ПРОГРАММА СБОРА ДАННЫХ О СТРУКТУРЕ ВЕБ-САЙТОВ

Потехин А. С., Александров А. А., Русецкий П. Н.
Кафедра информатики, Белорусский государственный университет информатики и радиоэлектроники
Минск, Республика Беларусь
E-mail: redrumofgod@gmail.com

В настоящее время всеобщие глобальные тенденции приближаются к тому, что все операции и торговые сделки будут проходить с использованием веб-ресурсов. Для того, чтобы успешно вести бизнес очень важно получать актуальные данные о движения рынка (динамика цен и товаров) и локальные новости, которые порой всецело влияют на формирование спроса, своевременно. Но необходимые данные не всегда легко доступны пользователю и чаще всего они не структурированы. Рассматривается приложение, которое будет обладать необходимым функционалом для сбора и структурирования данных с различных веб-ресурсов.

Введение

Анализ данных, представленных в Вебе, — распространенная на сегодняшнии день исследовательская задача. Инструменты, позволяющие собирать данные для исследовании из Веба, значительно развиты и позволяют не писать с нуля новые, но использовать и дорабатывать существующие [1, с.44]. Такие инструменты называются «веб-пауками» (web-spider), краулерами (web crawler) или скраперами (web scraper). В даннои статье рассматриваются некоторые имеющиеся на рынке краулеры, обосновывается решение написания собственного проекта.

I. Основная задача и ее ограничения

Целью исследования, требующего сбор данных из Сети Интернет, является сентимент-анализ данных с различных новостных сайтов. Данные должны содержать полную информацию о новости, включая заголовок, текст, дату и автора новости. Для того, чтобы обеспечить сбор указанной информации, необходимо реализовать инструмент – скрапер.

В широком понимании скрапер служит для сбора данных из различных интернет-ресурсов. Общий принцип его работы можно объяснить следующим образом: автоматизированный код выполняет запросы на целевой сайт и получая ответ, разбирает HTML-документ, ищет данные и преобразует их в заданный формат. Данные инструменты скрапинга позволяют вручную или автоматически извлекать новые или обновленные данные и сохранять их для последующего использования.

Для того чтобы выполнять эту задачу, инструмент должен поддерживать работу со следующими данными:

1. HTML, JavaScript, так как большинство сайтов построены с использованием этих технологий;
2. Plain text, PDF и другие форматы представления текстовых данных;
3. URLs, с возможностью построения на их основе графа веб-ресурсов.

Кроме того к краулеру предъявляется требование производительности. Согласно первичнои оценке объема данных, которые предстоит собрать и обработать, нужно извлечь данные с 100 новостных саитов, при условии того, что каждыи из них в среднем выпускает по несколько статей в день, в худшем случае необходимо будет обработать около 108000 страниц. При среднеи скорости обработки 2 стр/сек, которую обеспечивают краулеры со средней производительностью[4], нам потребуется около 20 суток, чтобы собрать указанные данные. Такое время обработки повлечет увеличение длительности процесса сбора и анализа данных, поэтому для исследовании требуется производительныи и надежныи краулер. Отдельным ограничением стоит отметить стоимость искомого инструмента, для исследовательских целеи нами рассмотрены только решений с открытым исходным кодом, т.к. они распространяются бесплатно и их исходныи код доступен для анализа и редактирования. Рассмотрим общии список требовании, которым должен отвечать искомыи инструмент.

II. Требования к инструменту

Приступая к выбору инструмента, необходимо обозначить основные требования к краулеру [1],[2],[3], на основании которых можно бы было проводить их сравнительную оценку:

1. Надежность – Веб содержит ресурсы, которые могут вводить скрапер в бесконечный цикл или недоступные сервисы, ожидать выполнения которых, он не должен. Скрапер должен быть устойчивым к таким ловушкам;
2. Вежливость – интернет-ресурсы имеют явные и неявные политики, регулирующие частоту, с которой скрапер может посетить их. Они описаны в файле robots.txt и эти политики должны соблюдаться;

3. Распределенность – скрапер должен иметь возможность выполняться в распределенном режиме на нескольких машинах;
4. Масштабируемость – скрапер должен поддерживать возможность увеличения производительности за счет добавления дополнительных вычислительных узлов, на которых он исполняется;
5. Производительность и эффективность – скрапер должен обеспечивать эффективное использование системных ресурсов, включая процессор, память и полосу пропускания сети;
6. Качество – скрапер должен уметь отделять спам-страницы от полезных и извлекать последние;
7. Актуальность – скрапер должен поддерживать обновление собранных данных;
8. Расширяемость – скрапер должен быть модульным, т.е. позволять добавлять новую функциональность, для анализа новых форматов данных, протоколов и т.д.

Помимо описанных общих требований для скраперов можно выделить основные требования для нашей задачи исследования:

1. Скрапер должен быть кросплатформенным, чтобы его можно было одинаково настраивать и конфигурировать на вычислительных узлах с разными операционными системами;
2. Скрапер должен обеспечивать производительность обработки порядка 100 стр/сек, чтобы время сбора описанного выше объема данных составляло часы, а не дни. В том случае если окажется, что данных для сбора и анализа больше предполагаемого, скрапер должен предоставлять возможность легко увеличить его производительность путем выделения ему для работы большего числа потоков или добавления дополнительных вычислительных узлов;
3. Скрапер должен быть интегрирован с базои данных для хранения собранной информации и полнотекстовым индексом, позволяющим быстро извлекать данные для последующего анализа, отвечающие указанным условиям;
4. Требуется скрапер для сбора данных в ширину и вертикального поиска, так как в указанной задаче необходимо извлечь информацию о конкретной предметной области, а не узкое множество фактов;

III. Обоснование выбора скрапера

Исходя из нескольких источников информации, дающих представление относительно того или иного продукта с открытым исходным кодом, не было найдено ни одного решения, в полной мере подходящей для данной задачи исследования. Многие из скраперов перестали поддерживаться сообществом и развиваться, многие не являются кросс-платформенными, не обеспечивают необходимую скорость скачки или являются проблематичными для настройки. Также были рассмотрены облачные сервисы, но для решения данной задачи они не подходят, так как не рассчитаны на хранение и обработку большого количества данных, на ограниченных вычислительных ресурсах или не имеет бесплатной опции доступа.

В результате проведенного анализа было принято решение реализовать свой собственный скрапер, который будет являться эффективным инструментом для поиска в Вебе, ядро будет написано на C++ с которым взимодействует Ruby-оболочка, будет поддерживаться граф связеи узлов, различные парсеры, фильтры и нормализаторы URL. Он будет позволять использовать различные хранилища данных, такие как Cassandra, Hbase и др. Скрапер также будет являеться масштабируемым (до 100 узлов в кластере) и легко настраиваеться и расширяться, в полнои мере являться «вежливым».

Заключение

В даннои работе рассмотрены требования и основные типы скраперов, которые на сегодняшнии день выделяют в литературе. Исходя этого не были найдены скраперы, удовлетворяющие условиям реализации исследовательского проекта и в результате было принято решение реализовывать свой собственный скрапер.

Список литературы

1. PAPAVASSILIOU V., PROKOPIDIS P., THURMAIR G. A modular open-source focused crawler for mining monolingual and bilingual corpora from the web // Proceedings of the 6th Workshop on Building and Using Comparable Corpora. – 2013.
2. AHUJA M.S., BAL J.S., VARNICA Web Crawler: Extracting the Web Data // International Journal of Computer Trends and Technology. – 2014.
3. YADAV M., GOYAL N. Comparison of Open Source Crawlers-A Review// International Journal of Scientific Engineering Research. – 2015.
4. Performance Benchmark [Электронныи ресурс]. URL: https://geekflare.com/web-scraping-frameworks/ (дата обращения: 03.06.2019)