МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра физики открытых систем

Разработка базы данных публикационной активности научного сообщества на основе международных информационно-аналитических систем научного цитирования

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студента__4__ курса __431___группы

Направления 09.03.02.«Информационные системы и технологии»

код и наименование направления

Факультета нелинейных процессов

наименование факультета

Постнова Антона Андреевича

фамилия, имя, отчество

Научный руководитель
д.ф.-м.н., профессор _____ О.И. Москаленко
должность, уч. степень, уч. звание дата, подпись инициалы, фамилия

Заведующий кафедрой
д.ф.-м.н., профессор _____ А.А. Короновский
должность, уч. степень, уч. звание дата, подпись инициалы, фамилия

Саратов 2020

Характеристика работы

Работа посвящена созданию базы данных публикационной активности научного сообщества СГУ на основе данных, полученных из системы Web Of Science, и написанию программы, выполняющей синтаксический анализ сайта Web of Science.

Целью настоящей бакалаврской работы является создание и наполнение базы данных информацией о статьях и авторах, когда-либо работавших в нашем университете.

В работе было использовано 8 литературных источников.

Содержание работы

Во Введении обосновывается актуальность работы, дается общее описание систем научного цитирования, наукометрических показателей, а также обозначаются основные цели разработки.

Говоря о системах научного цитирования, в первую очередь необходимо определить, о чем именно идет речь и какие именно данные нас интересуют. В рамках дипломной работы использовалась одна из наиболее распространенных систем научного цитирования – международная база данных Web of Science. Главными преимуществами данной системы являются свободный доступ к системе для студентов и сотрудников ВУЗа, получившего к ней доступ, с любого устройства и в любом месте, а также большое количество данных, которые система может предоставить. Эти преимущества выявлены в сравнении с двумя другими распространенными в научном сообществе нашей страны системами научного цитирования – Scopus (доступ к которой для студентов и сотрудников ВУЗа возможен только из сети университета) и eLibrary (которая не распространена в международном сообществе). Наукометрические показатели, получение которых из системы Web of Science является одной из целей дипломной работы, это:

- Цитируемость статьи
- Цитируемость автора

- Количество публикаций автора
- H-index* автора

С каждым годом количество статей, публикуемых ВУЗами, растет, и, соответственно, увеличивается количество публикующихся авторов. Актуальность работы связана, в первую очередь, со своевременным оцениванием результатов научной деятельности авторов, используя наукометрические показатели. Системы научного цитирования необходимы для удобства наблюдения как за наукометрическими показателями авторов, так и ВУЗов. Использование общераспространенных систем возможно, но, как говорилось выше, не всегда удобно. Наличие собственной базы данных хотя бы для одной системы научного цитирования значительно упрощает мониторинг информации из этой системы.

Разработка базы данных публикационной активности подразумевает три основных цели:

- Разработка базы данных для хранения информации
- Создание программы для наполнения базы данными
- Реализация способов взаимодействия с программой и с базой данных

В **разделе 1** работы описывается структура базы данных, алгоритм наполнения базы информацией и методы взаимодействующие с ней.

Для хранения и работы с большими объемами информации создание базы данных просто необходимо, так как скорость взаимодействия других способов недостаточна для активного использования хранимых данных. В ходе дипломной работы была спроектирована и реализована база данных, структура которой проиллюстрирована на рисунке 1.

*Индекс Хирша(h-index, критерий Хирша) – наукометрический показатель, предложенный в 2005 г. американским физиком Хорхе Хиршем. Показывает, что n-статей процитировали не менее n-раз.

Рисунок 1 – Диаграмма базы данных, построенная в Microsoft SQL Management Studio

База данных содержит в себе три таблицы:

- Таблица авторов
- Таблица статей
- Таблица связи

На рисунке 2 представлена блок-схема алгоритма наполнения базы данных информацией, разработанного в ходе дипломной работы.

Класс LINQ[1]-это класс, который содержит в себе все методы взаимодействия программы с базой, а именно внесение полученной в ходе синтаксического анализа информации в различные таблицы и получение из этих таблиц информации для определенного автора.

Если разделить все методы класса LINQ по области их работы, можно выделить две группы:

- Вносящие информацию в базу данных
- Запрашивающие информацию из базы данные

[1] *М. А. Райтман* Программирование на C# 5.0 // Эксмо 2014 С. 374

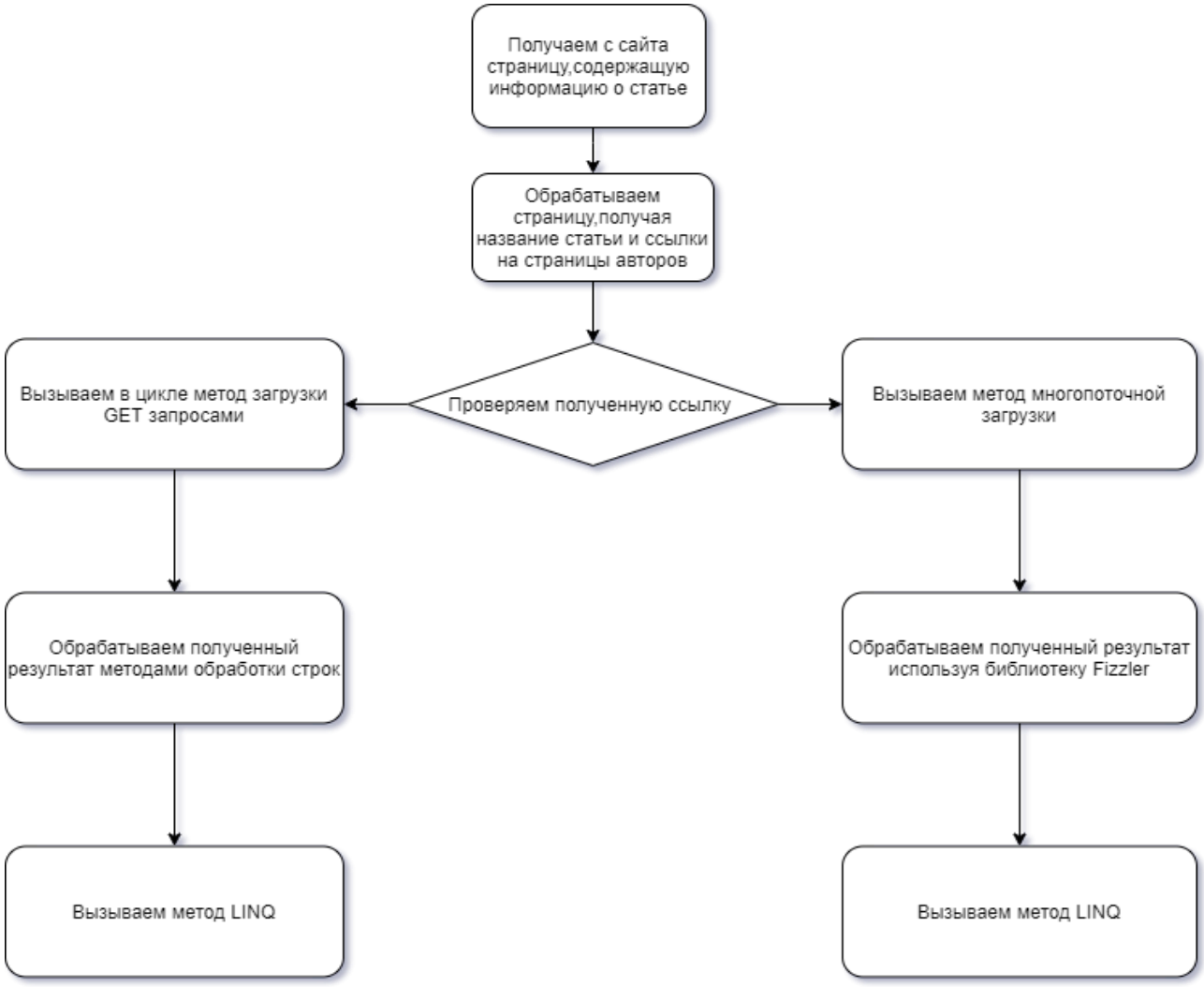

Рисунок 2 – Блок-схема разработанного подхода к наполнению базы данных

Ключевым отличиями этих двух групп методов, является то, что в первой группе методов нет возвращаемых значений, а во второй группе ни один из методов не вносит изменения в базу данных.

Кроме этих двух групп также был создан метод для очищения базы данных от всех записей, который не относится по выбранному выше критерию ни к одной из этих групп, но также используется в работе программы и выводится в интерфейс приложения ссылкой на отдельное представление.

В **разделе 2** описываются реализации многопоточной загрузки, метода прямых Get-запросов к сайту и главный цикл программы, выполняющий полный синтаксический анализ для выборки по нашему ВУЗу в системе Web Of Science.

При поиске информации о многопоточной загрузке в интернете, была найдена и доработана библиотека Spider[2]. На рисунке 3 представлено использование этой библиотеки.

Рисунок 3 – Алгоритм работы многопоточной загрузки

[2] *Сообщество профессионалов автоматизации [Электронный ресурс] : Многопоточный парсинг на C# и работа с базой данных URL: https://zennolab.com/discussion/threads/mnogopotochnyj-parsing-na-c-i-rabota-s-bazoj-dannyx.37704/*

Класс MultiThreadParser является доработанной версией найденной библиотеки Spider. Метод Start из нее отвечает за загрузку всех статей с сайта Web Of Science и вызывается при вызове метода Parsing контроллером. Использование вышеупомянутой библиотеки позволяет добавлять задачи на загрузку страниц в очередь, не дожидаясь окончания загрузки предыдущей страницы.

Совсем недавно сайт Web of Science сильно изменил структуру страниц авторов. Из-за этого возникли проблемы с обработкой страницы автора, решением которой является отправление прямых Get-запросов к сайту. При отправлении запроса на получение данных нужно только 3 значения:

- Общая ссылка для запроса
- Id автора
- Значение параметра «X-1P-WOS-SID»

Общая ссылка была найдена при рассмотрении запроса, отправляемого браузером при загрузке страницы автора, Id автора берется из ссылки на его страницу, при обработке страницы статьи, а значение параметра, которое нам необходимо, выделено жирным шрифтом в ссылке на главную страницу, выглядящую следующим образом:
https://apps.webofknowledge.com/summary.do?product=WOS&doc=1&qid=1&**SID=E3HNhMNJvfUwFynpIoz**&search_mode=AdvancedSearch&update_back2search_link_param=yes

После определения, как именно нужно отправить запрос для получения данных, был создан метод GetResponse, который вызывается в цикле для обработки всех авторов в статье, получает на входе ссылку на автора, затем извлекает из этой ссылки Id автора, и отправляет GET-запрос с параметром «X-1P-WOS-SID» на адрес «https://app.webofknowledge.com/api/rrc/author/ + id». В ответ на этот запрос программа получает текстовые данные, в которых содержится вся интересующая нас информация.

Под главным циклом подразумевается та часть программы, которая выполняет полный синтаксический анализ. Он содержит в себе два цикла: внешний и внутренний. С внешним циклом все достаточно просто: он содержит в себе проверку на переход на последнюю страницу, внутренний цикл и замену в ссылке на статьи параметра page при переходе на следующую страницу.

Внутренний цикл содержит в себе вызов унаследованного от класса MultiThreadParser, описываемого ранее при описании многопоточной загрузки, метод AddTask. Метод принимает два значения: ссылку для загрузки и название метода, в который это полученный документ передается (в данном случае в метод ArticlePageParser). Получая загруженный документ, метод ArticlePageParser вначале обрабатывает его, извлекая название статьи и ее цитирование, затем вызывается метод AddArticle, класса LINQ, и эти значения вносятся в базу данных, а затем, в цикле, происходит получение ссылок на страницы авторов и вызов метода GetResponse, который получает информацию об авторе, передает ее методу для обработки полученного текста, который затем вносит изменения в базу данных путем вызова одного из методов класса LINQ.

В **разделе 3** описывается шаблон приложения MVC(Model-View-Controller), реализация разделенных контроллеров, принцип их работы и некоторые моменты в реализации представлений.

MVC(Model-View-Controller) – это схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер. Принцип взаимодействия в них основывается на том, что пользователь взаимодействует с представлением, а контроллер взаимодействует с моделями и представлениями. Таким образом, контроллер отвечает за то, что сделает программа при определенном действии пользователя, и какая именно информация будет возвращена для пользователя обратно в представление (если она будет возвращена).

Схема работы приложения соответствует стандартам MVC приложений и состоит из 3 контроллеров: HomeController, обрабатывающий главное

представление приложения, ParsingConroller для обработки страницы загрузки статей, и ResultConroller, который обрабатывает страницу запросов к базе данных.

ParsingController содержит в себе 3 метода:

- ViewForParsing типа ViewResult, отвечающий за загрузку представления ViewForParsing при первой загрузке этой страницы или ее обновлении
- ViewForParsing типа ActionResult, отвечающий за вызов метода Parsing после введения ссылки на страницу с выборкой по ВУЗу
- ViewForClearDatabase типа ViewResult, который очищает базу данных от всех записей при переходе нажатии на соответствующую ссылку

ResultController также содержит в себе 3 метода:

- ViewFor Result типа ViewResult, отвечающий за загрузку страницы, созданной для запросов к базе данных
- ViewFor Result типа ActionResult, отвечающий за запрос к базе данных(вызов метода SelectAllAuthor)
- ViewForAllAuthorInfo типа ViewResult, который выводит всю информации о выбранном авторе при загрузке странице

Представления в программе также разделены по сути выполняемых ими задач на три группы:

- Главная страница
- Изменение в базе данных
- Получение результатов запросов к базе данных

Само представление является cshtml-документом, который, в свою очередь, представляет собой смесь html-кода и конструкций C#.

В **разделе 4** приведены скриншоты страниц приложения, которые загружаются браузером при выполнении кода программы, а также описаны функции, выполняемы при загрузке или действиях на этих страницах.

В **заключении** подводятся итоги проведенной работы и приводятся выводы, полученные в ходе разработки.

Общим результатом работы можно считать готовое приложение, способное загрузить все статьи для выборки по любому ВУЗу, будь то СГУ или МГУ, внести эти статьи и всю сопутствующую информацию в базу данных, из которой затем, через интерфейс этого же приложения, можно получить всю информацию по интересующему нас автору, когда-либо работавшего в выбранном ВУЗе или публиковавшегося в соавторствами с сотрудниками этой организации.