МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)
Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Отчет по лабораторной работе №7
«Ускорение. Прыжки по индексу»
по курсу
«Информационный поиск»

Группа: 80-106М
Выполнил: Демин И.А.
Преподаватель: Калинин А.Л.

Москва, 2019

Задание

В этом задании необходимо применить алгоритмы сжатия к координатным блокам. Исследовать изменения в размерах частей индекса, влияние на скорость индексации и поиска.

Реализация

Идея использования скип листов заключается в том, что бы при построении «пересечения» или другой операции, производимой для полученных списков doc_id, начинать хранить дополнительную информацию(место прыжка), которая позволит производить сравнение не последовательно, а пропуская некоторые значения.

Однако, данное ускорение является не максимальным, и можно получить еще большее ускорение. Для реализации ЛР по курсе был использован язык Python, а полученные doc_id для каждого слова или цитаты хранятся в структуре set(), а для того, что бы получить результат поискового запроса, применяются логические операции на set-ами.

Set() в Python является частным случаем dict(), который является реализацией хеш-таблицы. Таким образом set() - частный случай хеш-таблицы без значения. При использовании логических операций для 2-ух set-ов Python идет по элементам первого и проверяет его наличие во втором. Эта проверка производится оператором `in` за O(1). Таким образом операция & для 2-ух сетов длинной по N каждый будет проводится за O(N), а для набора set-ов — за O(max(len(set))).

Для подтверждения этого предположения были реализованы: сравнение «в лоб» за n^2, оптимизированное сравнение «в лоб», алгоритм SkipList, сравнение на основе set-ов. Тестовые данные — сгенерированные 4 списка с doc_id по 100000 элементов в каждом.

Результаты

```
(base) ivan@ivan-G5:~/study/search/search_MAI/l9_skip_list$ python compate_skiplist_speed.py
18728
Simple compare: 0:00:38.500024
18728
Optimized simple compare: 0:00:00.078726
18728
Skip list compare: 0:00:00.070578
18728
Python sets compare: 0:00:00.014338
```

Первая строка — сравнение за «квадрат», вторая — оптимизированное сравнение «в лоб», третья — SkipList, четвертая — set(). Из тестирования видно, что SkipList-ы работают достаточно быстро, но это зависит от сгенерированных данных. Независимо от данных использование set-ов самое быстрое.

Вывод

В ходе лабораторной работы я познакомился со SkipList-ами, более углубленно познакомился с устройством языка Python, а также провел сравнение различных методов для получения результата поискового запроса.