# **Визначення відстані Хеммінга в розподілених інфраструктурах для пошуку подібності.**

**Аннотація**. Сьогодні обсяги даних доступних в Інтернеті перевищують зетабайти (ZB). Ця умова визначає явище яке відоме в літературі як Big Data. Хоча традиційні бази даних дуже ефективні для пошуку та отримання потрібного контенту, вони не є ефективними в умовах роботи з великими обсягами даних, оскільки велика частина цих даних є неструктурованою та розподіленою в Інтернет. Таким чином для забезпечення пошуку подібності необхідні нові бази даних. Для того аби справлятися з даною ситуацією в даному розділі пропонується досліджувати схожість за Хеммінгом між ідентифікаторами контенту, що згенеровані з використанням функції випадкової гіперплощини хешування. Такі ідентифікатори забезпечують основу для побудови розподілених інфраструктур, що спрощує пошук подібності. В цьому розділі м представимо два різні підходи: P2P (Peer-to-Peer) (Hamming DHT (Distributed Hash Table)) та Data Center (HCube). Представлені оцінки що вказують на те, що обидва підходи здатні покращити результати пошуку подібності.

## **1. Вступ**

Згідно сучасного розуміння явища Big Data, користувачі стали джерелами даних, компанії зберігають незліченні обсяги інформації про клієнтів, мільйони сенсорів отримують інформацію з середовища, створюючи та обмінюючись даними в Інтернеті речей. Згідно дослідження International Data Corporation (IDC), опублікованого в травні 2010 [1], кількість даних доступних в Інтернеті перевищує 2 ZB в 2010, подвоююись кожні 2 роки і може перевищити 8 ZB в 2015. Дослідження також показало що близько 90% цих даних складаються з неструктурованих, гетерогенних та різних за природою даних як тексти, зображення та відео.

Нові технології такі як Hadoop [2] та MapReduce [3] є прикладами рішень створених щоб вирішувати проблеми які накладає Big Data – обсяг, різноманітність та швидкість. Використовуючи пераленьні обчислення в поєднанні з грід технологіями, чи, як нещодавно, використовуючи інфраструктуру хмарних обчислень, IT організації пропонують засоби для роботи з широкомасштабними, розподіленими та інформаційномісткими (такими що використовують великі об’єми даних) задачами. Зазвичай такі технології пропонують розподілену файлову систему та автоматизовані інструменти для налаштування в процесі роботи (“на льоту”) кількості серверів залучених в обробку задач. В цьому випадку великі об’єми даних пропускаються через мережу і підключаючись до серверів передають пари <ключ, значення> з маперів до ред'юсорів щоб отримати необхідні результати. В цьому випадку бажано мінімізувати потреби в передачі даних по мережі щоб пришвидшити роботу всієї обчислювальної задачі.

В той час як дані рішення є, беззаперечно, ефективними для роботи традиційних застосувань, як, наприклад, пакетної обробки великих об’ємів даних, вони не забезпечують достатню підтримку пошуку подібності [4], ціллю якої є вилучення наборів подібних даних за даним рівнем подібності. Наприклад, схожість даних може бути використана в системі рекомендацій заснованій на інформації соціальних профілів користувачів. В даному прикладі профіль користувача може бути визначених як набір характеристик що унікально впливають на те як користувачі приймають рішення. Користувачі зі схожими характеристиками, скоріш за все, матимуть схоже інтереси та вподобання.

Таким чином, щоб зробити систему пошуку подібності засновану на даних профілів користувачів, характеристики користувачів в соціальній мережі можуть бути поміщені в вектор і, використовуючи Vector Space Model (VSM), подібність між користувачами може бути визначена використовуючи метрики відстаней між векторами як, наприклад, Евклідова відстань, косинус між векторами та Хеммінгова відстань. Але, за виключенням Хеммінгової відстані, інші метрики піддаються прокляттю розмірності (значний ріст необхідних обчислень при рості розмірності задачі) [4]. Великі обчислювальні витрати внаслідок розмірності задачі – це виклик що постає перед системами поуку подібності в випадку Big Data.

В цьому розділі описано як підтримувати пошук подібності використовуючи Хеммінгову відстань як метрику подібності. Щоб це досягти, дані індексуються в базі даних використовуючи LSH (Locality Sensitive Hashing) функцію що називається функцією RHH (Random Hyperplane Hashing) [5]. RHH – це сімейство LSH функцій, що використовують косинус між векторами та Хеммінгову відстань як метрику відстані між згенерованими двійковими рядками, тобто чим більший косинус між парою векторів контенту, тим менша Хеммінгова відстань між двійковими рядками. Ці двійкові рядки представляють ідентифікатори даних чия подібність можу бути виміряна Хеммінговою відстанню. Кожен запит в цій базі даних вираховується використовуючи Хеммінгову відстань між ідентифікатором запиту та кожним ідентифікатором даних.

При пошуку подібності запит що обробляється складається з того ж набору характеристик контенту індексованого в базі даних. Кожен користувач системи пошуку подібності вводить бажані характеристики та рівень подібності згідно очікуваного об’єму відповіді на запит. Чим більший рівень подібності тим менша кількість даних отримується внаслідок більшої специфічності (націленості) запиту. Запит індексується в базі даних з використанням RHH функції та подібності за Хеммінгом між запитом та всіма обчисленими ідентифікаторами даних в базі даних. Всі профілі чия подібність за Хеммінгом задовольняє бажаному рівню подібності повертаються у відповідь на запит.

Щоб оцінити систему пошуку подібності були проведені наступні тести: взаємозв’язок подібності з використанням косинусу між векторами контенту та подібності за Хеммінгом їх ідентифікаторів представлені використовуючи чотири рівні подібності (0.7, 0.8, 0.9 та 0.95); частотних розподіл відстані за Хеммінгом між ідентифікаторами контенту відповідно до їх рівня подібності; також надані результати деяких запитів та відповідей на них. В наших експериментах вектори контенту представляють профілі користувачів в Adult Data Set репозиторію UCI [6].

В нашій попередній роботі [7] було представлено рішення з оверлеями для системи пошуку подібності що було створено на вершині DHT (Distributed Hash Table) структури. Головним чином була показана можливість зберігання подібних даних на серверах близько до логічного простору оверлейної мережі використовуючи примітив put(k,v) і що також можливо ефективно відновлювати набір подібних даних використовуючи один get(k,sim) примітив. В іншій попередній роботі [8] був показаний HCube – рішення для дата центру (центру обробки даних) для підтримки пошуку подібності в випадку Big Data, спрямоване на зменшення відстані щоб відновити схожий контент при пошуку подібності. В HCube подібні дані зберігаються на тому ж сервері чи на серверах розташованих близько в дата центрі.

Даний розділ організований наступним чином: частина 2 показує деякі основи технологій що використовуються в Hamming DHT та HCube. Частина 3 містить огляд літератури на пов’язані роботи з пошуку подібності в P2P (Peer-to-Peer) мережі та дата центрі. Частина 4 коротко показує рішення Hamming DHT та HCube. Частина 5 оцінює запропоновану систему пошуку подібності в розподіленому випадку. В частині 6 надані деякі ремарки та огляд майбутньої роботи.

## **2. Короткі теоретичні відомості**

Ця частина показує концепцію VSM (Vector Space Model) – модель представлення даних в вигляді векторів в багатовимірному просторі, функцію RHH – LSH функцію для генерування ідентифікаторів даних що зберігатимуть подібність між векторами контенту, та функцію подібності за Хеммінгом – функцію подібності що використовується для порівняння відстані за Хеммінгом між двійковими ідентифікаторами.

### 2.1 Модель векторів у просторі (VSM (Vector Space Model))

VSM – алгебраїчна модель представлення об’єктів як векторів. В загальному випадку, кожен вимір (кожна координата) цих векторів пов’язаний з характеристикою самого контенту як, наприклад, ключові слова в тексті, гістограма кольорів на зображенні чи атрибути профілю в соціальній мережі.

Набір векторів отриманих з Adult Data Set репозиторію UCI [6], використовується щоб описати процедуру перетворення таких атрибутів в вектор що може бути виміряний та порівняний з іншими векторами використовуючи алгебраїчні операції. В своїй суті цей набір даних містить інформацію про дорослих громадян що живуть в США включаючи наступні атрибути: вік, робочий клас, рівень освіти, кількість років проведених в школі, сімейний стан, вид діяльності, родинні зв’язки, расу, стать, фінансові здобутки та втрати за останній рік, кількість робочих годин на тиждень, країна походження, дохід за рік. Зразки цих профілів:

– ADULT1 - 43; Self-emp-not-inc; 5th-6th; 3;Married-civ-spouse;Craft-repair;Husband; White; Male; 0; 4700; 20; United-States; ≤50K

– ADULT2 - 56; Private; 10th; 6; Married-civ-spouse; Craft-repair; Husband;White; Male; 0; 0; 0.45; France; ≤50K

– ADULT3 - 50; Self-emp-inc; Prof-school; 15; Married-civ-spouse;Prof-specialty; Husband;White; Male; 0; 0; 36; United-States; ≥50K

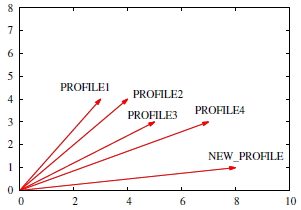
– ADULT4 - 30; Private; Prof-school; 15; Married-civ-spouse; Prof-specialty; Husband; White; Male; 0; 0; 30; United-States; ≥50K

Для експериментів повинні бути зроблені деякі перетворення над цими векторами. Числові атрибути представлені в векторах, такі як “вік”, повинні бути нормалізовані до границь [0..1]. Така нормалізація була зроблена шляхом ділення значення на найбільше значення в наборі. Координати що представляють дискретні атрибути (наприклад, стать, що може бути “чоловіча” чи “жіноча”) були розділені в 2 різні координати, кожна з яких в окремому вимірі відповідно до можливих значень. Для атрибуту “стать” було створено два виміри “чоловіча” та “жіноча”. Якщо людини – чоловік, відповідний вектор має значення “1” для виміру “чоловіча” та “0” для виміру “жіноча” і навпаки, якщо людина – жінка.

Як показано в [9] ці процедури були необхідні тому що поняття подібності чи відстані для дискретних даних не таке очевидні як для числових і тут це було основною задачею.Це виникає внаслідок того що різні значення дискретного атрибуту не упорядковані по свій суті і, таким чином, поняття впорядкованості для них не визначене. Також поняття подібності може відрізнятися в залежності від конкретного домену. Внаслідок цього кожен атрибут вектору має бути розширений на кількість вимірів відповідно до всіх значень які він [атрибут] містить. Використовуючи цю процедуру, вектор повинен бути розширений з 14 до 103 вимірів.

Можна виміряти подібність між векторами що представляють дані шляхом вирахування косинусу кута між ними (simcos). Як показано в [10], подібність за косинусом дає високоякісні результати в декількох доменах. Щоб показати це, на мал. 1 відображено застосування в якому профілі користувачів представлені двовимірними векторами, кожен вимір описує інтерес користувача до спорту та літератури, використовуючи масштаб в якому “0” означає відсутність інтересу, а “10” – повний інтерес до області. Розглянемо чотири профілі користувачів представлені кортежами PROFILE1(3,4) – ранг 3 для спорту та 4 для літератури, PROFILE2(4,4) – ранг 4 для спорту і літератури, PROFILE3(5,3) – ранг 5 для спорту та 3 для літератури та PROFILE4(7, 3) – ранг 7 для спорту та 3 для літератури.

Щоб дати розуміння використання подібності за косинусом (simcos), розглянемо розробку системи рекомендації друзів засновану на профілях користувачів відповідальну за вказування на друзів з подібними інтересами до профілю нового користувача представленого в вигляді кортежу NEW PROFILE(8,1) – ранг 8 для спорту та 1 для літератури. В цьому випадку система рекомендації запропонує користувачу NEW PROFILE наступний порядок для встановлення нових знайомств: 1) PROFILE4 чий simcos ≈ 0.96, 2) PROFILE2 чий simcos ≈ 0.9, 3) PROFILE3 чий simcos ≈ 0.8, and 4) PROFILE1 чий simcos ≈ 0.7.



Спорт

Мал. 1 Графічне представлення векторів профілей

### 2.2 RHH (Random Hyperplane Hashing) та подібність за Хеммінгом

Функції LSH зменшують розмірність векторів що представляють дані забезпечуючи те, що чим більш подібними два об’єкти є, тим більш подібними будуть хеш-значення їх векторів [4]. Кожне сімейство LSH функцій відноситься до певної функції подібності. RHH – приклад сімейства LSH функцій пов’язаних з подібністю за косинусом. В даному контексті, Чарікар [5] представляє техніку хешування резюмовану в даній частині.

Дано набір  з m векторів  кожна з координат яких випадковим чином отримана з стандартного нормального розподілу, та вектор . Хеш-функція  визначається наступним чином:



Для кожного  генерується один біт, а результат m об’єднується щоб створити m-бітний хеш-ключ для даного вектора . Для двох векторів даних  ймовірність згенерувати подібні ключі – значення косинусу кута між  та . Відповідно, чим більша подібність за косинусом, тим більш ймовірно що згенеровані ключі міститимуть спільні біти, приводячи схожість двох ідентифікаторів в відстані за Хеммінгом (тобто в кількості різних біт в двох двійкових рядках). І навпаки – подібність за Хеммінгом можна вирахувати як , де simh – подібність за Хеммінгом, Dh – Хеммінгова відстань, m – число біт в рядку що представляє ідентифікатор профілю користувача.

Наприклад, нехай застосування використовує 8-бітні ідентифікатори, повинна бути згенерована послідовність m=8 випадкових векторів  і повернені m  об'єднаних бітів щоб згенерувати 8-бітний ідентифікатор. В таблиці 1 показано приклад з порівнянням подібності за косинусом (simcos), відстані за Хеммінгом (Dh) та подібністю за Хеммінгом (simh) між 4 типовими 8-бітними ідентифікаторами профілів користувачів. В цьому прикладі “PROFILE B”, що найбільш подібний до “PROFILE A” на основі подібності за косинусом також має найменшу відстань за Хеммінгом (Dh) відповідну подібності за Хеммінгом (simh) в 0.875.

Таблиця 1. 8-бітні ідентифікатори профілів, відстань за Хеммінгом (Dh), подібність за Хеммінгом (simh) та подібність за косинусом (simcos) профілів користувачів

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Профілі користувачів | 8-бітні ідентифікатори | Dh | simh | simcos |
| PROFILE B,  PROFILE A | 01001010,  01101010 | 1 | 0.875 | 0.99 |
| PROFILE C,  PROFILE A | 01001000,  01101010 | 2 | 0.75 | 0.95 |
| PROFILE D,  PROFILE A | 01011000,  01101010 | 3 | 0.625 | 0.85 |

## **3. Огляд літератури**