Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторные работы по курсу Информационный поиск

Студент: Е. А. Коломытцева
Преподаватель: А. А. Кухтичев
Группа: М8О-406Б-22
Дата:
Оценка:
Подпись:

Москва, 2025

1 Цель работы

Целью данной работы является построение простейшей поисковой системы по большому корпусу документов: сбор корпуса, предварительная обработка текста (токенизация, стемминг), статистический анализ распределения частот (закон Ципфа), построение булева инвертированного индекса и реализация булевого поиска по нему. Дополнительно оцениваются производительность основных этапов (скорость токенизации, стемминга, индексации и выполнения запросов).

2 Описание данных

В качестве источника данных использована англоязычная версия Википедии. Корпус сформирован автоматизированно через MediaWiki API: выполнялся обход категорий, извлекались тексты статей в виде плоского текста без вики-разметки, а также сохранялись метаданные (заголовок и ссылка на источник). Для обеспечения тематической однородности корпус собирался из одной предметной области (категория, связанная с математикой).

Корпус хранится в формате: один документ — один файл `.txt` в кодировке UTF-8. Метаданные документов хранятся в файле `manifest.jsonl` (по одной JSON-строке на документ) и включают идентификатор, заголовок, URL и размер документа.

2.1 Объём корпуса

По результатам индексации было обработано:

- количество документов: **30000**;
- общий объём текста: **737 359 307 байт** (**720 077.4 KB**);
- общий объём токенов (до стемминга): **89 209 599**.

2.2 Предварительная обработка текста

На этапе токенизации текст разбивался на токены по правилу: токен — это максимальная последовательность ASCII букв/цифр (`[A-Za-z0-9]+`). Знаки пунктуации и прочие символы используются как разделители. Для всех токенов выполнялось приведение к нижнему регистру.

Достоинства выбранной токенизации:

- высокая скорость (линейный проход по байтам);
- простая и повторяемая реализация;
- достаточна для англоязычного корпуса и булевого поиска.

3 Закон Ципфа

Для анализа статистических свойств корпуса рассмотрен закон Ципфа, описывающий распределение частот термов. Перед подсчётом частот выполнялась нормализация:

- токенизация (разделители — не `[A-Za-z0-9]`);

- приведение к нижнему регистру;
- стемминг Porter для английского языка.

Для каждого уникального терма была подсчитана частота его вхождения во всём корпусе. Далее термы были отсортированы по убыванию частоты, каждому терму присвоен ранг. На основе полученных данных построен график зависимости частоты терма от его ранга в логарифмической шкале, а также наложена зависимость вида:

$$f(r) = \frac{C}{r},$$

где $C = f(1)$ — частота терма ранга 1.

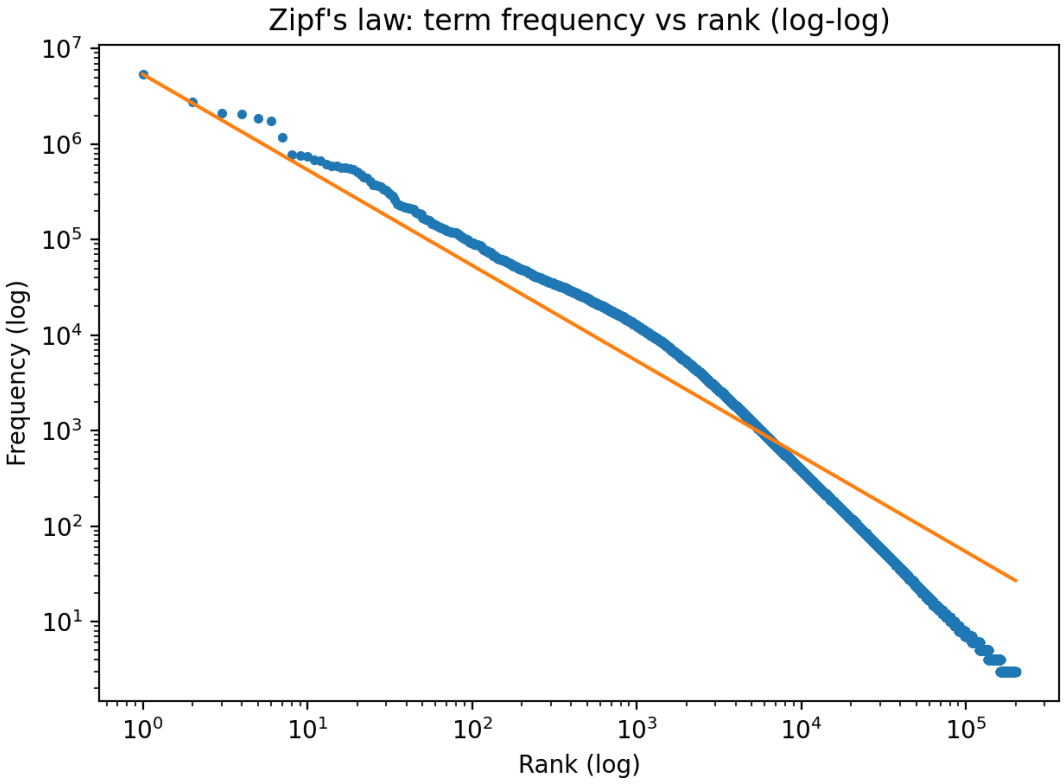

Рис. 1: Закон Ципфа для корпуса

Анализ графика показывает, что распределение близко к линейному в логарифмических координатах на значительном диапазоне рангов, что соответствует закону Ципфа. Отклонения наблюдаются:

- в начале распределения (для самых частотных термов), что связано с доминированием служебных слов;
- в хвосте распределения (для редких термов), что объясняется шумом токенизации, терминами из формул, именами собственными и ограниченной статистикой редких слов.

3.1 Наиболее частотные термы

В таблице 1 приведены наиболее частотные термы корпуса после токенизации, приведения к нижнему регистру и применения стемминга.

Таблица 1: Топ-10 наиболее частотных термов

Ранг	Терм	Частота
1	`the`	5 374 850
2	`of`	2 770 307
3	`a`	2 129 426
4	`and`	2 087 502
5	`in`	1 846 756
6	`to`	1 749 130
7	`is`	1 167 734
8	`for`	774 317
9	`on`	748 926
10	`that`	682 037

4 Примеры существующих поисковых систем

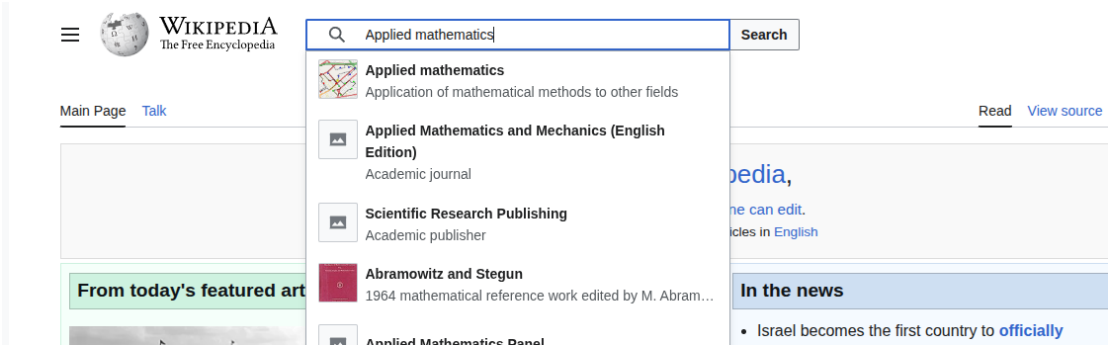

Рис. 2: Пример поиска в Википедии

5 Индексация и поиск

5.1 Булев инвертированный индекс

Реализован булев инвертированный индекс: каждому терму сопоставляется список идентификаторов документов (postings list), в которых терм встречается. Для уменьшения размера постинг-листов при индексации учитывалось только наличие терма в документе (то есть для каждого документа извлекалось множество уникальных термов, без учёта количества вхождений).

Для быстрого доступа индекс сохранён на диске в бинарном виде:

- `lexicon.bin` — словарь термов (для каждого терма: длина, df, смещение и длина постинг-листа);
- `postings.bin` — массив постинг-листов (отсортированные `uint32` doc_id);

- `docs.bin` — прямой индекс (заголовки и URL документов).

Ключевые характеристики индекса:

- количество термов: **671 038**;
- средняя длина терма: **7.585**;
- объём файла постингов: **95 299 976 байт**.

5.2 Операции булева поиска

Поддерживаются операторы:

- пробел или `&&` — логическое И;
- `||` — логическое ИЛИ;
- `!` — логическое НЕ;
- скобки для явного задания приоритета.

Поисковый запрос разбирается, преобразуется в последовательность операций над постинг-листами. Для выполнения используются стандартные алгоритмы над отсортированными списками:

- пересечение (AND) — «двухуказательный» проход;
- объединение (OR) — слияние двух отсортированных списков;
- отрицание (NOT) — разность относительно множества всех документов корпуса.

6 Пример работы поисковой системы

Пример запроса по одному терму:

```
echo 'function' | ./search_cli --index ./out --limit 5
Stdout:
1       Actuarial notation      https://en.wikipedia.org/wiki/
   Actuarial_notation
2       Actuarial present value https://en.wikipedia.org/wiki/
   Actuarial_present_value
3       Asymptotology   https://en.wikipedia.org/wiki/Asymptotology
4       Bass diffusion model    https://en.wikipedia.org/wiki/
   Bass_diffusion_model
5       Cartesian tensor        https://en.wikipedia.org/wiki/
   Cartesian_tensor
[STATS] query="function" hits=11834 shown=5 offset=0 time=0.000040
   sec
```

Пример булева запроса с отрицанием и скобками:

```
echo '(!network) (algorithm || proof)' | ./search_cli --index ./out
   --limit 5
Stdout:
9       Cryptanalysis   https://en.wikipedia.org/wiki/Cryptanalysis
13      Fractal derivative      https://en.wikipedia.org/wiki/
   Fractal_derivative
18      Least-squares spectral analysis https://en.wikipedia.org/wiki
   /Least-squares_spectral_analysis
23      Mathematics and art     https://en.wikipedia.org/wiki/
   Mathematics_and_art
30      Probabilistic numerics  https://en.wikipedia.org/wiki/
   Probabilistic_numerics
[STATS] query="(!network) (algorithm || proof)" hits=5857 shown=5
   offset=0 time=0.000388 sec
```

В выдаче отображаются идентификатор документа, заголовок и ссылка на источник (Википедия). Для постраничного вывода предусмотрены параметры `-limit` и `-offset` (следующие 50 результатов и т.д.).

7 Статистика работы системы

7.1 Токенизация

Результаты запуска:

- объём входных данных: 720 077.4 KB;
- количество токенов: 89 209 599;
- средняя длина токена: 4.756;
- время: 1.991 sec;
- скорость: 361 743.3 KB/s.

Токенизация реализована как линейный проход по буферу ввода, поэтому скорость близка к ограничению чтения данных и пропускной способности CPU. Ускорение возможно за счёт:

- чтения большими блоками (уже используется);
- распараллеливания по файлам (несколько потоков);
- SIMD-оптимизаций для классификации символов.

7.2 Стемминг

Результаты запуска:

- объём входных данных: 720 077.4 KB;
- raw_tokens: 89 209 599, avg_raw: 4.756;
- stem_tokens: 89 209 599, avg_stem: 4.149;

- changed: 28 552 194 (32.01%);
- время: 22.034 sec;
- скорость: 32 679.9 KB/s.

Стемминг значительно медленнее токенизации, так как для каждого токена выполняется набор правил и проверок суффиксов. Ускорение возможно за счёт:

- кэширования стеммов для частых токенов (LRU-словарь);
- распараллеливания по файлам;
- оптимизации ветвлений и работы со строками (меньше копирований).

7.3 Индексация

Результаты индексации:

- документы: 30 000;
- общий объём текста: 720 077.4 KB;
- токены: 89 209 599;
- среднее число уникальных термов на документ: 828.6;
- время: 217.39 sec;
- скорость: 3 312.3 KB/s.

Основные ограничения индексации:

- необходимость поддерживать в памяти структуру term $\rightarrow$ postings (временные блоки);
- стоимость сортировки и слияния блоков;
- стоимость чтения/разбора всех документов.

При увеличении объёма входных данных:

- в 10 раз — время приблизительно увеличится в 10 раз (при достаточной памяти);
- в 100 раз — потребуется внешняя сортировка/большее число блоков, возрастёт доля времени на merge;
- в 1000 раз — потребуется более оптимизированный формат (сжатие постингов), многопоточная индексация, распределённая обработка.

7.4 Скорость выполнения запросов

Примеры измерений (по выводам `search_cli`):

- запрос `function`: hits=11834, время порядка **десятков микросекунд**;
- запрос `(!network) (algorithm || proof)`: hits=5857, время порядка **0.5 мс**.

Сложные запросы замедляют работу из-за больших промежуточных множеств (например, выражения с частыми термами и несколькими OR). Оптимизация возможна за счёт:

- выполнения AND в порядке возрастания df (сначала пересекать самые редкие термы);
- сокращения промежуточных результатов;
- сжатия постинг-листов и SIMD-операций.

8 Заключение

В ходе выполнения работы реализован полный конвейер построения поисковой системы: сбор корпуса Википедии, токенизация, стемминг, статистический анализ частот (закон Ципфа), построение булева инвертированного индекса и выполнение булевых запросов по индексу.

Система корректно выполняет операции AND/OR/NOT и поддерживает скобки. Индекс хранится в бинарном виде и включает как инвертированный индекс, так и прямой индекс для вывода заголовков и ссылок. Полученные замеры показывают высокую скорость токенизации и приемлемую скорость поиска, при этом основными узкими местами являются стемминг и построение индекса.