

Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной
математики

Кафедра вычислительной математики и программирования

Лабораторная работа №2 по курсу «Информационный поиск»

Студент: А. Р. Белушкин
Преподаватель: А. А. Кухтичев
Группа: М8О-407Б-22
Дата:
Оценка:
Подпись:

Москва, 2025

1 Цель работы

Разработать веб-краулер на языке Python с использованием асинхронного программирования (`asyncio`), который способен:

- загружать тексты книг с сайтов Project Gutenberg и Standard Ebooks;
- хранить данные в MongoDB;
- обеспечивать контроль повторной проверки документов и уникальность контента;
- эффективно использовать параллельные запросы с ограничением конкуренции;
- расширять очередь загрузки ссылками, найденными на страницах.

2 Постановка задачи

Необходимо реализовать систему, которая:

1. Загружает тексты книг по заранее определенному диапазону идентификаторов (Gutenberg) или по стартовому URL (Standard Ebooks).
2. Хранит тексты в MongoDB с уникальным хэшированием содержимого (SHA-256).
3. Поддерживает асинхронную обработку множества заданий с использованием семафоров для ограничения числа одновременных запросов.
4. Обеспечивает возможность повторной проверки старых документов через заданный интервал (`recheck_interval`).
5. Позволяет добавлять новые ссылки из HTML-контента в очередь для дальнейшей обработки.

3 Используемые технологии

- **Язык программирования:** Python 3.11+
- **Асинхронные библиотеки:** `asyncio`, `aiohttp`
- **База данных:** MongoDB с асинхронным драйвером `motor`
- **Формат конфигурации:** YAML
- **Регулярные выражения:** `re` для фильтрации ссылок
- **Хэширование контента:** `hashlib.sha256`
- **Логирование:** `logging`

4 Структура кода

4.1 Основные компоненты

- `MongoCrawler` – основной класс краулера, инкапсулирующий логику работы с базой и HTTP-запросами.
- Методы класса:
 - `__init__()` – инициализация HTTP-сессии и индексов MongoDB.
 - `seed_gutenberg_range()` – заполнение очереди Gutenberg по ID книг.
 - `seed_standardebooks()` – добавление стартового URL Standard Ebooks.
 - `crawl_gutenberg_worker()` и `crawl_standardebooks_worker()` – асинхронные рабочие процессы для загрузки страниц.
 - `fetch_and_store()` – загрузка страницы, сохранение текста и хэша.
 - `enqueue_links_from_body()` – извлечение ссылок из HTML и добавление в очередь.
 - `recheck_scheduler()` – планировщик повторной проверки старых документов.
 - `run()` – запуск всех рабочих задач и планировщика.

4.2 Механизм семафоров

Для ограничения числа одновременных HTTP-запросов используется `asyncio.Semaphore`. Это позволяет контролировать нагрузку на сайты и локальные ресурсы.

```
1 async with self.sem:
2     async with aiohttp.ClientSession() as session:
3         async with session.get(url) as resp:
4             ...
```

4.3 Хэширование и проверка изменений

Каждый загруженный документ хэшируется с помощью SHA-256. Если хэш отличается от существующего в базе, документ обновляется.

```
1 content_hash = sha256_hex(body)
2 if not doc or doc.get('content_hash') != content_hash:
3     await docs_coll.update_one(
4         {'url': url},
5         {'$set': payload},
6         upsert=True
7     )
```

4.4 Очереди и повторная проверка

- Все URL хранятся в коллекциях очередей (`queue_collection`).
- Если документ устарел (`last_checked < cutoff`), его URL помещается обратно в очередь для повторной проверки.

5 Пример работы

1. Краулер инициализируется с конфигурационным файлом `config.yaml`.
2. Запускаются рабочие задачи для Gutenberg и Standard Ebooks.
3. Тексты загружаются и сохраняются в MongoDB.
4. Новые ссылки автоматически добавляются в очередь для последующего обхода.
5. Старые документы периодически проверяются на изменения.

6 Выводы

В ходе лабораторной работы был разработан эффективный асинхронный веб-краулер для загрузки текстов книг с популярных онлайн-библиотек.

- Реализована параллельная обработка с ограничением числа одновременных запросов.
- Использован хэш контента для отслеживания изменений.
- Поддерживаются автоматическое добавление ссылок и повторная проверка документов.
- Полученные данные сохраняются в MongoDB с индексами для быстрого поиска и уникальности.

Данный краулер может быть расширен для работы с другими источниками, добавления анализа текста и интеграции с веб-платформой по обработке литературы.