Осипчук Л. Рев’ю**[​](https://textbook.edu.goit.global/lms-python-homework/uk/docs/programming-core/hw-01/" \l "%D0%BD%D0%B0%D1%83%D0%BA%D0%BE%D0%B2%D0%B0-%D0%BF%D1%83%D0%B1%D0%BBi%D0%BA%D0%B0%D1%86i%D1%8F-1-%D1%80%D0%B5%D0%B2%D1%8E" \o "Direct link to heading) Методи та структури даних для реалізації бази даних рекомендаційної системи соціальної мережі**

У науковій публікації «**Методи та структури даних для реалізації бази даних рекомендаційної системи соціальної мережі**»від авторів В.В. Міхава, Є.В. Мелешка і С.В. Шимка з Центральноукраїнського національного технічного університету у місті Кропивницькому досліджено різноманітні методи та структури даних для створення бази даних рекомендаційної системи. Автори розглянули такі структури, як зв'язний список, розгорнутий зв'язний список, хеш-таблиця, B-дерево, B+-дерево та бінарна діаграма рішень. Основною метою цієї публікації було дослідження та реалізація програмних методів і структур даних для реалізації бази даних рекомендаційної системи з метою порівняння їх ефективності за затратами часу та пам'яті, що було здійснено шляхом проведення серії експериментів. Рекомендації формувалися за допомогою колаборативної фільтрації. Для моделювання рекомендаційної системи була розроблена програмна імітаційна модель, на якій здійснювались експерименти. В цій моделі були визначені три основні сутності: агент, сесія і предмет.

За результатами експериментів автори сформували таблиці і побудували графіки, із порівняння яких випливає, що найкращі результати за часом формування рекомендацій продемонстрували звичайний розгорнутий список та інвертований розгорнутий список. Крім того, непогані результати були отримані за допомогою структури даних B+-дерево. Аналогічно до результатів за затратами часу, найкращі показники за використаною пам'яттю для формування рекомендацій продемонстрували ті ж самі показники. Найгірші результати показала структура даних бінарна діаграма рішень як за затратами часу, так і за затратами пам’яті.

Цікавим є те, що структура розгорнутого списку відзначається своєю простотою, що в майбутньому дозволить використовувати багатопотокову роботу без блокувань. Розмір блоку впливає на швидкість роботи та обсяг використаної пам'яті. Зменшення розміру блоку дозволяє зменшити втрати пам'яті, але збільшує час доступу до елементів.

Для прискорення пошуку окремих елементів у розгорнутому списку після заповнення блоку можна відсортувати його елементи. Це відкриває можливість використовувати бінарний пошук замість лінійного та перевіряти лише ті блоки, де шуканий елемент належить до інтервалу, утвореного найменшим та найбільшим елементами блоку. Перевага розгорнутого списку перед іншими розглянутими структурами даних у використанні пам'яті значною мірою полягає у тому, що зберігається лише факт вподобання, без параметрів. При використанні 4 байтів на елемент для зберігання параметру витрати пам'яті наближаються до максимальних витрат бінарних діаграм рішень.

Начало формы

В результаті проведених експериментів автори дійшли висновків, що розгорнутий список продемонстрував найкращі показники швидкодії та ефективного використання пам'яті. Профілювання показало, що 75% часу роботи тесту з розгорнутим списком було витрачено на генерування випадкових даних для програмного імітаційного моделювання агентів та предметів рекомендаційної системи, що свідчить про високу ефективність цієї структури даних. Профілювання варіанту з інвертованим списком показало, що доступ до випадкових блоків займає більше часу через неможливість їх кешування. Тому, за умов реального навантаження, час вставки нових даних буде більшим, проте відносна ефективність застосування інвертованого списку може зрости. Для максимально ефективного використання пам'яті розмір блоку зв'язного списку повинен бути налаштований таким чином, щоб блоки були максимально заповнені. Хоча блоки малого розміру дозволяють зменшити втрати пам'яті, вони збільшують час обходу усіх елементів списку та накладні витрати пам'яті.