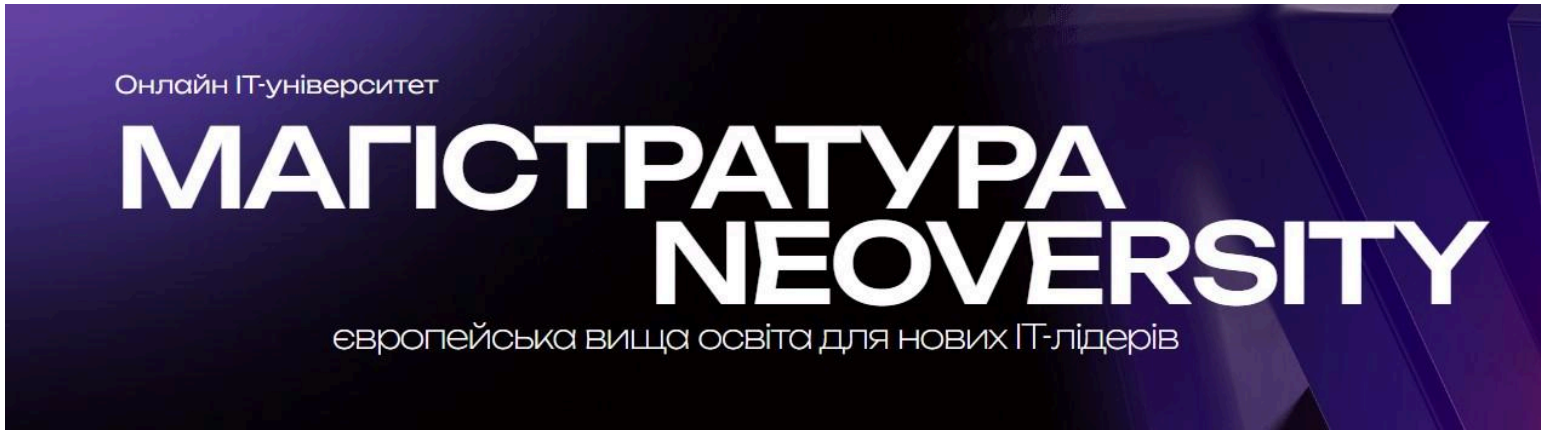


Faculty: Artificial Intelligence &
Machine Learning
Student: Oleksandr Skriabikov
Contact No.: +380 (63) 101-2000
E-mail: oskriabikov@gmail.com
LinkedIn: linkedin.com/in/askryabikov

Короткий огляд наукової публікації №3

Використання алгоритмів у бібліотеках мов програмування

Автори: Коваленко О.О., Корягіна Д.О. - Вінницький національний технічний університет

Вступ

Більшість бібліотек у програмуванні включають такі елементи: алгоритми, структури даних та інструкції для взаємодії з платформою. Усі види алгоритмів уже давно застосовуються або модифікуються під нові завдання, але кожен із них має власну швидкість роботи та підходить для певної структури даних. Саме про це йдеться в даній публікації.

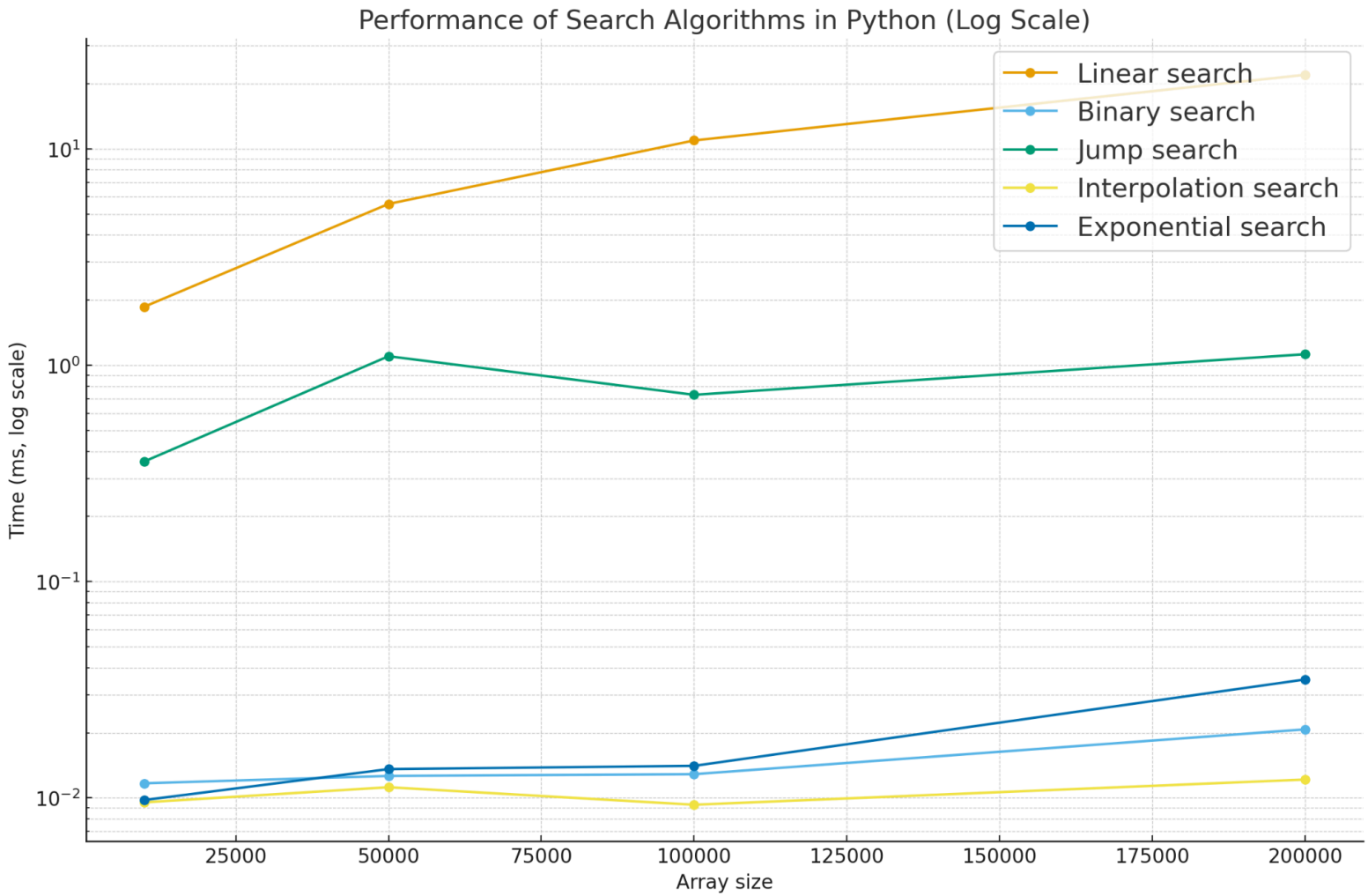
Методологія

Аналіз був побудований на вивченні окремого класу алгоритмів — пошуку. Цей клас призначений для знаходження інформації у структурах даних. До нього належать такі види пошуку: лінійний, двійковий, стрибкоподібний, інтерполяційний та експоненціальний. Також було розглянуто жадібний тип алгоритму, який можна розбивати на підзадачі.

Результати

Дослідивши код та швидкість виконання задач, автори визначили найбільш доречні сфери застосування для кожного алгоритму пошуку:

1. Лінійний (linear) — використовується в малих, не відсортованих структурах даних.
2. Двійковий (binary) — працює з відсортованими колекціями, де на кожній ітерації алгоритм ділить вхідні дані навпіл та шукає всередині потрібного сегмента.
3. Стрибкоподібний (jump search) — також застосовується у відсортованих колекціях; «стрибки» виконуються лише вперед, а при перевищенні шуканого значення запускається лінійний пошук у зворотному напрямку.
4. Інтерполяційний (interpolation) — ефективний у великих відсортованих масивах, де можна застосовувати формули інтерполяції для точнішого визначення позиції елемента.
5. Експоненціальний (exponential) — підходить для великих відсортованих колекцій, де пошук відбувається шляхом експоненційного збільшення індекса, доки не буде знайдено діапазон розташування елемента.



(графік 1) Порівняння продуктивності кожного алгоритму

На графіку чітко видно швидкість виконання різних алгоритмів пошуку: інтерполяційний виявився найшвидшим, тоді як лінійний — найменш ефективним.

Ключові інсайти

1. Перший висновок полягає в тому, що набуваючи знань про алгоритми, можна не лише користуватися готовими рішеннями, але й створювати нові, комбінуючи та модернізуючи попередні методи для складніших задач.
2. Розбиття алгоритмів на підзадачі стало для мене новим відкриттям, адже раніше я сприймав їх як повністю цілісні одиниці коду. У таких підходах кожна підзадача виконує свою конкретну функцію, незалежно від суміжних, прагнучи обробити свою частину максимально оптимально.

Висновок

Автори дійшли висновку, що вибір «найкращого» алгоритму пошуку в умовах розбиття задачі на підзадачі не завжди доцільний, і в багатьох випадках краще застосовувати простіші алгоритми. Вони підкреслюють важливість розуміння принципів роботи алгоритмів, включно з фундаментальними знаннями, оскільки це дозволяє ефективно розв'язувати задачі шляхом удосконалення вже існуючих рішень.