МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА ПОИСКОВОГО ПРИЛОЖЕНИЯ С ИСПОЛЬЗОВАНИЕМ ПОЛНОТЕКСТОВОГО ПОИСКА APACHE SOLR

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Лавринова Дмитрия Алексеевича

Научный руководитель
доцент, к. ф.-м. н. _____ И. А. Батраева

Заведующий кафедрой
доцент, к. ф.-м. н. _____ С. В. Миронов

Саратов 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Сбор и исследование больших объемов текстовой информации является актуальной задачей при разработке мультимедийных лингвокультурологических корпусов русской диалектной речи, которые полно представляют частные диалектные системы и основные типы русской диалектной речи. В существующем Саратовском электронном диалектологическом корпусе русского языка, предоставляющим репрезентативный материал для многостороннего изучения специфики диалектных подсистем, наблюдается проблема в скорости обработки объемных массивов текстовой информации, а так же связанных с ними мета информацией, ввиду чего вести исследования представляется затруднительной задачей.

Целью настоящей работы является внедрение платформы полнотекстового поиска Apache Solr в программное обеспечение Саратовского диалектологического корпуса русского языка. Внедряемая функциональность должна осуществлять:

- полнотекстовый поиск в большом объеме данных;
- обеспечение релевантного поиска;
- поиск точных форм;
- лексико-грамматический поиск;
- фасетный поиск;
- сортировку результатов;
- предсказание пользовательского ввода;
- работу с синонимами и стоп словами;
- панель администратора.

Для осуществления поставленной цели необходимо:

1. Создать и настроить локальный Solr сервер;
2. Разработать базовое приложение на технологии ASP.NET MVC 5.0;
3. Разработать модули для работы со статичными данными и динамическими страницами приложения;
4. Разработать модуль для загрузки и индексации документов;
5. Изучить и встроить в приложение набор библиотек SOLRNET, связав воедино все компоненты;
6. Настроить разработанное приложение для работы с произвольными документами;

7. Адаптировать приложение для работы в формате диалектологического корпуса русского языка.

В теоретической части работы описываются все используемые технологии и необходимые предварительные сведения для практической реализации.

В практической части работы приводятся примеры функционирования разработанного приложения.

1 Теоретическая часть

1.1 Apache Solr

Apache Solr — популярная, стремительно развивающаяся поисковая платформа с открытым исходным кодом, построенная на Apache Lucene. Solr обладает высокой надежностью, масштабируемостью и отказоустойчивостью, обеспечивая распределенное индексирование, репликацию и балансировку нагрузки, автоматическое восстановление после сбоев, централизованную конфигурацию и многое другое. Так как есть возможность распределенного поиска и репликации Solr хорошо масштабируем.

Solr работает как автономный полнотекстовый поисковый сервер. Он использует библиотеку поиска Lucene Java в своей основе для полнотекстового индексирования и поиска. Взаимодействие с сервером построено по принципу REST API, а обмен сообщениями происходит посредством XML и JSON, которые делают пригодным Solr для использования на самых популярных языках программирования. Внешняя конфигурация Solr позволяет адаптировать ее ко многим типам приложений без Java-кодирования и иметь архитектуру плагина для поддержки более детальной настройки.

Отличительные черты Solr:

- Открытый исходный код;
- Является очень гибким и настраиваемым решением;
- Позволяет выбирать какой тип функций поиска использовать как во время запроса, так и во время индексирования;
- Solr является Restful;
- Распределенный поиск через Sharding - позволяет масштабировать объем содержимого;
- Поддерживает кэширование;
- Имеется возможность совершать пространственный поиск на картах;
- Множество крупных компаний используют Solr в своих продуктах. Среди них присутствуют Netflix, NASA, Internet Archive, eBay и многие другие крупные компании.

Пользователи обычно взаимодействуют со слоем поиска через приложение или специальные службы, которые работают с его API. Этот слой называется слоем приложения, в котором располагается базовая логика, позволяющая создавать и посылать запросы, а так же получать и отображать результаты в

удобном для пользователя виде. В данной работе будет использоваться веб приложение, разработанное на языке программирования C# с использованием SolrNet в качестве связующего звена.

После того как запрос сформируется он отправляется в поисковую систему (search engine), где выполняется. Затем результаты возвращаются из индекса. Индекс содержит данные, среди которых происходит поиск. Данные могут подать в индекс различными способами. Внизу рисунка изображены примеры источников: базы данных, интернет, файловая система и т.д.

Чтобы иметь возможность извлекать данные из этих источников необходим коннектор (connector), который служит мостом для подключения к источникам данных. После чего данные извлекаются и посылаются в DPMS - конвейер обработки документов, который переводит данные в модель и предоставляет их в индекс по мере необходимости.

Apache Solr внутри использует Lucene, которая является реальной поисковой системой, что управляет корпоративным поиском. Работать напрямую с Lucene крайне сложно, оттого появляется острая необходимость в Solr, как в инструменте серверизации Lucene. Solr работает поверх Lucene и перерабатывает запросы для того, чтобы преобразовать их в то, что ожидает Lucene.

1.2 Поисковая система Lucene

Lucene — поисковая и индексирующая система с открытым исходным кодом, что находится под капотом у Solr. Lucene способна работать с любым документом, который имеет поля, что можно извлекать и индексировать. Однако поиск так же может осуществляться по изображениям, аудио-файлам, цепочкам ДНК и многому другому. Однако, Lucene имеет крайне специфический синтаксис, с которым сложно управиться человеку. Изначально Lucene была разработана на Java, но в дальнейшем была переписана на многие другие языки программирования.

Lucene способна осуществлять очень быстрый полнотекстовый поиск. Это возможно, потому используются индексы, а именно инвертированные. Вместо классического индекса, где для каждого документа формируется полный список слов (или терминов), которые он содержит, инвертированные индексы делают все наоборот: для каждого термина (слова) в документах формируется список всех документов, которые содержат этот термин. Это намного эффективнее при выполнении полнотекстового поиска.

Процесс создания индекса состоит из следующих шагов:

1. Содержимое документа считывается и преобразуется в текст;
2. Текст анализируется;
3. Текст разбивается на токены;
4. Происходит анализ токенов и преобразования их в слова (термины);
5. На основе полученных слов создается инвертированный индекс;
6. Результат процесса добавляется в базу.

1.3 ASP.NET

Приложение СарДК было разработано с использованием возможностей фреймворка ASP.NET, который позволяет разрабатывать клиент-серверное приложение на основе шаблона «Модель-Представление-Контроллер» или MVC. Благодаря использованию шаблона MVC появляется удобное обеспечение уровня абстракции, при котором слои бизнес логики и доступ к данным не зависят от представления, что позволяет заменять, дополнять и расширять их в ходе разработки.

Более подробно трехуровневая архитектура состоит из следующих уровней:

— Уровень представления — состоит из элементов пользовательского интерфейса и методов получения данных от пользователя. Помимо компонентов интерфейса слой включает модели представлений, контроллеры и объекты контекста запроса;
— Уровень бизнес логики — в него включаются методы, обрабатывающие полученные данные со стороны клиента и реализующие основную логику обработки данных.
— Уровень доступа к данным — отвечает за описание моделей, описывающих используемые сущности, а так же по средством различных технологий получения доступа к данным, например, Entity Framework, осуществление взаимодействия с базами данных.

На каждом из уровней присутствуют свои специфические объекты, помогающие выстраивать абстракцию для удобства работы и передачи данных между слоями:

— Модель — есть класс, который описывает сами данные. Как правило является простейшим классом с набором необходимых полей и используется для передачи данных между слоями. Модель возможно оснастить

некоторой простой логикой, но в идеале она должна лишь описывать содержимое, а не функционал.

— Представление — это то, что видит пользователь, а именно интерфейс. Как правило, представление это сформированная html страница, которая была построена ASP.NET на основе данных модели, что пришла на вход как параметр. В данной работе для представления был выбран движок Razor, так как данный инструмент позволяет генерировать коллекции HTML-элементов с помощью HTML-помощников, а также использовать различные инструкции на языке C# для вывода данных и элементов.

— Контроллер — это класс, что связывает представления, модели и логику. Именно этот класс принимает запросы от пользователя и на основе настроенной маршрутизации отправляет и обрабатывает данные. Класс контроллера тесно связан с URL по которым отправляются HTTP запросы пользователей. ASP.NET MVC использует специальную систему маршрутизации обработки запросов. Данная система ставит все входящие запросы в соответствие маршрутным схемам, которые указывают какой контроллер должен обработать данный запрос. По умолчанию используется встроенный маршрут, имеющий следующую структуру: контроллер, действие, необязательные параметры.

Интерфейс приложения создавался с использованием фреймворка Bootstrap 4.0 для обеспечения поддержки адаптивного разрешения экрана пользователя и используемой версии браузера. Фреймворк располагает богатым выбором готовых элементов стандартной веб-страницы: кнопки, блоки навигации, модальные окна, таблицы, подсказки и другое. Для стилизации элементов были использованы такие цветовые темы, как: primary, success, warning, danger. Изменение готовых стилей возможно за счет добавления своих собственных наборов CSS правил, которые переопределяют существующие. Технология сеток в Bootstrap дает возможность реализовать адаптивный дизайн страницы. Добавление новых элементов не нарушает общую структуру благодаря динамически изменяющейся сетке. Для размещения элементов на странице использовались теги контейнеры, что способствует сохранению дизайна страницы вне зависимости от размера окна браузера.

В качестве системы управления версиями была выбрана Git, так как данная система обладает высокой производительностью, удобным интерфейсом

и возможностью публиковать код на GitLab. Благодаря Git регистрируются изменения файлов, что обеспечивает возможность отследить обновления кода и возврат к предыдущим или альтернативным версиям сайта. GitLab — веб-приложение и система управления репозиториями программного кода для Git. Данный ресурс предоставляет платформу для хранения кода и совместной разработки масштабных программных продуктов. Репозитории включают в себя систему контроля версий для размещения различных цепочек разработки и веток, позволяющую разработчикам проверять код и откатываться к стабильной версии программного обеспечения в случае непредвиденных проблем. GitLab предоставляет возможность создания бесплатного приватного репозитория с управлением пользователями и группами, правами доступа к нему.

2 Практическая часть

В этом разделе располагается пошаговое описание создания и конфигурации Solr сервера и его дальнейшая интеграция в веб-приложение Саратовского диалектологического корпуса русского языка и культуры речи. Помимо этого для приложения были разработаны другие различные необходимые модули, поддерживающие и улучшающие его работу.

2.1 Раздел администратора

Помимо основного поискового функционала программное обеспечение СарДК должно обладать возможностью удобного и безопасного администрирования содержащихся внутри корпуса элементов, таких как наполнение сайта статической информацией и контроль доступов к ней. С этой целью был разработан раздел администратора. Для хранения долгосрочной информации была выбрана реляционная база данных MS SQL Server.

На странице «Настройка участников» можно управлять списком лиц, которые участвовали в создании корпуса. В верхней части страницы находится форма, позволяющая внести информацию об имени участника, его должности или вкладе, а так же указать ссылку на профиль. В нижней части отображается список добавленных на данный момент участников и ссылки на страницу с редактированием, перейдя по которой можно увидеть форму для редактирования участника, а так же кнопку полного удаления. Созданные таким образом участники отображаются на одной из главных страниц корпуса, странице «Участники».

С корпусом связано много публикаций, поэтому для их хранения и отображения была разработана отдельная страница «Настройка публикаций». Сверху страницы присутствует форма создания новой записи, которая позволяет указать авторов публикации, название и ссылку. Ниже находится таблица с уже добавленными публикациями. Страницы выполнены в унифицированном стиле для сохранения приятного и понятного пользовательского интерфейса.

Корпус умеет работать с различными подкорпусами, задать заранее определенные и их описание можно на странице «Настройка подкорпусов», где в верхней части страницы находится форма, позволяющая помимо названия и описания так же загрузить изображения. Одно из изображений содержит карту области в которой происходит изучение, а второе на выбор администратора:

что угодно, что как-то идентифицирует подкорпус среди других. Созданные на этой странице подкорпусы отображаются на одноименной странице сайта.

Помимо прочего добавлена возможность создавать и редактировать страницы сайта СарДК в режиме конструктора прямо из панели администратора. На странице создания можно выбрать в каком разделе сайта появится страница, ее заголовок для поисковых систем и вкладки браузера, а так же указать содержимое, используя полноценный язык разметки HTML. Созданные таким образом страницы мгновенно появляются на сайте и доступны пользователям для просмотра.

2.2 Создание и конфигурация Apache Solr сервера

С официального сайта Apache был загружен дистрибутив Apache Solr сервера. В данной работе используется версия 8.8.1 для Windows с поддержкой Java RE версии 1.8 или выше.

Был настроен и запущен графический интерфейс в виде панели администратора. Открыть его можно в любом браузере, набрав в строке ввода URL следующий адрес: localhost:8983. Далее дается краткое описание наиболее важных разделов панели администратора Solr сервера, что были сконфигурированы:

- **Dashboard.** Отображает общую информацию;
- **Logging.** Журнал логирования. Тут можно проанализировать правильность выполнения запросов и попробовать понять, почему тот или иной запрос выполняется медленно;
- **Core admin.** Здесь отображается информация о созданных ядрах. Отсюда необходимо перезагружать ядро после каждого изменения, что внесено в конфигурацию, для того, чтобы эти изменения пришли в силу;
- **Java properties.** Позволяет тонко настраивать Java Virtual Machine для управления выделенной памятью и т.п.;
- **Thread dump.** Позволяет посмотреть потоки, работающие на сервере и их состояния. Поток, отмеченный зеленой галочкой сейчас выполняется;
- **Analysis.** Очень важный раздел, в нем можно проверить как данные обрабатываются во время запроса или во время индексирования. Позволяет увидеть цепочку анализа, то есть шаги, которые Solr принимает для обработки запроса. В поле Index вводится фрагмент, в котором осуществляется поиск, а в поле Query – запрос. В раскрывающемся списке

Field type нужно выбрать тип поля, в котором осуществляется поиск. Обычно это значение устанавливается в Text;

- **Documents.** Позволяет выполнять команды индексации Solr в различных форматах непосредственно из браузера;
- **Query.** Здесь создаются запросы и выводятся результаты в том же виде, в котором их получит приложение.

В раздел «Files» было помещено множество важных конфигурационных файлов. Так в файле «stopwords.txt» был задан список слов, которые не должны попадать в индекс и участвовать в поиске и ранжировании. Сюда был включен символ ударения и двух косых черт, использующийся в разметке для выделения пауз в диалогах.

В файл «synonyms.txt» был записан словарь синонимов русского языка, чтобы поиск был гибче и более соответствовал ожиданиям пользователя. В файле «spelling.txt» были прописаны правила исправления орфографических ошибок, если такие возникают во время ввода поискового запроса. Файл «currency.xml» предоставит возможность описать мировые валюты с их возможными сокращениями, если это будет необходимо в поисковых запросах.

В папке «lang» располагаются файлы с предварительными настройками для различных мировых языков. В нем присутствуют настройки для арабских, европейских и славянских языков. В случае необходимости данные настройки можно дополнить, записав в файл соответствующие инструкции.

На страница «Plugins» отображаются подобранные и установленные расширения и библиотеки для Solr сервера, которые оказывают влияние на работу в целом и непосредственно поиск. На данный момент на сервер установлены расширения для администрирования, кеширования, проверки орфографии и репликации.

Репликация индекса распределяет полные копии главного индекса на один или несколько подчиненных серверов. Главный сервер продолжает управлять обновлениями индекса. Все запросы обрабатываются подчиненными. Такое разделение труда позволяет Solr масштабироваться, чтобы обеспечить адекватную реакцию на запросы при больших объемах поиска.

Главный сервер может обслуживать только определенное количество ведомых устройств, не влияя на производительность. Если каждое ведомое устройство загружает индекс из удаленного центра обработки данных, резуль-

тирующая загрузка может потреблять слишком много пропускной способности сети. Чтобы избежать ухудшения производительности в таких случаях, существует возможность настроить одно или несколько подчиненных устройств в качестве повторителей. Повторитель — это просто узел, который действует как ведущий и ведомый.

Solr реплицирует файлы конфигурации только тогда, когда реплицируется сам индекс. Это означает, что даже если файл конфигурации будет изменен на главном сервере, этот файл будет реплицирован только после того, как в индексе главного сервера будет произведена новая фиксация или оптимизация.

2.3 Создание ядра индексации

Было создано ядро индексации Solr, которое является экземпляром Lucene индекса, что содержит все файлы конфигурации Solr и связанный журнал транзакций. Это необходимо, что иметь возможность выполнять такие операции как индексация и отправка запросов. Более того, экземпляр Solr может содержать и более одного ядра, если потребуется, а индекс может быть распределен между несколькими узлами. Такое состояние называется коллекцией.

Первый файл, который был настроен после создания ядра — это «schema.xml». В нем было прописано какие поля и мета данные содержатся в индексируемых документах и каким образом их нужно обрабатывать как при индексации, так и во время выполнения запросов. Это делается средством объявления полей с указанием их имени, типа и дополнительных атрибутов.

2.4 Интеграция Solr в приложении СарДК

Все описанные выше команды исполняются через программный код автоматически при совершении определенных действий в программном обеспечении. Так, например, происходит автоматическая индексация документа в ядро Solr непосредственно при его загрузке на сайте. Загрузка новых документов осуществляется в разделе «Добавление документа в Solr индекс». Допустимо добавлять документы в xml формате с предварительно размеченными тегами и мета информацией текстами.

В случае, если загруженный документ будет неверно размечен или содержит другие непредвиденные ошибки, то на работу корпуса он не повлияет. Загружающему администратору будет отображено сообщение о возникшей ошибке и предложено удалить или поправить поврежденный документ.

В случае если документ был полностью размечен верно он сразу же попадает в индекс Solr и начинает участвовать в поиске. Далее исследователь может обратиться к таким разделам сайта как «Поиск точных форм (Solr)» и «Лексико-грамматический поиск (Solr)» в которых, используя разные алгоритмы, осуществить необходимый поиск.

Поиск точных форм принимает на вход запрос, что состоит из одного слова, и находит все его контексты. Результатом поиска является выдача мета информации о корпусе: объем документов, в которых осуществлялся поиск, объем словоупотреблений и количество удовлетворяющих запросу документов. Результирующая выдача формируется из найденных абзацев. Для удобства пользователя все найденные словоформы выделяются жирным шрифтом и цветом. Выдача происходит по 10 фрагментов текста на страницу.

Вместе с возможностью добавлять фильтры по лексико-грамматическим признакам так же была реализована функциональность поиска с использованием масок. Solr предоставляет возможность внедрения и конфигурирования различных масок, однако для данного проекты были выбраны и реализованы маски с символами «*» и «_», где символ звездочки означает любую, в том числе и пустую последовательность символов, а символ подчеркивания обозначает ровно один любой не пустой символ.

По мимо участия в поиске в виде отдельных абзацев, загруженные тексты хранятся в ядре Solr и доступны для просмотра и анализа целиком. В заголовке можно увидеть блок с мета информацией, содержащей такие важные характеристики документа как подкорпус, автор, регион, дата записи и образование. Внутри между абзацами присутствуют блоки с текстами, принадлежащими диалектологу.

2.5 Анализ прироста скорости работы поисковой системы

Необходимость использования Apache Solr в работе СарДК обусловлена наличием острых проблем с низкой скоростью как в случае документов в корпус, так и осуществлении поисковых запросов к ним. Внедрение Solr сервера с наличием обратного индекса позволило значительно ускорить оба этих аспекта работы корпуса. Как видно из графика время загрузки в случае с Sql Server растет экспоненциально, что было бы недопустимо в реальных рабочих условиях. Solr сервер же показывает стабильный результат в несколько секунд и рост времени поиска является линейным.

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы были реализованы все поставленные задачи и цель. Был развернут и настроен Solr сервер, а так же были разработаны модули для администрирования и работы с данным сервером. В программное обеспечение Саратовского диалектологического корпуса русского языка были внедрены модули для взаимодействия с ядром Solr, позволяющие исследователю совершать релевантный полнотекстовый поиск в большом объеме произвольных данных с возможностью фасетного поиска, сортировки и предсказания пользовательского ввода. Благодаря внедрению Solr удалось в разы улучшить скорость обработки массивов текстовых данных и тем самым увеличить производительность исследования корпусов русского языка.