Министерство образования и науки

Российской федерации

Федеральное агентство по образованию

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Омский государственный технический университет»

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

*Анохина Павла Викторовича*

**Разработка и исследование характеристик надежности системы**

**автоматического извлечения знаний из поисковых систем**

**Руководитель Студент**

КалекинД.В. Анохин П.В.

(фамилия, И., О.) (фамилия, И., О.)

К.т.н. , доцент ФЭОиМ, ИВМ–612

(уч. степень, уч. звание) (факультет, группа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата) (подпись, дата)

Омск, 2014 г.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 5](#_Toc388286557)

[1 Анализ текущего состояния информационного поиска в сети интернет 9](#_Toc388286558)

[1.1 Современные поисковые системы и тенденции их развития 11](#_Toc388286559)

[1.2 Проблемы в современном информационном поиске 14](#_Toc388286560)

[1.2.1 Особенности интернета с точки зрения хранения информации 15](#_Toc388286561)

[1.2.2 Гетерогенность сети интернет и семантическая неоднородность веб-страниц 19](#_Toc388286562)

[1.2.3 Надежность алгоритмов интеллектуального анализа данных при обработке содержимого веб-сайтов 21](#_Toc388286563)

[1.3 Использование поисковых систем как первого этапа в процессе поиска информации 25](#_Toc388286564)

[1.4 Выводы 27](#_Toc388286565)

[2 Анализ моделей и методов извлечения знаний 29](#_Toc388286566)

[2.1 Методы интеллектуального извлечения данных 30](#_Toc388286567)

[2.1.1 Задача системы извлечения данных 38](#_Toc388286568)

[2.1.2 Разработка метода разбиения содержания веб страниц на семантические блоки, основанного на анализе объектной структуры документа 39](#_Toc388286569)

[2.1.3 Анализ алгоритма сегментации веб-страниц, основанного на «компьютерном зрении» 42](#_Toc388286570)

[2.1.4 Разработка алгоритма извлечения знаний, основанного на шаблонах и их взаимном расположении 44](#_Toc388286571)

[2.1.5 Разработка алгоритма извлечения краткой описательной информации об объекте 48](#_Toc388286572)

[2.2 Анализ и выбор методов интеллектуального анализа данных 50](#_Toc388286573)

[2.2.1 Кластеризация и классификация на основе модели нейронной сети Кохонена. 51](#_Toc388286574)

[2.3 Разработка модели представления знаний в системе 53](#_Toc388286575)

[2.3.1 Иерархическая фреймовая модель 54](#_Toc388286576)

[2.3.2 Разработка алгоритма заполнения значений терминальных слотов фрейма-образца 56](#_Toc388286577)

[2.3.3 Разработка алгоритма сравнения двух фреймов-образцов 57](#_Toc388286578)

[2.4 Выводы 59](#_Toc388286579)

[3 Проектирование и разработка системы автоматического извлечения знаний из поисковых систем 61](#_Toc388286580)

[3.1 Анализ основных требований к системе в плане надёжности 63](#_Toc388286581)

[3.2 Разработка структуры системы 67](#_Toc388286582)

[3.2.1 Проектирование подсистемы извлечения данных с веб-сайтов 71](#_Toc388286583)

[3.2.2 Проектирование подсистемы интеллектуального анализа данных 74](#_Toc388286584)

[3.2.3 Проектирование базы данных и подсистемы работы с БД 77](#_Toc388286585)

[3.2.4 Проектирование подсистемы мониторинга 80](#_Toc388286586)

[3.3 Разработка программной реализации системы 82](#_Toc388286587)

[3.4 Выводы 89](#_Toc388286588)

[4 Проведение экспериментов и анализ результатов 90](#_Toc388286589)

[4.1 Исследование зависимости надежности алгоритмов извлечения данных от методов предварительной обработки. 90](#_Toc388286590)

[4.1.1 Условия проведения и входные данные экспериментов 91](#_Toc388286591)

[4.1.2 Автоматическое извлечение знаний с использованием алгоритма сегментации основанного на анализе объектной модели документа 92](#_Toc388286592)

[4.1.3 Автоматическое извлечение знаний с использованием алгоритма сегментации VIPS 94](#_Toc388286593)

[4.1.4 Сравнение результатов первых двух экспериментов 96](#_Toc388286594)

[4.1.5 Автоматическое извлечение знаний без использования алгоритма сегментации 99](#_Toc388286595)

[4.1.6 Сравнение результатов работы системы с использованием алгоритмов сегментации и без предварительной сегментации. 101](#_Toc388286596)

[4.2 Исследование процесса долгосрочного функционирования подсистемы извлечения данных на основе Марковской модели 103](#_Toc388286597)

[4.2.1 Условия проведения и входные данные экспериментов 105](#_Toc388286598)

[4.2.2 Проведение экспериментов и построение модели системы, основанной на Марковских процессах 105](#_Toc388286599)

[4.2.2 Анализ полученных результатов 113](#_Toc388286600)

[4.3 Выводы 114](#_Toc388286601)

[ЗАКЛЮЧЕНИЕ 116](#_Toc388286602)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 119](#_Toc388286603)

# ВВЕДЕНИЕ

Автоматическое извлечение знаний из различных источников является одной из достаточно молодых и актуальных на сегодняшний день областей науки. В современном мире информация, а также скорость ее получения и обработки играют важную роль во многих отраслях производства и бизнеса. Но для получения необходимой информации, часто приходится обрабатывать огромные массивы данных. Поэтому ведутся разработки различных алгоритмов, позволяющих автоматически извлекать знания из различных источников.

Автоматическое извлечение знаний из различных источников является молодым направлением в научной и инженерной деятельности. В современном мире информация, скорость ее получения и обработки играют важную роль в производственном и финансовом секторе бизнеса. Для оперативного и точного получения релевантной требуемому запросу информации приходится обрабатывать огромные массивы данных. Поэтому в США, России и странах Евросоюза ведутся разработки алгоритмов, повышающих эффективность извлечения знаний из множества источников.

Одним из наибольших массивов информации в современном мире является всемирная сеть интернет. В научной области существует отдельно направление занимающиеся проблемами извлечения данных и знаний с веб-страниц –WebMining.На сегодняшний день разработано много различных алгоритмов извлечения различных типов данных с веб страниц, но даже самые современные алгоритмы, реализованные на самых мощных аппаратных решениях, пока не могут соперничать по своей эффективности и надежности с анализом выполняемым человеком[1]. Именно поэтому актуальность исследований в данной области крайне высока.

Наибольшим массивом информации в современном мире является всемирная сеть интернет, для которого появилось отдельное направление WebMining, специализирующееся на проблемах извлечения знаний с веб-страниц. На сегодняшний день разработано много алгоритмов извлечения с веб-страниц данных разного характера, но даже самые современные алгоритмы, реализованные на самых мощных аппаратных решениях, пока не могут соперничать по своей эффективности и надежности с анализом выполняемым человеком[1]. Именно поэтому актуальность исследований в данной области крайне высока.

Наиболее эффективным инструментом для первичного поиска информации в интернете на сегодняшний день являются поисковые системы, которые позволяют уменьшить область поиска необходимой информации до небольшого множества сайтов. Именно поэтому при выполнении поиска источников необходимой информации чаще всего на первом шаге используют возможности поисковых систем[2].

Разработка системы автоматического извлечения и анализа знаний из интернет-ресурсов является одной из приоритетных задач для ООО «АИСистемс», где разрабатываются информационные продукты (такие как крупные интернет-порталы), основанные на базах данных большого объёма (гигабайты), содержащих информацию различного характера, сбор и последующий анализ которой – процесс весьма трудоемкий. Помимо того, собранные данные необходимо профильтровать (проверить на корректность), проанализировать их качество (в виде цифровых метрик), проверить на актуальность.

Вышеперечисленные задачи, как правило, решаются независимо друг от друга – для сбора информации разрабатываются отдельное программное обеспечение, аккумулирующее информацию из интернета на основе жестко заданных правил разбора веб-страниц, индивидуальных для каждого веб-ресурса. Далее для каждого полученного набора данных, в зависимости от его специфики, создаются различные узкопрофильные разработки для определения качества извлечения, анализа и коррекции полученных результатов. В процессе извлечения знаний используется множество инструментов, написанных на различных языках программирования, не имеющих общих интерфейсов взаимодействия. Очень часто для анализа данных, почти идентичных уже существующим, но имеющих небольшие отличия, приходится изменять инструмент (если его исходные коды доступны).

Сбор большого объема данных в сети Интернет – длительная по времени процедура (дни или недели непрерывного функционирования программно-аппаратного средства в зависимости от размера анализируемой выборки и скорости интернет соединения). Сбор требует постоянного контроля инженера, включает просмотр собранных данных и их выборочную проверку, что чрезвычайно трудоемко на больших объемах данных. Уменьшение роли человека при функционировании подобных программных средств позволит повысить объем извлекаемой выборки, а за счет уменьшения сроков их сбора и анализа – их актуальность (помимо сокращения экономических издержек).

Цель работы – разработать и исследовать характеристики надежности системы автоматического извлечения знаний из поисковых систем.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Разработать алгоритм функционирования системы извлечения знаний из поисковых систем.

2. Реализовать программное обеспечение на основе разработанного алгоритма.

3. Произвести тестирование разработанного ПО в нескольких режимах эксплуатации.

4. На основе сведений, полученных в результате тестирования произвести анализ характеристик надежности разработанной системы.

Основная идея разрабатываемой системы состоит в том, что алгоритмы интеллектуального извлечения и проверки не могут соперничать по степени эффективности с работой, выполняемой человеком, но весьма эффективны при использовании в роли первичного анализа больших объемов информации. Совмещение процесса автоматического извлечения данных с их последующей интеллектуальной обработкой позволяет создать систему, эффективную для обработки больших массивов и надежную с точки зрения качества извлечения.

В качестве одного из основных средств для получения первичных данных были выбраны глобальные информационно-поисковые системы, которые представляют собой мощное средство для поиска веб-ресурсов, релевантных запросу. Однако часто поисковые выборки, выдаваемые данными системами имеют достаточно большой объем, что затрудняет их ручной анализ. Система автоматического извлечения знаний позволит выполнить обход поисковой выборки размером даже в десятки или сотни ссылок, если данное действие необходимо в целях обеспечения требуемого качества извлекаемой информации.

Задачи по извлечению информации, которые могут быть поставлены перед подобной системой, могут требовать реализацию новых алгоритмов и методов извлечения знаний. Поэтому было принято решение о добавлении в систему механизма, реализующего возможность легкой модификации и модернизации системы в дальнейшем. Таковым механизмом является система подключаемых внешних модулей.

Зависимость надежности и эффективности некоторых алгоритмов по извлечению данных от объема и качества входной информации поставили задачу по добавлению в систему методов предобработки входящих данных для уменьшения объема лишней информации в процессе ее обработки. Для этого в систему были встроены два алгоритма сегментации веб-страниц. Один из них был разработан автором на основании алгоритмов выделения семантических блоков на веб-странице с помощью анализа DOM, а другой является разработкой компании Майкрософт и осуществляет разбиение веб-страниц на семантические блоки с помощью визуального анализа.

Данная система была реализована в виде серверного программного обеспечения и протестирована на наборе тестовых задач. На основании статистических данных были исследованы показатели надежности системы и сделаны выводы о дальнейших вариантах модернизации системы.

# Анализ текущего состояния информационного поиска в сети интернет

В настоящее время всемирная сеть интернет является основным хранилищем данных, из-за гигантских объемов которых остро встает проблема поиска нужной информации. Одним из способов первичного поиска информации в сети интернет является использование глобальных поисковых систем. Главной задачей, которую они выполняют - сформировать список документов, релевантных поисковому запросу, введенному пользователем.

Уже из самой формулировки данной задачи следует несколько вопросов, на которые сложно дать однозначные ответы. Главным среди них является вопрос оценки степени релевантности документов. Именно в этом направлении производили свои основные разработки создатели большинства популярных поисковых систем. В погоне за улучшением эффективности алгоритмов и методов по вычислению релевантности анализируемых страниц не учитывались следующие немаловажные вопросы:

* Насколько большим должен быть размер поисковой выдачи?
* Какие пределы релевантности для выдаваемых документов?
* Какова сложность составления поискового запроса для рядового пользователя?

Рассматривая эти вопросы, можно заметить слабые места в алгоритме функционирования современных информационно-поисковых систем. Увеличение размера итогового списка ссылок на документы, которые выдает система в ответ на запрос, обеспечивает большую информационную полноту. Рассмотрим процесс получения новых знаний в информационно-поисковых системах пользователем, по шагам:

1. Составление поискового запроса.
2. Поиск релевантных запросу документов.
3. Извлечение необходимых данных.
4. Анализ полученных данных, получение знаний.

Можно заметить, что следующим этапом после получения результатов выдачи поисковой системы будет обработка полученного списка релевантных документов пользователем. Следовательно, производительность данного алгоритма упирается в производительность человека, выполняющего анализ полученных документов на предмет получения новых знаний.

До недавнего времени поисковые системы никак не решали эту проблему, выдавая огромные объемы релевантных документов и предлагая пользователю анализировать эти большие объемы информации самостоятельно[2].

Основной причиной, вызвавшей появление этих проблем является то, что до недавнего времени ввиду слабой вычислительной мощности, компьютеры не могли выполнять тот анализ данных, который легко выполняется человеком. Как правило, это анализ слабоструктурированных и неструктурированных данных.

В настоящее время эта область стала интенсивно развиваться. Благодаря увеличению производительность современных ЭВМ стало возможно применение различных методов анализа текстов и методов интеллектуального анализа данных, для выполнения анализа различных данных больших объемов. И чем сильнее развивались данные области, тем больше становился круг практических задач, для решения которых они могли эффективно применяться. Именно это послужило толчком к переосмыслению того, какой должна быть идеальная информационно-поисковая система. Можно сформулировать несколько идей, которые должны отличать поисковые системы будущего от поисковых систем начала 21 века:

1. Поисковая система должна полностью покрывать сеть интернет (самая популярная на данный момент поисковая система Googleохватывает лишь 76.30% от общего числа сайтов в сети интернет[3], и лишь 0.04% от общего объема данных находящихся в нем[4]);
2. Поисковая система должна позволять пользователю составлять поисковые запросы на естественных языках;
3. Поисковая система должна максимально возможно автоматизировать процесс анализа информации, для выдачи минимальных объемов информации, но с сохранением полноты ответа на запрос пользователя;
4. Поисковая система должна использовать различные средства визуального представления информации для улучшения восприятия информации пользователем;

Достижение всех этих целей потребует огромных трудозатрат для инженеров в сфере информационного поиска, а также все более производительные ЭВМ.

## 1.1 Современные поисковые системы и тенденции их развития

В настоящее время поисковые системы, базирующиеся на поиске ключевых слов, являются наиболее распространенными в сети интернет. Функционируют они в полном соответствии с классической моделью ИПС.

Пользователь вводит запрос слов, которые должны присутствовать в найденных документах. Система, проходя по всему поисковому индексу находить документы, в которых встречаются такие слова, ранжирует их по определенным правилам и формирует итоговую выдачу ссылок на документы.

Несмотря на наличие определенного ряда проблем, информационно-поисковые системы, базирующиеся на поиске ключевых слов, являются эффективными механизмами поиска информации в интернете. Именно поэтому, когда появились первые сигналы о том, что поисковые системы не справляются с той функцией, которая на них возложена, многие специалисты в области информационного поиска стали искать решения этих проблем модифицируя стандартные алгоритмы.

Попытки технологического развития в рамках современной теории информационного поиска сегодня очень часто не улучшает, а ухудшает ситуацию[2]. Например, совершенствование технических аспектов информационно-поисковых систем лишь приводит к увеличению объемов релевантных наборов, которые зачастую не пригодны к употреблению.

Надежды, возлагавшиеся в свое время на идею последовательного уточнения поиска ("искать в найденном", "показать подобное" и т. п.) не оправдались по двум причинам. Во-первых, интересующий потребителя документ может просто не оказаться в первичной выборке, в силу чего последующие итерации теряют смысл, а во-вторых, составление уточняющего запроса, качественно отличающегося от исходного, представляет собой отнюдь не простую задачу, прямо скажем, непосильную для рядового пользователя. Пользователи не ищут в сетях то, что и так знают - в этом нет никакого смысла. Пользователямнеобходимо что-то, чего они не знают, и лишь пытаются объяснить машине, в каком сегменте информационного пространства это "что-то" должно находиться. Указание набора слов, с помощью которых этот сегмент можно локализовать, оказывается отнюдь не самым совершенным способом достижения поставленной цели. Очевидно, следует признать, что изначальная парадигма поисковых систем, сформированная десятки лет тому назад, уже не отвечает реальной ситуации. Поэтому возникает задача поиска новых концепций для проектирования качественно новых систем обработки больших и, что самое главное, динамически изменяющихся объемов данных.

Также не принесло серьезных успехов введение сложных формальных конструкций в язык формирования поисковых запросов, который понимает система. Изначально планировалось, что пользователь сможет с помощью подобных конструкций установить дополнительные связи между ключевыми словами, что позволит системе более точно найти необходимые ему документы. Как правило, язык поисковых запросов состоит из логических операторов, префиксов обязательности, возможности учета расстояния между словами, морфологии языка, регистра слов, расширенных операторов, возможностей расширенного поиска, уточнения поиска.

Однако после введения подобных средств в массово используемые информационно-поисковые системы выяснилось, что лишь малая доля пользователей поисковых систем используют данные возможности. Специалисты обосновывают слабую популярность данных поисковых возможностей их большой сложностью для неподготовленных людей.

Как видно, подобные пути решения проблемы, основанные на более детальном манипулировании пользователем над поисковым запросом, не привели к ожидаемому улучшению.

Следует обратить внимание на пару важных моментов:

1. Точные алгоритмы, по которым современные поисковые системы вычисляют релевантность сайта тому или иному запросу являются закрытой информацией, и постоянно изменяются.
2. Существует множество сторонних факторов, которые влияют на результаты сайта в поисковой выдаче –PageRank, GeoLocation, доверенность страницы и другие.

Следовательно, из-за наличия такой искусственной оценки релевантности документов как плотность ключевых слов, а также набора дополнительных параметров, возникает проблема по использованию поисковой системы человеком неосведомленным о принципах ее работы. Это позволяет искусственно повышать результаты сайта в выборке без изменения его основного содержания, что, по сути, и отображает всю неустойчивость данной модели.

## 1.2 Проблемы в современном информационном поиске

Одна из основныхпроблем современного информационногопоиска, это различие понятий "релевантность" и "пертинентность".

Наличие этих двух терминов свидетельствует о том, что различие было известно всегда, но в условиях ограниченных объемов данных им можно было пренебречь.Пользователь вручную анализируя всю релевантную выборку, мог отобрать необходимую ему информацию. В настоящее время, когда все чаще это становится невозможным, несовпадение релевантности с пертинентностьювсе сильнее влияет на результаты информационного поиска. Действительно, если из 10 тысяч предъявленных ИПС документов все являются пертинентными, то потребитель будет удовлетворен, по крайней мере, в первом приближении, прочитав любое их число. Остальные он может просто проигнорировать без особого ущерба для достижения поставленной цели. В некоторых областях отмеченная закономерность эффективно используется. Так, например, службы индикации новостей обслуживают своих клиентов, притом, что количество охватываемых источников информации практически у любой из них в настоящее время не превосходит 10 тысяч. При этом следует отметить, что проблему полноты новостной информации такой подход позволил решить, оставив, однако, нерешенной проблему формирования достаточного для пользователя объема информации.

Существующие ИПС изначально проектировались для обеспечения именно релевантности выборки по отношению к формальным запросам, и в этом их главная слабость в современных условиях. Низкий, а точнее говоря, неконтролируемый уровень пертинентности выборки с высоким уровнем ее релевантности порождает различные ситуации, допускающие более или менее общую типизацию.

### 1.2.1 Особенности интернета с точки зрения хранения информации

Интернет большую часть времени своего существования развивался не как хранилище информации, допускающее свободную обработку, а как носитель документов, используемых людьми. Что в свою очередь очень сильно отразилось на его структуре.

Многие современные исследователи в области глобальных сетей, выделяют в сети интернет две составляющие:

* SurfaceWeb[3] – область интернета доступная большинству современных поисковых систем. Представлена статическими или динамическими веб-сайтами, соединенными друг с другом с помощью гиперссылок.
* DeepWeb[4]– область, недоступная для большинства поисковых систем. Представляет собой FTP-сервера, интерфейсы работы с базами данных, а также динамически создаваемые страницы, требующие для своего создания большие наборы параметров.

Как правило, большинство пользователей даже не задумываются о подобном разделении интернета, однако следует заметить, что область “глубинного веба” растет в современных условиях намного интенсивнее чем “поверхностный веб”. Это вызвано появлением большого количество информации в интернете предназначенного не для обработки человеком: данные межмашинного взаимодействия, XML структуры и т.д.

Несмотря на то, что данные “глубинного веба” являются более структурированными, большинство современных поисковых систем игнорируют эти огромные объемы информации. Также отсутствует какая-либо стандартизация интерфейсов для работы с этими данными.

Если же рассматривать особенности “поверхностного веба” с точки зрения хранилища информации, то проблемы поиска в данной области вызывает в первую очередь отсутствие какой-либо метаинформации, которая бы позволяла достаточно эффективно производить анализ сайтов. Именно поэтому одним из путей решения проблем информационного поиска в сети интернет представляют изменение структуры большинства веб-страниц, и создания на каждой из них определенной структуры, состоящей из мета-тегов, позволяющей рассматривать каждую страницу как часть огромной семантической сети. Эта концепция получила название Web 3.0 [5]в продолжение концепции Web 2.0.

Важнейшим недостатком существующей структуры интернета, с точки зрения автоматического поиска информации, является то, что он практически не использует стандартов представления данных, удобных для понимания компьютером, а вся информация предназначена в первую очередь для восприятия человеком. Для решения проблем, связанных с попытками добавить служебную информацию о структуре описываемых объектов в текст предназначенный для восприятия человеком используются онтологии, позволяющие описать любую предметную область в понятных для машины терминах и эффективно использовать программных агентов.

При использовании данного подхода, дополнительно к видимой человеком информации на каждой странице имеются также и служебная информация, позволяющая эффективно использовать данные программными агентами с целью автоматического анализа.

Поэтому можно сказать что онтологии являются составной частью глобального перехода интернета на качественно новый уровень, называемый семантическая (SemanticWEB).

Для глобальной реорганизации всемирной сети требуется целый набор взаимосвязанных технологий. На рисунке 1.1 приводится структура понятий семантической паутины.

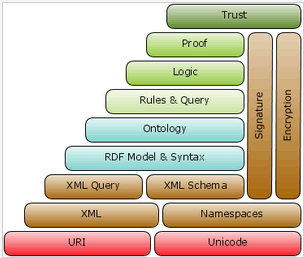


Рисунок 1.1. Основные понятие семантической паутины

Однако на настоящий момент эта концепция так и не получила реализации в современном интернете. По большей части это вызвано тем, что для ее внедрения потребуется изменение структуры почти всех существующих веб-страниц. Что в свою очередь может быть эффективно с точки зрения экономических показателей только в том случае, если принесет кардинальные улучшения в информационном поиске. Однако взять на себя обязательства начать этот процесс пока не решилась ни одна из ведущих компаний в сфере интернет поиска. Хотя, проекты, которые в будущем могут перерасти в глобальную семантическую сеть, уже разрабатываются.

В сообществе веб-разработчиков существуют и сомнения в возможности полной реализации семантической паутины. Основные тезисы в пользу сомнения в возможности создания эффективной семантической паутины:

• Человеческий фактор: люди могут врать, ленится добавлять мета описания, использовать неполные или просто неправильные метаданные. Как вариант решения данной проблемы можно использовать автоматизированные средства создания и редактирования метаданных.Однако на текущий момент развития алгоритмов и методов анализа естественной информации решение данной задачи практически неосуществимо.

• Излишнее дублирование информации: каждый документ должен иметь полное описание как для человека, так и для машины.

Это отчасти решается внедрением микроформатов.

Современные веб-сайты уже вышли за пределы семантических возможностей HTML, которые были предназначены для этого языка. У разработчиков закончились элементы и атрибуты HTML, которые обогащают семантическую разметку документов. Потому что HTML как семантический язык разметки страдает от фундаментального дефекта — его семантика фиксирована и не расширяема.

Это не просто теоретическая проблема. Многие разработчики используют атрибуты class и id для создания расширенной семантической разметки. При этом практически неизменно разработчики используют специальные словари, которые они сами же составляют, а не значения, взятые из существующих схем. В лучшем случае это псевдо семантика.

Многие страницы в Интернете используют микроформаты, чтобы добавить больше структурированной семантики, чем имеющийся небольшой набор HTML-элементов и атрибутов. В этом случае, значения, используемые для атрибута class, устанавливаются из согласованных словарей, иногда взятых из других стандартов, таких как hCard, а иногда из новых словарей, где нет твердого стандарта (как в hReview).

На данный момент самыми распространёнными микроформатами являются:

* hAtom — формат рассылок новостей;
* hCalendar — составление календаря и описание событий;
* hCard — описание людей, компаний, мест;
* hResume — формат описания резюме;
* hReview — внедрение обзоров;
* XFN — способ указания отношений между людьми;

Даже современный формат HTML5, ставящий своей целью расширить инструментарий для работы с семантикой не смог добиться каких-либо значимых успехов. [6]Так как предложил лишь небольшой набор дополнительных элементов для семантической разметки, что в свою очередь вызвало проблемы при обратной совместимости со старыми браузерами.

Большая часть информации в сети интернет представляет собой данные в виде текста на естественном языке. В связи с этим чтобы системы обработки информации, основанные на методах интеллектуального анализа данных, смогли извлечь требуемую информацию необходимо вначале извлечь полезные данные из текста, и записать их в базу данных для последующего анализа.

Основным направление занимающимся анализом неструктурированной информации является анализ текстов. Однако учитывая специфику данных сети интернет, выделяют отдельное направление – интеллектуальный анализ веб-сайтов. Основное отличие данного подхода в том, что большинство данных на естественном языке расположенных на веб-сайтах имеют некоторую степень структурированности в рамках HTMLмодели всего сайта.

### 1.2.2 Гетерогенность сети интернет и семантическая неоднородность веб-страниц

Веб-страницы, записи архивов FTP, базы данных различных форматов, видеоданные, информационные массивы данных межмашинного взаимодействия, электронная переписка, электронная коммерция, данные социальных сетей. Все это довольно разнородная информация, которая представлена в виде различных, мало согласованных друг с другом форматов данных: тексты, графическая и аудиоинформация и вообще все, что имеется в указанных хранилищах. Возникает задача - разработать информационно-поисковые системы, которые со всем этим могут работать.

Также существует большая (как лингвистическая, так и программная) проблема - многоязычие информационного пространства интернета. Примерно 50% информации в сети представлено на английском языке, вторая половина - на всех остальных, количество которых увеличивается по мере распространения сетевых технологий [7]**.**Эта проблема касается и обработки документов, и составления и обработки запросов, и собственно работы модулей поиска и выдачи информации.

Как правило, большинство поисковых систем ориентированы либо на английский язык как на мировой язык сети интернет, либо на язык той страны, на которую нацелена данная поисковая система.

Также говоря, о гетерогенности самого интернета нельзя не упомянуть о такой проблеме как гетерогенность содержимого самих веб-страниц. Долгое время считалось, что каждая веб страница представляет собой единичный семантический объект. Однако реалии современного интернета опровергают данное предположение. Обычно на веб-странице, кроме непосредственно тематического текста и данных, напрямую связанных с темой данной страницы, существует также некоторое количество информационных сущностей (блоков), семантически слабо связанных с главной темой веб-страницы.

Примером таких блоков, к примеру, является блок навигации по веб-странице. Или блок содержащий рекламу или же набор ссылок на похожие веб-страницы. Возможно множество вариантов их расположения и различное содержание побочных блоков.

В итоге это ведет к “зашумлению” основного контента веб-страницы информацией имеющей мало семантической связанности с главной темой веб-страницы. В итоге при анализе содержания веб-страницы с помощью алгоритмов семантического анализа получаются искаженные результаты, то в свою очередь может отразиться на дальнейшей позиции страницы в поисковой выдаче.

Очевидно, необходимо каким-либо образом разделять веб-страницу на семантические объекты, оценивать их информационную ценность для данной веб-страницы и производить анализ только тех семантических блоков, которые релеванты, как основной теме веб-странице, так и тому поисковому запросу, по которому поисковая система анализирует данный документ.

Основной проблемой при решении данной задачи является неопределенность в том, какими критериями необходимо оперировать и как вычислять, к примеру, меры “похожести” двух семантических блоков. Большинство алгоритмов, разработанных в данной области являются эвристическими, в виду отсутствия строгих формальных моделей.

Однако данные алгоритмы могут обеспечивать достаточных уровень качества при анализе групп однородных по структуре сайтов, что позволяет применять данные методы для построения поисковых систем с семантическим анализом документов.

### 1.2.3 Надежность алгоритмов интеллектуального анализа данных при обработке содержимого веб-сайтов

Как уже было отмечено, веб-страница представляет собой объект со смешанной структурой: основной каркас страницы основан на модели DOM, которая является интерфейсом для взаимодействия с языком разметки HTML. Изначально язык разметки должен был применятся для структурирования и форматирования документов без привязки к средствам воспроизведения. Но увеличивающиеся потребности в мультимедийном и графическом оформлении привели к тому что основной задачей языка разметки стало именно графическое оформление страницы.

Внутри элементов HTMLразметки может содержатся текст на естественном языке. Причем сама разметка не содержит никаких данных о находящемся внутри нее тексте. В итоге веб-страницу можно охарактеризовать как слабоструктурированный объект. Несмотря на это, много раз предпринимались попытки использовать для анализа веб-страниц алгоритмы, основанные исключительно на анализе DOMмодели. Но чем более визуально изощренными становились веб-страницы со временем, тем менее эффективными становились данные методы.

Поэтому для анализа веб-страниц необходимо использовать алгоритмы для анализа естественных текстов. Данные алгоритмы намного более ресурсоемкие, и надежность извлечения правильных данных на данный момент не достигает высоких значений, но данное направление непрерывно совершенствуется, и вместе с ростом вычислительных мощностей персональных компьютеров растут и эффективность, и надежность данных алгоритмов. Однако при применении алгоритмов Textmining в области анализа веб-страниц возникают специфические трудности, которые ощутимо влияют на надежность данных алгоритмов.

Как уже было сказано, веб-страница содержит текст на естественном языке внутри элементов языка разметки. Однако группировка текста происходит не по семантическому признаку, а для улучшения визуального восприятия веб-страницы человеком. Именно поэтому текст внутри одного узла может быть частью большего текста и не иметь самостоятельной семантики. А текст полученный путем объединения содержимого нескольких элементов, находящихся на одном уровне DOMдерева, может быть сильно неоднородным, и не иметь целостной семантики.

Для примера рассмотрим часть DOMдерева сайта:

<div id=”main”>

<div class=”title”>Недвижимость</div>

<div class=”content”>...</div>

<div class=”title”>Страхование</div>

<divclass=”content”>...</div>

</div>

Рисунок 1.2. Часть структуры веб-страницы на языке HTML

Если анализировать структуру разметки, представленную на рисунке 1.2, то четыре внутренних элемента принадлежат одному уровню иерархии:



Рисунок 1.3. Часть дерева документа построенного на основе анализа DOM.

Но даже беглый анализ содержимого данных блоков позволяет убедиться в том, что их содержимое совсем не однородно. Часть блоков являются заголовками, а часть содержат информацию, относящуюся к данному заголовку. В случае если бы это был текстовый документ сами заголовки бы выделялись с помощью различных текстовых методов (Новая строка, заглавные буквы, увеличенный размер шрифта). В данном случае выделение происходит за счет каскадных стилей, которые применяются для визуального оформления веб-страницы. И проанализировать их на основе htmlразметки невозможно. Это существенно повлияет на надежность современных алгоритмов textmining, так как анализ заголовков один из эффективных методов установления основной темы блока текста.

Рассмотрим, как выглядит структура данного документа, если добавить в нее элементы семантического (или визуального) обобщения:



Рисунок 1.4 Часть дерева документа, построенного с учетом семантического анализа.

В данном варианте каждый рассматриваемый блок текста имеет отдельный заголовок, четкие границы и обладает значительной большей семантической однородностью. Используя определенные признаки наличия искомой информации в рассматриваемом блоке, можно добиться значительного сокращения объема данных которые будут обрабатываться основной частью алгоритма анализа. За счет этого можно использовать более медлительные, но также более надежные методы интеллектуального анализа.

Именно проблема перехода от представления структуры документа, заданного в HTMLразметки (рисунок 1.3), к структуре содержащей данные о внутренней семантике данных находящихся на веб-странице (рисунок 1.4) является ключевой в интеллектуальном анализе интернет ресурсов. Ее решение позволило бы применять методы анализа структурных данных для отдельных семантических блоков страницы, а для содержимого блоков использовать методы анализа естественных текстов. Сейчас же, в виду того что большинство алгоритмов семантической сегментации веб-страницы имеют малую надежность, итоговые данные чаще всего остаются смешанные по своей структуре. Что сказывается на эффективности дальнейшего анализа.

В настоящее время предпринимаются попытки создания алгоритмов семантического разбиения веб страниц, на основе различных подходов, и каждый их них обладает своими преимуществами и недостатками. Наиболее распространенные и эффективные подходы к семантической сегментации веб-страниц будут проанализированы в следующем разделе.

## 1.3 Использование поисковых систем как первого этапа в процессе поиска информации

Как уже было отмечено, наиболее развитыми системами интеллектуального информационного поиска являются глобальные поисковые системы крупных компаний – Google, Yandex, Bring. На создание и разработку данных систем потрачены огромные трудовые и финансовые ресурсы. Именно поэтому при разработке систем, решающих задачи, связанные с интеллектуальным анализом и поиском информации в сети интернет крайне эффективным шагом является использование в процессе анализа результатов, предоставленных в поисковой выдаче поисковых систем. Все крупные поисковые системы предоставляют внешние APIдля использования их в автоматическом режиме. Для исследовательских целей достаточно того количества запросов с одного IPадреса которые предлагаются для бесплатного пользования. Если же требуется более интенсивный опрос поисковой системы, то возможно заключить договор на необходимое количество запросов.

В списке проблем современного информационного поиска упоминалась такая проблема, как разрастание поисковой выборки. В итоге возникает необходимость в инструменте, который выполняет автоматический анализ данных из поисковой выборки.

По своему существу поисковая выборка представляет список веб-страниц, на которых находятся релевантные поисковому запросу данные. Как правило, каждый поисковый запрос связан с какой-то сущностью, информацию о которой ищет пользователь. К примеру, запрос “Как проехать от кинотеатра к автовокзалу” связан с сущностью типа “путь от кинотеатра к автовокзалу”. Существует множество сущностей данного типа, но пользователя, как правило, интересует та, которая имеет наименьшее значение атрибута “расстояние”. Или запрос “рецепт домашнего пирога из свеклы” который связан с сущностью типа “рецепт домашнего пирога” одним из атрибутов которого является “состав” включающий в себя “свекла”.

В большинстве случаев задачей при анализе поисковой выборки является нахождение всех значений атрибутов всех сущностей, связанных с искомым типом, и на основании анализа данных значений выбор наиболее оптимальных для конкретного случая сущностей.

Частным случаем данной задачи является анализ множества веб-сайтов. Но именно получение списка веб-страниц с релевантной информацией часто является весьма нетривиальной задачей. И наиболее эффективным решением часто является использование результатов, выданных поисковыми системами, с учетом рейтинга релевантности (позиции в поисковой выборке).

Именно поэтому в данной работе было решено использовать поисковые системы как первый этап в локализации области поиска необходимой информации. В данной работе уже упоминались наиболее популярные поисковые систему, но также стоит учитывать и узкоспециализированные поисковые системы, предназначенные для поиска в определенной области знаний, а также мета-поисковые системы, реализующие объединение нескольких поисковых выборок с различных систем. Чем точнее будет поисковая выборка, тем меньше шумовой информации придется обработать системе, именно поэтому выборка поисковой системы также является важной задачей.

## 1.4 Выводы

В данномразделе была дана оценка текущей ситуации в сфере информационного поиска в глобальной сети интернет. Исходя из тех тенденций, которые можно наблюдать в современном информационном поиске можно предположить, что в дальнейшем развитие информационного поиска в интернете будет развиваться одним из двух возможных способов:

1. Различные методы интеллектуального анализа данных будут все интенсивнее использоваться в глобальных информационно-поисковых системах, что в итоге позволит выполнять эффективный интеллектуальный поиск на той структуре интернета и веб-страниц которые существуют на сегодняшнее время.
2. С целью увеличения эффективности информационного поиска в сети интернет будет изменяться сама структура интернета и веб-страниц. Что позволит использовать механизмы интеллектуального поиска, опираясь на какую-то дополнительную структуру (например, в случае семантической сети, данной структурой могут быть метатеги или семантические атрибуты).

Современное состояние интернет поиска является граничным состоянием между методами классического поиска и интеллектуальными поисковыми системами. Современные разработчики систем информационного поиска в сети интернет неизбежно сталкиваются с огромными трудностями, вызванными тем, что область интеллектуального анализа больших объемов слабоструктурированных данных является мало изученной на текущее время.

Также в данномразделерассматривается проблема влияния смешанной структуры данных веб-страницы на надежность методов интеллектуального анализа. Для решения этой проблемы в данной работе предлагается использовать предварительную обработку данных веб-страницы с помощью алгоритмов семантической сегментации. Это позволит получить семантически структурированное дерево документа, и увеличит семантическую однородность отдельных его элементов. Что в итоге должно увеличить надежность алгоритмов интеллектуального анализа.

На основании проведенного анализа вдиссертации будет предложен проект системы, выполняющей извлечение знаний из поисковых систем, с выполнением предварительной сегментации содержимого веб-страниц. В системе будут использоваться разработанные на данный момент различные комбинации моделей и алгоритмов интеллектуального поиска с целью эффективного извлечения информации из интернета.

# 2 Анализ моделей и методов извлечения знаний

Часто способы извлечения данных разделяют условно на три группы, в зависимости от степени автоматизации процесса.

1. Ручной. Выполняется полностью человеком оператором. Раньше, когда вычислительные мощности компьютеров были недостаточными для реализации, каких-либо методов обработки неструктурированных данных, решением этой задачи занимались люди.
2. Полуавтоматический. К этому типу относятся решения, которые, после некоторой настройки пользователем, способны автоматизировать процесс извлечения данных. К данному типу относятся программы для автоматического извлечения данных с конкретных веб-сайтов – парсеры (от англ. Parse- разбор). Основным недостатком таких программ является зависимость от структуры целевых веб-сайтов и сложность адаптации программы под новые источники данных (чтобы подключить новый веб-сайт в виде источника данных, необходимо программно задать методы обхода всей структуры сайта, с помощью визуального интерфейса или набора регулярных выражений). Если рассматривать коммерческие продукты, то наиболее популярными и универсальными являются VisualWebRipper иMozenda, которые позволяют пользователю, затратив небольшие усилия, настроить программу-парсер для разбора конкретного веб-сайта.
3. Автоматический (интеллектуальный). Основным отличием подобных систем является автоматическое определение структуры разбираемого сайта, а также нахождение сущностей в естественных текстах. В настоящее время не существует универсальных систем, которые бы справились с анализом сайта любой структуры и выполняли бы поиск по сущностям любой тематики. Наиболее близкими к подобным системам является поисковые механизмы таких компаний как Google иMicrosoft.

Наибольший интерес на сегодняшний день представляют системы, находящиеся на границе между 2 и 3 типом. А именно системы-конструкторы, используя которые пользователь может создать необходимый ему инструмент для извлечения данных, подходящий под рассматриваемую задачу. Кпримеру,встатье [8], ееавторамиописываетсяспециальнуюбиблиотеку программных инструментов для быстрого создания инструментов извлечения знаний. Однако в данном случае пользователь должен обладать минимальными знаниями в области программирования для использования разрабатываемого инструмента.

## 2.1 Методы интеллектуального извлечения данных

Как упоминается в статье [9], для выполнения извлечения данных с веб-страницы с учетом ее структуры необходимо решит следующие пять задач:

1. Решение проблемы по поиску и получению целевых страниц (проблема навигации),
2. Решение проблемы по распознаванию участков с целевыми данными (проблема распознавания данных),
3. Решение проблемы по определению структуры найденных данных (проблема поиска общей структуры данных),
4. Решение проблемы по обеспечению однородности извлекаемых данных (проблема определения извлекаемых атрибутов)
5. Решение проблемы по объединению данных с различных источников (проблема объединения данных).

В настоящее время эти задачи решаются множеством различных методов, в зависимости от того, какой требуется результат. Далее рассмотрим каждую задачу в отдельности и наиболее распространенные методы их решения.

**Проблема навигации (Navigationproblem)**

Для того чтобы приступить к процессу извлечения знаний, необходим доступ к начальным данным. Интернет это распределенный ресурс, не предоставляющий единого механизма для получения доступа ко всем ресурсам в нем расположенным. Эта задача решается поисковыми системами с помощью программ-краулеров (от англ. crawler- паук). Программы выполняют обход веб-сайтов в глобальной сети и заносят некоторые данные о посещенных веб-сайтах в базу поисковой системы. Для обхода всех страниц веб-сайта эти программы анализируют структуру сайта на предмет наличия ссылок на страницы текущего ресурса.

Такие программы, как правило, имеют ограничение на максимальную глубину, на которую система анализирует веб-сайт, а также защиту от зацикливания для избегания циклического перехода по взаимосвязанным веб-страницам.

Как правило, поисковые базы собираются глобальными поисковыми системами на протяжении длительного времени, поэтому в задачи программ-краулеров созданных для систем извлечении данных, чаще всего входит лишь анализ ссылок внутри отдельного веб-ресурса. Однако при реализации подобного механизма возникает множество трудностей – необходимость исполнения JavaScript, динамическая загрузка данных с помощью технологии AJAX, ограничения по количеству или частоте запросов с одного IPадреса, сохранения cookies, необходимость авторизации, наличие капчи, отправка форм, анализ Flash данных, определение логического дублирования страниц.

В виду вышесказанного сложность данного компонента системы зависит от того какие требования предъявляются к результату. Если достаточно простого получения HTML контента со всех страниц сайта, данный механизм будет весьма прост в реализации. Если же анализируемые сайты имеют множество web 2.0 функций, то реализация данного механизма является весьма нетривиальной задачей.

В разрабатываемой системе будет использоваться широко распространенный механизм рекурсивного обхода веб-сайта по ссылкам, с обнаружением петель. При сохранении результатов, страницы, различающиеся лишь параметрами в адресной строке, будут считаться одной и той же страницей.

**Проблема распознавания данных (Dataextractionproblem)**

Анализируя содержание большинства веб страниц крупных веб-порталов, можно заметить, что помимо полезной информации, непосредственно связанной с темой данной страницы, на ней также представлена своего рода «шумовая» информация слабо связанная или вообще не связанная с темой. Это могут быть рекламные блоки, блоки для навигации по сайту, блоки с информацией по всем веб-ресурсу дублируемые на каждой его странице.

Чаще всего в современных системах автоматического извлечения данных эта задача сводится к задаче поиска повторяющихся структур данных. Причем повторение структуры не обязательно должно происходить в рамках одной веб-страницы или даже в рамках одного веб-сайта. Алгоритмы по нахождению регулярных структур данных это огромная тема, которую невозможно охватить в рамках одной работы. На данный момент существует несколько различных подходов к нахождению повторяющихся структур данных:

* Подход основанный на семантичностиhtml разметки (с учетом семантических атрибутов из различных стандартов, таких как schema.orgили hCard)
* Подход основанный на определении позиции в DOM дереве
* Подход основанный на визуальном анализе.

В данной работе предложен собственный алгоритм разбиения содержимого сайта на семантические блоки с помощью анализа структуры DOMдерева, который описывается далее. Данный алгоритм с достаточной эффективностью позволяет сегментировать веб страницы, имеющие простую DOMструктуру, однако в случае «плоского» дерева, происходит чрезмерная сегментация.

Однако DOM структура документа не всегда полностью отражает визуальную модель сайта. Но именно с визуальными данными и взаимодействует человек в ходе работы с веб-сайтом. Следовательно, в виду невозможности однозначного отображения DOMструктуры в визуальный образ сайта и обратно, алгоритмы, основанные исключительно анализе HTML, не всегда будут корректно определять структуру данных.

Для решения данной задачи компанией Microsoftбыл разработан алгоритм VIPS[10], для выполнения сегментации страниц веб-сайтов на визуальные блоки.

Как правило, чем сильнее в алгоритме делается упор на анализ визуальной составляющей, тем более ресурсоемким становится его программная реализация, но тем более точными становятся его результаты.

**Проблема поиска общей структуры данных (Structuresynthesisproblem)**

Чаще всего при извлечении данных, собирается информация об одном типе сущностей, к примеру, персональная информация о человеке, данные о фирме, о географическом объекте и т.д. Именно на основе данных об этом абстрактном образе, который описывает структуру всех реальных объектов, которые к нему относятся, и выполняется разбор атрибутов. В случае если же тип сущности заранее не известен, данная задача становится крайне сложной, так как в сети интернет, в частности на отдельном взятом сайте, не гарантируется полнота данных об извлекаемой сущности. Именно поэтому поиск общей структуры представляет собой нетривиальную задачу, требующую анализ большого количества данных, причем универсального решения на сегодняшний день не существует.

В данной работе мы будем рассматривать случай, когда система имеет данные о структуре типа анализируемых сущностей, что существенно упрощает данную задачу.

**Проблема сопоставления извлекаемых данных (Datamappingproblem)**

После выполнения предыдущего шага по извлечению данных, система имеет некоторый набор полезных данных, и сведения о структуре, в которую необходимо эти данные поместить (так называемая «сетка атрибутов»). Эту задачу можно разделить на две – задача по определению атрибута, и задача по нормализации значений атрибутов.

В ходе решения первой задачи системе необходимо принять решение о соответствии некоторых полученных данных семантике атрибута рассматриваемой сущности. К примеру, имея в данных слово «Microsoft», система должна соотнести его с атрибутом «имя компании». Как правило это выполняется на основании соответствия некоторых заданных признаков, которые указывают на каждый атрибут сущности. К примеру, в случае с именем компании одним из таких признаков является написание слова с заглавной буквы в любой части предложения. Еще одним признаком можно рассматривать как присутствие данного слова в заголовках верхних уровней на странице, а также в заголовке главной страницы веб-сайта.

Далее необходима нормализация полученных значений атрибутов. Этот шаг является крайне важным для осуществления последующего анализа извлеченных данных. Чем более точно будет выполнена нормализация атрибутов, тем эффективнее будет последующий анализ данных.

В случае с атрибутами, представленными строковыми значениями, нормализация, как правило, ограничивается сохранением определенного формата регистра букв, а также исключением определенных символов или стоп-слов. Однако если тип атрибута представляет собой число или дату, нормализация становится весьма нетривиальной. Числовые данные могут быть записаны в текстовом виде, что требует определенного лексического анализа для их преобразования в цифровую форму. Форматов записи даты существует огромное множество, поэтому система должна приводить все разобранные даты к одному формату. Очень часто нормализация основывается на семантическом значении атрибута – например, если атрибут представляет собой обозначение расстояния без конкретного указания единиц измерения, то этот параметр должен быть определен в зависимости от целевой аудитории данного сайта, а также стандартных границах этого параметра для объектов подобного класса (К примеру, если извлекаемой сущностью является расстояние от одного города до другого, а сайт ориентирован на русскоговорящую аудиторию то следует предположить, что значение атрибута указано в километрах).

Каких-либо универсальных алгоритмов по нормализации значений атрибутов не существует, поэтому система должна реализовывать нормализацию тех типов атрибутов, с которыми она будет непосредственно взаимодействовать.

В разрабатываемой системе будут присутствовать следующие типы атрибутов:

1. Строкове значение
2. Целочисленное значение
3. Вещественное значение

Нормализация данных типов уже реализована в большинстве языков программирования. Если атрибут представляет собой более сложный тип, то за его нормализацию будет отвечать модуль извлечения.

Одними из таких атрибутов являются терминальные атрибуты фрейма «физический адрес». Будут использоваться следующие правила нормализации:

Для названия города:

1. Начинается с заглавной буквы.
2. Присутствует в базе данных с названиями городов мира, сравнение происходит с расчетом метрики Дамерау-Левенштейна, граница для признания соответствия равна 1.
3. Если начинается с «г.» или «город», то эти слова удаляются.

Для названия улиц:

1. Начинается заглавной буквы.
2. Если содержит слова «ул.», «улица», «проспект» и др. то они сохраняются.
3. Может содержать несколько цифр за которыми следует окончание (пример «1ая Поселковая», «2ой Литейный»)

**Проблема объединения данных (Dataintegrationproblem)**

Как уже было отмечено выше данные на отдельно взятой веб-странице или веб-сайте, не могут гарантировать полноты данных об анализируемой сущности. Именно поэтому крайне важно решение задачи объединения данных, в ходе которой система определяет, что две извлеченные ею сущности в реальности представляют собой один и тот же объект, и выполняет слияние массивов атрибутов этих сущностей в один.

Тут сразу возникает две проблемы:

1. Как системе определить что сущность , извлеченная из сайта , и сущность , извлеченная из сайта - являются одним и тем же объектом?
2. Как реагировать если при слиянии массива атрибутов обеих сущностей, будут обнаружены одинаковые атрибуты, но имеющие разные значения?

Решение каждой из данных проблем является весьма нетривиальной задачей, и очень часто применяемый метод решения зависит от специфических особенностей частной задачи.

Наиболее популярным способом определения того что извлеченные сущности имеют общее происхождение является определение равенства наиболее значимых свойств данных объектов. К примеру, при извлечении данных о компаниях, решение о слиянии двух сущностей следует принимать, в случае если совпадают названия фирмы и регион регистрации.

Вторая проблема обычно решается на основе введения ранга источников, данных – атрибут значение, которого получено из более достоверного источника сохраняет свое значение. Также возможно определение более полного значения атрибута, но данный метод требует, чтобы значения атрибута можно было как-то сравнивать друг с другом в плане их полноты.

В данной работе сущности будут представлены фреймами-образцами, а значения их атрибутов значениями терминальных слотов фреймов. Алгоритм сравнения двух фреймов-образцов будет представлен далее.

### 2.1.1 Задача системы извлечения данных

На основании рассмотренных задач, можно сформулировать задачу, которую должна решать система по извлечению данных. Однако перед этим определим основные моменты системы, которая будет разработана в рамках данной работы:

1. Для решения задачи навигации в глобальной сети интернет, разрабатываемая система будет использовать интерфейс взаимодействия с глобальными поисковыми системами.
2. Общая структура извлекаемых данных будет задаваться пользователем, в виде иерархического фрейма, имеющего набор терминальных атрибутов и их типов.
3. С каждой сущностью ассоциирован непустой массив поисковых запросов.

С учетом данных ключевых моментов запишем задачу системы извлечения данных:

Обозначим множество извлекаемых сущностей , каждая сущность представляет собой кортеж , где множество базовых поисковых запросов для данной сущности, множество атрибутов сущности (терминальных слотов фрейма класса) размером . Каждый атрибут являетсякортежем уникальное имя атрибута, тип атрибута, значение атрибута. Пусть при принятии поискового запроса поисковая система выдает множество сайтов в поисковой выдаче, релевантных запросу ,число которых может быть ограничено сверху предельным значением количества ссылок которые система обрабатывает с одной поисковой выдачи. Источником данных для каждой сущности будем называтьобъединение множеств .

Решением задачи системы интеллектуального извлечения знаний для обработки сущности является множество значений атрибутов ,полученное на основании интеллектуального анализа системой источника данных , с последующей обработкой полученных значений атрибутов с помощью алгоритмов интеллектуального анализа и визуализации результатов.

### 2.1.2 Разработка метода разбиения содержания веб страниц на семантические блоки, основанного на анализе объектной структуры документа

Главная идея алгоритма заключается в предположении, что семантически схожие элементы веб-страницы чаще всего имеют общего предка в DOM структуре. Из работ [11] и [12] можно сделать вывод что основным фактором при принятии решении о разбиении элемента на дочерние, является параметр «вес». В данных работах предложены различные формулы для вычисления веса каждого элемента. В разрабатываемом алгоритме мы будем использовать следующие свойства блока при расчете его веса:

*l –* Длина текстового содержимого;

*s –*Размер элемента;

*d –*Глубина данного элемента в дереве;

– Длина текстового содержимого родителя данного элемента;

– количество текстовой информации в служебных элементах (таких как input, a, textarea);

Где *–*настраиваемыекоэффициенты;

Если элемент не может содержать в себе текст (к примеру тег IMG), то его параметр *l*рассчитывается как*.*Где – средняя площадь 1 символа текста (Определяется в зависимости от Font’а тега BODY).

На рисунке 2.1 представлена схема данного алгоритма. В качестве порогового значения для количества семантических блоков был выбран предложенный в [12] , где число элементов на самом нижнем уровнеDOM. Т.е. квадратный корень из числа элементов не имеющих дочерний.



Рисунок 2.1. Алгоритм разбиения содержимого страницы на блоки с анализом структуры DOM.

При подсчете внутренних параметров элемента не учитываются повторяющиеся пробелы и escape-последовательности. Также со страницы удаляются вместе со всем своим содержимым такие теги как <STYLE>и <SCRIPT>.

Также данный алгоритм осуществляет поиск заголовка для каждого семантического блока (это позволяет ускорить поиск термов на веб странице если известно, что они должны быть заголовками некоторых блоков)[13].

Заголовок ищется по следующим правилам, основанным на предположениях из [11]:

1. В первую очередь анализируются заголовочные элементы (<H1>—<H6>)
2. Если заголовочных элементов не обнаружено, то анализируются элементы с максимальным шрифтом
3. Если все элементы имеют максимальный шрифт, то анализируются элементы с выделенным шрифтом (boldи italic)
4. Если заголовок блока не был найден, то заголовком служит первое предложение в контенте блока (или первые 30 символов если в тексте отсутствуют точки).

### 2.1.3Анализ алгоритма сегментации веб-страниц, основанного на «компьютерном зрении»

VIPS (VisionbasedPageSegmentationAlgorithm) - Алгоритм сегментации страниц основанный на «компьютерном зрении», разработанный компанией Microsoft в некоторой степени похож на предыдущий алгоритм, однако в нем намного больше внимания уделяется визуальному представлению веб-страницы. Здесь также анализируется структура DOM, однакорешениео разбиении документа на отдельные блоки определяется их визуальным отображением. В отличие от предыдущего, данный алгоритм способен более эффективно разбивать на блоки различные меню и галереи с изображениями. Также он позволяет сегментировать блоки содержащие преимущественно текст, но с различными стилями текста.

Механизмегоработы изображеннарисунке 2.1. Исходная веб-страница (Рисунок 2.2а) на основе правил сегментации разбивается на множество непересекающихся блоков, каждый из которых в свою очередь может быть также множество блоков более низкого уровня (Рисунок 2.2б). В итоге формируется дерево визуальных блоков (Риунок 2.2с).

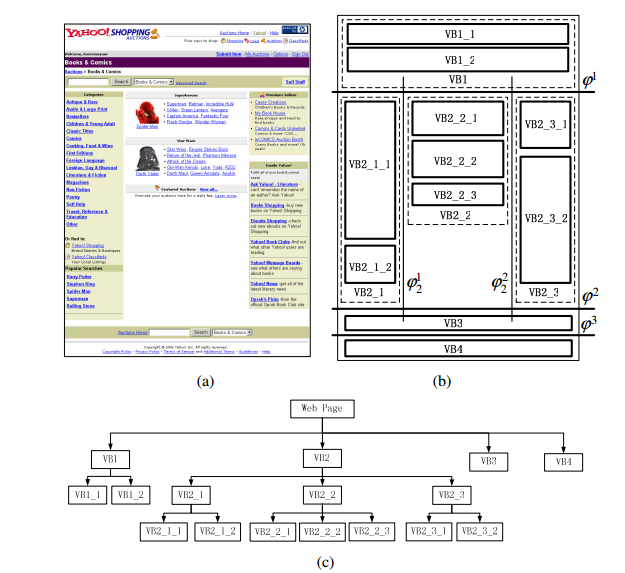


Рисунок 2.2 основные этапы алгоритма VIPS

Далее для каждого блока рассчитывается степень когерентности (DegreeofCoherence), которая определяет степень однородности визуальной информации в блоке.

Данный алгоритм является комбинацией рассматриваемых подходов, и его модификации используются в различных системах по извлечению данных с веб-сайтов.

### 2.1.4Разработка алгоритма извлечения знаний, основанного на шаблонах и их взаимном расположении

Многие свойства объектов представляют собой наборы значений свойств более низкого уровня. Примерам подобной структуры является физический адрес или технические характеристики объекта. В большинстве случаев данная информация располагается либо в табличном виде, либо представляет собой текст на естественном языке, но имеющий определенные границы и порядок.

В случае табличного представления извлечение данной информации основывается на анализе структуры таблицы:

1. Анализируя содержание таблицы, на основании совпадения ключевых слов, принимается решение содержит ли данная таблица необходимые данные;
2. К каждой ячейки таблицы поочередно применяются все основные методы интеллектуального извлечения данных данного алгоритма;
3. Если ячейка содержит данные которые распознаны как заголовок строки или столбца, то анализируются соседние (одна ячейка снизу и одна справа) сохраняя информацию о данных в первой ячейке.
4. Если соседняя ячейке распознана как содержащая данные относящиеся к исходной (титульной) ячейки, то анализируется весь ряд (или вся строка).
5. Если в таблице отсутствуют ячейки с заголовками, то выполняется поочерёдный анализ всех ячеек.

Основой данного подхода является предположение о том, что каждая ячейка таблицы содержит минимально неделимое значение свойства объекта.

Табличный способ представления информации встречается довольно часто, однако игнорируя иные способы представления комплексных свойств объектов невозможно добиться высоких показателей эффективности алгоритмов извлечения данных.

Вторым наиболее часты способом представления информации о комплексных свойствах объекта является запись на естественном языке, но имеющая определенные закономерности, зависящие исключительно от анализируемого свойства.

Для примера рассмотрим описание свойства физический адрес расположенное на одном из веб-сайтов «Наш адрес: г. Омск, ул. Энтузиастов, д. 2, корп. 1, ТЦ "О\'Кеу"». Можно выделить следующие признаки которые позволяют установить принадлежность данного текста к информации об адресе:

1. Вводные слова «Наш адрес:».
2. Наличие ключевых слов, используемых для обозначения адреса «ТЦ», «г.», «ул.», «корп.».
3. Взаимное расположение и близость данных ключевых и вводных слов.
4. Значение свойства «Омск», которое явно ассоциируется со свойством «город» и является частью данного комплексного свойства.

Для извлечения знаний из текстов на естественном языке, но имеющих слабую структурную зависимость от извлекаемого типа свойств в рамках данной работы был разработан алгоритм представленный на рисунке 2.3.

В данном алгоритме используется следующая гипотеза: все единичные значение свойств относящиеся к искомому комплексному свойству расположены в тексте так, что расстояние между двумя самыми крайними значениями составляет не более чем слов.

Данная граница является уникальной для каждого свойства, и зависит не только от количества слов в значениях единичных слов, но и от количества возможных незначащих слов. Наличие данной границы позволяет извлекать значения тех единичных свойств, значения которых трудноотличимы от всего остального текста, т.е. невозможно создать шаблон по поиску данных значений в большом тексте. Также это позволяет использовать методы «анализа окрестности», которые заключаются в том, что границы значения текущего единичного свойства определяется не на основе каких-либо знаков препинания или других способов выделения части текста, а с помощью анализа соседних слов от уже извлеченных значений и принятии решений о том может ли данной слово являться значение другого единичного свойства.

Чем меньше заданное максимальное расстояние , тем больше вероятность что анализ окрестности извлечет те значения для которых отсутствуют шаблоны поиска, однако это может привести к тому что случае если в записи присутствует большое количество незначащих слов, некоторые значения буду распознаны как не относящиеся к данному экземпляру комплексного свойства.

Оптимальное значение определяется индивидуально для каждого комплексного свойства и может корректироваться в зависимости от характеристик анализируемых текстов. Если известно, что знания будут извлекаться из страниц с краткими описаниями технических устройств, то разумнее будет взять небольшую границу, так как в техническом описании как правило многие параметры записаны простым перечислением. Если же требуется произвести анализ веб-страниц с текстами, написанными пользователями то данный параметр рекомендуется взять с запасом, так как возможно появление большого количества незначащих слов.



Рисунок 2.3 Алгоритм поиска комплексных свойств в тексте основанный на шаблонах единичных свойств и их близости.

### 2.1.5 Разработка алгоритма извлечения краткой описательной информации об объекте

Краткое описание объекта представляет собой небольшой объем текста на естественном языке. Автоматическое извлечение знаний из подобных описательных текстов весьма затруднительно в том случае если неизвестны приблизительные шаблоны по которым данное описание составлено.

В данной работе предлагается алгоритм по извлечению текстовых блоков содержащих описания объектов. В дальнейшем они могут использоваться для проведения ручного анализа.

Основные проблемы при извлечении описательных текстов представлены ниже.

1. Проблема в определении является ли анализируемый текст описательной информацией искомого объекта.
2. Проблемы нахождения границ данного текста.

Вторая проблема в данной работе решается с помощью применения алгоритмов сегментации. Примем предположение что если система приняла решение что в семантическом блоке содержится описательный текст, то границы этого текста равны границам данного семантического блока.

Основная идея предлагаемого алгоритма заключается в том, что наиболее полное описание какого-либо объекта находится в блоке с наибольшим текстовым наполнением и максимальным параметром значение которого рассчитывается на основании некоторых правил.

Выполнение следующих условий() увеличивает значение .

* 1. Если в полном пути к анализируемой веб-страницы содержится слово ‘about’ или анализируемая страница является первой страницей сайта (index.html без параметров).
  2. Если адрес веб-страниц представляет собой шаблон {постоянная часть} {название искомого объекта}.
  3. Если в анализируемом блоке присутствуют слова из массива наиболее часто встречающихся слов использующихся для описания искомых сущностей.
  4. Если фрагмент с наибольшим шрифтом в блоке содержит слова из массива наиболее часто встречающихся слов в заголовках описания искомых сущностей.

Следующие условия () понижают значение .

1. Если в анализируемом блоке присутствуют стоп-слова.
2. Если в анализируемом блоке количество HTMLразметки больше чем 30% от общего числа внутреннего текста.
3. Присутствие в анализируемом блоке активных HTMLэлементов (кроме ссылок), то есть тэги FORM, INPUT, BUTTON.

Массивы наиболее встречающихся слов, а также массив стоп-слов формируются отдельно для каждой области в которой производится извлечение данных.Полная формула по которой рассчитывается параметр для анализируемого блока представлена ниже.

Где – настраиваемые коэффициенты, значения которых зависят от особенностей описательных текстов в каждой конкретной области.

Алгоритм извлечения описательной информации об объекте состоит из следующих шагов.

1. Загружаем содержимое веб-страницы связанной с искомым объектом (адрес получен либо из поисковой выдачи, либо найден на уже разобранной веб-странице)
2. Для 3 блоков с наибольшим текстовым содержанием рассчитываем параметр и берем блок с максимальным значением.
3. Если сохраненный блок отсутствует, то сохраняем полученный блок и переходим к пункту 5.
4. Если найденного блока выше чем сохраненного блока то заменяем сохраненный блок на найденный.
5. Если список нерассмотренных веб-страниц не пуст, то переходим к пункту 1, иначе алгоритм завершается.

## 2.2 Анализ и выборметодов интеллектуального анализа данных

В процессе интеллектуального извлечения данных, знания, полученные из различных источников, сохраняются в базу данных. Однако возможен дальнейший анализ полученных данных с целью нахождения «скрытых» знаний. Данный процесс анализа носит название интеллектуальный анализ данных.

Целью интеллектуального анализа данных (англ. Datamining, другие варианты перевода - "добыча данных", "раскопка данных") является обнаружение неявных закономерностей в наборах данных

Исходными данными для анализа чаще всего является множество некоторых сущностей, каждая из которых обладает некоторым набором атрибутов, которые его характеризуют. В процессе анализа система пытается найти взаимосвязь между различными параметрами. Зависимый параметр может иметь уже известное значение в начальной выборке данных, или быть вычисляемым на основе других параметров.

Одними из основных задач интеллектуального анализа данных являются:

1. Задача классификации. Суть данной задачи состоит в том, чтобы разделить все множество анализируемых сущностей на некоторое количество классов. Само множество классов может быть либо известным заранее, либо находится в процессе кластеризации.
2. Задача кластеризации состоит в нахождении некоторого числа классов на множестве анализируемых сущностей.
3. Задача регрессии во многом схожа с задачей классификации, но в ходе ее решения производится поиск шаблонов для определения числового значения.
4. Задачей прогнозирования является нахождение новых значений на основании уже известных значениях, между которыми наблюдается корреляция.

В данной работе подсистема интеллектуального анализа данных будет изначально реализовывать метод интеллектуального анализа данных основанный на нейронной сети Кохонена. Данная нейронная сеть способна выполнять кластеризацию и классификацию полученных наборов данных.

В виду того, что существует огромное множество других моделей и методов интеллектуального анализа данных, которые позволяют увеличивать эффективность работы разрабатываемой системе интеллектуального поиска, в системе будет реализован механизм для легкой интеграции новых алгоритмов анализа. Причем возможна будет не только модификация процесса интеллектуальной обработки данных, но также и процесса извлечения данных.

### 2.2.1 Кластеризация и классификация на основе модели нейронной сети Кохонена.

Сеть Кохонена – это одна из разновидностей нейронных сетей, которые используют неконтролируемое обучение (или обучение “без учителя”). При таком обучении обучающее множество состоит лишь из значений входных переменных, в процессе обучения нет сравнивания выходов нейронов с эталонными значениями[14].

Структура сети Кохонена представлена на рисунке 2.4.

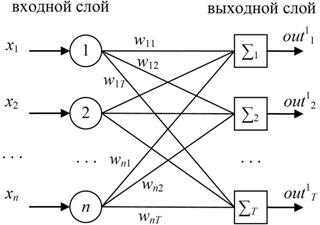


Рисунок 2.4. Структура сети Кохонена

Принцип работы уже обученной нейронной сети Кохонена следующий. Нейроны входного слоя (на рисунке 2.4 обозначены кружочками) не выполняют никаких вычислений и служат лишь точками разветвления. Каждый нейрон входного слоя соединенс каждым нейроном скрытого слоя, который называют слоем Кохонена, отдельным весом, так, например, первый нейрон входного слоя соединен с первым нейроном слоя Кохоненавесом . Нейроны слоя Кохонена складывают поступающие значения ивыдают их.

Как правило, на выходе нейронной сети еще устанавливают некоторый интерпретатор, который определяет номер нейрона с наибольшим выходным сигналом и выдает его номер. Или же делает выходной сигнал нейрона–победителя равным 1, а выходные сигналы всех остальных нейронов – 0.

В разрабатываемом модуле к системе интеллектуального анализа, реализуем вариант нейронной сети с интерпретатором, выдающим номер нейрона победителя.

Будем использовать стандартный метод обучения сети[15]:

Задан некоторый начальный набор параметров нейронов. Далее сети предъявляется один объект . Находим нейрон, выдавший максимальный сигнал. Пусть номер этого нейрона . Тогда параметры нейрона модифицируются по следующей формуле:

Где –коэффициент скорости обучения;

Коррекция весов записана в виде векторного выражения. Часто используют обучение, когда монотонно убывает.

Алгоритм выполняется до тех пор, пока веса не перестанут меняться или же достигнута требуемая точность:

Где – допустимая погрешность обучения;

В разрабатываемой системе данная модель нейронной сети будет представлена отдельным модулем, который имеет возможность встраиваться как в систему извлечения данных, так и в систему анализа. Эффективность данной модели в системе извлечения данных обуславливается тем, что данная нейронная сеть способна обрабатывать потоки данных, постепенно на них обучаясь. Это позволит использовать данную модель в алгоритмах извлечения данных, в зависимости от степени того насколько обучена сеть – пока сеть не добилась устойчивого положения всех ядер классов, она не будет выполнять кластеризацию и ее результаты не будут влиять на итоговый результат. Однако, когда достигнутая точность будет достигнута, система автоматически включит данную модель в процесс извлечения данных.

## Разработка модели представления знаний в системе

В разрабатываемой системе будет использоваться иерархическая фреймовая модель представления знаний. Главной структурной единицей модели является фрейм, представляющий собой формализованную модель извлекаемой информационной сущности. Описание структуры используемых фреймов задается пользователем с помощью создания фрейма-класса, являющегося шаблоном для описания сущностей.

### 2.3.1 Иерархическая фреймовая модель

В процессе извлечения система будет создавать фреймы-образцы используя заданный пользователем шаблон, и записывать извлеченные данные в значения их терминальных слотов.

В отличие от обычной фреймовой модели, иерархическая фреймовая модель обладает строгой иерархией и имеет лишь один фрейм верхнего уровня, т.е. представляет собой древовидную структуру.«Сетка атрибутов» объекта будет соответствовать множеству терминальных слотов фрейма (листам иерархии). На рисунке 2.5представлен пример фреймовой модели, используемой в системе.

При описании требуемого фрейма-класса пользователь способен создавать слоты указатели на фреймы, новые фреймы, а также терминальные слоты различных типов.

В данной модели знаний фрейм представляет собой тип данных вида:

Где *N* – имя объекта или подобъекта.

множество терминальных слотов, определяющих свойства фрейма (декларативная семантика)

множество слотов указателей на фрейм, связывающих данный фрейм с другими (семантическая связь)

множество процедур для извлечения, сохранения, и преобразования значений слотов данного фрейма.



Рисунок 2.5.Пример фреймовой модели для использования в программе.

В отличие от канонической фреймовой модели, в модели используемой в данной работе, некоторые процедуры привязываются не к отдельным слотам фрейма, а к всему фрейму целиком. Т.е. при обращении к фрейму в одном из режимов (режим извлечения данных\заполнения фрейма, режим сравнения двух фреймов-образцов, режим анализа содержимого фрейма), происходит запуск всех процедур, привязанных к фрейму. Данные процедуры выполняют предобработку данных, а также представляют связанность слотов в пределах фрейма.

В простейшем случае, процедуры привязываются к терминальным слотам для двух основных режимов – извлечения значения слота, и сравнения его значения с другим.

### 2.3.2 Разработка алгоритма заполнения значений терминальных слотов фрейма-образца

Процесс заполнения фрейма-образца основанный на извлечении данных с веб-ресурса выглядит следующим образом:

1. С каждым информационным объектом представленным фреймом верхнего уровня связано некоторое множество источников, данных (веб-страниц);
2. Заполнения начинается с фрейма верхнего уровня;
3. Выполняются все процедуры предобработки содержимого источника прикрепленные к фрейму;
4. Запускаются экстракторы всех терминальных слотов данного фрейма, и записывают извлеченные значения в слоты фрейма;
5. Если какой-либо экстрактор не смог извлечь нужные данные, то данный терминальный слот принимает значение undefined, и устанавливается один из служебных флагов указывающий на причину отсутствия данных:

NO\_INPUT\_DATA - экстрактор не поступили входные данные (это возможно в случае если включены различные методы предобработки данных, к примеру алгоритмы блочной сегментации)

INCORRECT\_DATA – данные полученные экстрактором не подходят для его работы (к данному типу относится также слишком маленький объем данных)

ERROR\_IN\_PROCESSING – во время работы экстрактора произошла ошибка, более детальное описание ошибки должно быть сохранено экстрактором в отдельный файл или запись в журнал

1. Если все фреймы обработаны, то алгоритм заполнения прекращается;
2. Для каждого из слотов указателей на фрейм данного фрейма выполняются пункты 3-7;

После заполнения значений всех слотов фрейма (даже если экстрактор не смог извлечь нужные данные слот получает значение undefined), все данные о фрейме-экземпляре заносятся в базу данных. В начальном варианте системы используется реляционная БД которая содержит все данные о значениях терминальных слотов отдельно от структуре фрейма. При необходимости восстановление фреймовой модели из БД, вначале создается структура фрейма по фрейму-классу, а следом заполняется данными значения всех терминальных слотов. Все прикрепленные к фреймам процедуры хранятся во фрейме-классе, а процедуры, прикрепленные к отдельным слотам в таблице фреймов-образцов.

### 2.3.3 Разработка алгоритма сравнения двух фреймов-образцов

Так как в данной системе для одной информационной сущности возможно установить множество источников данных, то, как уже было упомянуто выше, остро встает вопрос о нахождении объектов-дубликатов. Для этого в данную фреймовую модель введен коэффициент подобия двух фреймов-образцов.

Два фрейма-образца признаются равными, то есть описывающими один и тот же реальный объект, в том случае если их коэффициент подобия больше порогового значения, устанавливаемого заранее.Расчет численного значения коэффициента подобия двух фреймов вычисляется по следующей формуле:

(1)

Где число терминальных слотов фрейма;

степень идентичности значений *i*-го терминального слота;

коэффициент важности терминального *i*-го слота;

число слотов с указателями на фреймы;

коэффициент важности *g*-го фрейма-слота;

корневой фрейм

Степень идентичности значенийтерминального слота вычисляется в зависимости от типа данных в данном слоте. То есть для каждого типа данных задается своя метрика, позволяющая производить операции сравнения. Для числовых типов используется типовое расстояние следовательно степень идентичности будет равна:

Где предельное значение расстояния подобия.

Для строковых значений используется метрика Дамерау-Левенштейна:

Задаваемые пользователем при создании фрейма-класса, коэффициенты важности терминальных слотов и слота-указателя из формулы (1), используются для выделения наиболее важных частей дерева. Это позволяет задать наиболее важные для идентификации свойства объекта.

Если два фрейма-образца признаны идентичными, то в том случае если в одном из них присутствуют данные которых нет в другом (значения undefined), то эти данные копируются. В остальном, если обнаружено различие в значениях некоторых терминальных слотов, то они в зависимости от типа либо принимают усредненное значение, либо просто берется значение одного из фреймов. После чего один из фреймов удаляется.

## 2.4 Выводы

В данномразделе были рассмотрены основные задачи, которые ставятся перед системами интеллектуального извлечения и анализа данных. Проанализировав данные задачи и существующие на данный момент методы их решения, а также их эффективность и ресурсоемкость, следует сделать следующие выводы:

1. Ни один из существующих алгоритмов не может пока достичь точности результатов хоть в какой-то степени сравнимых с выполнением анализа человеком. А наиболее эффективные алгоритмы являются крайне ресурсоемкими и требующими огромных вычислительных мощностей. Именно поэтому при разработке подобных систем интеллектуального извлечения или анализа данных используются алгоритмы наиболее эффективные для частной задачи.
2. Надежность и эффективность многих алгоритмов на разных объемах и типах данных крайне трудно определить теоретически, без непосредственного моделирования на ЭВМ. Поэтому многие разработчики таких систем, вынуждены, опираясь лишь на свой профессиональный опыт, выбирать те модели и алгоритмы, которые они считают наиболее оптимальными для поставленных перед системой задач, и с помощью исследованияразличных моделей находить наиболее эффективную для данной задачи.

Сделан вывод, что для обеспечения максимальной универсальности системы, она должна иметь возможность легкого изменения тех алгоритмов и моделей интеллектуального извлечения и анализа которые в ней используются.

Для увеличения надежности методов извлечения знаний в данной работе используются алгоритмы блочной сегментации, такие как сегментация с анализом DOM и VIPS. Теоретически, применение данных алгоритмов позволит уменьшить количество лишней информации, поступающей на функции, выполняющие извлечение знаний.

Выбранная иерархическая фреймовая модель обеспечивает простоту создания информационного представления сущности, извлечение знаний о которой необходимо выполнить системе. В рамках данной модели используется зависимость извлечения значений терминальных слотов фреймов от положения данного фрейма в общей иерархии. Данный подход основывается на том что когда человек осуществляет поиск информации на веб-странице, он вначале пытается найти информационный блок содержащий информацию данного типа, а потом анализирует содержание этого блока.

Для увеличения надежности алгоритмов сравнения двух фреймов-образцов, в данную модель были добавлены коэффициенты важности, которые позволяют повысить влияние подобия наиболее важных свойств объекта и уменьшить влияние малозначащих свойств.

# 3 Проектирование и разработка системы автоматического извлечения знаний из поисковых систем

Разрабатываемая система автоматического извлечения знаний из поисковых систем должна выполнять следующие функции:

1. Возможность отправки запросов на различные поисковые системы, и получение результатов поисковой выборки.
2. Обход веб-сайтов адрес которых получен из поисковой выдачи или задан в БД. Получение всех внутренних ссылок с каждого сайта и обход сайта «в глубину», величина которой регулируется пользователем.
3. Применение к обрабатываемому содержимому веб-страницы определенного числа функции предварительной обработки и извлечения признаков.
4. Сохранение промежуточных результатов в БД.
5. Применение к полученному набору данных различных модулей интеллектуального анализа (выполнение кластеризации или классификации).
6. Визуализация полученных результатов.

Модульное строение системы, а также потоки данных между модулями системой и внешними источниками изображено на рисунке 3.1.

У данной системы имеются два внешних источника данных, это поисковые системы, для работы с которыми должны быть настроены соответствующие модули системы, а также множество веб-сайтов в сети интернет доступные с помощью URLполученных от поисковых систем.

Исходными данными для работы системы является настроенный пользователем фрейм-класс хранящий информацию о строении извлекаемых сущностей, а также множество поисковых фраз, каждая из которых ассоциируется с одной концептуальной сущностью.



Рисунок 3.1 Диаграмма модулей системы и потока данных в системе.

Основным моментом в проектировании системы является то, что по причине существования большого числа различных алгоритмов для выполнения интеллектуального извлечения и анализа данных, ни один из которых не является достаточно эффективным для решения всех возможных задач интеллектуального анализа, в настоящее время невозможно построить универсальную систему, подходящую под любые задачи.

Соответственно для обеспечения максимальной универсальности системы, следует при разработке системы обеспечить механизмы для возможности легкого и быстрого изменения механизмов интеллектуального анализа системы, путем добавления в процесс анализа различных алгоритмов и моделей. Это позволит пользователю в кратчайшие сроки и с минимальными затратами адаптировать систему под конкретную задачу.

Однако следует отметить, что если уделить универсальности слишком много внимания, то итоговый продукт будет требовать от пользователя слишком больших затрат при настройке системы. Примером такого случая могут служить Framework’и разработанные для конструирования систем решающих задач DataMining.

Следовательно, необходимо тщательно произвести анализ тех требований, которые предъявляются к системе, рассмотреть какие из них могут служить ограничениями при попытке унификации системы.

Также следует проанализировать наиболее ожидаемые условия функционирования системы, с целью разработки структуры системы, в которой изначально заложена идея оптимизации работы системы в ожидаемых режимах работы.

## 3.1 Анализ основных требований к системе в плане надёжности

Как уже было отмечено в предыдущих частях работы, современные методы интеллектуального извлечения и анализа данных еще не достигли тех уровней эффективности и надежности, при которых они могли бы конкурировать с ручным анализом выполняемым человеком. Однако если рассматривать поверхностный анализ, то здесь преимущество автоматических систем очевидно. Именно поэтому основной идеей данной системы автоматического извлечения знаний является достижение поставленной эффективности анализа на больших объема данных. Из этого следует, что временные параметры процесса анализа, а в первую очередь время выполнение извлечения знаний по всему объему требуемых запросов будут иметь достаточно большие значения.

В большинстве случаев надежность алгоритмов интеллектуального анализа пропорциональна времени исполнения данного алгоритма. Однако время получения знаний также является весьма важным фактором при разработке.

Для уточнения этих предположений, был разработан небольшой прототип системы, реализующий простейший алгоритм анализа, основанный на простейших регулярных выражениях. В данном прототипе была реализована лишь та часть функционала, которая относится к извлечению данных. Также в созданном прототипе присутствовал лишь один поток извлечения данных, не было никакого функционала распределения задач. В ходе проведения нескольких экспериментов были получены данные, отображенные в таблице 3.1.

Так как данный прототип реализовывал лишь извлечение данных, о его эффективности можно судить лишь по числу корректно извлеченных атрибутов. Как видно из таблицы 3.1, время обработки растет почти линейно, и несмотря на увеличивающуюся доли необработанных запросов (вызванные различными отказами в системе, от некорректности URL, 404 и 500 ошибок, до временного отключения доступа в интернет компьютера, на котором был развернут прототип), число извлеченных атрибутов также показывает почти линейный рост.

Таблица 3.1. Данные при тестировании прототипа системы

|  |  |  |  |
| --- | --- | --- | --- |
| Число обрабатываемых запросов, штук | Число успешно обработанных запросов, штук | Число извлеченных атрибутов, штук | Время процесса обработки, ч |
| 1000 | 954 | 3242 | 1 |
| 5000 | 4798 | 13410 | 6 |
| 50000 | 44112 | 120974 | 65 |

Однако важным обстоятельством является то, что в реализуемой системе помимо подсистемы извлечения данных, также будет реализована подсистема анализа полученных данных. Чья эффективность напрямую зависит от объема данных извлеченных в ходе работы первой подсистемы. Из этого следует, что чем более точные данные требуются получить пользователю системы в результате, тем больше должен быть массив входных данных, на основе которых система сформирует множество запросов, и соответственно тем больше будет время выполнения анализа.

Перед разрабатываемой системой уже поставлены некоторые задачи, на решение которых она должна быть ориентирована в первую очередь. Объемы запросов в данных задачах имеют значения от 100 000 до 1 000 000 запросов. Если принять во внимание оптимизацию разрабатываемой системы по сравнению с прототипом, а также с другой стороны усложнение алгоритмов анализа, то можно сделать предположение что время обработки данных итоговой системой не будет отличаться от времени обработки данных прототипом более чем на порядок (а в случае применения ресурсоемких алгоритмов анализа скорость работы итоговой системы может быть ниже скорости работы прототипа).

Исходя из этих предположений, можно сделать вывод, что для выполнения данных задач системе ориентировочно потребуется от 12 до 1 200 часов.

В виду того что система ориентировочно будет функционировать достаточно длительное время, одним из важных факторов, показывающих ее эффективность, будет надежность системы при продолжительном функционировании в режиме извлечения и анализа.

Можно выделить три основных момента влияющих на надежность разрабатываемого ПО:

1. Организация ПО и качество его разработки.
2. Входные данные.
3. Внешние воздействия на аппаратную часть.

Третий пункт не будет рассматриваться в данной работе, так как обеспечения надежности аппаратной составляющей системы, является отдельной темой в разработке надежных серверных систем.

Рассматривая первый момент, можно заметить, что при устранении дефектов, которые обнаружены в процессе опытной эксплуатации, а также с учетом того что новые дефекты не вносятся, то интенсивность отказов ПО уменьшается. Это означает что ПО является «молодеющей» системой.

Поскольку разрабатываемая система должна извлекать контент с любого сайта в сети интернет, то можно утверждать, что ограничения на входные данные крайней небольшие:

* Загрузка происходит по протоколу HTTP
* При получении заголовка протокола из множества 2хх, сигнализирующих об успешной передачи данных, содержимое страницы считается доступным
* Кодировка страницы определяется либо из заголовка HTTPзапроса“ Content-Type”, либо из мета-тега с указанием “charset”. Причем в случае если кодировка в заголовке отличается от кодировки в мета-теге, используется кодировка из заголовка. Если кодировка не указана, содержимое сайта признается некорректным и извлечение прекращается.

Если рассматривать отдельно подсистему извлечения данных, то можно прийти к выводу что при извлечении данных с веб-сайтов может произойти множество отказов вызванных некорректной передачей данных, прекращением функционирования веб-сайтов, а также другими внешними причинами. Основным моментом является то, что для обеспечения наибольшей надежности система должна иметь возможности восстанавливаться после данных сбоев, и продолжать функционировать без вмешательства оператора.

Если рассматривать возможности внутренних отказов в системе, то наиболее вероятным отказом (согласно испытаниям на прототипе) является некорректные значения атрибутов. Из-за подобных отказов многие алгоритмы анализа, реализуемые в системе, могут прекратить функционирования или выдавать ложные данные. Для минимизации вероятности возникновения подобных отказов следует применять соглашения для различных типов атрибутов, что не позволит системе записать в значение атрибута некорректные данные.

## 3.2 Разработка структуры системы

Структура разрабатываемой система представлена на рисунке 3.2 и состоит из четырех основных подсистем:

1. Подсистема извлечения данных;
2. Подсистема интеллектуального анализа;
3. Подсистема работы с БД;
4. Подсистема мониторинга;

Подсистемы извлечения данных и интеллектуального анализа могут работать независимо друг от друга, и выход из строя одной из подсистем не вызовет отказа другой.

Слабым местом в плане надежности является подсистема работы с базой данных. Если данная подсистема выйдет из строя, то работа всей системы будет невозможна. Для того чтобы в дальнейшем данный недостаток было возможно устранить было реализовано следующее:

Подсистема работы с БД реализована в виде внешнего унифицированного интерфейса работы с базой данных, который не зависит от использованного типа базы данных. Взаимодействие с БД на более низком уровне осуществляется за счет модулей-адаптеров. В исходном варианте системы реализован только один модуль для работы с СУБД MySQL. В случае если возникнет необходимость повышения надежности системы за счет введения информационной избыточности в БД или использование других СУБД, то необходимо создание и подключение новых модулей в подсистему работы с БД.

За выполнение контроля за состояние всей системы в процессе работы будет отвечать подсистема мониторинга. Данная подсистема включает в себя контроль за состоянием внутренних потоков обработки в системе, а также записи в журнал событий всех необходимых данных. Выбор использования журнала событий вместо системы с визуальным мониторингом был сделан на основе требований к эксплуатации системы – ввиду эксплуатации системы достаточно продолжительное время без непосредственного наблюдения оператора, журнал событий способен предоставить все необходимые данные о работе системы после ее завершения или отказа.



Рисунок 3.2 Структура системы извлечения знаний из поисковых систем

Помимо внутреннего мониторинга, также в системе будет реализован внешний процесс слежения за состояние процесса системы. Это будет реализовано за счет отдельной программы которая не имеет интерфейса и может запускаться в фоновом режиме. С помощью системных функций данная программа будет осуществлять периодический контроль за состоянием процесса основного приложения. Реакция данной программы на пребывание процесса в состояние «Не отвечает» или в случае полного прекращения работы процесса без сигнала об успешном завершении, будет попытка перезапуска системы с восстановлением предыдущей сессии. Если перезапуск прошел успешно, то внешняя программа мониторинга продолжает функционировать в штатном режиме. Если же возникли какие-то проблемы при перезапуске системы, то принимается заключение о критическом отказе системы, с соответствующей записью в журнал событий. После чего программа внешнего мониторинга прекращает свою работу.

### 3.2.1 Проектирование подсистемы извлечения данных с веб-сайтов

Функциональная схема данной подсистемы изображена на рисунке 3.3.Данная подсистема выполняет следующие основные функции:

1. Отправление поискового запроса на сервер поисковой системы, и получение ответа.
2. Анализ полученной поисковой выдачи и формирование списка источников для извлечения данных.
3. Последовательный обход по полученным источникам и получение содержимого необходимых веб-страниц.
4. Сегментация содержимого каждой веб-страницы выбранным пользователем алгоритмом сегментации.
5. Анализ полученных семантических блоков, с целью определения блоков для последующего извлечения атрибутов.
6. Извлечение атрибутов сущностей с помощью алгоритмов реализованных в отдельных модулях подсистемы извлечения атрибутов.

Получение исходных поисковых запросов происходит с помощью запросов к подсистеме управления БД и выборки их из специальной таблицы. Сохранение данных об обработанном поисковом запросе, во-первых, обеспечивает возможность кэширования, а во-вторых обеспечивает дополнительные данные об нахождении анализируемых источников в поисковой выдаче, что позволит в дальнейшем, в случае необходимости, произвести анализ данной информации.

В исходном варианте в модуль взаимодействия с поисковыми системами реализована поддержка лишь одной поисковой системы «Yandex». Однако ввиду наличия интерфейса обобщающего работы с данным модулем, в случае необходимости возможно добавление отдельного модуля для взаимодействия с другими поисковыми системами. Этот интерфейс инкапсулирует в себя функции для пересылки поисковых запросов на сервер поисковой системы и модификацию их с учетом служебных команд используемой поисковой системы. А также разбор данных поисковой выдачи.

Работа данной подсистемы связана с получением данных с различных интернет источников. Как правило, беспрерывная работа этих источников не гарантируется, они могут отключаться на технические работы, быть недоступны вследствие неполадок в сети. Все это может привести к получению некорректных данных, которые в дальнейшем могут негативно повлиять на работу системы. Поэтому система должна отслеживать и адекватно реагировать на различные ошибки при получении данных от интернет серверов.



Рисунок 3.3 Схема работы подсистемы извлечения данных

Работа данной подсистемы связана с получением данных с различных интернет источников. Как правило, беспрерывная работа этих источников не гарантируется, они могут отключаться на технические работы, быть недоступны вследствие неполадок в сети. Все это может привести к получению некорректных данных, которые в дальнейшем могут негативно повлиять на работу системы. Поэтому система должна отслеживать и адекватно реагировать на различные ошибки при получении данных от интернет серверов.

Данный модуль содержит в себе механизмы для реагирования на основные виды отказов при получении данных от поисковой системы или при загрузке содержимого веб-страницы. Это позволяет подсистеме извлечения данных не использовать полученные данные в дальнейшем анализе, что могло бы вызвать различные ошибки, а пытаться получить корректные данные иным способом – запрос может быть отложен на некоторое время, запрос может быть занесен в специальную таблицу, для того чтобы обслуживающий инженер самостоятельно мог проверить причину ошибки, или могут быть изменены некоторые параметры запроса.

Также данная подсистема выполняет проверку извлеченных данных на соответствие допустимым значениям типа данных атрибута. Это позволяет избежать ошибок входных данных в подсистеме интеллектуального анализа.

### 3.2.2Проектированиеподсистемыинтеллектуального анализа данных

Функциональная схема данной подсистемы представлена на рисунке 3.4. Данная подсистема выполняет следующие функции:

1. Выполнение подготовительного анализа в поточном режиме (для классификаторов на основе нейронной сети Кохонена требуется предварительное обучение, и так как данная нейронная сеть обучается без учителя, то для обеспечения параллельной работы подсистемы извлечения данных и подсистемы анализа данных была реализована данная функция).
2. Извлечение подготовленных данных из БД.
3. Выполнение кластеризации и классификации данных на основе настроек, полученных от пользователя перед началом анализа.
4. Отображение текущего состояние процесса извлечения (помимо общего процента готовности возможно отображение доли извлеченных признаков)
5. Отображение результатов, полученных при выполнении классификации

Эта подсистема использует для своей работы данные которые были получены в ходе работы подсистемы извлечения данных. Поэтому их корректность и пригодность для последующей обработки гарантируется условиями типизации значений атрибутов. Именно поэтому число отказов в данной подсистеме минимально и не имеет сильной зависимости от внешних входных данных.

В качестве механизма для кластеризации и классификации был выбран эмулятор нейронной сети Кохонена разработанный автором данной работы. Так как для работы данного эмулятора необходимы численные входные данные, то в данном варианте системы возможна кластеризация и классификация лишь атрибутов с типами Int, Double, Bool.

В дальнейшем возможно добавлений модулей для преобразования текстовых значений атрибутов в численные эквиваленты на основе их семантики.



Рисунок 3.4 Схема работы подсистемы интеллектуального анализа данных

### 3.2.3 Проектирование базы данных и подсистемы работы с БД

В разрабатываемой системе будет использоваться реляционная модель базы данных. Ее схема представлена на рисунке 3.5.

В базе данных присутствует небольшая информационная избыточность, необходимая, во-первых, для увеличения надежности БД. Если одна из таблиц, описывающих структуру фрейма класса будет повреждена, то с помощью избыточных данных из других таблиц возможно частичное восстановление структуры фрейма класса. Во-вторых, данная избыточность позволяет увеличить скорость параллельной обработки базы данных несколькими потоками. Это позволяет подсистеме извлечения данных и подсистеме интеллектуального анализа данных работать с разными таблицами, но при этом сохраняется целостность и непротиворечивость данных. В-третьих, это позволяет производить горизонтальное масштабирование базы данных, к примеру, разместить ее на нескольких серверах.

Структура фрейма класса описывающего извлекаемую сущность хранится в следующих таблицах:

Class\_frame – таблица содержащая идентификатор данного фрейма класса, ссылку на коренной фрейм, а также комментарий описывающий данный класс.

Frames – таблица хранящая информацию о всех фреймах в системе. Здесь хранится идентификатор фрейма родителя (если фрейм является корневым в фрейме классе, то данное поле имеет значение NULL), а также список идентификаторов присоединенных демон-функций (GUID’ы модулей предобработки данных).

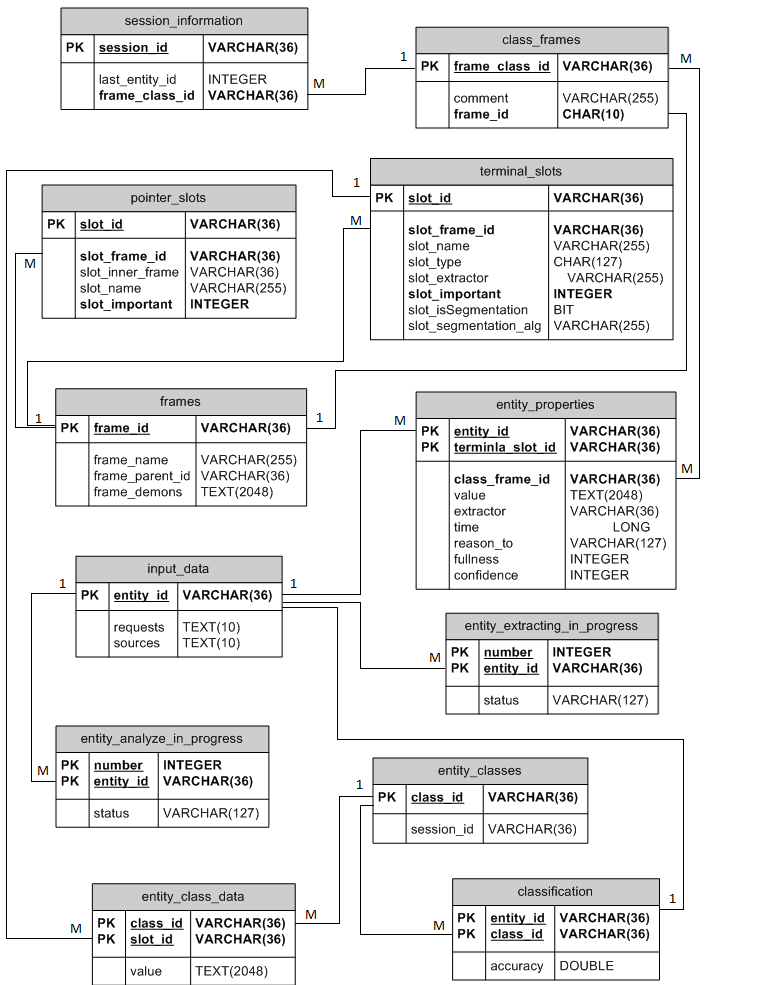


Рисунок 3.5 Схема базы данных системы.

Pointer\_slots – содержит информацию о фреймах-указателях. Включает в себя идентификатор на внутренний фрейм, имя слота и значение параметра важности слота.

Terminal\_slots – содержит информацию о терминальных фреймах. Здесь хранятся тип данных данного слота, значение (в текстовом виде), идентификатор экстрактора (GUIDмодуля экстрактора), время извлечения значения, причина по которой экстрактор установил данное значение (если значение является undefined, то данное поле содержит флаг указывающий на причину этого), а также значения параметров полноты и уверенности извлеченных значений.

Таблицыentity\_extracting\_in\_progressиentity\_analyze\_in\_progressслужат буферами для хранения промежуточных данных о ходе выполнения процессов извлечения и анализа данных. Также в случае многопоточной обработки они выполняют роль общей памяти для потоков обработки. В случае если работа программы завершается в аварийном режиме, то с помощью данных этих двух таблицах система может продолжит выполнение двух основных операций – извлечение данных и анализ, с прерванного места.

Таблица entity\_properties хранит в себе результат извлечения данных с веб-страниц (в виде фрейма-образца). Данные хранятся в виде значений терминальных слотов для каждой отдельной сущности. Так как слоты указатели не содержат внутри себя каких-либо данных, а служат лишь узлами для структуры фрейма, то они не дублируются в данной таблицы. Данная таблица является исходными данными для подсистемы интеллектуального анализа.

Втаблицахentity\_classes, entity\_class\_dataиclassification хранятся данные полученные в результате кластеризации и классификации полученных данных. А именно идентификаторы полученных в процессе кластеризации или заданных пользователем классов, значения терминальных слотов ядер класса, а также итоговую классификацию сущностей по классам.

### 3.2.4 Проектирование подсистемы мониторинга

Так как данная система должна отвечать требованиям высоким требованиям надежности, причем большую часть времени функционирования система находится без наблюдения оператора, необходима реализация подсистемы слежения за состоянием работы системы.

Данная подсистема будет включать в себя внешний модуль мониторинга за состоянием процесса работы основной системы. Анализ выполняется с интервалом в 1 минуту.

Анализируемые состояние основного процесса:

* Running – Процесс находится в активном состоянии.
* Suspend – Процесс приостановлен. Подобное состояние может возникать в следующих ситуациях:
* ОС выполняет освобождение памяти, требуемое для процесса, в случае если используется файл подкачки данный процесс занимает значительное время.
* ОС остановила фоновый поток из-за возникших внутренних проблем.
* Процесс был остановлен пользователем.
* ОС использует распределение времени выполнения для нескольких процессов, работающих с общей памятью
* Родительский процесс выполнил приостановку дочернего процесса
* Blocked – процесс заблокирован до тех пор, пока не произойдут операции ввода\вывода

Основными событиями на которые необходимо реагировать внешней системе мониторинга являются либо отсутствие процесса в списке процессов (если произошло его завершение по каким-то причинам), или если процесс долгое время находится в состояниях Suspend или Blocked.

Если за определённый отрезок времени процесс не вернулся в состояние Running, то внешний модуль мониторинга прекращает работы процесса и его перезапуск. Для завершения процесса в ОС Windowsиспользуется команда «taskkillpid», где pid–идентификатор завершаемого процесса. Далее выполняет запись в журнал мониторинга (для данного модуля существует свой, отдельный файл с журналом событий), если необходимо делается сохранение образа БД в резервный файл, после чего система запускается заново с параметром –r, который означает автоматическое продолжение предыдущего сеанса работы. Если при перезапуске возникли какие-то неполадки, происходит запись об этом в журнал событий системы, и система прекращает работу. Если же система была удачно перезапущена модуль внешнего мониторинга возвращается к своему стандартному функционированию.

Внутренняя часть подсистемы мониторинга отвечает за слежение за потоками обработки в подсистемах извлечения и анализа данных. Слежение за состояниями потоков выполняется аналогично слежению за процессом, с тем отличием, что если поток обработки был приостановлен или экстренно завершился, то прежде чем выполнить его перезапуск необходимо произвести откат в состоянии той части БД, над которой в настоящий момент работал данный поток. Это осуществляется с помощью флагов над обрабатываемыми сущностями. Система находит сущность которая помечена как обрабатываемая данным потоком и стирает всю информацию из БД относящуюся к данной сущности. После чего данная сущность помечается как необработанная.

## 3.3 Разработка программной реализации системы

Для разработки данного ПО были выбраны следующие средства разработки:

* Язык программирования C#
* Framework .NET version 4.5
* СредаразработкиMicrosoft Visual Studio 2012

Разрабатываемое ПО написано в рамках парадигмы ООП, а также модульной реализации. Каждый тип модуля имеет Unit-тест для контроля за его работоспособностью в процессе разработки и последующей модификации.

Диаграмма классов подсистемы извлечения знаний представлена на рисунке 3.6.

Абстрактные классы ModueWithOptions и DataModuleвыполняют роль интерфейса для подключаемых модулей с которыми работает данная подсистема. Она использует 3 следующих типа модуля:

1. VBDataExtractor– модуль для извлечения атрибутов. Данные модули привязываются к терминальному слоту и выполняют извлечение атрибута в соответствии с установленными настройками. Также они выполняют преобразование извлеченных значений к установленному типу слота.
2. VBDataPreprocessor – модуль предварительной обработки данных. Реализует различные методы для очистки входных данных. В данной работе представлен модулем TagRemover, которыйвыполняет удаление всех HTMLтегов и преобразование HTML-сущностей в их символьное представление.
3. VBDataSegmentor – модуль сегментации содержимого веб-страницы на семантические блоки. На вход данный модуль получает объект класса HtmlElement или его строковое представление, далее выполняет разбиение полученного фрагмента HTMLна семантические блоки и возвращает их в видел HtmlElementCollection. В данной работе реализациями данного класса являются DomAnalyzerSegmentorи VIPS.

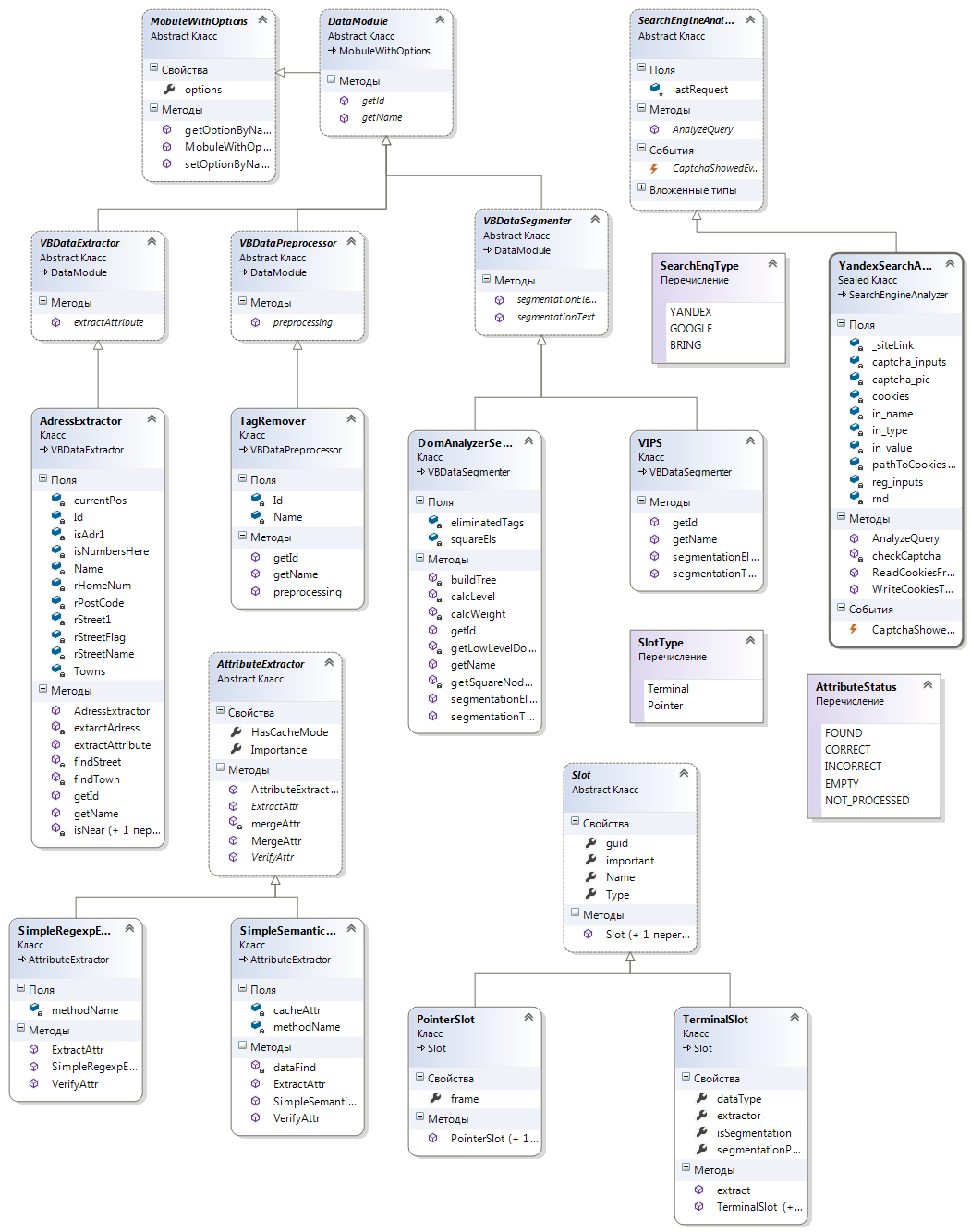


Рисунок 3.6 Диаграмма классов системы извлечения знаний.

DomAnalyzerSegmentor– класс реализующий алгоритм сегментации основанный на анализе DOMи его содержимого.

VIPS – класс реализующий алгоритм сегментации на основе алгоритма визуальной сегментации.

Класс Slotпредставляет собой абстрактный слот фрейма, классами-реализациями которого являются PointerSlot и TerminalSlot.

TerminalSlot – реализует терминальный слот фреймовой структуры. Содержит информацию о типе значения атрибута, экстракторе который применяется для его извлечения, а также используемые сигналы для выбора анализируемых семантических блоков.

SearchEngineAnalyzer – абстрактный класс представляющий интерфейс взаимодействия с модулями взаимодействия с поисковыми системами. В данной работе его реализацией является класс YandexSearchAnalyzer инкапсулирующий в себе логику работы с поисковой системой Яндекс.

Диаграмма классов подсистемы извлечения знаний представлена на рисунке 3.7.

Здесь присутствуют три интерфейса – INeuron, INeuronWithOut, INeuronLayer.

INeuron – интерфейс который должен реализовывать любой нейрон

INeuronWithOut – интерфейс который должен реализовывать любой “неполный” нейрон, например нейрон входного слоя, который не имеет функции активации или синаптических весов, однако также может образовывать слои, и передавать значения сигналов.

INeuronLayer – интерфейс который должен реализовывать любой слой нейронов.

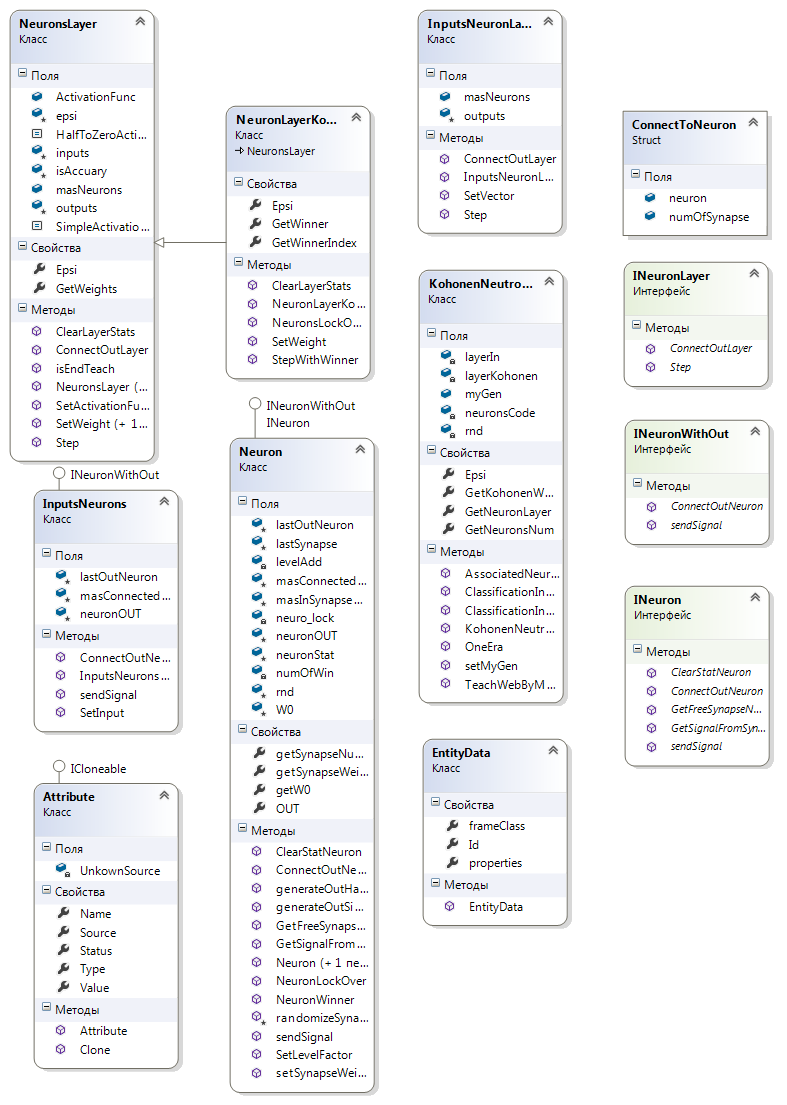


Рисунок 3.7 Диаграмма классов подсистемы интеллектуального анализа данных.

Класс NeuronLayer служит для представления слоя нейронов, в отличии от интерфейса класс содержит в себе функции для подсоединения к другому слою, стандартные функции обучения, которые могут быть переопределены в потомках.

Класс NeuronLayerKohonen – является уже представлением слоя Кохонена. Он переопределяет функции своего предка в соответствии с алгоритмами сети Кохонена.

Класс InputNeuronLayer – представляет входной слой нейронов.

Класс Neuron – это основной класс, который представляется объект – формальный нейрон. Внутри этого класса возможна настройка функций активации, от простых ступенчатых до сигмоидальных. Этот класс реализует функции по подключению одного нейрона к другому, изменению его весов и т.д.

Класс KohonenNetworkWeb – представляет собой законченную нейронную сеть Кохонена, состоящую из 2 слоев – входного слоя и слоя Кохонена. Этот класс реализует алгоритм обучения Кохонена, и может выполнять классификацию входных данных, возвращая вектор весов нейрона-победителя на каждый поступающий входной вектор.

Диаграмма классов подсистемы работы с БД представлена на рисунке 3.8.

Абстрактный класс DBAdapterявляется реализацией интерфейса прямого взаимодействия с различными базами данных. Он представляет набор функций для подключение к БД и выполнения запросов.

Его класс-потомок DBWorkerпредставляет набор функций для работы с разработанной структурой базы данных. Но он не включает в себя их реализацию, зависящую от конкретных особенностей БД. Его класс наследник MySqlWorkerинкапсулирует в себе всю логику по взаимодействию с БД MySQL.

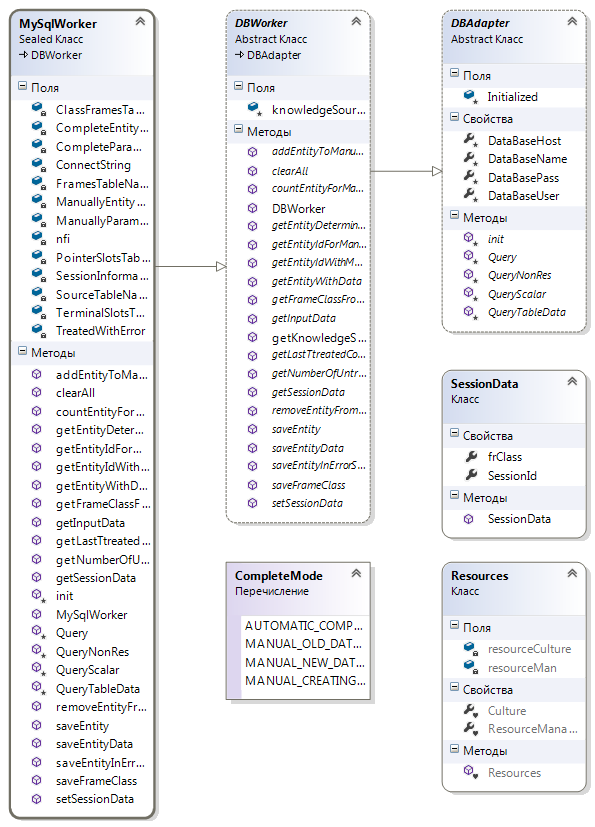


Рисунок 3.8 Диаграмма классов подсистемы работы с БД.

## Выводы

В данном разделе была спроектирована и разработана система извлечения знаний из поисковых систем. Разработанная система реализует алгоритм извлечения данных основанный на иерархической фреймовой модели и использующий алгоритмы сегментации веб-страниц для первоначальной обработки входных данных.

В ходе проектирования системы и тестирования прототипов было установлено, что чаще всего отказы, связанные с ошибочным извлечением знаний, происходят при функционировании подсистемы извлечения данных. Данное предположение также основывается на том факте, что подсистема извлечения знаний, во-первых, взаимодействует с внешними источниками, надежность и доступность которых неизвестна, а во-вторых, оперирует «сырыми» данными, которые могут содержат непредусмотренные разработчиком ошибки.

Источником входных данных для подсистемы интеллектуального анализа является разработанная БД, содержащая нормализованные значения атрибутов. Так как современные БД обладают высокими показателями надежности из этого можно сделать вывод, что доля отказов в работе подсистемы интеллектуального анализа будет существенно меньше, чем в подсистеме извлечения данных.

Даже используя современные методики тестирования, нельзя полностью исключить вероятность ошибок в программном коде, которые могут привести с аварийному завершению программы, поэтому в систему была включена подсистема мониторинга. С помощью этого система в случае если это возможно, сможет восстанавливаться после отказов, связанных с ее аварийным завершением, в том случае если данные о ходе работы системы не были повреждены.

# 4 Проведение экспериментов и анализ результатов

При проведении исследования программной реализации системы извлечения знаний из поисковых систем будет проведена серия экспериментов с различными входными данными и на разных аппаратных конфигурациях. В зависимости от задач, поставленных перед системой в ходе эксперимента, будут функционировать различные модули системы. Это позволит упростить сбор статистики по работе отдельных модулей.

## 4.1 Исследование зависимости надежности алгоритмов извлечения данных от методов предварительной обработки.

Целью данного исследования является определение наиболее эффективного, с точки зрения отношения времени выполнения к надежности извлечения данных, метода сегментации входных данных.

Основными задачами исследования являются определения временных параметров процесса извлечения данных и надежности алгоритмов распознавания при использовании различных методов сегментации, а также в случае использования системы без предварительной сегментации входных данных.

В данном исследовании будут проведены три эксперимента:

1. Извлечение атрибутов при предварительной обработке входных данных с помощью алгоритма сегментации, основанного на анализе объектной модели документа.
2. Извлечение атрибутов при предварительной обработке входных данных с помощью алгоритма визуальной сегментации VIPS.
3. Извлечение атрибутов без использования сегментации входных данных.

В данной серии экспериментов будут использоваться следующие модули подсистемы извлечения данных:

* Модуль извлечение физического адреса
* Модуль определения названия сущности
* Модуль извлечение телефонного номера
* Модуль извлечения описания

Подсистема интеллектуального анализа данных в этой серии экспериментов использоваться не будет, так как исследуемые алгоритмы реализованы исключительно в подсистеме извлечения данных.

### 4.1.1 Условия проведения и входные данные экспериментов

Система будет функционировать на персональном компьютере со следующими техническими характеристиками:

* Процессор:Intel i7-4820K 3.70GHz
* Объем оперативной памяти: 16Gb
* Интернет канал (прием, отдача): 32Mbytes/s, 32Mbytes/s
* Операционная система: Windows 7 Professional x64

Входными данными для системы будет список веб-сайтов состоящий из 500 позиций. Все адреса были проверены на доступность и соответствие тем требованиям, которые предъявляет система. Алгоритм определения кодировки содержания веб-сайта на данной выборке всегда определял верную кодировку. Это позволило исключить из эксперимента отказы системы, связанные с недоступностью веб-ресурса или некорректной обработки его содержимого.

Дадим определения отказов, которые будут учитываться в данном эксперименте:

1. Полный отказ алгоритма сегментации – происходит в случае если алгоритм сегментации выдал заведомо некорректные данные – один из блоков не имеет содержимого или максимальный и минимальный веса полученных блоков отличаются более чем на порядок (Это свидетельствует о неправильной сегментации).
2. Некорректное извлечения атрибута при анализе блока, выбранного для извлечения – происходит, когда блок был выбран как содержащий необходимую информацию, но алгоритм извлечения не извлек данные, либо они некорректны.
3. Некорректный выбор блока для извлечения – если алгоритм выбрал блок не содержащий необходимые данные для извлечения.
4. Некорректная сегментация – когда значение атрибута было разделено на несколько блоков.

Для сбора информации о состояниях системы в период ее работы были проанализирован журнал событий в который система выводил всю информацию о процессах обработки.

### Автоматическое извлечение знаний с использованием алгоритма сегментации основанного на анализе объектной модели документа

Время работы системы составило 1 час0 минут 21 секунду.

Таблица 4.1. Данные о времени нахождения системы в различных состояниях при извлечении данных с сегментацией, основанной на анализе объектной модели документа.

|  |  |
| --- | --- |
| **Состояние системы** | **Время, сек** |
| Загрузка контента | 1489 |
| Сегментация | 277 |
| Извлечение адреса | 502 |
| Извлечение названия | 319 |
| Извлечение телефона | 353 |
| Извлечение описания | 551 |
| Остальное | 130 |

Для удобства ниже представлена круговая диаграмма, показывающая долю времени каждого состояния от всего времени работы системы (Рисунок 4.1).

Наиболее долгим процессом в работе системы ожидаемо является загрузка контента с веб-сайта. Данное значение зависит только от пропускной способности входящего интернет соединения. Так как современные сервера имеют широкополосные каналы доступа в интернет, то при использовании данной системы на сервере доля времени по загрузке содержимого сайтов может существенно уменьшится.

Рисунок 4.1 Доля времени пребывания системы в различных состояниях

В процессе работы в системе были получены следующие данные:

Таблица 4.2 Итоговые данные эксперимента

|  |  |
| --- | --- |
| Название параметра | Число, шт. |
| Обработанные сущности | 500 |
| Обработанные атрибуты | 2000 |
| Извлечено атрибутов | 1579 |
| Корректных атрибутов | 1232 |
| Некорректно извлечённые атрибуты | 347 |
| Отказы в алгоритме сегментации | 13 |

Из данных таблицы 4.2 видно, что система произвела извлечение 78.95% из всего множества атрибутов. 17.35% были некорректными данными. Следовательно, надежность алгоритма извлечения данных в данном случае составила 61.5%, т.е. вероятность корректного извлечения атрибута составляет .

Вероятность отказа в алгоритме сегментации в данном эксперименте составила .

### Автоматическое извлечение знаний с использованием алгоритма сегментации VIPS

Время работы системы составило 1 час 11 минут 53 секунд.

Таблица 4.3. Данные о времени нахождения системы в различных состояниях при извлечении данных с VIPSсегментацией

|  |  |
| --- | --- |
| Состояние системы | Время, сек |
| Загрузка контента | 1501 |
| Сегментация | 803 |
| Извлечение адреса | 539 |
| Извлечение названия | 301 |
| Извлечение телефона | 323 |
| Извлечение описания | 670 |
| Остальное | 176 |

Рисунок 4.2 Доля времени пребывания системы в различных состояниях при сегментации алгоритмом VISP

По завершению работы системы получены следующие результаты:

Таблица 4.4 Итоговые данные экспериментаcиспользование сегментации VIPS

|  |  |
| --- | --- |
| Название параметра | Число, шт. |
| Обработанные сущности | 500 |
| Обработанные атрибуты | 2000 |
| Извлечено атрибутов | 1470 |
| Корректных атрибутов | 1229 |
| Некорректно извлечённые атрибуты | 241 |
| Отказы в алгоритме сегментации | 41 |

По полученным данных из таблицы 4.4можно определить, что система произвела извлечение 73.5% из всего множества атрибутов. 13.05% были признаны некорректно извлеченными. Следовательно, надежность алгоритма извлечения данных в эксперименте составила 61.45%, т.е. вероятность корректного извлечения атрибута составляет .

Вероятность отказа в алгоритме сегментации в данном эксперименте составила .

### Сравнение результатов первых двух экспериментов

Если сравнивать полученные данные о временных параметрах процесса работы системы из второго эксперимента (Таблица 4.3 и рисунок 4.2) с тем что были получены в ходе первого эксперимента (Таблица 4.1 и рисунок 4.1) можно сделать следующие выводы:

1. Общее время работы системы во втором эксперименте возросло на 19.11% по сравнению с первым экспериментов.
2. Время, затрачиваемое системой на сегментацию содержимого веб-сайтов, возросло почти в 3 раза (289.89%).
3. Время извлечения атрибутов «адрес» и «описание» также увеличилось. Время атрибутов «название» и «телефон» почти не изменилось.

Увеличение времени извлечения некоторых атрибутов связано с тем, что алгоритм сегментации VIPSразделяет веб-страницу на менее равномерные, по объему текстового содержимого, блоки чем алгоритм основанный на анализе объектной модели документа. А так как в большинстве случае при извлечении данных атрибутов анализируется блок с наибольшим объемом текстовой информации, то и время анализа возрастает.

При сравнении результатов работы системы видно, что надежность алгоритма извлечения во втором эксперименте почти не изменилась (61.5% и61.45%) по сравнению с первым. Однако при этом уменьшился процент некорректно извлечённых атрибутов от общего числа извлеченных атрибутов. Также сильно возросло число отказов самого алгоритма сегментации(процент отказов возрос с 2.6% до 8.2%).

Для более детального анализа рассмотрим причины случаев, когда атрибуты были не обнаружены системой или извлечение было некорректным.

Таблица 4.5 Причины по которым некоторые атрибуты не были извлечены

|  |  |  |  |
| --- | --- | --- | --- |
|  | Атрибуты, не извлеченные из-за полных отказов в алгоритме сегментации, шт. | Атрибуты, не извлеченные из-за неверной сегментации, шт. | Атрибуты, не извлеченные из-за недостатков алгоритмов извлечения, шт. |
| Сегментация с анализом объектной модели | 52 | 20 | 349 |
| Сегментация алгоритмом VIPS | 164 | 11 | 355 |

|  |  |  |
| --- | --- | --- |
|  | Некорректно извлеченные атрибуты по причине неверной сегментации, шт. | Некорректно извлеченные атрибуты из-за недостатков алгоритмов извлечения, шт. |
| Сегментация с анализом объектной модели | 133 | 214 |
| Сегментация алгоритмом VIPS | 61 | 200 |

Таблица 4.6 Причины по которым некоторые атрибуты были некорректно извлечены.

Анализируя данные из таблиц 4.5 и 4.6 можно сделать следующие выводы:

1. Алгоритм сегментации VIPSобеспечивает более корректную сегментацию, по сравнению с алгоритмом сегментации, основанным на анализе объектной модели.
2. Низкая надежность данного алгоритма сводит к нулю преимущества более корректной сегментации.

В сочетании с большими временным затратами на сегментирование, алгоритм VIPSв данной реализации уступает простому алгоритму анализа объектной модели документа в общей эффективности.

Однако основная причина слабой надежности алгоритма визуальной сегментации в том, что он сильно зависит от правил визуализации содержимого веб-страниц. За время, прошедшее с создания данного алгоритма произошли значительные изменения в области визуального отображения веб-сайтов, и новые подходы к созданию красивых мультимедийных веб-страниц никак не учитываются данным алгоритмом.

### Автоматическое извлечение знаний без использования алгоритма сегментации

Время работы системы составило 1 час 22 минуты 52 секунд.

Таблица 4.7. Данные о времени нахождения системы в различных состояниях при извлечении данных без предварительной сегментации

|  |  |
| --- | --- |
| Состояние системы | Время, сек |
| Остальное | 141 |
| Загрузка контента | 1533 |
| Извлечение адреса | 990 |
| Извлечение названия | 339 |
| Извлечение телефона | 362 |
| Извлечение описания | 1651 |

Рисунок 4.3 Доля времени пребывания системы в различных состояниях без использования алгоритма сегментации.

После завершения работы системы были получены следующие данные:

Таблица 4.8 Итоговые данные экспериментапо извлечению данных без использования сегментации

|  |  |
| --- | --- |
| Название параметра | Число, шт. |
| Обработанные сущности | 500 |
| Обработанные атрибуты | 2000 |
| Извлечено атрибутов | 1770 |
| Корректных атрибутов | 1131 |
| Некорректно извлечённые атрибуты | 639 |

По результатам эксперимента система произвела извлечение 88.5% из всего множества атрибутов. 31.95% были признаны некорректно извлеченными. Следовательно, надежность алгоритма извлечения данных в эксперименте составила 56.55%, т.е. вероятность корректного извлечения атрибута составляет .

### 4.1.6 Сравнение результатов работы системы с использованием алгоритмов сегментации и без предварительной сегментации.

Сравнительные диаграммы надежности алгоритмов извлечения и времени работы системы представлены на рисунках 4.4 и 4.5

Рисунок 4.4 Вероятность корректного извлечения атрибута.

Рисунок 4.5 Время работы системы.

При сравнении вероятности корректного извлечения атрибутов можно сделать вывод что использование предварительной сегментации содержимого веб-страницы увеличивает надежность алгоритмов извлечения данных.

Также из данных графиков видно, что когда система не использует сегментацию данных то время ее работы увеличивается по сравнению с теми экспериментами где использовались алгоритмы сегментации. Более информативно это описывает сравнение диаграмм, изображённых на рисунках 4.1, 4.2 и 4.3. Здесь явно заметно увеличение общей доли времени, затрачиваемое системой на извлечение таких атрибутов как «адрес» и «описание». Это вытекает из того, что данные атрибуты извлекаются с помощью механизма регулярных выражений. Причем оба атрибута при извлечении используют анализ большого числа шаблонов для наиболее полного извлечения. В то время как для определения содержания адреса или описания в семантическом блоке используются менее ресурсоемкие методы, основанные на статистических оценках содержимого блока, а также нахождения некоторого числа слов-сигналов.

В итоге при применении сегментации общий объем данных который необходимо проанализировать модулям по извлечению данных атрибутов значительно снижается, что и вызывает ускорение в работе системы.

Более интересен тот факт, что несмотря на наличие в обоих первых экспериментах отказов алгоритмов сегментации и наличие случаев некорректной сегментации, системы с предварительной сегментацией показывают больший процент корректности извлечения.

Это объясняется тем, что корректное извлечение таких атрибутов как «описание» и «адрес» сильно зависит от общего объема анализируемого текста. К примеру, границы блока «описание» рассчитываются исходя из стандартных подходов для анализа текстов – наличие символом табуляции или переноса строки. Так как при переходе от HTMLструктуры к простому текстовому содержанию, если удаляемые элементызаменяются на перенос строки, это приводит к урезанию этих атрибутов. Если же удаляемые атрибуты заменяются на пробелы есть вероятность что следом будет следовать текст не имеющий никакого отношения к данному атрибуту. Именно в этом случае использование сегментации позволяет в большинстве случаев приблизительно определить текстовые границы данного атрибута.

Это также объясняет наличие большого числа некорректно извлечённых атрибутов – при анализе большого объема текста и не имея возможности разделить его, алгоритмы извлечения этих атрибутов ориентируются лишь на сигнальные слова, что вызывает большое число ложных срабатываний.

## 4.2 Исследование процесса долгосрочного функционирования подсистемы извлечения данных на основе Марковской модели

В разработанной системе предусмотрены процедуры восстановления после наиболее частых отказов, которые были зафиксированы во время тестирования системы. Они позволяют сохранять работоспособность системы в тех случаях, когда это возможно. Но на восстановление работоспособности системы тратится время, во время которого система не выполняет свои основные функции по обработки данных.

Целью данного исследования является определение доли времени, которое система выполняет полезную работу. Под полезной работой в данном случае будем понимать работу, связанную непосредственно с загрузкой, извлечением и анализом данных.

Задачи данного исследования:

* 1. Провести эксперименты над различными модулями системы для сбора информации о возможных отказах.
  2. На основании тех отказов которые были выявлены в ходе данных экспериментов построить модель данной системы в виде графа надежности системы.
  3. Провести эксперименты по восстановлению работоспособности системы из различных от казовых состояний.
  4. Основываясь на статистических данных полученных в ходе проведенных экспериментов определить интенсивности отказов и восстановления.
  5. Для стационарного состояния системы рассчитать предельные вероятности состояний.
  6. Провести контрольный эксперимент по функционированию системы в течение длительного времени и собрать информацию о времени нахождения системы в каждом состоянии.
  7. Сравнить полученные при моделировании результаты с практическими.
  8. Сделать вывод на основе полученных результатов.

### Условия проведения и входные данные экспериментов

Во время экспериментов система будет функционировать на сервере со следующими техническими характеристиками:

* Процессор:Intel i5-3470 3.0GHz
* Объем оперативной памяти: 8Gb
* Интернет канал (прием, отдача): 32Mbytes/s, 32Mbytes/s
* Операционная система: Windows 7 Professional x64

При проведении экспериментов использовалась конфигурация системы с включенной сегментацией содержимого веб-сайтов с помощью разработанного алгоритма основанного на анализе объектной модели документа.

В данных экспериментах нет никаких сведений о достоверности входного массива, также возможно отсутствие веб-сайта по искомому поисковому запросу.

Для получения статистической информации об отказах системы, связанных с ошибками при загрузке данных тестировался модуль загрузки содержимого веб-сайтов. Исходными данными для выполнения загрузки являлся массив с поисковыми запросами размером 30 000 позиций.

Для получения информации об отказах в модуле обработки содержимого веб-сайтов (входящего в подсистему извлечения данных), проводился эксперимент по обработке массива данных полученного в результате предыдущего эксперимента.

### Проведение экспериментов и построение модели системы, основанной на Марковских процессах

Общее время функционирования модуля загрузки содержимого веб-сайтов составило 13 часов 39 минут 13 секунд. Общее время функционированиямодуля обработки содержимого 22 часа 21 минуту 9 секунд.

Ни одного полного отказа, после которого автоматическое восстановление системы было бы невозможно не произошло. Основными причинами для возникновения подобных отказов являются аппаратные отказы или сбои в БД. Однако интенсивности отказов у современных аппаратных частей компьютера, ровно, как и у БД – величины более низких порядков чем интенсивности рассматриваемых отказов. Поэтому при построении модели данные отказы рассматриваться не будут.

Были зафиксированы следующие отказы.

1. Отказ алгоритма сегментации – алгоритм сегментации либо произвел неравномерное разбиение веб-страницы (веса блоков отличаются более чем на порядок), либо выделил незначащий блок (пустое текстовое содержание).
2. Отказ вызванный некорректными входными данными – либо получены данные не текстового формата, либо при первичной обработке входных данных не была распознана кодировка, либо в ответе отсутствуют HTMLтеги.
3. Отказ главного процесса – возникновение критической ошибки в главном процессе системы, который повлек за собой его экстренное завершение и последующую перезапуск системы с помощью внешней программы мониторинга.
4. Отказ запрашиваемого ресурса – получение HTTPкода состояния отличного от 2xx, но не свидетельствующего о неполадках в интернет соединении.
5. Отказ в установлении соединения вызванный блокировкой – блокировка HTTPзапроса антивирусом или иной программой.
6. Отказ в установлении соединения вызванный неполадками в сети – получение HTTPкода состояния из множества {404,503,504,505}, который свидетельствует о возможных неполадках в интернет соединении.
7. Отказ интернет соединения – отсутствие устойчивого канала связи с сетью интернет.

#### Основные допущения при моделировании разработанной системы с помощью Марковских процессов

Так как вероятности появления отказов, после которых система не сможет восстановить свою работоспособность автоматически, являются величинами меньших порядков чем вероятности появления рассматриваемых отказов, то можно рассматривать данный процесс как стационарный.

Исходя из того, что для определения настоящих законов распределения времен безотказной работы системы до каждого типа отказа отдельно, необходимы большие статистические исследования (выборки, полученные в результате данной работы не обладают достаточной репрезентативностью в виду малых размеров), для описания процесса функционирования разработанной системы с помощью Марковских цепей ведены следующие допущения.

1. Все отказы в системе являются независимыми.
2. Время безотказной работы системы отдельно до каждого из рассмотренных отказов и время восстановления системы после каждого отказа имеют экспоненциальное распределение.
3. Интенсивность переходов не зависит от времени .
4. Восстановление системы при каждом типе отказа начинается сразу же после возникновения отказа.

#### Граф состояний системы

На основе анализа результатов эксперимента построен граф состояний системы изображенный на рисунке 4.6.



Рисунок 4.6 Граф состояний системы.

На данном графе изображены следующие состояния системы.

S0 – Основное активное состояние системы. В данном состоянии система выполняет работу по извлечению данных.

S1 –Отказ алгоритма сегментации.

S2 – Отказ вызванный некорректными входными данными.

S3 – Отказ главного процесса.

S4 – Отказ запрашиваемого ресурса.

S5 – Отказ в установлении соединения вызванный блокировкой.

S6 – Отказ в установлении соединения вызванный неполадками в сети.

S7 – Отказ интернет соединения.

#### Определение интенсивностей переходов αи µна основе статистических данных

В ходе проведения экспериментовтестируемые модули генерировали файлы журнала, в котором были отмечены все моменты времени по переходам в различные состояния отказа. На основании анализа этих файлов были получены следующие данные.

Таблица 4.9 Статистические данные полученные в результате экспериментов

|  |  |  |  |
| --- | --- | --- | --- |
| Тип отказа | Количество отказов за эксперимент, шт. | Интенсивность отказа, 1/сек | Интенсивность восстановления после отказа |
| Отказ алгоритма сегментации | 208 | 0,000811644 | 0,99047619 |
| Отказ вызванный некорректными входными данными | 157 | 0,000611178 | 0,332540352 |
| Отказ главного процесса | 4 | 0,000015609 | 0,093023256 |
| Отказ в установлении соединения вызванный неполадками в сети | 347 | 0,00135274 | 0,333013128 |
| Отказ запрашиваемого ресурса | 248 | 0,000966243 | 0,332821301 |
| Отказ в установлении соединения вызванный блокировкой | 106 | 0,000414104 | 0,50057759 |
| Отказ интернет соединения | 5 | 0,004803074 | 0,00310559 |

#### Составление и решение системы уравнений, описывающих стационарный режим работы системы.

Так как представленный граф состояний системы является связанным, т.е. из каждого состояния система может перейти в любое другое за конечное число шагов, то существуют предельные стационарные вероятности состояний.

Составим систему линейных однородных уравнений для их расчета.

(2)

Также необходимо нормировочное условие.

(3)

Подставим значения интенсивностей, полученные из статистических результатов, а также заменим седьмое уравнение условием нормировки.

(4)

Для решения системы (4) используем матричный метод. Необходимым и достаточны условием для использования данного метода является где матрица основная матрица системы.

Используя теорему Лапласа для вычисления определителя*n-*го порядка, получим определитель данной матрицы.

Так как полученный определитель матрицы не равен 0, то допустимо применение метода обратной матрицы. Найдем обратную матрицу по следующей формуле.

Где транспонированнаяматрица алгебраических дополнений соответствующих элементов матрицы.

Умножив полученную матрицу на столбец свободных членов (0, 0, 0,0,0,0,0,1) получим решение данной системы уравнений.

, ,

, , ,

,

#### Сравнение результатов моделирования с практическими данными полученными в ходе контрольного эксперимента

Для проверки адекватности построенной модели был проведен контрольный эксперимент по функционированию системы в течение длительного времени, с полным профилированием всех состояний системы. Входными данными для эксперимента являлся массив поисковых запросов размером 52 000 позиций. Время проведения эксперимента составило 72 часа 12 минут 11 секунд (259931 секунд). Были получены следующие данные.

Таблица 4.10 Результаты контрольного эксперимента

|  |  |  |
| --- | --- | --- |
| Состояние системы | Время нахождения системы в данном состоянии, сек | Относительная доля времени нахождения системы в данном состоянии, % |
| Основное активное состояние системы | 255537 | 98,2 |
| Отказ алгоритма сегментации | 213 | 0,082 |

Окончание таблицы 4.10

|  |  |  |
| --- | --- | --- |
| Отказ вызванный некорректными входными данными | 470 | 0,181 |
| Отказ главного процесса | 49 | 0,019 |
| Отказ запрашиваемого ресурса | 777 | 0,299 |
| Отказ в установлении соединения вызванный блокировкой | 228 | 0,088 |
| Отказ в установлении соединения вызванный неполадками в сети | 1039 | 0,4 |
| Отказ интернет соединения | 1618 | 0,623 |

Сравнив полученные результаты с результатами моделирования получим, что средняя относительная ошибка модели составляет 4,84%.

Это свидетельствует о том, что построенная модель достаточно точно описывает стационарный режим работы системы.

### Анализ полученных результатов

Исходя из полученных результатов моделирования видно, что даже учитывая небольшую погрешность в модели (~5%), система находится в основном активном состоянии около 98% времени функционирования.

Из этого можно сделать вывод что используемые методы для восстановления работоспособности системы не оказывают существенного влияния на снижение общей производительности системы, при рассматриваемых интенсивностях отказов. Это обеспечивается, во-первых, наличием подсистемы мониторинга за состоянием основной системы, включающей в себя внешний модуль наблюдения за состоянием рабочего процесса системы. Во-вторых, за счет возможности системы анализировать полученные ответы по протоколу HTTPна предмет наличия кодов состояний, свидетельствующих об ошибке, и производить диагностику интернет соединения, в случае если возникают неполадки при загрузки веб-страниц.

## Выводы

В данной главе были проведены исследования по эффективности и надежности разработанной системы извлечения знаний из поисковых систем.

В первом исследовании было проанализировано влияние различных алгоритмов сегментации содержимого веб-страниц на временную эффективность системы и на надежность алгоритмов извлечения данных. Было установлено что при использовании алгоритмов сегментации система имеет большую производительность чем без использования сегментации. Также использование сегментации увеличивает надежность используемых алгоритмов извлечения данных.

При сравнении двух различных алгоритмов сегментации – алгоритма, основанного на анализе объектной структуры документа и алгоритма, основанного на «компьютерном зрении», было установлено следующее.

1. Надежность алгоритмов извлечения данных почти не изменилась при смене алгоритма сегментации.
2. Алгоритм, основанный на анализе объектной структуры документа, имеет большую производительность (время обработки на 19.11% меньше).

Во втором исследовании было проведено моделирование стационарного режима работы системы с помощью Марковской модели. Построенная модель была рассчитана на основе статистических данных полученных в ходе серии экспериментов. Для проверки адекватности модели был проведен контрольный эксперимент на долгосрочное функционирование системы.

Анализируя полученные результаты, можно сделать вывод, что используемые в разработанной системе процедуры восстановления из состояний отказа не оказывают сильного негативного влияния на производительность системы (система занимается полезной работой по загрузке и обработке данных около 98% от всего времени функционирования).

# ЗАКЛЮЧЕНИЕ

Паш, сравни, что ставилось и что получено. Все, в принципе, есть – только нужно перефразировать.

Цель данной работы – разработать и исследовать характеристики надежности системы автоматического извлечения знаний из поисковых систем.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Разработать алгоритм функционирования системы извлечения знаний из поисковых систем.

2. Реализовать программное обеспечение на основе данного алгоритма.

3. Произвести тестирование разработанного ПО в нескольких режимах эксплуатации.

4. На основе данных полученных в результате тестирования произвести анализ характеристик надежности разработанной системы.

В данной магистерской диссертации был разработан вариант системы извлечения знаний из веб-ресурсов использующий поисковые сети как средство для первичного поиска информации в сети интернет.

Процесс извлечения знаний в данной системе разделен на два основных этапа. Первый этап заключается в извлечении данных из веб-ресурсов с помощью отдельных модулей, каждый из которых реализует извлечение отдельного типа данных. Второй этап заключает в анализе полученных данных, в том числе установление идентичности между полученными фреймами−образцами если они описывают один концептуальный объект.

В разработанной системе предлагается использовать для решения задачи кластеризации и классификации извлеченных данных нейронную сеть Кохонена. Этот выбор основывается на возможности обучения и работы данной сети на потоке данных, что позволяет выполнять анализ до завершения процесса извлечения.

Для представления декларативных знаний, извлеченных из веб-страниц использовался модифицированный вариант иерархической фреймовой структуры, в которой демон−функции присоединяются к фрейму и выполняют роль предобработки данных при извлечении из анализируемого содержимого веб-страницы значений слота данного фрейма.

Было предложено использовать сегментацию содержимого веб-страницы на семантические блоки для увеличения производительности и надежности алгоритмов извлечения данных. Для этого был разработан алгоритм семантической сегментации веб-страницы основанный на анализе объектной модели документа. С целью сравнить влияние типа алгоритма сегментации на параметры системы также был рассмотрен алгоритм разделения веб-страницы на визуальные блоки, основанный на «компьютерном зрении».

После проведенного исследования установлено, что использование сегментации содержимого веб-страницы увеличивает производительность и надежность тех методов извлечения данных, в которых возможно реализовать функции, сигнализирующие о наличии требуемых данных в рассматриваемом блоке. Вероятность корректного извлечения атрибутов в случае системы с сегментацией на 5% выше, чем без ее использования.

При сравнении двух различных алгоритмов сегментации было установлено что тип сегментации почти не влияет на надёжность используемых методов извлечения данных, однако алгоритм визуального анализа уступает в производительности алгоритму, анализирующему только HTMLструктуру. Следовательно, в качестве основного алгоритме для выполнения сегментации содержимого веб-страниц был принят разработанный алгоритм основанный на анализе объектной модели, что дало уменьшение времени извлечения данных на 38.52% в сравнении с вариантом системы без сегментации.

С целью обеспечения долгосрочного функционирования в разработанном ПО реализованы специальные средства по восстановлению работоспособности системы в случае наиболее частых отказов. К ним относятся система внешнего слежения за состоянием рабочего процесса системы, фильтрация входных данных, модуль контроля состояния интернет соединения. Данный функционал направлен лишь на восстановление системы из состояний отказа с сохранением необходимой диагностической информации.

Было исследовано, какую долю времени функционирования разработанная система занимается полезной работой по загрузке, извлечению и анализу данных, а какую тратит на восстановление из возникающих отказов. Была предложена Марковская модель описывающая стационарное состояние системы. С помощью контрольного эксперимента было установлено, что данная модель адекватно описывает рассмотренный режим работы системы.

В результате установлено, что около 98%процентов времени функционирования система занята выполнением полезной работы и лишь оставшиеся 2% расходуются на сохранение работоспособности системы.

В дальнейшем предполагается развитие данной системы, основанное на добавлении новых модулей извлечения или анализа данных. Возможно также введение дополнительных связей между подсистемами с целью использования результатов анализа уже извлеченных данных для корректировки процесса дальнейшего извлечения данных.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Современные информационные потоки: актуальная проблематика[Электронный ресурс]. –Режим доступа: http://visti.net/~dwl/art/nti05/index1.html(датаобращения: 09.05.13)
2. Ландэ Д. В., Снарский А. А., Безсуднов И. В. Интернетика: Навигация в сложных сетях: модели и алгоритмы. — M.: Либроком (Editorial URSS), 2009. — 264 с.
3. Lalitha K. Sami Understanding the Deep Web [Electronic resource]. –Режимдоступа: http://www.webpages.uidaho.edu/~mbolin/iffat-sami.htm (датаобращения: 11.11.13)
4. Mining the Deep Web Web [Electronic resource]. –Режимдоступа: http://www.learnthenet.com/how-to/search-the-deep-web/(датаобращения: 20.11.13)
5. Web 3.0, the "official" definition. [Electronicresource]. –Режимдоступа: http://calacanis.com/2007/10/03/web-3-0-the-official-definition/(датаобращения: 21.10.13)
6. The harsh truth about HTML5′s structural semantics [Electronic resource]. –Режимдоступа: http://www.webdesignerdepot.com/2013/01/the-harsh-truth-about-html5s-structural-semantics-part-2/(датаобращения: 22.10.13)
7. Usage of content languages for websites [Electronic resource]. –Режимдоступа: http://w3techs.com/technologies/overview/content\_language/all (датаобращения: 19.10.13)
8. Extracting Information from Semi-structured Web Documents: A Framework/ NasrullahMemon, Abdul Rasool Qureshi, David Hicks, Nicholas Harkiolakis // APWeb 2008 International Workshops: BIDM, IWHDM, and DeWeb Shenyang, China. – 2008. –pp. 54-64
9. Основные проблемы при извлечении знаний из веб-ресурсов. [Электронный ресурс]. –Режимдоступа:http://habrahabr.ru/post/188078/(датаобращения: 7.12.13)
10. Extracting Content Structure for Web Pages based on Visual Representation / Deng Cai, Shipeng Yu, Ji-Rong Wen, Wei-Ying Ma // Technical Report on MSR-TR, Redmond, WA. – 2007. – 29 p.
11. Разбиение веб-страниц на семантические блоки[Электронный ресурс]. –Режимдоступа:http://habrahabr.ru/post/210824/(датаобращения: 10.02.14)
12. КосиновД. И. / Метод разбиения веб-страниц на семантические блоки с целью выявления схожих документов / Д.И. Косинов//Вестник ВГУ, Серия: Системный анализ и информационные технологии, – 2008. –Вып. 2. – С. 123-126
13. Агеев М.С., Вершинников И.С., Добров Б.В. / Извлечение значимой информации из web-страниц для задач информационного поиска. “Интернет-Математика-2005”: семинар в рамках Всеросс. науч. конф. RCDL’2005 — Яндекс, 2005. — С. 283—301.
14. Kohonen T. Self-Organizing Maps / TeuvoKohonen. Springer: Berlin – Heidelberg, 1997.
15. Заенцев, И. В. Нейронные сети: основные модели / И.В. Заенцев. – Воронеж: ВГУ, 1999. – 74 с.