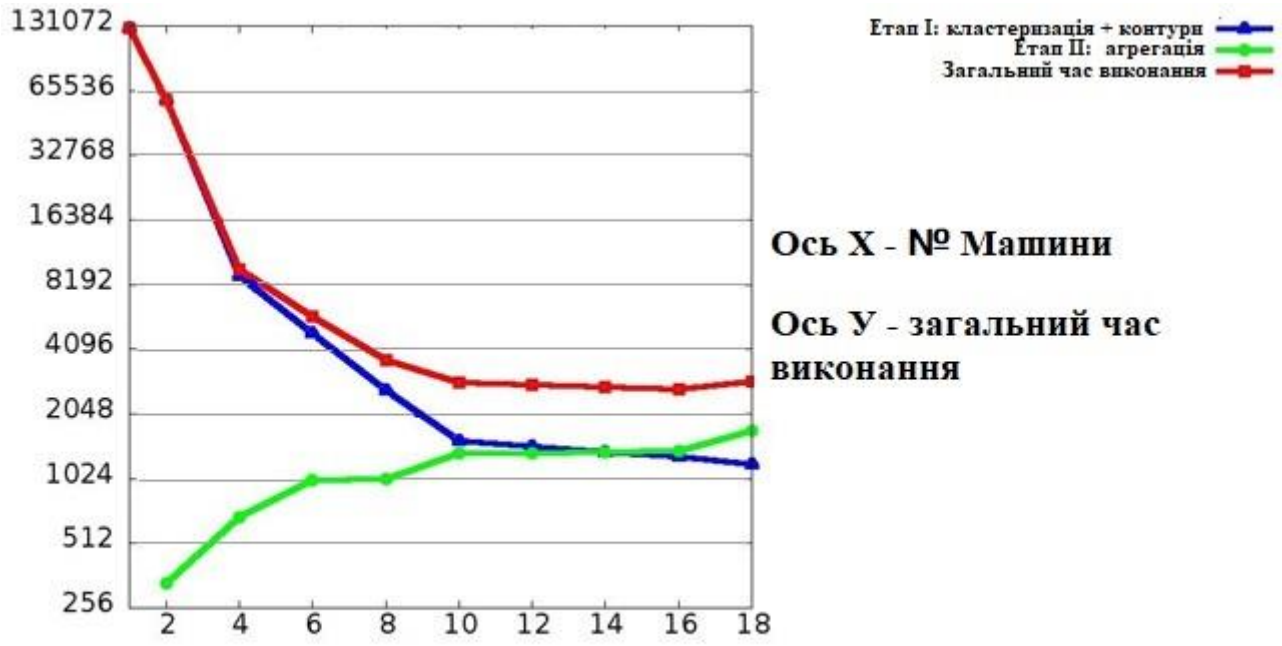


Рисунок 4.7 - Масштабованість PK-DBSCAN з використанням D_2

Результати проведених експериментів показали, що застосування розробленої системи надає можливість досягти підвищення ефективності кластеризації для видобутку просторових даних на рівні 98.2%, що на 0.5 – 3.9% вище в порівняні з відомими системами.

4.3 Висновок

В четвертому розділі розроблено розподілену динамічну систему кластеризації для видобутку просторових даних та вивчено ефективність підходу з точки зору часу виконання в реальній розподіленій системі. Розподілена система використовує обчислювальну потужність розподіленої платформи, максимізуючи паралельність та мінімізуючи комунікації та головним чином обсяг даних, якими обмінюються вузли в системі. Система реалізована з використанням як синхронних, так і асинхронних комунікацій. Система застосовує методи стиснення даних, що зменшує обсяг обміну даними між компонентами розподіленох системи. Результати проведених експериментів показали, що

застосування розробленої системи надає можливість досягти підвищення ефективності кластеризації для видобутку просторових даних на рівні 98.2%, що на 0.5 – 3.9% вище в порівняні з відомими системами.

ВИСНОВКИ

В даній роботі за результатами виконаних теоретичних та практичних досліджень було розроблено розподілену динамічну систему кластеризації для видобутку просторових даних.

У першому розділі була досліджена предметна область, досліджені відомі методи кластеризації для видобутку просторових даних.

У другому розділі удосконалено та представлено модель процесу розподіленої динамічної кластеризації для видобутку просторових даних, яка, на відміну від відомих моделей, використовує алгоритми стиснення та агрегації даних;

У третьому розділі представлено удосконалений метод розподіленої динамічної кластеризації для видобутку просторових даних, який, на відміну від відомих методів, ґрунтується на побудованій моделі розподіленої динамічної кластеризації для видобутку просторових даних та застосовує методи стиснення даних та динамічної кластеризації. Застосування розробленого методу надає можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами.

У четвертому розділі на основі представленого методу розроблено розподілену динамічну систему кластеризації для видобутку просторових даних. Проведено експериментальні дослідження, результати яких показали, що застосування розробленої системи надає можливість досягти підвищення ефективності кластеризації для видобутку просторових даних на рівні 98.2%, що на 0.5 – 3.9% вище в порівняні з відомими системами.

Отже, впровадження удосконаленого методу розподіленої динамічної кластеризації для видобутку просторових даних дозволяє підвищити ефективність кластеризації для видобутку просторових даних.

За темою дипломної роботи опублікован статтю у фаховому виданні.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бобровнікова, К. Ю., Е. В. Товстуха. Методи забезпечення енергоефективності та енергозбереження в системі розумного будинку. *Computer Systems and Information Technologies*, Том 1 № 1, 2020, с. 54-59.

2. Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal .Morgan Kaufmann Data Mining: Practical machine learning tools and techniques,2016,25-27p.

3. Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *ACM Sigmod Record*. 2020, volume 29, 1–12 p.

4. Yannis Manolopoulos, Apostolos N Papadopoulos, and Michael Gr Vassilakopoulos. Spatial databases: technologies, techniques and trends. *IGI Global* - 2017.

5. Krzyszt of Koperski, Junas Adhikary, and Jiawei Han. Spatial data mining: progress and challenges survey paper. *Proc. ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, Montreal, Canada, 2016, 1–10 p.

1. Arvind Sharma, HS Jat, and RK Gupta. A survey of spatial data mining approaches: Algorithms and architecture.*International Journal of Computer Technology and Electronics Communication*, 2017, 1:15–22 p.

2. Stewart Fotheringham and Peter Rogerson. Spatial analysis and GIS. *CRC Press*, 2018.

3. Dennis Wheeler, Gareth Shaw, and Stewart Barr. Statistical techniques in geographical analysis. *Routledge*, 2019.

6. Sujni Paul. Parallel and distributed data mining. *In New Fundamental Technologies in Data Mining*. Tech, 2017.

1. Rakesh Agrawal, Tomasz Imieli ´nski, and Arun Swami. Mining association rules between sets of items in large databases. *Acm sigmod recor*d, 2016. volume 22, 207–216 p.

7. Mrs Bharati and M Ramageri. Data mining techniques and applications. *Indian Journal of Computer Science and Engineering*, 1, 2020.

1. Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. *Proc. 20th int. conf. very large data bases*, VLDB, 2020, volume 1215, 487–499 p.

8. Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *ACM Sigmod Record*, 2020, volume 29, pages 1–12 p.

9. Martin Ester, Hans-Peter Kriegel, and Jörg Sander. Spatial data mining: A database approach. *Advances in spatial databases*, 2017, 47–66 p.

10. Gennady L Andrienko and Natalia V Andrienko. Data mining with c4. 5 and interactive cartographic visualization. *User Interfaces to Data Intensive Systems, IEEE, 2020. Proceedings*, 162–165 p.

11. J Ross Quinlan. C4. 5: programs for machine learning. *Elsevier*, 2017.

12. Usama M Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy. Advances in knowledge discovery and data mining, volume 21. *AAAI press Menlo Park*, 2016.

13. Casey Cleve, Maggi Kelly, Faith R Kearns, and Max Moritz. Classification of the wildland–urban interface: A comparison of pixel-and object-based classifications using high-resolution aerial photography. *Computers, Environment and Urban System*s, 2018, volume32(4), 317–326 p.

14. Baris M Kazar, Shashi Shekhar, David J Lilja, Ranga R Vatsavai, and R Kelley Pace. Comparing exact and approximate spatial auto-regression model solutions for spatial data analysis. *In International Conference on Geographic Information Science*, 2020, 140–161 p.

15. Oleg Smirnov and Luc Anselin. Fast maximum likelihood estimation of very large spatial autoregressive models: a characteristic polynomial approach. Computational Statistics & Data Analysis, 2016, volume 35(3), 301–319 p.

21. Jacob Kogan. Introduction to clustering large and high-dimensional data. *Cambridge University Press*, 2017.

22. Jen-Wei Huang, Su-Chen Lin, and Ming-Syan Chen. Dpsp: Distributed progressive sequential pattern mining on the cloud. *Advances in Knowledge Discovery and Data Mining*, , 2019, 27–34 p.

23. Christopher Moretti, Jared Bulosan, Douglas Thain, and Patrick J Flynn. Allpairs: An abstraction for data-intensive cloud computing. *In International Symposium on Parallel and Distributed Processing (IPDPS)*, IEEE, 2020, 1–11 p.

24. Brandyn White, Tom Yeh, Jimmy Lin, and Larry Davis. Web-scale computer vision using mapreduce for multimedia data mining. *In Proceedings of the Tenth International Workshop on Multimedia Data Mining, ACM*, 2017, 9 p.

25. Yaobin He, Haoyu Tan, Wuman Luo, Shengzhong Feng, and Jianping Fan. Mrdbscan: a scalable mapreduce-based dbscan algorithm for heavily skewed data. *Frontiers of Computer Science*, 2018, 83–99 p.

26. Bi-Ru Dai and I-Chang Lin. Efficient map/reduce-based dbscan algorithm with optimized data partition. I*n 5th International Conference on Cloud Computing (CLOUD), IEEE*, 2018, 59–66 p.

27. Liang-Chi Hsieh, Guan-Long Wu, Yu-Ming Hsu, and Winston Hsu. Online image search result grouping with mapreduce-based image clustering and graph 152 REFERENCES construction for large-scale photos. *Journal of Visual Communication and Image Representation*, 2017, 384–395 p.

28. Yaobin He, Haoyu Tan, Wuman Luo, Huajian Mao, Di Ma, Shengzhong Feng, and Jianping Fan. Mr-dbscan: an efficient parallel density-based clustering algorithm using mapreduce. *In 17th International Conference on Parallel and Distributed Systems (ICPADS), IEEE*, 2019, 473–480 p.

29. Yan Xiang Fu, Wei Zhong Zhao, and Hui Fang Ma. Research on parallel dbscan algorithm design based on mapreduce. *In Advanced Materials Research, volume 301, Trans Tech Publ*, 2019, 1133–1138 p.

30. Younghoon Kim, Kyuseok Shim, Min-Soeng Kim, and June Sup Lee. Dbcuremr: An efficient density-based clustering algorithm for large data using mapreduce. *Inf. Syst.*, June 2018, 15–35 p.

31. Yaobin He, Haoyu Tan, Wuman Luo, Huajian Mao, Di Ma, Shengzhong Feng, and Jianping Fan. Mr-dbscan: an efficient parallel density-based clustering algorithm using mapreduce. *In International Conference on Parallel and Distributed Systems (ICPADS), IEEE*, 2011, 473–480 p.

32. John F Roddick, Kathleen Hornsby, and Myra Spiliopoulou. An updated bibliography of temporal, spatial, and spatio-temporal data mining research. *In Temporal, Spatial, and Spatio-Temporal Data Mining*, Springer, 2016, 147–163 p.

33. Jyrki Kivinen and Heikki Mannila. The power of sampling in knowledge discovery. In Proceedings of the thirteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, *ACM*, 2016, 77–85 p.

34. N-A. Le-Khac, M. Bue, M. Whelan, and M-T.Kechadi. A knowledgebased data reduction for very large spatio-temporal datasets. *International Conference on Advanced Data Mining and Applications, (ADMAâA˘Z2010)* , November 2019, 19-21 p.

35. Eshref Januzaj, Hans-Peter Kriegel, and Martin Pfeifle. Dbdc: Density based distributed clustering. *Advances in Database Technology-EDBT*, 2018, 529– 530 p.

36. J-F Laloux, N-A. Le-Khac, and M-T. Kechadi. Efficient distributed approach for density-based clustering. *Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 20th IEEE International Workshops, 27-29 June 2019, 145–150 p.

37. M.J. Fadilia, M. Melkemib, and A. ElMoataza. Pattern Recognition Letters:Nonconvex onion-peeling using a shape hull algorithm, 15 October 2016, volume 24.

38. A.Ray Chaudhuri, B.B. Chaudhuri, and S.K. Parui. A novel approach to computation of the shape of a dot pattern and extraction of its perceptual border. *Computer vision and Image Understranding*, 03 December 2016, 257–275 p.

39. Mahmoud Melkemi and Mourad Djebali. Computing the shape of a planar points set. *Elsevier Science*¸S1436, 9 September 2018.

40. Herbert Edelsbrunner, David G Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *Information Theory*, IEEE Transactions on, 2016, 551–559 p.

41. Adriano Moreira and Maribel Yasmina Santos. Concave hull: A k-nearest neighbours approach for the computation of the region occupied by a set of points. In

Int'l. Conf. on Computer Graphics Theory and Applications (GRAPP-2017), Barcelona, Spain, 8-11 March 2017, 61–68 p.

42. Matt Duckhama, Lars Kulikb, Mike Worboysc, and Antony Galtond. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. Elsevier Science Inc. New York, NY, USA, 15 March 2019.

43. N-A. Le-Khac, M. Bue, M. Whelan, and M-T.Kechadi. A knowledgebased data reduction for very large spatio-temporal datasets. *International Conference on Advanced Data Mining and Applications, (ADMAâA˘Z2010)*, 19-21 November 2018.

44. Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A densitybased algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, 2016, 226–231 p.

45. J-F Laloux, N-A. Le-Khac, and M-T. Kechadi. Efficient distributed approach for density-based clustering. Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), *20th IEEE International Workshops*, 27-29 June 2019, 145–150 p.

46. Roberto Solar, Francisco Borges, Remo Suppi, and Emilio Luque. Improving communication patterns for distributed cluster-based individual-oriented fish school simulations. Procedia Computer Science, 2018.

47. E Cortese. Benchmark on jade message transport system. URL: http://jade. cselt. it/doc/tutorials/benchmark/JADERTTBenchmark. htm, 2018.

48. Fabio Bellifemine, Federico Bergenti, Giovanni Caire, and Agostino Poggi. JadeâA˘Ta java agent development framework. *In Multi-Agent Programming, Springer*, 2017, 125–147 p.

49. Malika Bendechache and M-Tahar Kechadi. Parallel and distributed clustering framework for big spatial data mining. International Journal of Parallel, Emergent and Distributed Systems, 2017.

50. Malika Bendechache and M-Tahar Kechadi. Distributed clustering algorithm for spatial data mining. *In 2nd International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM)*, IEEE, 2019, 60–65 p.

51. Malika Bendechache, Nhien-An Le-Khac, and M-Tahar Kechadi. Efficient large scale clustering based on data partitioning. *In International Conference on Data Science and Advanced Analytics (DSAA)*, IEEE, 2016, 612–621 p.

52. Malika Bendechache, Nhien-An Le-Khac, and M-Tahar Kechadi. Hierarchical aggregation approach for distributed clustering of spatial datasets. I*n 16th International Conference on Data Mining Workshops (ICDMW)*, IEEE, 2016, 1098–1103 p.

53. Malika Bendechache, Nhien-An Le-Khac, and M-Tahar Kechadi. Performance evaluation of a distributed clustering approach for spatial datasets. *In 15th International Conference on Australasian Data Mining Conference (AusDM*). CRPIT, 2017.

54. Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. Data Mining: Practical machine learning tools and techniques. *Morgan Kaufmann*, 2016.

55. J. Han and M. Kamber. Data Mining: Concepts and Techniques. *Morgan Kaufmann Publisher*, 2nd edition, 2018.

56. Sujni Paul. Parallel and distributed data mining. *In New Fundamental Technologies in Data Mining*. InTech, 2019.

57. Usama M Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy. *Advances in knowledge discovery and data mining*, *volume 2*1. AAAI press Menlo Park, 2016.

58. Roberta Siciliano and Claudio Conversano. *Decision tree induction.*, 2016.

59. Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 2017.

60. Guoqing Chen, Hongyan Liu, Lan Yu, Qiang Wei, and Xing Zhang. A new approach to classification based on association rule mining. *Decision Support Systems*, 2017.

61. Brandon K Vaughn. Data analysis using regression and multilevel/hierarchical models, by gelman, a., & hill, j. J*ournal of Educational Measurement*, 2017.

62. Finn V Jensen. An introduction to Bayesian networks, volume 210. *UCL press London*, 2019.

63. Janet Kolodner. Case-based reasoning. Morgan Kaufmann, 2019.

64. Frederic Dreier. Genetic algorithm tutorial, 2018.

65. Rakesh Agrawal, Tomasz Imieli ´nski, and Arun Swami. Mining association rules between sets of items in large databases. *In Acm sigmod record*, volume 22, ACM, 2016, 207–216 p.

66. Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. *In Proc. 20th int. conf. very large data bases*, VLDB, volume 1215, 2016, 487–499 p.

67. Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In ACM Sigmod Record, volume 29, *ACM*, 2018, 1–12 p.

68. Yannis Manolopoulos, Apostolos N Papadopoulos, and Michael Gr Vassilakopoulos. Spatial databases: technologies, techniques and trends. *IGI Global*, 2016.

69. Krzysztof Koperski, Junas Adhikary, and Jiawei Han. Spatial data mining: progress and challenges survey paper. In Proc. ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, Montreal, Canada, *Citeseer*, 2016, 1–10 p.

70. Arvind Sharma, HS Jat, and RK Gupta. A survey of spatial data mining approaches: Algorithms and architecture. *International Journal of Computer Technology and Electronics Communication*, 2017.

71. Stewart Fotheringham and Peter Rogerson. Spatial analysis and GIS. *CRC Press*, 2018.

72. Dennis Wheeler, Gareth Shaw, and Stewart Barr. Statistical techniques in geographical analysis. *Routledge*, 2017.

73. Jiawei Han, Yandong Cai, and Nick Cercone. Data-driven discovery of quantitative rules in relational databases. *IEEE transactions on Knowledge and Data Engineering*, 2016.

74. Gregory Piateski and William Frawley. Knowledge discovery in databases. MIT press, 2016.

75. Jeremy Mennis and Diansheng Guo. Spatial data mining and geographic knowledge discoveryâA Tan introduction. Computers, Environment and Urban Systems, 2018.

76. Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. Annals of Data Science, 2018.

77. Oded Maimon and Lior Rokach. Data Mining and Knowledge Discovery Handbook. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2016.

78. Donato Malerba. Mining spatial data: Opportunities and challenges of a relational approach, 2017.

79. Oded Maimon and Lior Rokach. Data Mining and Knowledge Discovery Handbook. *Springer Publishing Company, Incorporated*, 2nd edition, 2016.

80. WR Tobler. Cellular geography. *In Philosophy in geography*, Springer, 2018, 379–386 p.

81. Mrs Bharati and M Ramageri. Data mining techniques and applications. *Indian Journal of Computer Science and Engineering*, 1, 2018.

82. Yanbin Ye and Chia-Chu Chiang. A parallel apriori algorithm for frequent itemsets mining. *In Fourth International Conference on Software Engineering Research, Management and Applications, IEE*E, 2017, 87–94 p.

83. Ning Li, Li Zeng, Qing He, and Zhongzhi Shi. Parallel implementation of apriori algorithm based on mapreduce. *In 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD), IEEE*, 2019, 236–241 p.

84. Raffaele Perego, Salvatore Orlando, and P Palmerini. Enhancing the apriori algorithm for frequent set counting. *In International Conference on Data Warehousing and Knowledge Discovery*, Springer, 2016, 71–82 p.

85. Krzysztof Koperski and Jiawei Han. Discovery of spatial association rules in geographic information databases. *In Advances in spatial databases*, Springer, 2016, 47–66 p.

86. Jeremy Mennis and Junwei Liu. Mining association rules in spatio-temporal data. I*n Proceedings of the 7th International Conference on GeoComputation*, 2017, 642–646 p.

87. Jiawei Han and Yongjian Fu. Mining multiple-level association rules in large databases. *IEEE Transactions on knowledge and data engine*ering, 2016.

ДОДАТОК А
(обов'язковий)

Фрагменти лістингу програмної реалізації розподіленої динамічної системи кластеризації для видобутку просторових даних

Clustering.py

```
from sklearn.cluster import KMeans, DBSCAN, MeanShift, AgglomerativeClustering
from ..config import log


@log('- K-Means clustering started.')
def kmeans(X, attrs):
    return KMeans(n_clusters=int(attrs['kNumber']), n_init=100, algorithm='full', n_jobs=-1).fit(X)


@log('- DBSCAN clustering started.')
def dbscan(X, attrs,):
    return DBSCAN(min_samples=int(attrs['minSamples']), eps=float(attrs['eps']), n_jobs=-1).fit(X)


@log('- Mean Shift clustering started.')
def meanshift(X, attrs):
    return MeanShift(cluster_all=attrs['clusterAll'], min_bin_freq=int(attrs['binNumber']), n_jobs=-1).fit(X)


@log('- Hierarchical clustering started.')
def hcluster(X, attrs):
    return AgglomerativeClustering(n_clusters=int(attrs['kNumber']), linkage=attrs['linkage'], affinity=attrs['affinity']).fit(X)


cluster_algorithms = {
    'kmeans': kmeans,
    'dbscan': dbscan,
    'hcluster': hcluster,
    'meanshift': meanshift
}
```

```
def get_vectorized_text(dataframe, columns, attrs):
    """Vectorize the text columns of a dataframe. Append them in a single matrix"""
    vec = get_vectorizer(attrs)
    stem = attrs['stemming']

    X = []
    for column in columns:
        logger.info('-- Vectorizing field: {}...'.format(column))
        text = dataframe[column].values

        if stem:
            text = stem_text_input(text)

        X_new = vec.fit_transform(text)
```

```
def get_all_vectors(df, attrs):
    """Returna a matrix of the numerical and text vectors (text after vectorisation)."""
    num_columns, text_columns = get_dataframe_fields(df, attrs['fields'])
    X = df[num_columns].values  # Get the numerical fields (ready to use)

    # Check if there are any text columns, and vectorize them
    return hstack([X, get_vectorized_text(df, text_columns, attrs)]) \
        if text_columns else X


def projection_exists(attrs):
    return '{}.pkl'.format(attrs['decomposition']) in os.listdir(TEMP_PATH)

def get_projection(attrs):
    """Project the data in a 2D space."""
    df = load_df()

    X = get_all_vectors(df, attrs)
    logger.info('- Data shape original: {}'.format(X.shape))


@log('Clustering process started')
def cluster_data(attrs):
    """Retrieve items, cluster, and return the result dict."""
    if projection_exists(attrs):
        df = load_df()
        X = load_X()
    else:
        X, df = get_projection(attrs)

    # Execute clustering
    model = cluster_algorithms[attrs['algorithm']](X, attrs)
    labels = model.labels_ \
        if hasattr(model, 'labels_') \
        else model.predict(X)

    # Save clusters for tfidf and similar
    df['clx_cluster'] = labels

    save_df(df)
    save_cluster_model(model)
    return scatterplot(X, labels, df['clx_id'])
```

```
@log('- Using: PCA.')
def pca(X):
    return PCA().fit(X)


@log('- Using: SVD.')
def svd(X):
    return TruncatedSVD().fit(X)


@log('- Using: t-SNE')
def tsne(X, metric):
    return TSNE(learning_rate=100, perplexity=15, metric=metric).fit(X)


@log('Decomposition started.')
def dimension_reduction(X, decomp, metric):
    """Dimension reduction and saving of the model."""
    if X.shape[1] > 2:
        if decomp == 'pca':
            model = pca(X)
        elif decomp == 'tsne':
            model = tsne(X, metric)
        elif decomp == 'mds':
            model = mds(X, metric)
        else:
            model = svd(X)

        save_decomposition(model, decomp)
        X = model.transform(X) \
            if hasattr(model, 'transform') \
            else model.embedding_

    save_X(X)
    return X
```

```
to_float = lambda l: [float(i) for i in l]


@log('- Search started.')
def search(query):
    results = sqldf('SELECT clx_id FROM df WHERE {} ;'.format(query),
                    {'df': load_df()}).values
    return [int(i[0]) for i in results]


def get_node_details(_id):
    df = load_df()
    fields = df[df['clx_id'] == _id].columns.values
    details = [detail[:20] for detail in df[df['clx_id'] == _id].values[0]
               if isinstance(detail, str)]
    return fields, details


def get_field_data(fields):
    df = load_df()
    fields += ['clx_id', 'clx_cluster']
    return [dict(zip(fields, to_float(row)))
            for row in df[fields].values]
```

```
TOKEN_PATTERN = r'\b\w+\b'  # Keeps single letter attrs
STEM_REGEX = r'\b[^\d\W]+\b'

stop = lambda attrs: stopwords if attrs['stopwords'] else None
norm = lambda attrs: None if attrs['norm'] == 'none' else attrs['norm']
join_by_cluster = lambda cl, df, tc: ' '.join(df[df.clx_cluster == cl][tc].apply(' '.join, axis=1).values)

analyzer = StemmingAnalyzer(expression=STEM_REGEX, minsize=1) | CharsetFilter(accent_map)


def get_vectorizer(attrs):
    """Returna a vectorizer with the user options."""
    stop_ = stop(attrs)
    norm_ = norm(attrs)
    features = int(attrs['featureNumber'])
    vectorizers = {
        'hashing': HashingVectorizer(n_features=features, non_negative=True, norm=norm_, stop_words=stop_),
        'tfidf': TfidfVectorizer(max_features=features, norm=norm_, token_pattern=TOKEN_PATTERN, stop_words=stop_),
        'count': CountVectorizer(max_features=features, token_pattern=TOKEN_PATTERN, stop_words=stop_)
    }

    logger.info('- Vectorizer created: {}'.format(attrs['vectorizer']))
    return vectorizers[attrs['vectorizer']]
```

```
@log('- TF-IDF scoring started.')
def tfidf(attrs, clusters=None):
    """
    Returns the TF-IDF scores of words, separated by cluster in the form of:
    {
        '0':[{'term': 'x', 'score': y}, {'term': 'x', 'score': y}, ...],
        '1': [...]
    }
    """
    df = load_df()
    text_columns = list(df.select_dtypes(include=['object']).columns)
    attrs['vectorizer'] = 'tfidf'  # monkey-patch to use the API

    # Only get tfidf in text attributes
    if not text_columns:
        return

    # Consider each cluster a document, so get the text from each cluster separately
    # If an cluster list is given, get results for those clusters
    if not clusters:
        clusters = df.clx_cluster.unique()

    text_by_cluster = [join_by_cluster(cluster, df, text_columns) for cluster in clusters]

    # Feature matrix
    vec = get_vectorizer(attrs)
    tfidf_matrix = vec.fit_transform(text_by_cluster).toarray()
    feature_names = vec.get_feature_names()


     # Get the top 10 TF-IDF scores for each cluster
     tfidf_by_cluster = {}
     for cluster, doc in zip(clusters, tfidf_matrix):
         max_10_indices = doc.argsort()[-10:][::-1]
         tfidf_by_cluster[str(cluster)] = [{'term': feature_names[i], 'score': doc[i]}
                                           for i in max_10_indices]
     return tfidf_by_cluster


@log('- Stemming started')
def stem_text_input(text):
    """Stem the text corpus."""
    parallel = Parallel(n_jobs=-1, backend='multiprocessing', verbose=1)
    return parallel(delayed(_stem)(row) for row in text)
```

```
 def _stem(row):
     """Process a single item, to be used in parallel."""
     return ' '.join([token.text for token in analyzer(row)])

projections = Blueprint('projections', __name__)


@projections.route('/get_clustering_results', methods=['POST'])
def get_clustering_results():
    """
    Returns a dict with the plot coordinates of the clusted data,
    and the TF-IDF scores, if we have text attributes.
    """
    attrs = get_attrs(request)
    return jsonify(**{'results': cluster_data(attrs)})


@projections.route('/get_projection', methods=['POST'])
def get_projection_results():
    """
    Returns a dict with the plot coordinates of the clusted data,
    and the TF-IDF scores, if we have text attributes.
    """
    attrs = get_attrs(request)
    coords, df = get_projection(attrs)
    return jsonify(**{'results': scatterplot(coords, [0 for _ in range(len(df))], df['clx_id'])})


 @projections.route('/predict_cluster', methods=['POST'])
 def predict_cluster():
     """Predict the cluster of the imput row."""
     # attrs = get_attrs(request)
     # return jsonify(**{'cluster_results': {'nodes': predict_data(attrs)}})
     pass


 @projections.route('/scatterplot_matrix', methods=['POST'])
 def get_scatterplot_coordinates():
     attrs = get_attrs(request)
     return jsonify(**{'results': get_field_data(attrs['scatterFields'])})
```

```
results = Blueprint('results', __name__)


@results.route('/search', methods=['POST'])
def search_data():
    """Render algorithm options."""
    query = request.form.get('query')
    return jsonify(**{'results': search(query)})


@results.route('/tfidf', methods=['POST'])
def tf_idf():
    """Returns the TF-IDF score per cluster."""
    attrs = get_attrs(request)
    clusters = json.loads(request.form.get('clusters'))
    return jsonify(**{'results': tfidf(attrs, clusters)})


@results.route('/node_details', methods=['POST'])
def node_details():
    _id = int(request.form.get('id'))
    fields, details = get_node_details(_id)
    return render_template('node_details.html', data={'fields': fields, 'details': details})
```

Index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Clusterix</title>
    <!-- CSS -->
    <link rel="stylesheet" type="text/css" href="../static/css/bootstrap.min.css">
    <link rel="stylesheet" type="text/css" href="../static/css/fileinput.css">
    <link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.1.8/semantic.min.css">

    <link rel="stylesheet" type="text/css" href="../static/css/clusterix.css">
    <link rel="stylesheet" type="text/css" href="../static/css/clusterix_colors.css">

    <!-- JS LIB -->
    <script src="../static/js/jquery-2.1.4.min.js"></script>
    <script src="../static/js/bootstrap.js"></script>
    <script src="../static/js/d3.min.js"></script>
    <script src="../static/js/fileinput.js"></script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.1.8/semantic.min.js"></script>

    <!-- JS -->
    <script src="../static/scripts/plotting/scatterplot.js"></script>
    <script src="../static/scripts/plotting/scatterplot_brush_init.js"></script>
    <script src="../static/scripts/plotting/scatterplot_brushed_area.js"></script>
    <script src="../static/scripts/plotting/scatterplot_matrix.js"></script>
    <script src="../static/scripts/plotting/tfidf_chart.js"></script>
```

```
    <script src="../static/scripts/search.js"></script>
    <script src="../static/scripts/router.js"></script>
    <script src="../static/scripts/misc.js"></script>
    <script src="../static/scripts/init.js"></script>
</head>
<body>


<!---------------------------------->
<!-- NAVBAR AND SEARCH -->
<!---------------------------------->
<nav class="navbar">
    <div class="container-fluid no-padding-left">

        <!-- LOGOS -->
        <div class="navbar-header">
            <a class="brand-logo inspirehep-logo pull-left" href="#"><img src="../static/img/inspirelabs.svg"/></a>
            <a class="clusterix-logo pull-left" href="#"><img src="../static/img/clusterix.svg"/></a>
        </div>

        <!-- SEARCH BAR -->
        <div class="navbar-collapse collapse">
            <form class="navbar-form navbar-right" role="search">
                <div class="form-group">
                    <input type="text" id="search" class="form-control white" placeholder="Search...">
                </div>
            </form>
        </div>
    </div>
</nav>


     <!---------------------------------->
     <!-- PROCESS AND DIAGRAM SPACE -->
     <!---------------------------------->
     <div class="row">

         <!---------------------------------->
         <!-- PROCESS SPACE -->
         <!---------------------------------->
         <div class="col-lg-3 processing-container">
             <h2 class="space-title">Data Processing</h2>

             <!-- DATA INPUT PANEL -->
             <div class="panel grey-background">
                 <div class="panel-heading row no-padding-left light-blue">
                     <h3 class="panel-title col-lg-11">Data Input</h3>
                 </div>
                 <div class="panel-body grey-background" id="data-input-body">
                     <label class="control-label">Select File</label>
                     <input id="data-input" type="file">
                 </div>
             </div>
```

```
    <!---------------------------------->
    <!-- DIAGRAM SPACE -->
    <!---------------------------------->
    <div class="col-lg-9 diagram-container grey-background">

        <!-- VISUALIZATION -->
        <div class="row">
            <h2 class="space-title">Visualization Swatchboard</h2><div id="vizualization-area" class="row"></div>
        </div>

        <!-- BRUSH, TFIDF -->
        <div id="brush-results" class="row padding-top-20 no-display" >
            <div class="col-md-6">
                <h3 class="text-left space-title">Selected Area</h3><div id="brushed-area"></div>
            </div>
            <div class="col-md-6">
                <h3 class="text-left space-title">TF-IDF by cluster</h3><div id="tf-idf-results"></div>
            </div>
        </div>


        <!-- SCATTERPLOT MATRIX -->
        <div id="scatterplot-matrix" class="row padding-top-20 no-display">
            <h3 class="text-left space-title">Scatterplot Matrix</h3><div id="matrix-area" class="row"></div>
        </div>



<!-- LOADING SCREEN -->
{% from 'loading.html' import loading %}
{{ loading() }}

</body>
</html>
```

Dbscan.html

```
<!-- DBSCAN CLUSTERING -->
<!-- EPS -->
<div class="form-group">
    <label class="control-label" for="eps">Min. Bin Frequency</label>
    <input class="form-control input-sm" id="eps" type="number" value="0.5">
</div>

<!-- MIN SAMPLE -->
<div class="form-group">
    <label class="control-label" for="min-samples">Min. Samples</label>
    <input class="form-control input-sm" id="min-samples" type="number" value="20">
</div>
```

Hcluster.html

```
<!-- K NUMBER -->
<div class="form-group">
    <label class="control-label" for="kmeans-num">K Number</label>
    <input class="form-control input-sm" id="kmeans-num" type="number" value="1">
</div>

<!-- LINKAGE METHODS -->
<div class="form-group row">
    <label for="linkage-selection" class="no-padding-left col-lg-3">Linkage Methods</label>
    <select id="linkage-selection" class="ui dropdown selection col-lg-9">
        <option value="">Choose...</option>
        <option value="complete">Complete</option>
        <option value="average">Average</option>
        <option value="ward">Ward</option>
    </select>
</div>

<!-- AFFINITY METHODS -->
<div class="form-group row" id="affinity-container">
    <label for="affinity-selection" class="no-padding-left col-lg-3">Affinity</label>
    <select id="affinity-selection" class="ui dropdown selection col-lg-9">
        <option value="">Choose...</option>
        <option value="euclidean">Euclidean</option>
        <option value="manhattan">Manhattan</option>
        <option value="cosine">Cosine</option>
    </select>
</div>
<script>
    $('#linkage-selection').dropdown();
    $('#affinity-selection').dropdown();
</script>
```

Meanshift.html

```
<!-- BIN NUMBER -->
<div class="form-group">
    <label class="control-label" for="bin-frequency">Min. Bin Frequency</label>
    <input class="form-control input-sm" id="bin-frequency" type="number" value="5">
</div>

<!-- CLUSTER ALL DATA -->
<div class="form-group row">
    <div class="ui checkbox" id="cluster-all">
        <input type="checkbox"><label>Cluster all data</label>
    </div>
</div>
<script>$('.checkbox').checkbox();</script>
```

Algoritms.html

```
<!-- ALGORITHMS AND DEFINITIONS PANEL -->
<div class="panel-heading row no-padding-left orange">
    <h3 class="panel-title col-lg-11">Algorithm Options</h3>
</div>

<div class="panel-body grey-background" id="algorithms-body">
    <!-- ALGORITHM SELECTION -->
    <div class="form-group row">
        <label for="algorithms-selection" class="no-padding-left col-lg-3">Algorithm</label>
        <select id="algorithms-selection" class="ui dropdown selection col-lg-9">
            <option value="">Choose...</option>
            <option class="small-font" value="kmeans">K-Means</option>
            <option class="small-font" value="hcluster">Hierarchical Clustering</option>
            <option class="small-font" value="meanshift">Mean Shift</option>
            <option class="small-font" value="dbscan">DBSCAN</option>
        </select>
    </div>

    <!-- OPTIONS CONTAINER -->
    <div id="algorithm-options">
        <!-- DEFAULT - KMEANS -->
        <div class="form-group">
            <label class="control-label" for="kmeans-num">K Number</label>
            <input class="form-control input-sm" id="kmeans-num" type="number" value="1">
        </div>
    </div>

    <!-- CLUSTER BUTTONS -->
    <div class="row">
        <a href="#" class="btn btn-warning btn-lg btn-block" id="get-results">
            <span class="glyphicon glyphicon-play"></span> Get Cluster Results</a>
    </div>
</div>
```

Decomposition.html

```
<!-- STEMMING/STOPWORDS -->
<div class="form-group row">
    <div class="ui checkbox" id="stem-checkbox">
        <input type="checkbox" name="stemming"><label>Stemming</label>
    </div><br>
    <div class="ui checkbox" id="stop-checkbox">
        <input type="checkbox" name="stopwords"><label>Stopwords removal</label>
    </div>
</div>
```

Config.py

```
# App configuration
TEMP_PATH = 'temp/'
CLUSTERER_PATH = 'temp/clusterer.pkl'
DATAFRAME_PATH = 'temp/dataframe.pkl'
DECOMPOSITION_MODEL_PATH = 'temp/{}.pkl'
X_PATH = 'temp/X.npy'

stopwords = ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours', 'yourself',
             'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself',
             'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'these',
             'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do',
             'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while',
             'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before',
             'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again',
             'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each',
             'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than',
             'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now']

# Logging
logging.basicConfig(stream=sys.stdout, level=logging.INFO,
                    format='%(asctime)s - %(levelname)s\t%(message)s',
                    datefmt='(%Y-%m-%d %H:%M:%S)')
logger = logging


def log(msg):
    """Logger decorator."""
    def decorator(func):
        def wrapper(*args):
            logger.info(msg)
            return func(*args)
        return wrapper
    return decorator
```

Utils.py

```
import os
import json
import pandas as pd
import numpy as np
from werkzeug.utils import secure_filename
from sklearn.externals import joblib
from .config import TEMP_PATH, DECOMPOSITION_MODEL_PATH, \
    DATAFRAME_PATH, CLUSTERER_PATH, X_PATH, log
```

```
# Clustering
@log('- Cluster model saved.')
def save_cluster_model(model):
    """Pickle cluster model."""
    joblib.dump(model, CLUSTERER_PATH)


@log('- Cluster model loaded.')
def load_cluster_model():
    """Load pickled cluster model."""
    return joblib.load(CLUSTERER_PATH)


# Decomposition
@log('- Decomposition model saved.')
def save_decomposition(model, name):
    """Pickle dec. model."""
    joblib.dump(model, DECOMPOSITION_MODEL_PATH.format(name))


@log('- Decomposition model loaded.')
def load_decomposition(name):
    """Load pickled dec. model."""
    return joblib.load(DECOMPOSITION_MODEL_PATH.format(name))


# Dataframe
@log('- Dataframe saved.')
def save_df(df):
    """Pickle dataframe."""
    pd.to_pickle(df, DATAFRAME_PATH)


@log('- Dataframe loaded.')
def load_df():
    """Load pickled dataframe."""
    return pd.read_pickle(DATAFRAME_PATH)

# X
@log('- X saved.')
def save_X(X):
    """Pickle X."""
    np.save(X_PATH, X)
```

```
def get_attrs(req):
    """Get the data attributes."""
    return json.loads(req.form.get('data'))


@log('- File loaded.')
def save_file(file):
    """Save file to disk."""
    try:
        file_path = os.path.join(TEMP_PATH, secure_filename(file.filename))
        file.save(file_path)
    except KeyError:
        pass  # no file sent, abort


def process_and_save_dataframe(filename):
    """
    Create a dataframe from the file, and save the model to the disk.
    Returns a field-type dict, and a boolean of the appearance of text data.
    """
    df = read_file(filename)
    save_df(df)

    fields = df.drop(['clx_cluster', 'clx_id'], axis=1).columns
    types = list(map(str, df.dtypes))
    has_text = 'object' in types  # 'object' represents the text fields
    numeric_fields = [field for field in fields if df[field].dtype != 'object']

    return dict(list(zip(fields, types))), numeric_fields, has_text


def process_and_save_dataframe(filename):
    """
    Create a dataframe from the file, and save the model to the disk.
    Returns a field-type dict, and a boolean of the appearance of text data.
    """
    df = read_file(filename)
    save_df(df)

    fields = df.drop(['clx_cluster', 'clx_id'], axis=1).columns
    types = list(map(str, df.dtypes))
    has_text = 'object' in types  # 'object' represents the text fields
    numeric_fields = [field for field in fields if df[field].dtype != 'object']

    return dict(list(zip(fields, types))), numeric_fields, has_text
```

```
def read_file(filename):
    """Process a file and return a dataframe."""
    df = pd.read_csv('{}{}'.format(TEMP_PATH, filename),
                     error_bad_lines=False,
                     encoding='latin-1')

    df.rename(columns=lambda x: x.strip().replace(" ", ""), inplace=True)
    df.fillna(df.median(), inplace=True)
    df.fillna(u'NaN', inplace=True)

    df['clx_id'] = df.index                             # index for search, etc
    df['clx_cluster'] = [0 for _ in range(len(df))]     # cluster init 0
    return df


@log('- Plot coordinates created.')
def scatterplot(coords, labels, ids):
    """
    Get a dictionary that will be converted to json, and contains all the scatterplot visualization data.
    Return example:
        {'nodes': [
               {'clx_cluster': 0, 'clx_id': 2, 'x': 2.658062848683116e-16, 'y': 1.7320508075688774},
               {'clx_cluster': 1, 'clx_id': 1, x': 1.5034503553765397, 'y': 0.0},
                ..............]}
    """
    nodes = [{
        'x': coord[0],
        'y': coord[1],
        'clx_cluster': int(label),
        'clx_id': int(_id)
        } for coord, label, _id in zip(coords, labels, ids)]

    return nodes
```

ДОДАТОК Б
(обов'язковий)

Презентація

УДК 004.896

Розподілена динамічна система кластеризації для видобутку просторових даних

Науковий керівник: к. т. н. Бобровнікова К.Ю.
Доповідач: Товстуха Е.В.

- **Об’єктом дослідження** є процес розподіленої динамічної кластеризації для видобутку просторових даних.
- **Предмет дослідження** є модель, метод розподіленої динамічної кластеризації для видобутку просторових даних та розподілена динамічна система кластеризації для видобутку просторових даних.
- **Метою роботи** є підвищення ефективності розподіленої динамічної кластеризації для видобутку просторових даних шляхом розроблення розподіленої динамічної системи кластеризації для видобутку просторових даних.

Задачі дослідження:

1) провести огляд відомих методів розподіленої динамічної кластеризації для видобутку просторових даних та визначити їх недоліки;

3) розробити метод розподіленої динамічної кластеризації для видобутку просторових даних, який би усував недоліки відомих методів та надав можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами;

3) на основі розробленого методу створити розподілену динамічну систему кластеризації для видобутку просторових даних, застосування якої надасть можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими системами.

Наукова новизна:

1) Удосконалено модель процесу розподіленої динамічної кластеризації для видобутку просторових даних, яка, на відміну від відомих моделей, використовує алгоритми стиснення та агрегації даних;

2) Удосконалено метод розподіленої динамічної кластеризації для видобутку просторових даних, який, на відміну від відомих методів, ґрунтується на побудованій моделі розподіленої динамічної кластеризації для видобутку просторових даних, застосовує методи стиснення даних та динамічної кластеризації та є основою розподіленої динамічної системи кластеризації для видобутку просторових даних. Застосування розробленого методу дозволить підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами.

Практична цінність дипломної роботи полягає в розробленні розподіленої динамічної системи кластеризації для видобутку просторових даних, яка надає можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими системами.

.

Актуальність задачі

1) Веб-дані дуже динамічні і змінюються надзвичайно швидкими темпами, тому видобуток цим способом робити не варто в більшості випадків.
2) В більшості алгоритмів видобутку даних припустимо, що дані безшумні. Очевидно, це дуже вагоме припущенняа,так як реальні дані далеко не вільні від шуму, включаючи веб-дані звичайно. Це значно порушує точність результатів.
3) Інтернет дані дуже багатовимірні, і простір пошуку зазвичай зростає експоненційно з кількістю розмірностей. Це дуже погано впливає на продуктивність багатьох алгоритмів майнінгу.
4) Інтелектуальний аналіз даних, частіше, використовує машинне навчання та статистичні методи для аналізу даних та / або даних фази попередньої обробки. Однак більшість із цих методів не є розробленими для дуже великих наборів даних, як у Big Data. Їх складність або експоненціальний або середній порядок многочлена. Норма для аналізу великих даних алгоритмів - це лінійна складність. Вибірка, кластеризація та розділення – це інші поширені методи, які використовуються для зменшення розміру набору даних, який буде видобуто. вони створюють інші проблеми, такі як повнота.

Метод розподіленої динамічної кластеризації для видобутку просторових даних

Основні кроки запропонованого методу:

1. Кожен вузол має набір даних, що представляє частину даних або загальний набір даних.

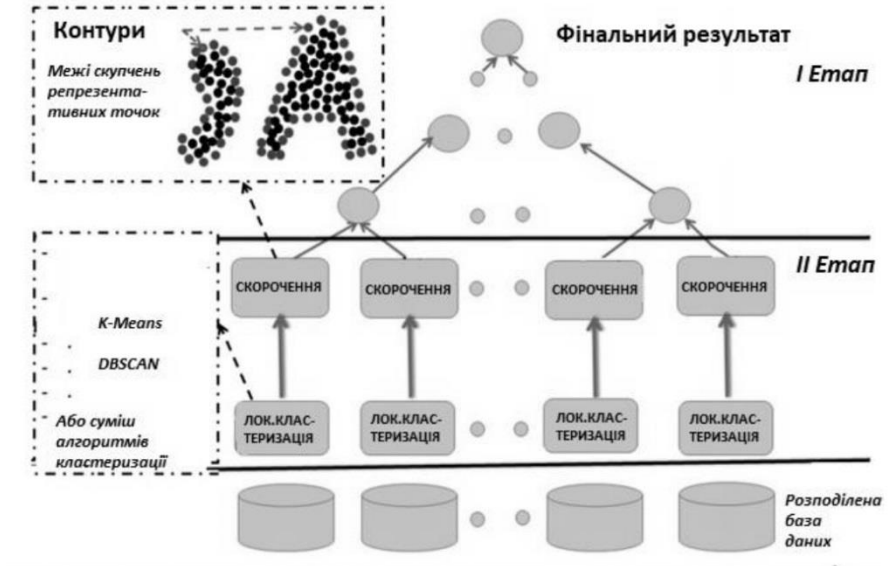
2. Кожен листовий вузол виконує локальний алгоритм кластеризації зі своїм входними параметрами.

3. Кожен вузол ділиться своїми кластерами із сусідами, щоб утворити більші кластери з використанням методу накладання даних.

4. Провідний вузол збирає результати своїх груп.

5. Повторювати пункти 3 та 4, поки не будуть сформовані всі глобальні кластери.

Метод розподіленої динамічної кластеризації для видобутку просторових даних



Модель процесу розподіленої динамічної кластеризації для видобутку просторових даних

Представимо функцію φ_1 наступним чином:

$$\varphi_1 = \langle s_1, s_2 \rangle,$$

де s_1 – етап розподілення просторових даних між компонентами розподіленої динамічної системи;

s_2 – етап агрегації результатів кластеризації.

Модель процесу розподіленої динамічної кластеризації для видобутку просторових даних

Опишемо множину алгоритмів кластеризації наступним кортежем:

$$\Pi = \langle \Xi, C, \varphi_2, \varphi_3 \rangle,$$

Ξ – множина об'єктів кластеризації;

C – множина кластерів, $C = \{c\}_{i=1}^{N_c}$, де N_c – кількість кластерів;

φ_2 – функція відстані до центру кластера;

φ_3 – функція кластеризації, $\Xi \xrightarrow{\varphi_3} C$.

Модель процесу розподіленої динамічної кластеризації для видобутку просторових даних

Представимо модель процесу кластеризації для видобутку просторових даних у вигляді кортежу:

$$\Psi = \langle \Omega, \Pi, O, \Theta, \varphi_1 \rangle,$$

де Ω – множина компоненів розподіленої динамічної системи,

$\Omega = \{\omega\}_{i=1}^{N\omega}$, де $N\omega$ - кількість компонентів розподіленої динамічної системи;

Π – множина алгоритмів кластеризації;

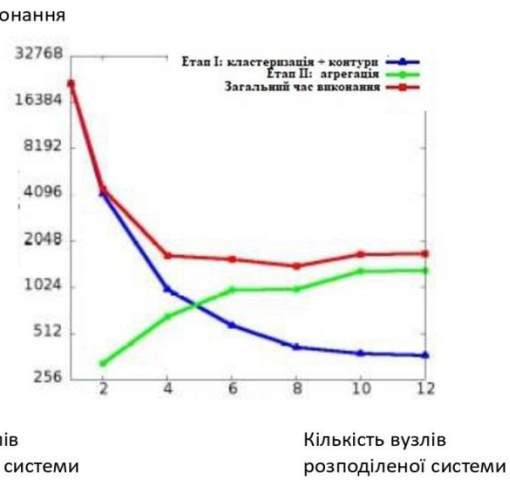
O – множина просторових даних, $O = \{o\}_{i=1}^{p}$, де p – кількість розділів просторових даних;

Θ – множина алгоритмів стиснення даних, $\Theta = \{\theta_1, \theta_2\}$, де θ_1 – алгоритм побудови α-форми, θ_2 – алгоритм побудови вектору балансування;

φ_1 – функція розподілення просторових даних між компонентами розподіленої динамічної системи з множини Ω та агрегації результатів кластеризації.

Результати експериментів

Використання K-means

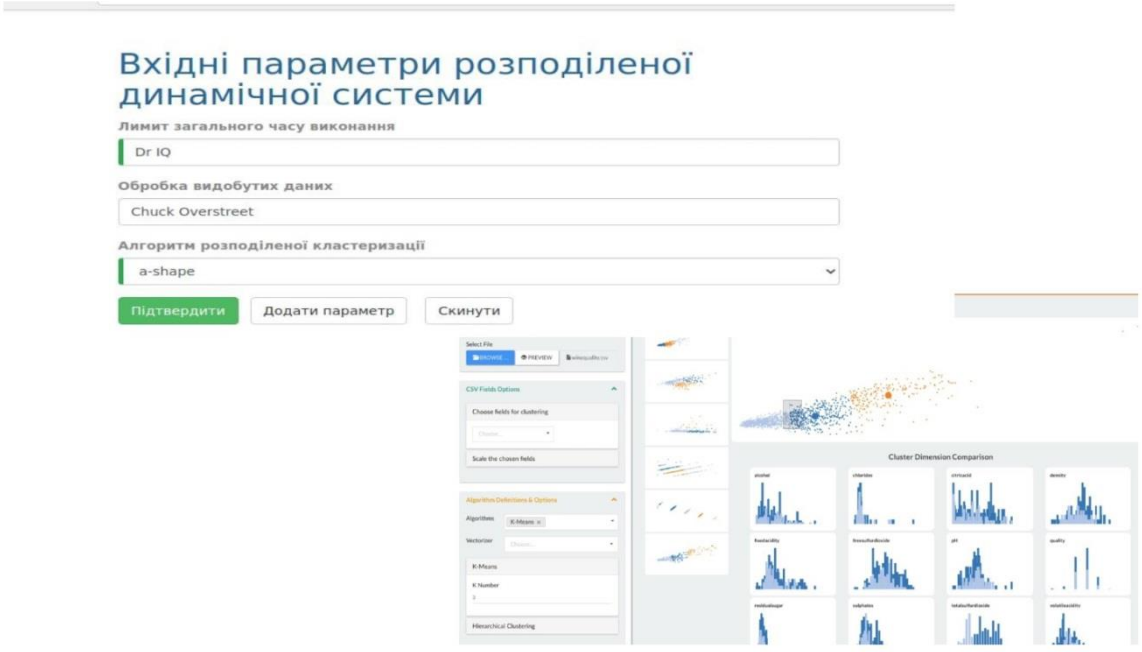


Результати експериментів

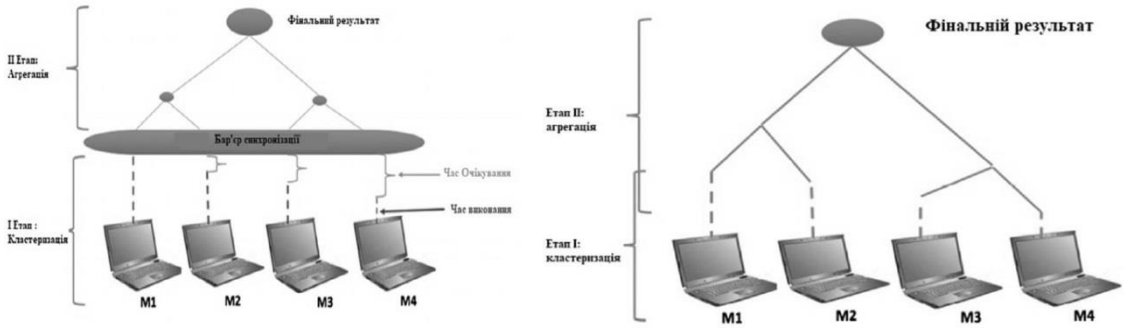
Використання K-means

Параметри експерименту		Синхронний			Асинхронний		
Машина	Розмір набору даних	Крок 1	Крок 2	Час	Крок 1	Крок 2	Час
M1	10 000	21 270	973	22 243	21 270	595	21 865
M2	1250	215	21 775	21 990	215	518	733
M3	1250	640	21383	22 023	640	20 100	20 740
M4	1250	304	21 730	22 034	304	497	801
M5	1250	161	22 034	22 195	161	394	555
M6	1250	171	21 856	22 027	170	286	456
M7	1250	245	21 918	22 163	245	509	754
M8	1250	185	21 854	22 039	185	858	1043
		Загальний час виконання		22 243	Загальний час виконання		21 865

Результати експериментів

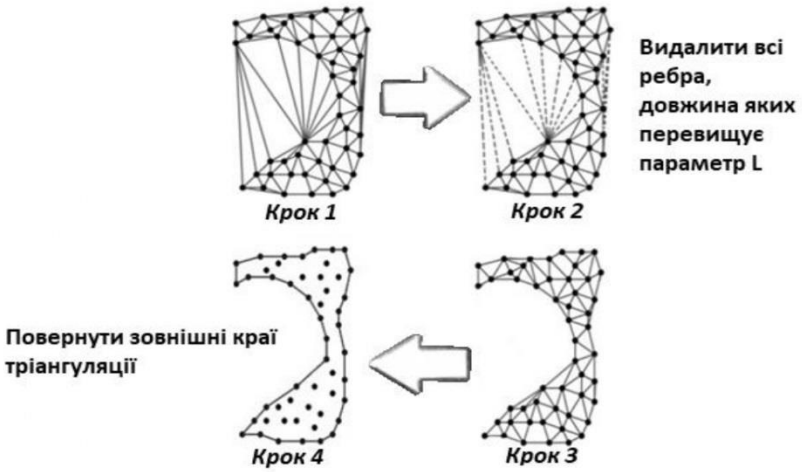


Метод розподіленої динамічної кластеризації для видобутку просторових даних



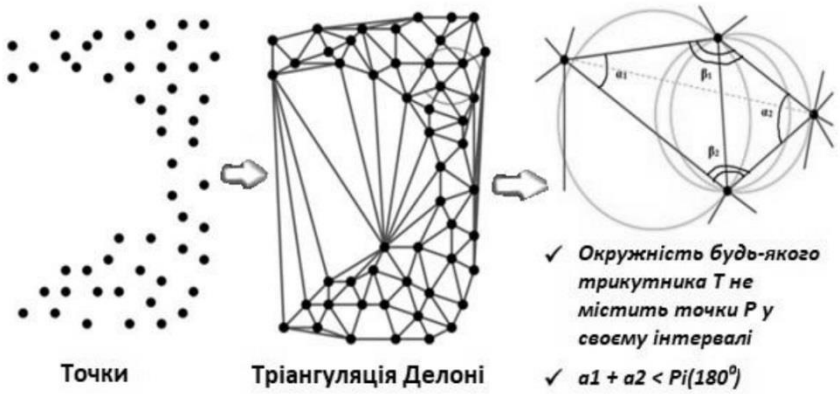
Метод розподіленої динамічної кластеризації для видобутку просторових даних

Кроки алгоритму побудови α-форми



Метод розподіленої динамічної кластеризації для видобутку просторових даних

Приклад побудови триангуляції Делоне



Метод розподіленої динамічної кластеризації для видобутку просторових даних

Кроки алгоритму побудови α-форми θ_1:

1. Сформувати триангуляцію Делоне з набором вхідних точок (P).

2. Видалити з триангуляції найдовший зовнішній край так, щоб:

2.1. Край, який потрібно видалити, був довший за параметр довжини l

2.2. Зовнішні краї результуючої тріангуляції утворюють межу простого багатокутника (скупчення точок).

3. Повторюйвати другий крок до тих пір, поки не буде видалено всі краї.

4. Повернути багатокутник (контур), утворений зовнішніми краями триангуляції.

Результати експериментів

Використання DBSCAN

Параметри експерименту		Синхронний			Асинхронний		
Машина	Розмір набору даних	Крок 1	Крок 2	Час	Крок 1	Крок 2	Час
M1	10 000	21 270	35978	57 248	21 270	905	22 175
M2	10 000	21 590	34 869	56 459	21 590	11513	33 103
M3	10 000	53 005	3008	56 013	53 005	3292	56 297
M4	10 000	32 424	24 691	57 115	32 424	6,996	39 420
M5	10 000	17 364	38 493	55857	17 364	4612	21 976
M6	10 000	15841	41 237	57 078	15841	2066	17 907
M7	10 000	38 732	18 483	57 215	38727	18 459	57186
M8	1250	185	56 915	57 100	184	16 077	16 261
		Загальний час виконання		57 248	Загальний час виконання		57186

Висновки

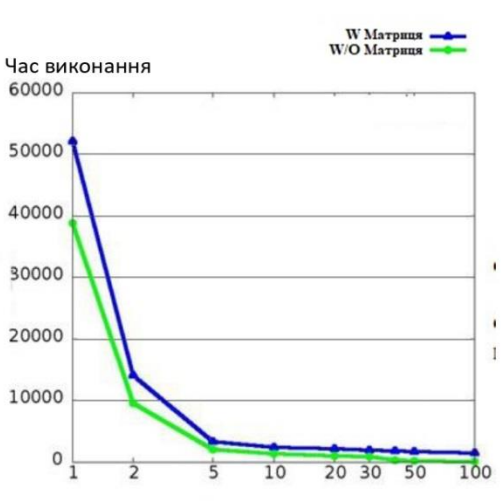
1) Проведено огляд відомих методів розподіленої динамічної кластеризації для видобутку просторових даних та визначено їх недоліки;

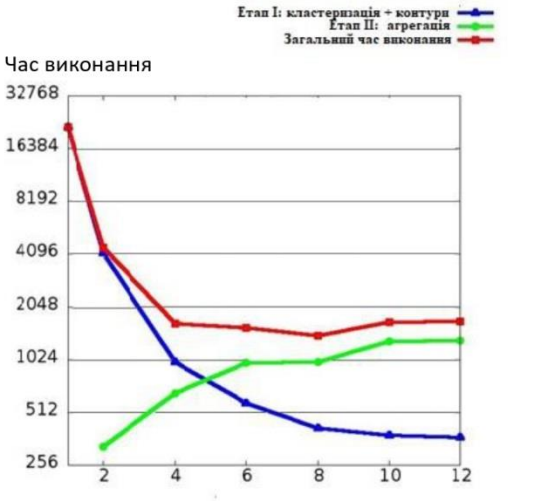
3) Розроблено метод розподіленої динамічної кластеризації для видобутку просторових даних, який, на відміну від відомих методів, використовує алгоритми стиснення та агрегації даних та усуває недоліки відомих методів, що надає можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами;

3) На основі розробленого методу створено розподілену динамічну систему кластеризації для видобутку просторових даних, застосування якої надало можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими системами.

Результати експериментів

Використання DBSCAN





Дякую за увагу!

ДОДАТОК В
(обов’язковий)

Стаття, опублікована у фаховому виданні

ISSN 2710-0766
DOI 10.31891/CSIT

THE INTERNATIONAL SCIENTIFIC JOURNAL

COMPUTER SYSTEMS AND INFORMATION TECHNOLOGIES

Volume 1 ***No 1-2020***

МІЖНАРОДНИЙ НАУКОВИЙ ЖУРНАЛ

КОМП'ЮТЕРНІ СИСТЕМИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

2020

КОМП'ЮТЕРНІ СИСТЕМИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

МІЖНАРОДНИЙ НАУКОВИЙ ЖУРНАЛ

Засновано в 2020 р. *Виходить 2 рази на рік*

Хмельницький, 2020, № 1 (1)

Засновник і видавець: Хмельницький національний університет

Журнал включено до наукометричних баз:

Google Scholar https://scholar.google.com.ua/citations?hl=uk&user=HW1XpMsAAAAJ

Рекомендовано до друку рішенням Вченої ради Хмельницького національного університету, протокол № 1 від 26.08.2020

Адреса редакції: **Україна, 29016,**
м. Хмельницький, вул. Інститутська, 11,
Хмельницький національний університет
редакція журналу "Компютерні системи та інформаційні технології"

☎ **(0382) 67-51-08**

e-mail: csit.khnu@gmail.com

web: http://csitjournal.khmnu.edu.ua/
http://lib.khnu.km.ua/csit_khnu.htm

Зареєстровано Міністерством юстиції України
Свідоцтво про державну реєстрацію друкованого засобу масової інформації
Серія КВ № 24512-14452Р від 20 липня 2020 року

ЗМІСТ

International scientific journal **«Computer systems and information technologies»**

УДК 004.896
DOI: 10.31891/CSIT-2020-1-7

БОБРОВНІКОВА К. Ю., ТОВСТУХА Е. В.
Хмельницький національний університет

МЕТОДИ ЗАБЕЗПЕЧЕННЯ ЕНЕРГОЕФЕКТИВНОСТІ ТА ЕНЕРГОЗБЕРЕЖЕННЯ В СИСТЕМІ РОЗУМНОГО БУДИНКУ

Розумний будинок (smart home) – це система управління основними процесами життєзабезпечення як невеликих систем (комерційні, офісні приміщення, квартири, котеджі), так і великих автоматизованих комплексів (торгові і промислові комплекси). Однією з важливих задач, на вирішення яких спрямована концепція сучасного розумного будинку, є проблема енергоефективності і енергозбереження. Ефективний контроль опалення, вентиляції, системи кондиціонування повітря, більш ефективне використання традиційних приладів та впровадження енергоефективного обладнання в будівлі мають важливе значення для забезпечення продуктивної, здорової та безпечної праці та життя, відіграють важливу роль у запобіганні втрат енергії, а також зменшують вплив на навколишнє середовище. Крім того, підвищення ефективності управління споживанням енергії є єдиним підходом забезпечення енергоефективності та енергозбереження багатьох існуючих будівель, які не можуть бути модернізовані згідно з вимогами сучасних будівельних технологій. В роботі представлено огляд сучасних методів і технологій, спрямованих на забезпечення енергоефективності та енергозбереження в системі розумного будинку.

Ключові слова: енергоефективність, енергозбереження, розумний будинок, Інтернет речей, автоматизація будівель.

BOBROVNIKOVA K., TOVSTUKHA E.
Khmelnitsky National University

METHODS FOR ENERGY EFFICIENCY AND ENERGY SAVING IN THE SMART HOME SYSTEM

Today, the efficient use of energy resources is one of the most important tasks. The fastest growing sector of energy consumption in the world is electricity, which is projected to grow by 56% by 2035, and in developed countries almost all the growth is due to the generation and consumption of electricity and heat. Further growth of energy consumption by the population is also expected. At the same time, almost a third of the total energy consumption is made up of certain losses, ie energy is consumed for other purposes. Against the background of global growth in energy consumption, the rate of further accumulation of CO_2 emissions will increase. That is why the European Union, United Nations bodies, international financial organizations and the International Energy Agency give priority to energy efficiency issues. To this end, a set of mechanisms and practical tools for economic stimulation of measures to implement modern energy-saving technologies is used at the international level.

Smart home is a system for managing the basic life support processes of both small systems (commercial, office premises, apartments, cottages) and large automated complexes (commercial and industrial complexes). One of the important tasks to be solved by the concept of a modern smart home is the problem of energy efficiency and energy saving. Effective control of heating, ventilation, air conditioning, more efficient use of traditional appliances and the introduction of energy-efficient equipment in the building are important to ensure productive, healthy and safe work and life of residents, play an important role in preventing energy loss and reduce impact on the environment. In addition, improving the efficiency of energy management is the only approach to ensuring the energy efficiency and energy saving of many existing buildings that cannot be upgraded according to the requirements of modern construction technologies. The paper presents an overview of modern methods and technologies aimed at ensuring energy efficiency and energy saving in the smart home system.

Keywords: energy efficiency, energy saving, smart home, Internet of things, building automation.

Вступ. На сьогодні ефективність використання енергетичних ресурсів є однією з найважливіших задач. Найбільш швидкозростаючим сектором споживання енергоресурсів в світі є електроенергетика, де прогнозується 56 % приросту до 2035 р., причому у розвинених країнах майже весь приріст припадає на генерацію та споживання електро- та теплоенергії [1]. Також очікується подальше продовження зростання енергоспоживання населенням. При цьому майже третину від загального споживання енергії складають ті чи інші втрати, тобто енергія витрачається не за призначенням [2]. На фоні загальносвітового зростання споживання енергетичних ресурсів зростатимуть темпи подальшого накопичення викидів CO_2. Тому Європейський Союз, органи Організації Об'єднаних Націй, Міжнародні фінансові організації та Міжнародне енергетичне агентство надають питанням енергоефективності пріоритетного значення. З цією метою на міжнародному рівні задіяно комплекс механізмів та практичних інструментів економічного стимулювання заходів із впровадження сучасних енергозберігаючих технологій [1].

В [3] термін «енергозбереження» визначено як діяльність (організаційна, наукова, практична, інформаційна), яка спрямована на раціональне використання та ощадливе витрачання первинної та перетвореної енергії і природних енергетичних ресурсів і яка реалізується з використанням технічних, економічних та правових методів. Згідно [4] енергетична ефективність будівлі – це властивість будівлі, що характеризується кількістю енергії, необхідної для створення належних умов проживання та/або життєдіяльності людей у такій будівлі. Енергоефективний проект спрямований на скорочення енергоспоживання, а саме: реконструкція мереж і систем постачання, регулювання і облік споживання води, газу, теплової та електричної енергії, модернізація огороджувальних конструкцій та технологій виробничих процесів [4]. Проблеми ефективного енергоспоживання в будівлях регламентовані ДСТУ Б EN 15232:2011 «Енергоефективність будівель – Вплив автоматизації, моніторингу та управління будівлями» [5]. Цей стандарт є тотожним перекладом EN 15232:2007 «Energy performance of buildings – Impact of Building Automation, Controls and Building Management», розробленого Європейським технічним комітетом стандартизації CEN/TC 247 Building Automation, Controls and Building Management (Автоматизація,

моніторинг та управління будівлями), та містить вимоги відповідно до чинного законодавства України. В [5] викладено основні аспекти та нормативи енергетичних характеристик будівель, їх автоматизації та менеджменту: (1) структурний перелік функцій автоматизації, моніторингу та управління будівлями, а також технічного управління інженерними системами, які мають вплив на енергоефективність будівель; (2) метод визначення мінімальних вимог щодо використання функцій автоматизації, моніторингу та управління, які повинні бути впроваджені в будівлях різного рівня складності; (3) деталізовані методи оцінювання впливу вказаних функцій для конкретних будівель. Ці методи надають можливість ввести вплив вказаних функцій у розрахунки рейтингів енергоефективності будівель, а також показників, що розраховуються згідно з чинними стандартами; (4) спрощений метод для отримання первинної оцінки впливу вказаних функцій для типових будівель. Стандарт регламентує такі складові автоматизації будівель як: (1) управління та моніторинг опалення та охолодження; (2) управління та моніторинг систем вентиляції; (3) управління та моніторинг систем освітлення; (4) управління та моніторинг систем жалюзі.

Підвищення уваги до проблем енергоефективності та енергозбереження сприяє розвитку концепції сучасного розумного будинку. Якщо спочатку ця концепція полягала у підключенні давачів, приладів та пристроїв через мережу з метою віддаленого моніторингу, доступу та контролю житлового середовища та надання необхідних послуг користувачам, то на сучасному етапі передбачає також оптимальне використання енергії в будинках. Згідно [6] ідея розумного будинку полягає в тому, що комп'ютерне програмне забезпечення, яке грає роль інтелектуального агента, сприймає стан фізичного середовища та мешканців за допомогою давачів, а потім вживає дій для досягнення визначених цілей, таких як максимізація комфорту мешканців, мінімізація споживання ресурсів, збереження здоров'я та безпека будинку та мешканців.

Аналіз останніх досліджень та публікацій. Сьогодні в наукових джерелах широко представлені різноманітні підходи, спрямовані на забезпечення енергоефективності та енергозбереження в системі розумного будинку. В [7] проведено огляд сучасної концепції розумного будинку. Зазначено, що в останні роки основним напрямком політики в галузі енергоефективності було сприяння використанню більш ефективних приладів та компонентів. Проте контроль автоматизації будинку відіграє важливу роль для ефективної та сталої експлуатації. Роль автоматизації будівель особливо важлива в операціях, спрямованих на стабільний комфорт у приміщенні та енергоефективне управління системою будинку: (1) шляхом виявлення та усунення втрат енергії; (2) шляхом використання енергії лише в потрібній кількості, місці та лише в той час, коли вона потрібна; (3) шляхом здійснення правильного контролю функціонального рівня системи для правильного застосування в потрібному місці.

В [8] проведено аналіз переваг та ризиків технологій розумного будинку з різних точок зору. Основними ризиками визначено необхідність поступитися автономністю та незалежністю в будинку за умов посилення технологічного контролю. Іншим ризиком є недостатня увага розробників в галузі технологій розумного будинку до заходів з підвищення рівня довіри споживачів до безпеки та конфіденційності даних. Правильна політика керування ризиками може відіграти важливу роль у пом'якшенні цих ризиків та підтримці потенціалу підвищення енергоефективності в галузі технологій розумного будинку. Заходи політики щодо підтримки розвитку ринку технологій розумного будинку включають стандарти проектування та експлуатації, вказівки щодо даних та їх конфіденційності, контроль якості та регіональні дослідницькі програми.

В [9] розглянуто вимоги до енергетичних показників та проблеми модернізації розумного будинку в умовах європейського холодного клімату. Зазначено, що у європейському секторі житлових будинків енергоспоживання систем опалення та охолодження займає близько 80% від загального споживання. Також обговорюються заходи щодо зниження, контролю та моніторингу енергоспоживання в будинку. В роботі розглянуто різні підходи до інтелектуального управління системами будинку, а також проаналізовано, як управління та автоматизація системи управління будинком можуть зменшити споживання енергії будівлі. Одним з підходів для підвищення продуктивності системи управління будинком та зменшення енергоспоживання є контроль з прямим зв'язком (feed-forward control) [9]. Така система безпосередньо компенсує фактори завад, такі як зовнішня температура, вітер, сонячне випромінювання, внутрішній приріст тепла, шляхом виміру коефіцієнтів завад в режимі реального часу для здійснення відповідних заходів на основі відомих параметрів. Іншим підходом є предиктивне керування на основі моделей (model-based predictive control – MPC) [9] – це структурований підхід, який прогнозує майбутню поведінку системи на основі моделей і відповідно коригує систему. Така система добре підходить для оптимізації, наприклад, для збалансування витрат та енергоефективності шляхом мінімізації пікових навантажень тощо. Недоліком MPC є складність моделювання, збору, обробки та моніторингу даних, що робить застосування підходу нерентабельним для житлових приміщень та невеликих комерційних будівель.

Управління на основі нечіткої логіки не потребує складної математичної моделі для управління системою і може базуватися безпосередньо на якісному досвіді користувача. Недоліком підходу є складність визначення оптимальних правил та функцій приналежності для таких систем [9]. Іншим відомим підходом для побудови систем управління будинку є штучні нейронні мережі (artificial neural networks, ANN), які широко застосовуються для моделювання і прогнозування використання енергії в будівлях. Штучні

нейронні мережі здатні моделювати нелінійні процеси, постійно пристосовуватися до нових даних та вчитися на основі цих даних з метою вирішення складних задач [9]. Також відомі гібридні підходи, засновані на використанні нечіткої логіки та штучних нейронних мереж, які поєднують переваги обох підходів – наслідування людській логіці та здатність до навчання. Адаптивні нейронечіткі системи (adaptive neuro-fuzzy – ANF) реалізують алгоритми навчання нейронної мережі для налаштування функцій приналежності в нечіткій системі. В системі управління на основі агентів, які є віртуальними або фізичними модулями, агенти співпрацюють з навколишнім середовищем шляхом сприймання та впливу на параметри за допомогою штучного інтелекту [9]. Такі системи здатні знаходити компроміс між споживанням енергії, витратами і комфортом шляхом вимірювань та взаємодії з середовищем і контролювання систем опалення, вентиляції та кондиціонування повітря та електричних приладів. Мультиагентні системи спроможні вирішувати складні задачі шляхом навчання за шаблонами та реагувати на зміни у приміщенні в режимі реального часу.

Зазначено, що найважливішими завданнями системи управління є мінімізація показників [9]: (1) вплив завад на вихідну змінну; (2) відхилення (похибка) між вихідною змінною і опорною змінною (задана точка); (3) час реакції, коли виявлена помилка. Зазвичай системи управління потребують налаштування для забезпечення оптимальної продуктивності, що може потребувати витрат часу, проте адаптивні контролери мають можливість постійно самоналаштовуватися, пристосовуючись до різних умов у будівлях.

В дослідженні [10] проведено аналіз відомих систем управління енергією будинку з метою виявлення ключових відмінностей щодо їх функціональності та якості. Визначено можливості для енергозбереження (як поведінкового, так і експлуатаційного), проте зазначено, що у багатьох випадках потенційні переваги, пов'язані із зручністю, комфортом чи безпекою, можуть обмежувати реалізацію сценаріїв заощадження енергії. Це пов'язано з нестачею інформації, що стосується енергозбереження, яка надається користувачам, а також недостатнім розумінням способів взаємодії користувачів з додатковою інформацією та наданою їм можливістю контролю.

В роботі [11] представлено систему для управління енергоспоживанням (Energy Management System – EMS) для розумного будинку. В запропонованій системі кожен домашній пристрій поєднується з модулем збору даних, що представляє собою об'єкт IoT з унікальною IP-адресою, і всі пристрої утворюють велику бездротову мережу. Модуль збору даних (System on Chip – SoC) збирає дані про споживання енергії з кожного пристрою, з кожного розумного будинку, з усіх житлових районів, та передає дані на централізований сервер для подальшої обробки та аналізу. Ця інформація накопичується на сервері як великі дані (Big Data). Для управління споживанням енергії та задоволення попиту споживачів EMS використовує пакети програмного забезпечення Business Intelligence (BI) та Big Data analytics. В якості тестового обладнання для експериментального дослідження роботи запропонованої системи було використано прилади для опалення, вентиляції та кондиціонування.

В [12] запропоновано енергоефективну мультисенсорну кібер-фізичну систему розумного будинку для людей похилого віку, яка використовує потенціал хмарних обчислень та технологій великих даних. Зважаючи на те, що літній людині може знадобитися допомога у виконанні складних дій з підтримання енергоефективності, розроблено програмне забезпечення розумного мультимедійного помічника, що дозволяє людині спостерігати за різними енергоефективними процесами, керувати розумними побутовими приладами за допомогою жестів, отримувати сповіщення щодо статусів пристрою тощо. Для обробки даних на сервері вилучаються низькорозмірні ознаки, доступні з давачів та мультимедіа. Для сприяння прийняттю рішень також на сервері автоматично класифікуються події. З метою заощадження енергії розроблено автоматичну систему управління увімкненням / вимкненням за допомогою мови або жестів. В якості команд управління використовуються ключові слова, які вилучаються безпосередньо з мови. Експериментальні результати показують обнадійливий потенціал для розгортання запропонованого фреймворку у великих масштабах.

Підвищення ефективності споживання енергії, як правило, впливає на рівень комфорту користувача. В [13] представлено адаптивно-інтелектуальний інструмент управління енергією (Adaptive-Smart Energy Management Tool, A-SEM) для збалансування рівня комфорту та ефективності споживання енергії. Підхід використовує алгоритм адаптивного обмеження енергії, який побудований на основі визначення середньодобового споживання енергії за 30 днів. На споживання енергії впливає поведінка користувача, яка контролюється системою. Система використовує різні давачі для контролю споживання енергії та розпізнає поведінку користувача, щоб адаптувати обмеження для забезпечення комфорту. Користувач розумного будинку може встановити свій щомісячний бюджет споживання енергії, який впливає на початковий рівень щоденного обмеження енергії. A-SEM здійснює моніторинг та контроль у режимі реального часу, головним чином враховуючи показники, оцінені на етапі тестування.

Обмеження споживання енергії розраховується за формулою:

$$EL_n = \frac{(EL_{n-1} \times n) + EC_{n-1}}{m}, \quad (1)$$

де EL_n – обмеження енергії в день n;

EL_{n-1} – обмеження енергії в день $n-1$;

EC_{n-1} – споживання енергії в день $n-1$;

n – кількість днів;

$m = n + 1$ – з максимальним значенням n, рівним 30.

Оцінки енергозбереження, рівня комфорту та ефективності обчислюються за наступними формулами відповідно:

$$Energy\ saving\ (\%) = \left(1 - \frac{\sum_1^{30} Em}{\sum_1^{30} En}\right) \times 100\%,\ Comfort\ level = \frac{\sum_1^{30} Em - \sum_1^{30} EL}{\sum_1^{30} En - \sum_1^{30} EL},\ Efficiency\ level = \frac{\sum_1^{30} En - \sum_1^{30} Em}{\sum_1^{30} En - \sum_1^{30} EL}. \quad (2)$$

Проведені експерименти показали, що алгоритм обмеження споживання енергії «адаптивного граничного рівня» успішно врівноважує рівень комфорту та енергоефективності, що відповідає належному рівню комфорту користувача, а також досягає певного заощадження енергії. Застосування підходу надає можливість забезпечити заощадження енергії на рівні 2,62 %, що може збільшуватися або зменшуватися в залежності від поведінки користувача.

В [14] представлено систему IntelliHome, яка спрямована на зниження споживання електричної енергії в розумному будинку. З цією метою система використовує технології аналізу великих даних, методи машинного навчання та статистики, щоб надати користувачам корисну інформацію про їх звички споживання електроенергії та активно залучити їх до процесу енергозбереження за допомогою інформації, що надається в режимі реального часу, та рекомендацій щодо енергозбереження.

На сьогоднішній день має місце нестача операційних систем, які б надавали можливість інтеграції пристроїв, що складають середовище розумного будинку. Проблема пов'язана з тим, що розумні пристрої засновані на модулях самообслуговування та використовують незалежні IoT платформи, розроблені різними виробниками. Це призводить до необхідності керування кожним приладом окремо, що знижує енергоефективність будинку та збільшує кількість трафіку в мережі. Для вирішення цієї проблеми в [15] запропоновано інтегровану систему управління, яка поєднує IoT пристрої в єдину систему. Для ефективного управління розроблено три інтелектуальні моделі в якості сервісів додатку IoT платформи для розумного будинку: (1) інтелектуальне врахування особливостей задачі (intelligence awareness target, IAT), що слугує для збору даних з середовища та використовує інтелектуальне навчання для набуття ситуаційної обізнаності про значення даних, генерованих сенсорами; (2) інтелектуальна енергоефективність (intelligence energy efficiency, IE^2S), що виконує роль сервера і заснована на платформі з відкритим кодом Mobius та відкритій програмній бібліотеці для машинного навчання TensorFlow для обробки даних, зібраних IAT, а також аналізу шаблонів з метою автоматичного надання послуг; (3) інтелектуальний сервіс TAS (IST), який слугує для контролю та управління на етапі обслуговування користувача. Ці три інтелектуальні моделі дозволяють взаємодіяти пристроям IoT у розумному будинку і вирішувати проблеми перевантаженості мережі та втрат енергії за рахунок скорочення непотрібних задач та використання енергії шляхом керування задачами за відповідними шаблонами.

В [16] пропонується гібридна інтелектуальна система, яка використовує туманні обчислення для досягнення енергоефективності в розумному будинку. Гібридна архітектура системи поєднує інтелект реагування для швидкої адаптації до середовища, що постійно змінюється, та деліберативний інтелект для виконання комплексного навчання та оптимізації. Така архітектура надає можливість системі бути адаптивною, реагуючи в режимі реального часу на відповідні події, що відбуваються в навколишньому середовищі, і в той же час постійно вдосконалювати свою ефективність, вивчаючи потреби користувачів. Результати експериментів показують, що система забезпечує значне заощадження енергії при управлінні розумним середовищем, одночасно задовольняючи потреби та вподобання користувачів. Баланс між виконанням переваг користувача та заощадженням енергії встановлюється відповідно до рівняння:

$$F = \alpha \times energy\_saving + (1-\alpha) \times user\_satisfaction, \quad (3)$$

де $energy\_saving \in [0,1]$ – інтервал, визначений як нормалізована різниця між верхньою межею допустимого споживання енергії та поточним витрачанням енергії. Якщо значення α близьке до 0, то система надає пріоритет перевагам користувача; якщо значення α близьке до 1, то збільшується важливість заощадження енергії. Основними недоліками системи є: непридатність для керування сценаріями, розрахованими на багатьох користувачів, а також у випадках, коли сценарії різних користувачів можуть впливати один на одного; протиріччя індивідуальних потреб користувача; необхідність збору вихідних даних для налаштування нечітких контролерів для визначення нечітких функцій приналежності інтелекту реагування.

В [17] запропоновано комплексний підхід, який полягає у побудові частково-цілочисельної моделі квадратичного програмування для схеми керування з прогнозуванням, заснованої на температурній моделі

будівлі та системі управління енергією будівлі. Ключовим аспектом запропонованої моделі прогнозуючого контролера є обчислення оптимальних рішень під час опрацювання неперервних двійкових обмежень, а також змінних та врахування як теплової, так і електричної складової розумного будинку. Система включає також прогнозування завад, функцій будівлі та індивідуальних ваг користувачів, причому прогнозування функцій будівлі базується на неконтрольованому методі. Система надає можливість здійснювати управління такими приладами як опалення, акумулятор, морозильна камера, посудомийна машина, фотоелектричні системи, а також здійснювати покупки та продажі через розумну мережу. Оптимальне використання теплоємності будівлі допомагає мінімізувати необхідну ємність акумулятора.

В [18] представлена система HEMS-IoT для управління енергією розумного будинку з метою забезпечення домашнього комфорту, безпеки та збереження енергії, заснована на великих даних та машинному навчанні. Система використовує алгоритм машинного навчання J48 та API Weka для вивчення поведінки користувачів та моделей споживання енергії, а також класифікації будинків відповідно до споживання енергії. Для створення енергозберігаючих рекомендацій і одночасного збереження комфорту та безпеки розумного будинку, заснованих на вподобаннях користувачів, використовуються RuleML та Apache Mahout.

Висновки. В роботі проведено огляд сучасних методів забезпечення енергоефективності та енергозбереження в системі розумного будинку. Огляд літератури показав, що проблема енергоефективності та енергозбереження є надзвичайно актуальною. Відомі підходи, спрямовані на вирішення цієї проблеми, мають ряд недоліків та обмежень, такі як: складність моделювання, збору, обробки та моніторингу даних; складність визначення оптимальних правил та функцій приналежності; потреба в попередньому налаштуванні, що може вимагати значних витрат часу; необхідність поступитися автономністю, незалежністю та комфортом умов проживання за умов посилення технологічного контролю; потенційні переваги, пов'язані із зручністю, комфортом чи безпекою, можуть обмежувати реалізацію сценаріїв заощадження енергії; протиріччя індивідуальних потреб користувача. Отже, можна зробити висновок, що існує необхідність у розробленні нових методів забезпечення енергоефективності та енергозбереження в системі розумного будинку, які б усували недоліки відомих підходів.

Література

1. Національна енергетична компанія «Укренерго». Аналіз законодавства провідних зарубіжних країн та України щодо ефективного використання енергетичних ресурсів – [Електронний ресурс] – Режим доступу: https://ua.energy/wp-content/uploads/2018/01/1.-Efektyvne-vykorystannya-energoresursiv.pdf. – 2.07.2020 р.

2. U.S. Energy Information Administration (EIA). International Energy Outlook 2019 – [Електронний ресурс] – Режим доступу: https://www.eia.gov/outlooks/ieo/. – 2.07.2020 р.

3. Верховна Рада України. Законодавство України. Про енергозбереження. Закон України [Документ 74/94-ВР, чинний, поточна редакція – Редакція від 23.07.2017, підстава - 2095-VIII]. – [Електронний ресурс] – Режим доступу: https://zakon.rada./gov.ua/laws/show/74/94-%D0%B2%D1%80#Text. – 2.07.2020 р.

4. Верховна Рада України. Законодавство України. Про енергетичну ефективність будівель. Закон України від 22.06.2017 № 2118-VIII [Чинний, поточна редакція — Прийняття від 22.06.2017]. – [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/laws/show/2118-19#Text. – 2.07.2020 р.

5. Енергоефективність будівель – Вплив автоматизації, моніторингу та управління будівлями (EN 15232:2007, IDT) : ДСТУ Б EN 15232:2011. – [Чинний з 2012-04-01]. – К.: Мінрегіон України, 2012. – 115 с. –(Національний стандарт України).

6. Cook, D. J. How smart is your home? – Science.– 2012. – Vol. 335, Issue 6076. – pp. 1579-1581.

7. Fabi, V. Insights on smart home concept and occupants' interaction with building controls / V. Fabi, G. Spigliantini, S. P. Corgnati // Energy Procedia. – 2017. – Vol. 111. – pp. 759-769.

8. Wilson, C. Benefits and risks of smart home technologies / C. Wilson, T. Hargreaves, R. Hauxwell-Baldwin // Energy Policy. – 2017. – Vol. 103. – pp. 72-83.

9. Felius, L. C. Retrofitting towards energy-efficient homes in European cold climates: a review / L. C. Felius, F. Dessen, B. D. Hrynyszyn // Energy Efficiency. – 2020. – Vol. 13, Issue 1. – pp. 101-125.

10. Ford, R. Categories and functionality of smart home technology for energy management / R. Ford, M. Pritoni, A. Sanguinetti, B. Karlin // Building and environment. – 2017. – Vol.123. – pp. 543-554.

11. Al-Ali, A. R. A smart home energy management system using IoT and big data analytics approach / A. R. Al-Ali, I. A. Zualkernan, M. Rashid, R. Gupta, M. Alikarar // IEEE Transactions on Consumer Electronics. – 2017. – Vol. 63, Issue 4. – pp. 426-434.

12. Hossain, M. S. Cyber–physical cloud-oriented multi-sensory smart home framework for elderly people: An energy efficiency perspective / M. S. Hossain, M. A. Rahman, G. Muhammad // Journal of Parallel and Distributed Computing. – 2017. – Vol. 103. – pp. 11-21.

13. Isnen, M. A-SEM: An adaptive smart energy management testbed for shiftable loads optimisation in the smart home / M. Isnen, S. Kurniawan, E. Garcia-Palacios // Measurement. – 2020. – Vol. 152, 107285.

14. Paredes-Valverde, M. A. IntelliHome: An internet of things-based system for electrical energy saving in smart home environment / M. A. Paredes-Valverde, G. Alor-Hernández, J. L. García-Alcaráz, M. D. P. Salas-Zárate, L. O. Colombo-Mendoza, J. L. Sánchez-Cervantes // Computational Intelligence. – 2020. – Vol. 36, Issue 1. – pp. 203-224.

15. Jo, H. Intelligent smart home energy efficiency model using artificial TensorFlow engine / H. Jo, Y. I. Yoon // Human-centric Computing and Information Sciences. – 2018 – Vol. 8, Issue 1. – pp. 1-18.

16. De Paola, A. A fog-based hybrid intelligent system for energy saving in smart buildings / A. De Paola, P. Ferraro, G. L. Re, M. Morana, M. Ortolani // Journal of Ambient Intelligence and Humanized Computing. – 2020. – Vol. 11, Issue 7. – pp. 2793-2807.

17. Killian, M. Comprehensive smart home energy management system using mixed-integer quadratic-programming / M. Killian, M. Zauner, M. Kozek // Applied energy. – 2018. – Vol. 222 . – pp. 662-672.

18. Machorro-Cano, I. HEMS-IoT: A Big Data and Machine Learning-Based Smart Home System for Energy Saving / I. Machorro-Cano, G. Alor-Hernández, M. A. Paredes-Valverde, L. Rodríguez-Mazahua, J. L. Sánchez-Cervantes, J. O. Olmedo-Aguirre // Energies. – 2020. – Vol. 13, Issue 5. – pp. 1097.

References

1. Nacionalna energetichna kompaniya «Ukrenergo». Analiz zakonodavstva providnih zarubizhnih krayin ta Ukrayini shodo efektivnogo vikoristannya energetichnih resursiv – [Elektronnij resurs] – Rezhim dostupu: https://ua.energy/wp-content/uploads/2018/01/1.-Efektyvne-vykorystannya-energoresursiv.pdf. – 2.07.2020 r.
2. U.S. Energy Information Administration (EIA). International Energy Outlook 2019 – [Elektronnij resurs] – Rezhim dostupu: https://www.eia.gov/outlooks/ieo/. – 2.07.2020 r.
3. Verhovna Rada Ukrayini. Zakonodavstvo Ukrayini. Pro energozberezhennya. Zakon Ukrayini [Dokument 74/94-VR, chinnij, potochna redakciya – Redakciya vid 23.07.2017, pidstava - 2095-VIII]. – [Elektronnij resurs] – Rezhim dostupu: https://zakon.rada.gov.ua/laws/show/74/94-%D0%B2%D1%80#Text. – 2.07.2020 r.
4. Verhovna Rada Ukrayini. Zakonodavstvo Ukrayini. Pro energetichnu efektivnist budivel. Zakon Ukrayini vid 22.06.2017 № 2118-VIII [Chinnij, potochna redakciya — Prijnyattya vid 22.06.2017]. – [Elektronnij resurs] – Rezhim dostupu: https://zakon.rada.gov.ua/laws/show/2118-19#Text. – 2.07.2020 r.
5. Energoefektivnist budivel – Vpliv avtomatizaciyi, monitoringu ta upravlinnya budivlyami (EN 15232:2007, IDT) : DSTU B EN 15232:2011. – [Chinnij z 2012-04-01]. – K.: Minregion Ukrayini, 2012. – 115 s. –(Nacionalnij standart Ukrayini).
6. Cook, D. J. How smart is your home? – Science.– 2012. – Vol. 335, Issue 6076. – pp. 1579-1581.
7. Fabi, V. Insights on smart home concept and occupants' interaction with building controls / V. Fabi, G. Spigliantini, S. P. Corgnati // Energy Procedia. – 2017. – Vol. 111. – pp. 759-769.
8. Wilson, C. Benefits and risks of smart home technologies / C. Wilson, T. Hargreaves, R. Hauxwell-Baldwin // Energy Policy. – 2017. – Vol. 103. – pp. 72-83.
9. Felius, L. C. Retrofitting towards energy-efficient homes in European cold climates: a review / L. C. Felius, F. Dessen, B. D. Hrynyszyn // Energy Efficiency. – 2020. – Vol. 13, Issue 1. – pp. 101-125.
10. Ford, R. Categories and functionality of smart home technology for energy management / R. Ford, M. Pritoni, A. Sanguinetti, B. Karlin // Building and environment. – 2017. – Vol.123. – pp. 543-554.
11. Al-Ali, A. R. A smart home energy management system using IoT and big data analytics approach / A. R. Al-Ali, I. A. Zualkernan, M. Rashid, R. Gupta, M. Alikarar // IEEE Transactions on Consumer Electronics. – 2017. – Vol. 63, Issue 4. – pp. 426-434.
12. Hossain, M. S. Cyber–physical cloud-oriented multi-sensory smart home framework for elderly people: An energy efficiency perspective / M. S. Hossain, M. A. Rahman, G. Muhammad // Journal of Parallel and Distributed Computing. – 2017. – Vol. 103. – pp. 11-21.
13. Isnen, M. A-SEM: An adaptive smart energy management testbed for shiftable loads optimisation in the smart home / M. Isnen, S. Kurniawan, E. Garcia-Palacios // Measurement. – 2020. – Vol. 152, 107285.
14. Paredes-Valverde, M. A. IntelliHome: An internet of things-based system for electrical energy saving in smart home environment / M. A. Paredes-Valverde, G. Alor-Hernandez, J. L. Garcia-Alcaraz, M. D. P. Salas-Zarate, L. O. Colombo-Mendoza, J. L. Sanchez-Cervantes // Computational Intelligence. – 2020. – Vol. 36, Issue 1. – pp. 203-224.
15. Jo, H. Intelligent smart home energy efficiency model using artificial TensorFlow engine / H. Jo, Y. I. Yoon // Human-centric Computing and Information Sciences. – 2018 – Vol. 8, Issue 1. – pp. 1-18.
16. De Paola, A. A fog-based hybrid intelligent system for energy saving in smart buildings / A. De Paola, P. Ferraro, G. L. Re, M. Morana, M. Ortolani // Journal of Ambient Intelligence and Humanized Computing. – 2020. – Vol. 11, Issue 7. – pp. 2793-2807.
17. Killian, M. Comprehensive smart home energy management system using mixed-integer quadratic-programming / M. Killian, M. Zauner, M. Kozek // Applied energy. – 2018. – Vol. 222 . – pp. 662-672.
18. Machorro-Cano, I. HEMS-IoT: A Big Data and Machine Learning-Based Smart Home System for Energy Saving / I. Machorro-Cano, G. Alor-Hernandez, M. A. Paredes-Valverde, L. Rodriguez-Mazahua, J. L. Sanchez-Cervantes, J. O. Olmedo-Aguirre // Energies. – 2020. – Vol. 13, Issue 5. – pp. 1097.

Надійшла / Paper received: 10.08.2020
Надрукована / Paper Printed : 02.09.2020

UNICHECK

Ім'я користувача:
Кафедра КІ

ID перевірки:
1008142123

Дата перевірки:
02.06.2021 15:40:44 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
02.06.2021 15:42:36 EEST

ID користувача:
100005591

Назва документа: Розподілена динамічна система кластеризації для видобутку просторових даних

Кількість сторінок: 81 Кількість слів: 14675 Кількість символів: 107344 Розмір файлу: 2.25 MB ID файлу: 1008222700

0.81%
Схожість

Найбільша схожість: 0.63% з джерелом з Бібліотеки (ID файлу: 1008095688)

0.26% Джерела з Інтернету	16	Сторінка 83
0.81% Джерела з Бібліотеки	77	Сторінка 83

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	25

Wed Jun 02 14:57:24 EEST 2021, Медзатий Дмитро Миколайович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальное совпадение с одним документом 0.0%

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 7%**

	Документ		Суммарное совпадение по Базе Данных	
ID: 92054 Название: Розподілена динамічна система кластеризації для видобутку просторових даних Добавлено в БД: 2021-06-02 Авторы: Товстуха Е.В. Руководители: Бобровнікова К.Ю. Консультанты: Оппоненты:	Символы	Лексемы	Символы	Лексемы
	94002	828	533 (1%)	8 (1%)

Источник плагиата

ID	Описание	Наличие плагиата в документе	
		Символы	Лексемы

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА ДИПЛОМНУ РОБОТУ

Дипломник: Товстуха Едуард Володимирович

Тема: Розподілена динамічна система кластеризації для видобутку просторових даних

Спеціальність**:** 123 «Комп'ютерна інженерія»

Обсяг дипломної роботи:

Кількість листів креслень ___; кількість сторінок записки 79

1. Короткий зміст роботи та прийнятих рішень В дипломній роботі удосконалено модель процесу розподіленої динамічної кластеризації для видобутку просторових даних, що використовує алгоритми стиснення та агрегації даних. На основі розробленої моделі побудовано метод розподіленої динамічної кластеризації для видобутку просторових даних та створено розподілену динамічну систему кластеризації для видобутку просторових даних, яка надає можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими системами

2. Висновок про відповідність роботи дипломному завданню Дипломна робота відповідає виданому завданню як в теоретичній, так і в практичній частині.

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи: Розділ 1 – проведено огляд відомих методів розподіленої динамічної кластеризації для видобутку просторових даних та визначено їх недоліки. Розділ 2 – удосконалено модель процесу розподіленої динамічної кластеризації для видобутку просторових даних. Розділ 3 – удосконалено метод розподіленої динамічної кластеризації для видобутку просторових даних та проведено експериментальні дослідження методу. Розділ 4 – розроблена розподілена динамічна система кластеризації для видобутку просторових даних та представлено результати роботи розробленої системи. В загальному усі розділи відповідають завданню.

4. Позитивні сторони роботи: Застосування удосконаленого методу розподіленої динамічної кластеризації для видобутку просторових даних надає можливість підвищити ефективність кластеризації для видобутку просторових даних, в порівнянні з відомими методами.

5. Негативні сторони роботи: Надмірна кількість теоретичного матеріалу. В рамках дипломної роботи варто було приділити більшу увагу задачі розпізнавання просторових образів.

6. Оцінка графічного оформлення та пояснювальної записки роботи Пояснювальна записка відповідає нормам оформлення та виконана на задовільному рівні.

7. Відгук про роботу в цілому: В загальному дипломна робота заслуговує задовільної оцінки. Дипломна робота присвячена вирішенню актуальної задачі розроблення розподіленої динамічної системи кластеризації для видобутку просторових даних. Усі розділи роботи йдуть у вірній послідовності, що дозволяє розуміти викладений матеріал.

8. Інші зауваження: _____

_____

_____

_____

_____

9. Оцінка дипломної роботи: Розглянувши позитивні та негативні сторони представленої дипломної роботи, можна зробити висновок, що вона заслуговує на оцінку «задовільно». _____

Рецензент (прізвище, ім’я, по батькові, посада, місце роботи) Мартинюк Валерій Володимирович, д.т.н., професор, завідувач кафедрою Автоматизації, комп'ютерно-інтегрованих технологій і телекомунікацій_____

“25” травня 2021 р.  (підпис)

Завідувачу кафедри КІСП
д-р.техн.наук, проф. Говорущенко Т. О.

Товстухи Едуарда Володимировича
ПІБ здобувача вищої освіти

ФПКТС, 2 курсу, групи КІ2м-19-1

ЗАЯВА

З правилами чинного Положення «Про дотримання академічної доброчесності в Хмельницькому національному університеті» від 26.09.2020 (зі змінами від 26.11.2020), згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений (а). Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіатоповіщений (а) та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

25.05 2021 року

дата

підпис

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА СИСТЕМНОГО ПРОГРАМУВАННЯ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Розподілена динамічна система кластеризації для видобутку просторових даних

Автор: Товстуха Едуард Володимирович

Спеціальність: 123 – Комп'ютерна інженерія та програмування

Освітня програма: освітньо-наукова

Науковий керівник: Бобровнікова К.Ю., к.т.н.

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) запозичення розміщені в розділах аналізу існуючих аналогів та прототипів, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;
2) усі запозичення фрагментарні, або мають належним чином оформленні посилання;
3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 0.81% і адресується до 93 першоджерел, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ К. Ю. Бобровнікова

Гарант ОП _____ О. С. Савенко

Завідувач кафедри КІСП _____ Т. О. Говорущенко