Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский авиационный институт» (Национальный исследовательский университет)

Институт: №8 «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Отчет по лабораторным работам

по предмету «Информационный поиск»

Группа: М8О-412Б-22
Студент: **Андриянов Эрик Вячеславович**
Оценка: _____
Дата сдачи: _____

Москва, 2025 г.

1 Сбор и состав корпуса документов

1.1 Постановка задачи

Первым этапом разработки поисковой системы является формирование корпуса документов. Основными требованиями к корпусу в рамках данной работы были:

1. Достаточный объем (не менее 30 000 документов).
2. Тематическая однородность (область искусственного интеллекта, LLM и AI-агентов).
3. Наличие минимум трех независимых источников данных.
4. Поддержка двуязычности (русский и английский языки).

1.2 Реализация сбора данных

В качестве источников данных были выбраны:

- **Habr.com** (статьи по тегам AI, нейросети, Python).
- **Arxiv.org** (научные препринты в категории CS.AI).
- **TechCrunch.com** (новости технологий и стартапов в сфере AI).

Для сбора данных был разработан многопоточный поисковый робот на языке Python с использованием библиотек `requests` и `BeautifulSoup4`. Процесс включал очистку текста от HTML-разметки, навигационных элементов и рекламы.

1.3 Итоговая статистика

В результате работы робота был сформирован корпус из 29 609 документов.

- Формат хранения: текстовые файлы `doc_XXXXX.txt`.
- Общий объем словаря: 736 796 уникальных словоформ (после токенизации).

2 Архитектура поискового робота

2.1 Алгоритм работы

Разработанный робот функционирует на основе алгоритма обхода в ширину (BFS). Для обеспечения уникальности и предотвращения повторных загрузок реализована система отслеживания состояния:

- `visited_urls.txt`: файл-реестр всех обработанных ссылок.
- Discovery: автоматическое извлечение новых ссылок из текстов статей для углубления поиска.

2.2 Контроль актуальности

Робот поддерживает механизм возобновления работы: при перезапуске он считывает список уже посещенных URL и продолжает сбор с последнего места, что критично при работе с большими объемами данных.

3 Лингвистическая обработка

3.1 Токенизация

Реализована на языке C++. Программа разбивает текст на токены, удаляет пунктуацию и спецсимволы, а также корректно обрабатывает многобайтовые последовательности UTF-8 для приведения кириллицы к нижнему регистру.

3.2 Стемминг

Для приведения слов к нормальной форме реализован алгоритм Портера на языке C++. Особенность реализации: алгоритм выполнен без использования стандартной библиотеки шаблонов (STL) для обеспечения максимальной производительности и соблюдения академических требований. Поддерживаются правила как для русского, так и для английского языков.

4 Анализ по закону Ципфа

Для верификации естественности собранного текста был проведен статистический анализ. С помощью скрипта на Python было подсчитано распределение частот слов. Результатом стал график распределения в логарифмических координатах, подтвердивший линейную зависимость. Наиболее частотные слова — предлоги и технические термины (ии, ai, модел, gpt), что подтверждает репрезентативность корпуса.

5 Построение инвертированного индекса

Финальным этапом стало создание структуры данных, позволяющей выполнять мгновенный поиск. Реализация выполнена на языке C++ без использования STL. В качестве структуры данных используется собственная реализация хеш-таблицы с методом цепочек для разрешения коллизий. Формат индекса: бинарный файл `index.bin`, содержащий словарь терминов и списки документов (postings lists). Объем словаря в индексе составляет 736 796 слов.

6 Результаты тестирования поиска

Реализована система булева поиска с поддержкой операторов AND и OR. Система автоматически применяет стемминг к запросу пользователя.

Примеры поисковых запросов:

- Запрос: `AND нейросеть обучение` — найдено 31 документ.
- Запрос: `AND gpt почему` — найдено 625 документов.
- Запрос: `OR интеллект алгоритм` — найдено более 2500 документов.

Скорость обработки запроса составляет менее 0,01 секунды благодаря использованию бинарного индекса и алгоритма Merge Join для пересечения списков.

7 Заключение

В ходе выполнения цикла лабораторных работ была разработана полнофункциональная локальная поисковая система. Пройдены все этапы: от сбора сырых данных из интернета до реализации высокопроизводительного поискового движка на языке C++. Особое внимание было уделено низкоуровневой реализации алгоритмов (стемминг, хеш-таблицы) без использования сторонних библиотек, что позволило добиться высокой скорости работы и глубокого понимания механизмов индексации данных.