Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «МЭИ»

На правах рукописи

Зейн Али Нажи

ИССЛЕДОВАНИЕ И РАЗРАБОТКА МЕТОДОВ АВТОМАТИЧЕСКОЙ КЛАСТЕРИЗАЦИИ ИНТЕРНЕТ-ПОЛЬЗОВАТЕЛЕЙ И ИНТЕРНЕТ-РЕСУРСОВ ДЛЯ ПЕРСОНАЛИЗАЦИИ ПОИСКА

Специальность 05.13.11 –
Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель
кандидат технических наук
доцент Мороховец Ю.Е.

Москва 2014

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Интернет в 21-ом веке является неотъемлемой частью повседневной жизни. Экономическая, социальная и научная деятельность человечества в той или иной степени связана с Интернет-технологиями. В наши дни можно проводить переговоры с партнёрами по бизнесу, денежные переводы, онлайн консультации, обучение и многое другое не выходя из дома. Мобильный Интернет привязал человека к виртуальному миру – в любой момент времени и в любом месте на земном шаре, имея доступ к Интернету, можно быть в курсе всего, что происходит в реальном мире.

В 2011 году, по данным исследования аналитической компании *Royal Pingdom* [65], более двух миллиардов жителей планеты пользовались Интернетом. Для подключения к Интернету достаточно иметь вычислительное (персональный компьютер) или мобильное (телефон или планшет) устройство с возможностью подключения к каналу передачи данных. Для «серфинга по просторам Интернета» пользователь обычно использует простую программу – браузер (*browser*). Указывая адрес конкретной *web*-страницы или переходя по цепочке гиперссылок, пользователь получает экранный образ *HTML*-кода требуемой веб-страницы, включающий различные визуальные компоненты (тексты, картинки, гиперссылки и т.д.), образующие «окно в мир».

Огромное количество ресурсов и содержащейся в них информации превратило всемирную паутину в грандиозное хранилище плохо организованных, неструктурированных данных. Поиск информации в сети Интернет стал уделом человечества. Средняя аудитория поисковой системы Яндекса составляет более 20000000 человек в сутки [54]. В течение суток эта поисковая система обрабатывает до 150000000 запросов, выдавая Интернет-пользователям более 10000000000000 ссылок на Интернет-ресурсы [9]. К сожалению, фактом является то, что большинство найденных ресурсов не содержат информации, отвечающей поисковым интересам пользователей. Например, если любитель природы включит

в поисковый запрос слово «ягуар», то в первых позициях поисковой выдачи будут автодиллеры, которые занимаются продажей или сервисом автомобилей марки *Jaguar*. Для владельца или потенциального покупателя автомобиля такой результат является достаточно релевантным, но обычному пользователю Интернета он вряд ли нужен. Огромное количество «мусора», выдаваемого поисковыми системами, делает актуальной проблему персонализации Интернет-поиска, адаптации поисковых систем к запросам отдельных пользователей или их групп. Мечтой становится положение, которое можно сформулировать так: «каждому пользователю свой поисковик, свой Интернет».

Интуитивно любой ИП формирует свою систему классификации и отбора веб-ресурсов для удовлетворения собственных потребностей в информации. Пользователь Интернета имеет свой личный психологический портрет и посещает конкретные, «любимые» им веб-страницы. Если говорить о поведении человека в сети Интернет, то можно выделить кратковременные (сессионные) действия ИП, которые связаны с поиском конкретной информации в течение одной или несколько поисковых сессий. Когда пользователь находит релевантную информацию, он прекращает свой поиск и даже может выйти из сети. Кроме сессионных действий пользователей можно выделить их рутинное поведение в сети, например, ежедневный утренний обзор новостей о спорте или общение в социальных сетях в обеденное время.

Крупные поисковые системы (Яндекс, *Google* и т.д.) пользуются персональной информацией и файлами *cookie* из браузеров для персонализации результатов поиска – маркетологи, например, подбирают рекламу в зависимости от поисковой истории или в зависимости от пола и возраста ИП. Удачнее всего применяется региональный или географический таргетинг – люди думают, что Яндекс действительно поумнел и сказать что, это не так, нельзя. На самом деле Яндекс хорошо работает с региональными запросами при поиске магазинов/товаров местного пользования/потребления.

Программисты работают над алгоритмами, повышающими релевантность документов запросам с помощью расчёта весов поисковых терминов, что

позволяет отбирать релевантные результаты и предпочтения пользователей. В компании Яндекс кроме лингвистического анализа контента, индекса цитирования, функции *DCG* (*Discounted cumulative gain*) [39], системы машинного обучения Матрикснет [37] и фильтров негативных признаков в число таких методов входят и различные процедуры учета и обработки первичной персональной информации. Когда пользователи выдают запросы Яндексу, примерно в 20% случаев они формулируют запросы неоднозначно [39]. Технология компании Яндекс, названная «Спектр» умеет учитывать множество неявных целей пользователей и показывать соответствующие ответы. В основе работы Спектра лежит статистика поисковых запросов ИП.

Социально-демографическая (далее соц-дем) классификация – основной метод классификации ИП после их авторизации на Интернет-сайтах – обеспечивает учет половых и возрастных различий, другой статической атрибутивной информации пользователя [10]. Соц-дем классификация на сайтах применяется, например, для таргетирования рекламных кампаний, но при этом поведение пользователей никак не применяется во внимание. Проводимая на стороне сайтов персонализация пользователей далека от совершенства, так как сайты работают по принципу «клиент всегда прав», то есть акцент делается на рекламодателе, вложившим большие денежные средства в продвижение товара – отсюда и хромают результаты поиска на стороне пользователей.

Хорошие результаты, за счет применения ассоциативных методов классификации [63, 83], достигнуты для товаров, реализуемых через Интернет-магазины. Классификация позволяет увеличивать продажу товаров, когда при покупке одного товара система предлагает приобрести сопутствующий товар или набор сопутствующих аксессуаров. Как показывает практика, покупатели достаточно часто приобретают несколько товаров из одной классификационной группы. Однако неизвестно, на сколько удачно можно применять ассоциативные методы для классификации ИП и ИР с целью персонализации Интернет-поиска?

В последние годы в информационных источниках можно встретить общие сведения о применении методов кластеризации для классификации ИП и ИР.

Декларируются различные цели применения методов кластерного анализа к Интернет-объектам, однако в подавляющем большинстве случаев детали этих методов и способов их применения не разглашаются. Так в работах [26, 38] отмечается, что для кластеризации текстовой информации могут использоваться методы *TF* и *TIDF*, а также их модификации. Эти методы действительно подходят для кластеризации текстов газет, учебников, научных статей и других информационных ресурсов со статичным содержанием. В своей работе [28] Куралёнок И.Е. упомянул, что векторные и вероятносные модели, которые применяются поисковыми системами показывающие хорошие результаты на одних данных, оказываются много хуже тех же классических моделей иа других данных. Можно ли с их помощью добиться приемлемых результатов для кластеризации ИП и, в особенности, для кластеризации современных высоко динамических ИР остается неизвестным.

Актуальность темы исследования.

Приведённые аргументы свидетельствуют о необходимости дальнейшего приспособления Интернета к нуждам пользователей и, в частности, за счет персонализации Интернет-поиска. Повышение уровня персонализации поиска, в свою очередь, может быть достигнуто за счет разработки перспективных методов классификации ИП и ИР, основанных на кластерном анализе, внедрения этих методов в существующие поисковые системы.

Степень разработанности проблемы.

Проблема: отсутствие эффективных методов и средств, обеспечивающих персонализацию поиска информации в Интернете.

О персонализации поиска жаркие дискуссии идут уже почти 20 лет – все заинтересованы в том, чтобы результаты поиска в Интернете были как можно более релевантными пользовательским запросам. Однако недостаточная научная проработанность проблемы, закрытость большинства практически реализованных решений ведущими компаниями поставщиками Интернет-услуг обусловила необходимость исследования теоретических и практических вопросов применения методов кластерного анализа для персонализации поиска.

По теме кластерного анализа существует обширная литература. Она охватывает общие вопросы математического описания объектов и алгоритмы их кластеризации. Кластеризация объектов со статическими свойствами широко применяется повседневно в основном в аналитической деятельности. Однако, методы кластеризации динамических объектов, таких как ИР, недостаточно разработаны и, кроме того, мало кто из исследователей рассматривал идею обобщенного представления объектов разной природы, обладающих подобными свойствами.

Цели и задачи исследования.

Целью диссертационной работы является применение методов классического кластерного анализа для классификации ИП и ИР, для персонализации информационного поиска в Интернете. Для достижения поставленной цели требуется решить следующие основные задачи.

1. Проанализировать существующие некластерные методы классификации ИП и ИР. Проанализировать существующие методы кластерного анализа ИП и ИР, показать их преимущество по сравнению с некластерными методами.

2. Предложить адекватное математическое описание объектов исследования – ИП и ИР, обеспечивающее применение существующих алгоритмов кластеризации.

3. Выбрать алгоритм кластеризации ИП и ИР из числа известных методов кластерного анализа, позволяющий управлять результатом с помощью входных параметров.

4. Определить масштаб влияния информационной динамики Интернет-объектов на результаты их кластерного анализа. Предложить методы устранения динамических факторов при кластеризации ИП и ИР.

5. Разработать и применить оригинальный подход, основанный на принципе обобщения и одновременной кластерной обработки ИП и ИР.

6. Разработать программные средства для наблюдения за активностью ИП и сбора данных о страницах ИР, а также кластеризации ИП и ИР, оценки эффективности предлагаемых методов.

7. Оценить эффективность применения предлагаемых методов для персонализации Интернет-поиска, с точки зрения релевантности получаемых данных.

8. Разработать корпоративную систему персонализации поиска (КСПП) предприятия, использующую предлагаемые методы классификации ИП и ИР. Реализовать КСПП как виртуальную программную систему, позволяющую, в том числе, провести сравнительную оценку предлагаемых методов.

Научная новизна.

В диссертации представлены оригинальные методы, направленные на решение проблемы персонализации и повышения качества результатов поиска в Интернете. Эти методы позволяют использовать существующие классические алгоритмы кластерного анализа для Интернет-объектов – ИП и ИР – с учетом особенностей их математического описания. Для математического описания ИП и ИР предложено использовать характеристические вектора, числовые координаты, которых расположены в том же порядке, что и термины в глобальном словаре терминов поисковой системы. Характеристические вектора строятся на основе данных, полученных в результате сбора уникальных терминов как на стороне ИП, так и на стороне ИР. Переход от вербальных данных к числовому представлению координат векторов происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений в наблюдаемый текст. После векторизации Интернет-объектов происходит расчёт меры близости между ними и в конечном итоге формируются кластеры с использованием одного из известных алгоритмов.

Как результат натурных экспериментов проведен выбор алгоритма кластеризации ИП и ИР, обеспечивающего наилучшие показатели кластерной структуры – им оказался алгоритм *k*-средних.

Сама по себе задача персонализации поиска достаточно старая и в коммерческих целях уже широко применяются не кластерные методы классификации ИП, основанные на статической информации, и ассоциативные методы классификации ИР. Однако эти методы не учитывают интересы ИП и качество классификации ИР, оставляют желать лучшего. Существующие методы

кластеризации текстов не берут во внимание особенности современных ИР: не учитываются динамические компоненты *DOM*-моделей ИР. Наконец, задача поиска оптимального подхода к кластеризации должна учитывать, как поведение ИП, так и динамику ИР. Следует обратить внимание на тот факт, что кластеризация ИП и ИР сейчас проводится раздельно. Предложенный в диссертации метод обеспечивает совершенно новый подход и даёт новую математическую модель обобщения ИП и ИР как единого объекта исследования. Изложенный метод может быть применён не только для персонализации поиска в Интернете, но и для решения широкого круга задач, где имеется взаимодействие человека с множеством подобных объектов, которые необходимо классифицировать в соответствии с его предпочтениями.

Теоритическая и практическая значимость полученных результатов.

В диссертации разработаны и программно реализованы методы, обеспечивающие выполнение кластерного анализа для персонализации поиска в Интернете. Программная реализация указанных методов осуществлена в виде виртуальной корпоративной системы персонализации поиска – КСПП. Одна часть программных средств, поддерживающих предлагаемые методы, реализована на языке *C#* в среде разработки *Microsoft Visual Studio* 2010 в виде соответствующего инструментария и набора инфоботов, позволяющих запускать и выполнять задания по получению текстового содержания ИР и сканирования их *DOM*-моделей, а также отслеживать поисковую активность ИП. Другая часть программных средств реализована на языке *T-SQL* в среде *Microsoft SQL Server* 2012. Этими средствами поддерживается вся аналитическая часть проекта, выполняется кластеризация Интернет-объектов. Используя указанные инструменты, эксперт-аналитик на основе результатов кластерного анализа получает чёткую картину о распределении ИП и ИР по кластерам в зависимости от нескольких входных параметров: продолжительности периода наблюдения за активностью ИП, числа кластеров, значений коэффициентов усиления и минимальной длины терминов. При достаточно низком (<40%) показателе коэффициента попадания в целевую группу он (эксперт) может принять решение

о целесообразности выполнения кластеризации с новыми входными параметрами. В случае, когда указанный показатель становится чересчур высоким (>60%), эксперт может зафиксировать входные параметры и запустить в автоматическом режиме кластеризацию объектов на более длительный период. Предложенный подход безусловно требует значительных вычислительных затрат, но при наличии локального дата-центра или корпоративного грида может дать существенную отдачу, повысив уровень персонализации Интернет-поиска. Таким образом, в диссертационной работе наряду с указанными выше целевыми методами, предложен целостный подход к их практическому использованию.

Объект исследования.

Объектом исследования являются методы персонализации Интернет-поиска, основанные на изучении и классификации ИП и ИР при помощи кластерного анализа.

Предмет исследования.

Предметом исследования являются способы математического описания ИП и ИР, процедуры сбора и обработки информации об этих Интернет-объектах, позволяющие эффективно применять аппарат классического кластерного анализа для целей персонификации Интернет-поиска.

Методы исследования.

В основе диссертационного исследования лежат методы статистического и кластерного анализа, теория графов, *web mining* и *web*-технологии. При проведении исследований и при изложении полученных материалов применяется систематический подход, базирующийся на анализе экспериментальных результатов. На каждом этапе работы, после проведения сравнительного анализа полученных результатов делаются выводы и выбираются наиболее рациональные подходы для продолжения исследований.

Положения, выносимые на защиту.

1. Метод снижения влияния динамических элементов *DOM*-модели ИР, основанный на применения числовых коэффициентов усиления. Анализ состояния кластерной структуры с помощью степени принадлежности объектов к

кластерам.

2. Метод трёхтактной кластеризации ИР с обратной связью, основанный на выявлении динамических элементов *DOM*-модели с последующим исключением их контента из кластерного анализа.

3. Математическое представление объектов исследования и применение метода обобщённых объектов: обобщённый словарь терминов, обобщённый характеристический вектор и обобщённая кластеризация.

4. Метод структуризации содержания ИР, определяющий структуру базы данных, содержащей информацию о поисковой активности ИП и текстовом контенте ИР.

Степень достоверности и апробация результатов.

Основные результаты диссертационного использования представлены на научно-практических конференций, среди которых: VI international research and practice conference «European Science and Technology» (Munich, 2013), IV international research and practice conference «Science, Technology and Higher Education» (Westwood, 2014), международная научно-практической конференция (Уфа, 2014) и международная научно-техническая конференции «Тенденции и инновации современной науки» (Краснодар, 2014).

Положения и результаты диссертационной работы использовались в производственной деятельности компании «ЗАО ТНС Гэллап Эдфакт» при обработке нестандартных рекламных баннеров с последующей кластеризации динамических элементов, принадлежащие одному баннеру, но полученные в разные моменты времени, что подтверждается актом о внедрении.

Область применения разработанных методов.

Предложенные методы могут быть применены для исследования и разработки поисковых систем общего и специального назначения, имеющих высокий уровень персонализации поиска. Примерами таких систем могут являться социальная поисковая система, работающая на уровне узкоспециализированных групп пользователей, или корпоративная система персонализации поиска, формирующая поисковый результат в зависимости от

поисковой направленности отделов предприятия – бухгалтерии, финансового отдела, отдела маркетинга и т.д.

Следует отметить, что разработанные в диссертации методы имеют определённые ограничения по применению. Очевидно, что их нецелесообразно применять в условиях, когда одним компьютером (браузером) не пользуется более одного человека, так как в этом случае необходима настройка автоматической очистки истории поиска и *cookie* в самом браузере. Каждый ИП имеет свои поисковые интересы и ведёт свой образ жизни в киберпространстве, поэтому разным ИП свойственны разные интересы и, как следствие, они могут попадать в разные кластеры.

Результаты, полученные автором.

В рамках диссертации лично автором получены следующие основные результаты:

1. Предложена и реализована процедура лингвистической обработки текстов, основанная на использовании двухуровневого словарь терминов и лемм с возможностью применения открытых словарей. При необходимости предусмотрена возможность обращения к «лингвистическому эксперту» для лемматизации новых, не включённых в словарь терминов.

2. Для достижения стабильности кластерной структуры и устранения динамического эффекта, разработан метод наблюдения за ИП, основанный на применения временного окна. С этой же целью для наблюдения за ИР разработан метод анализа содержания (сканирования) *DOM*-модели ресурса с последующим применением числовых коэффициентов усиления.

3. С целью выявления и фильтрации динамических компонентов *DOM*-модели предложена трёхтактная схема кластеризации ИР с обратной связью. Реализация схемы позволяет превращать динамичные ИР в статические ИР, и применять к последним стандартные алгоритмы кластерного анализа.

4. Предложено решение задачи формирования характеристических векторов ИП и ИР, числовые координаты которых, расположены в порядке, соответствующем лексикографическому порядку следования термином в

глобальном словаре системы. Переход от вербального к числовому представлению координат происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений в текст поисковых запросов или текстовый контент статических компонентов *DOM*-модели ИР.

5. Введено понятие обобщённого Интернет-объекта, применение которого позволяет одновременно проводить кластерный анализ как ИП, так и ИР. Унификация объектов наблюдения делает кластерный анализ более транспарентным и глобальным. Информация об ИП, в зависимости от их интересов, может храниться вместе с информационно-релевантными ей данными об ИР.

6. Разработан набор программных модулей (программная система) для слежения за активностью ИП и получения текстового содержания ИР с учётом их *DOM*-модели.

7. В среде *MS SQL Server* 2012 разработаны специальные хранимые процедуры, выполняющие все необходимые расчёты – от формирования словарей терминов до конечного распределения объектов по кластерам.

Публикации в журналах ВАК.

1. Зейн А.Н., Мороховец Ю.Е. Персонализация поиска: статическая или динамическая кластеризация? // Журнал «Вестник МЭИ». – М.: Издательство МЭИ. – 2014. – № 2. – С. 76-81.

2. Мороховец Ю.Е., Зейн А.Н. Трехтактная кластеризация динамичных Интернет-ресурсов с применением *DOM*-моделей. // Международный журнал «Программные продукты и системы». – Тверь: НИИ Центрпрограммсистем. – 2014. – № 3. – С. 58-63. URL: http://swsys.ru/index.php?page=article&id=3861 (01.12.2014 г.).

Другие публикации.

1. Зейн А.Н. Статические и динамические явления в кластерной структуре Интернет-объектов. // Сборник научных трудов «Новый взгляд. Международный научный вестник». Выпуск 2. – Новосибирск: ЦРНС. – 2013. – С. 51-60.

2. Zein A. N. Clusterization of web-sites using numeric coefficients based on

DOM-model. // Materials of the VI international research and practice conference «European Science and Technology». Vol. 2. – Munich: Vela Verlag Waldkraiburg, 2013. – PP. 372-375.

3. Зейн А.Н. Динамическая активность Интернет-ресурсов в кластерной структуре. // Сборник статей «Международной научно-практической конференции». – Уфа: РИЦ БашГУ. – 2014. – С. 123-127.

4. Зейн А.Н. Интернет-ресурсы: новый подход для оптимизации результатов поиска. // Материалы XII международной научно-технической конференции «Тенденции и инновации современной науки». – Краснодар: Априори. – 2014. – С. 54.

5. Зейн А.Н. Персонализация поиска: кластеризация Интернет-пользователей и Интернет-ресурсов. // Электронный журнал «Вычислительные сети: теория и практика». – М.: НИУ МЭИ. – 2014. – №1. URL: http://network-journal.mpei.ac.ru/cgi-bin/main.pl?l=ru&n=24&pa=6&ar=1 (01.12.2014 г.).

Объём и структура диссертации.

Общий объём диссертации – 303 страницы, из которых 181 страница основного текста. Диссертация содержит 76 рисунков, 27 таблиц и состоит из введения, пяти глав, заключения, списка литературы и приложений.

Первая глава посвящена обзору существующих подходов и методов как некластерной, так и кластерной классификации ИП и ИР. Эти методы широко применяются во многих отраслях, в том числе и в Интернет-индустрии.

Вторая глава посвящена лингвистической обработки терминов из запросов Интернет-пользователей и текстов Интернет-ресурсов с применением специальных динамических словарей, формируемых в процессе лемматизации терминов.

В третьей главе предлагаются два новых метода, снижающих влияние динамических компонентов на стабильность кластерной структуры – метод, базирующийся на применении числовых коэффициентов усиления, и метод трёхтактной кластеризации Интернет-ресурсов с фильтрацией, основанной на анализе *DOM*-моделей.

В четвертой главе предлагается подход к обобщению Интернет-объектов на базе вводимого здесь же понятия обобщённого характеристического вектора. Результаты интерпретируются с использованием графовой модели. Применение обобщения делает Интернет-ресурсы более социальными: ресурсы, ассоциированные с одними пользователями, могут предлагаться другим пользователям, относящимся к одному и тому же кластеру.

Пятая глава посвящена разработке структуры поисковой системы, использующей предлагаемые в диссертации методы классификации ИП и ИР и программной реализации предложенных методов.

В заключении приводится перечень основных результатов работы, показываются направления её развития.

1. АНАЛИЗ СУЩЕСТВУЮЩИХ МЕТОДОВ КЛАССИФИКАЦИИ ИНТЕРНЕТ-ПОЛЬЗОВАТЕЛЕЙ И ИНТЕРНЕТ-РЕСУРСОВ, ПРИМЕНЯЕМЫХ ДЛЯ ПЕРСОНАЛИЗАЦИИ ПОИСКА

При попытке получения знаний из *web*-а мы не можем ориентироваться на строгие структуры и компоненты, так как в Интернете присутствует огромное количество распределённой, гетерогенной, неструктурированной и динамически изменяющейся информации. Несмотря на это, ИР научились быть ближе к ИП, перестали быть изолированными от них. Как только ИП заходит на ИР, он сразу оставляет свой след: становятся известны его местоположение (география), персональные данные (пол, возраст и т.д.), его история поиска. С учетом этого, в первой главе диссертации дан обзор существующих подходов и методов классификации ИП и ИР, которые широко применяются для персонализации поиска в Интернете.

1.1. Примеры использования информации о пользователях и их активности в социальных сетях для решения задач персонализации

В настоящее время персональная информация ИП представляет огромный интерес, как для Интернет-площадок, так и для рекламодателей. Дело в том, что любой ИР заинтересован в обработке личной информации ИП, посещавших его страницы. Это важно для статистической обработки посещаемости с целью продажи рекламы. Можно чётко разделить мужские и женские сайты, спортивные или новостные сайты. Для примера, возьмём один из крупных Интернет-порталов России – *mail.ru*. По данным исследовательской компании *TNS* Россия, количество пользователей за апрель 2012 года по всему порталу *mail.ru* составило примерно 47 миллионов российских пользователей [82], а на главной странице *mail.ru* их было примерно 12 миллионов за тот же период.

Начнём с использования персональной информации пользователя компанией *mail.ru* для персонализации контента. Работа пользователя начинается

с регистрации почтового ящика (рисунок 1.1) на главной регистрационной страницы сайта (*http://e.mail.ru/cgi-bin/signup*). На этой странице пользователь оставляет ценнейшую информацию о себе: дату рождения, пол, город и страну проживания.

Рисунок 1.1 – Регистрационная форма для создания почтового ящика *mail.ru*

После того, как ИП оставит персональную информацию при регистрации нового почтового ящика (рисунок 1.1), эта информация становится доступной большому числу специалистов (программистам, маркетологам и т.д). С этого момента начинают работать различные алгоритмы для персонализации информационного Интернет-потока с портала *mail.ru*. Достаточно понаблюдать за баннерной рекламой на главной странице *mail.ru* после авторизации пользователя. Итак, сразу после заполнения регистрационной формы персональной информацией, ИР начинает её использовать для подбора рекламы. Например, пользователю-мужчине старше восемнадцати лет ИР показывает рекламу пивной продукции (рисунок 1.2 и 1.3. Скриншоты были получены 7 мая 2012 года, т.е. до появления изменений к требованиям Федерального закона о рекламе, вступивших в силу с 1 сентября 2012 г.)

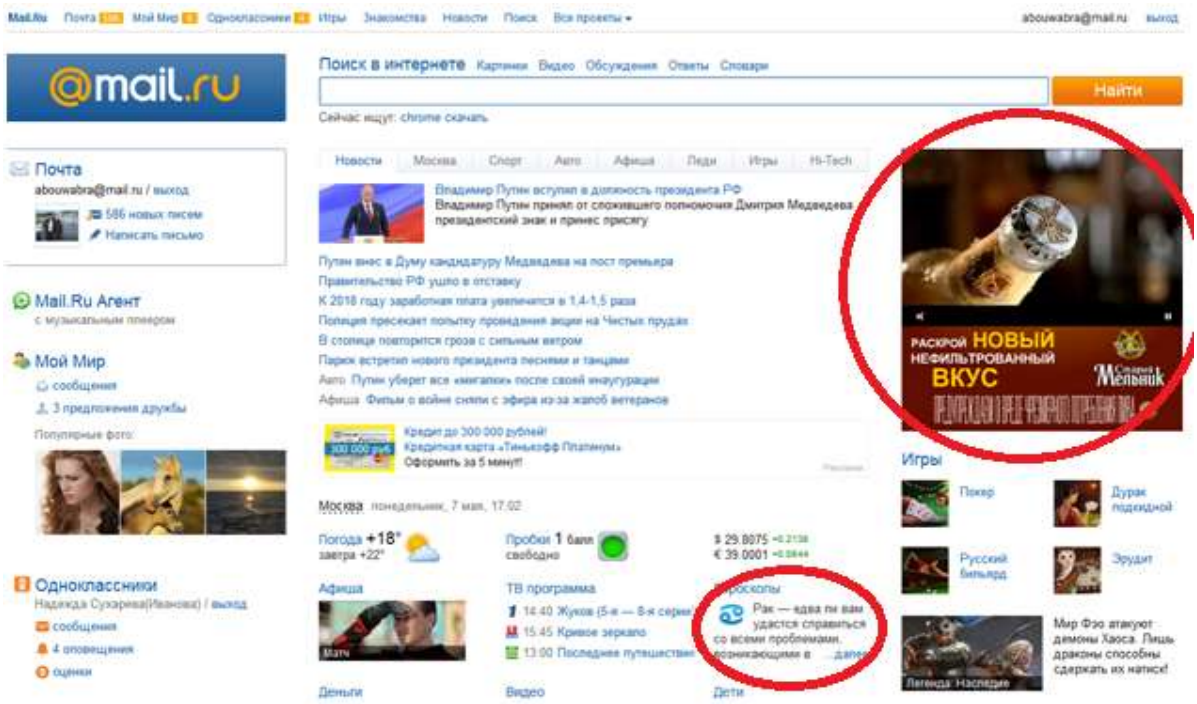

Рисунок 1.2 – Пиво «Старый мельник» таргетированная баннерная реклама

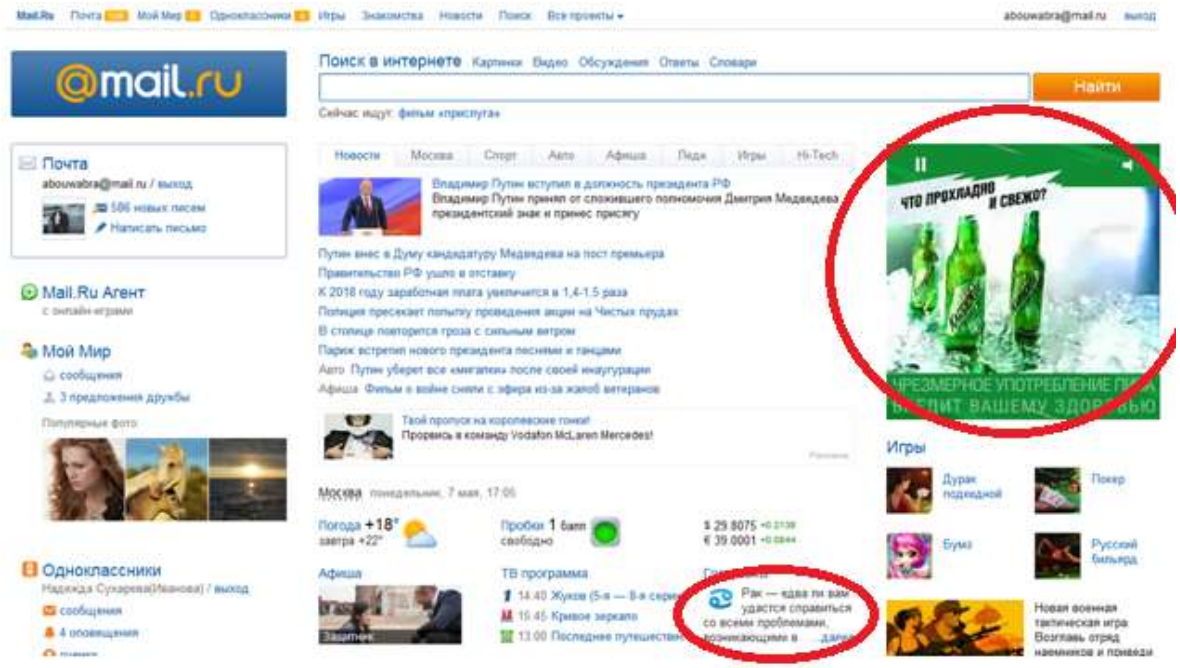

Рисунок 1.3 – Пиво «Клинское» таргетированная баннерная реклама

На рисунках 1.2 и 1.3 можно обратить внимание на персонализацию рекламы пользователя, который заполнил регистрационную форму: дата рождения 29 июня 1984 г. Также был подобран соответствующий гороскоп. Если очистить файлы *cookie* и зайти повторно на тот же ИР без авторизации, то отсутствие персонализации ИР сразу становится заметно (рисунок 1.4).

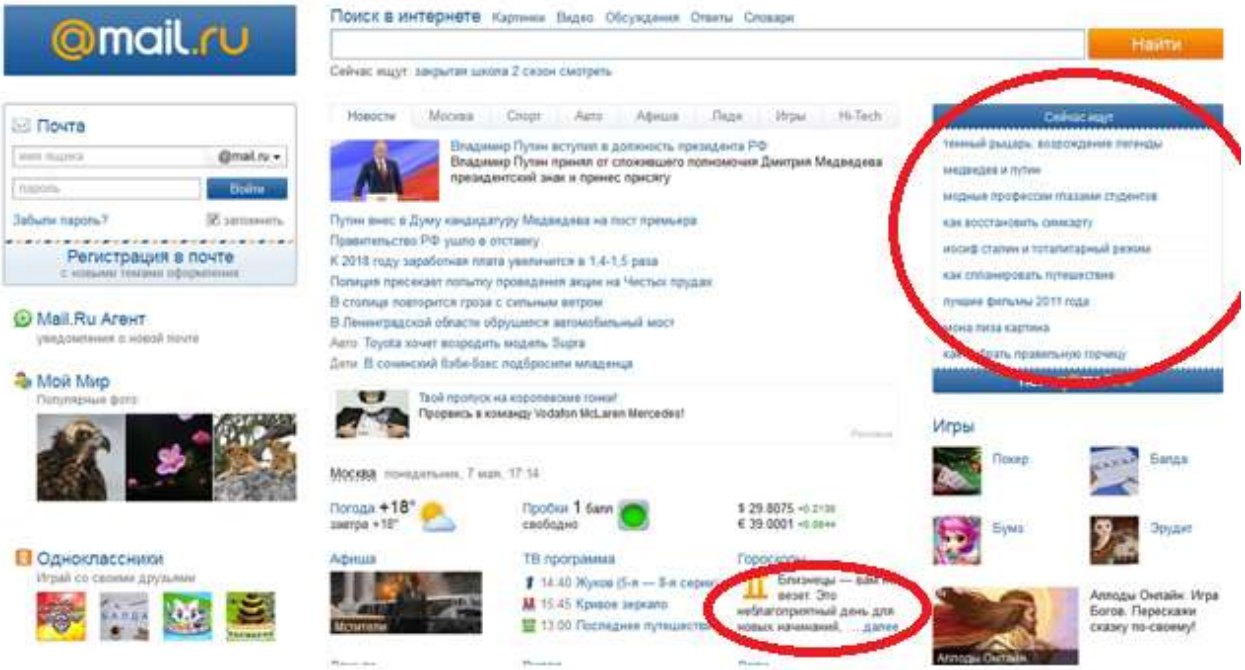

Рисунок 1.4 – Главная страница *mail.ru* для неавторизованного пользователя

Приведённый пример показывает один из подходов автоматической классификации авторизованных пользователей для персонализации контента ресурса. Алкогольная продукция (в том числе и пиво), а также табачные изделия не будут показываться несовершеннолетним. Кроме рекламы, можно обратить внимание на гороскопы, которые чётко соответствуют личной информации ИП. На рисунках 1.2 и 1.3 автоматически проставляется гороскоп, в соответствии с датой рождения авторизованного ИП (29 июня – знак зодиака Рак). Для неавторизованного пользователя (рисунок 1.4) гороскопическая информация выводится случайным образом.

Если остановиться на представленном подходе к персонализации контента, то можно заметить, что применяемый алгоритм достаточно примитивен – персональная информация ИП используется лишь для таргетирования Интернет-рекламы и составления гороскопов.

Человек испокон веков жил в социальной среде и поэтому по своему поведению является социально зависимым субъектом. Его активность распространялась на семью, друзей и соратников, работу и другие жизненные сферы. Сейчас, в век телекоммуникаций и компьютерных технологий, социальная активность человека перешла в киберпространство. С появлением социальных

сетей (*Facebook*, Одноклассники, ВКонтакте и др.), б*о*льшая часть социально активных людей перешла к общению в виртуальном мире. Люди смогли найти своих коллег, одноклассников и друзей, с которыми давно была потеряна связь. Увеличение числа людей с одинаковыми интересами и взглядами привело к рождению идеи создания специализированных групп (например: *http://vk.com/club*2545214 группа «выпускники МЭИ»).

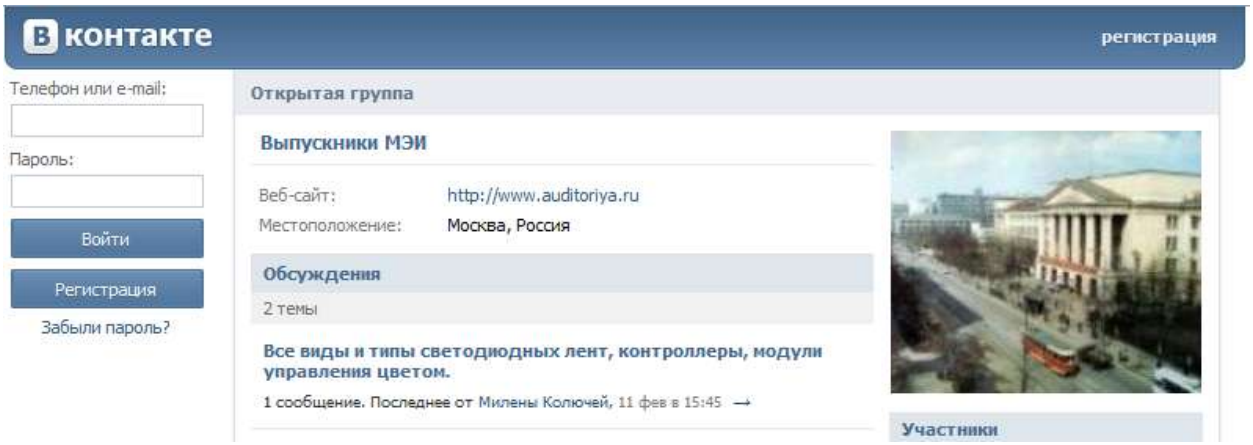

Рисунок 1.5 – Группа «выпускники МЭИ» в социальной сети «ВКонтакте»

В рамках одной и той же специализированной социальной группы участники обсуждают конкретные темы и проводят онлайн беседы с людьми, имеющими одинаково направленное мышление. Практика показывает, что в социальных сетях и в блогах информация распространяется гораздо быстрее, чем на новостных сайтах. В социальных сетях человек оставляет гораздо больше информации о себе и о своём поведении, чем на каком-либо другом Интернет-ресурсе: школа, институт, работа и конечно комментарии. Стоит отметить очень важный момент, присутствующий в социальных сетях и связанный с так называемыми *like*-ами: авторизованный пользователь может ставить оценки другим пользователям, их статьям, комментариям, фотографиям и т.д., отметив понравившееся *like*-ом.

Для моментального отражения статуса пользователей и их поведения в рамках группы применяются нереляционные базы данных, работающие с высокоскоростной оперативной памятью. Это так называемые *in memory database* или базы данных в памяти. Основная идея таких систем – хранение данных не на

дисковом накопителе, а прямо в памяти. Применение такого рода БД уменьшает время отклика системы и позволяет практически моментально переключаться по группам интересов пользователей социальных сетей.

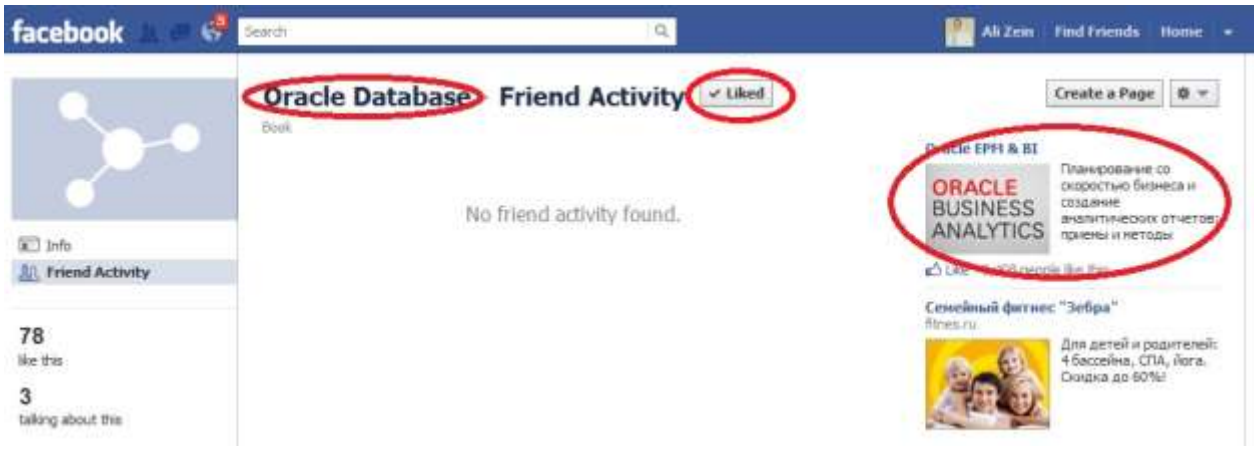

Рисунок 1.6 – Использование *like*-ов для показа рекламы в социальной сети.

Социальные группы являются хорошим примером для классификации ИП. Дело в том, что люди формируют группы интересов, могут пользоваться общими ресурсами и делиться опытом в том или ином направлении. При этом любая сформированная группа будет достаточно специализированной и может быть легко индексирована для быстрого поиска в области интересов пользователей.

1.2. Методы некластерной классификации Интернет-пользователей и Интернет-ресурсов

Ассоциативный метод.

Ассоциативный метод классификации широко применяется в Интернет-магазинах, когда содержание покупных корзин определенного множества покупателей анализируется и образуется некая вероятностная закономерность покупок.

Проведём анализ релевантности между элементами вектора с помощью ассоциативных правил. В БД Интернет-запросов от 20 декабря 2012 г. случайно выбранные пять ИП выполняли поиск товара на сайте *market.yandex.ru*. Представим таблицу поиска с векторами поиска, состоящими из множества товаров {фотоаппарат, регистратор, навигатор, память}. Поисковые множества,

представленные в таблице 1.1, были получены в условиях, когда время между поисковыми запросами не превышало четырех часов.

Таблица 1.1 – Таблица транзакций поиска товаров ИП

TID	**Транзакции**
1	{фотоаппарат, регистратор}
2	{фотоаппарат, навигатор, память}
3	{навигатор, память}
4	{регистратор}
5	{фотоаппарат, память}

Для начала необходимо распределить элементы {фотоаппарат, регистратор, навигатор, память} в промежуточную таблицу попаданий (таблица 1.2).

Таблица 1.2 – Таблица попаданий

Набор элементов	**Транзакции**	**Число попаданий**
{}	{1,2,3,4,5}	5
{фотоаппарат}	{1,2,5}	3
{регистратор}	{1,4}	2
{навигатор}	{2,3}	2
{память}	{2,3,5}	3
{фотоаппарат, регистратор}	{1}	1
{фотоаппарат, навигатор }	{2}	1
{фотоаппарат, память}	{2,5}	2
{навигатор, память}	{2,3}	2
{фотоаппарат, навигатор, память}	{2}	1

Столбец «Набор элементов» формируется с помощью отдельных элементов и возможных комбинаций этих элементов, в соответствии с реальными результатами таблицы транзакций (таблица 1.1). Столбец «Транзакции» формируется с помощью набора транзакций, в котором присутствовала комбинация элементов в *i*-ой строке. Значение столбца «Число попаданий» формируется на основании числа элементов столбца «Транзакции».

Теперь можно построить ассоциативную таблицу элементов (таблица 1.3):

Таблица 1.3 – Ассоциативная таблица элементов

Ассоциативный набор	**Число попаданий**	**Процент вероятности**
{фотоаппарат} → {память}	2	2/3 = 67%
{регистратор} → {фотоаппарат}	1	1/2 = 50%

{навигатор} → {фотоаппарат}	1	1/2 = 50%
{навигатор} → {память}	2	2/2 = 100%
{память} → {фотоаппарат}	2	2/3 = 67%
{память} → {навигатор}	2	2/3 = 67%
{фотоаппарат, навигатор} → {память}	1	1/1 = 100%
{фотоаппарат, память} → {навигатор}	1	1/2 = 50%
{навигатор, память} → {фотоаппарат}	1	1/2 = 50%
{навигатор} → {фотоаппарат, память}	1	1/2 = 50%

По ассоциативной таблице элементов проводится расчет вероятности появления события {память}, если событие {фотоаппарат} имело место и т.д. На примере {фотоаппарат} → {память} в числителе будет находиться число случаев, когда в транзакции присутствуют оба элемента {фотоаппарат} и {память}: это вторая и пятая транзакция в таблице 1.1. В знаменателе будет число случаев, когда в транзакции только присутствует элемент {фотоаппарат}. Таким образом, в числителе будет *count*({фотоаппарат, навигатор, память}, {фотоаппарат, память}) = 2, в знаменателе будет *count*({фотоаппарат, регистратор}, {фотоаппарат, навигатор, память}, {фотоаппарат, память}) = 3. Отсюда, вероятность появления события {память}, если наступило событие {фотоаппарат} будет равна $P(\{фотоаппарат\} - - > \{память\}) = \frac{2}{3} \times 100 = 67\%$.

Метод пересечений.

Метод пересечений (*Éclat mining*) основывается на пересечении элементов на разных транзакциях. Вернёмся к таблице попаданий (таблица 1.2), и из этой таблицы выберем одиночные наборы элементов, формируя таблицу одиночных наборов элементов (таблица 1.4) {фотоаппарат}, {регистратор}, {навигатор} и {память}.

Таблица 1.4. – Таблица одиночных наборов элементов

№№	{фотоаппарат}	{регистратор}	{навигатор}	{память}
1	+	+	-	-
2	+	-	+	+
3	-	-	+	+
4	-	+	-	-
5	+	-	-	+

Из таблицы одиночных наборов элементов можно составить таблицу

двойных наборов элементов (таблица 1.5).

Таблица 1.5 – Таблица двойных наборов элементов

№№	{фотоаппарат, регистратор}	{фотоаппарат, навигатор}	{фотоаппарат, память}	{регистратор, навигатор}	{регистратор, память}	{навигатор, память}
1	+	-	-	-	-	-
2	-	+	+	-	-	+
3	-	-	-	-	-	+
4	-	-	-	-	-	-
5	-	-	-	-	-	-

На этапе формирования таблицы двойных наборов элементов происходит выпадение конкретных комбинаций: выпадают {регистратор, навигатор} и {регистратор, память}. Для расчёта вероятности, необходимо сначала определиться с порядком следования элементов в формируемых парах – {фотоаппарат, регистратор} или {регистратор, фотоаппарат}, т.к. результат расчёта вероятностей будет разным:

$$P(\{фотоаппарат\} - - > \{регистратор\}) = \frac{1}{3} \times 100 = 63\%$$

$$P(\{регистратор\} - - > \{фотоаппарат\}) = \frac{1}{2} \times 100 = 50\%$$

Из таблицы двойных наборов элементов можно составить таблицу тройных наборов элементов (таблица 1.6).

Таблица 1.6 – Таблица тройных наборов элементов

№№	{фотоаппарат, регистратор, навигатор}	{фотоаппарат, регистратор, память}	{фотоаппарат, навигатор, память}
1	-	-	-
2	-	-	+
3	-	-	-
4	-	-	-
5	-	-	-

На этапе формирования таблицы тройных наборов элементов происходит

выпадение конкретных комбинаций: выпадают комбинации {фотоаппарат, регистратор, навигатор} и {фотоаппарат, регистратор, память}. Для расчёта вероятности необходимо также определиться с порядком следования элементов в тройках.

Для выполнения алгоритма необходимо выполнить перерасчёт всех возможных комбинаций (рисунок 1.7)

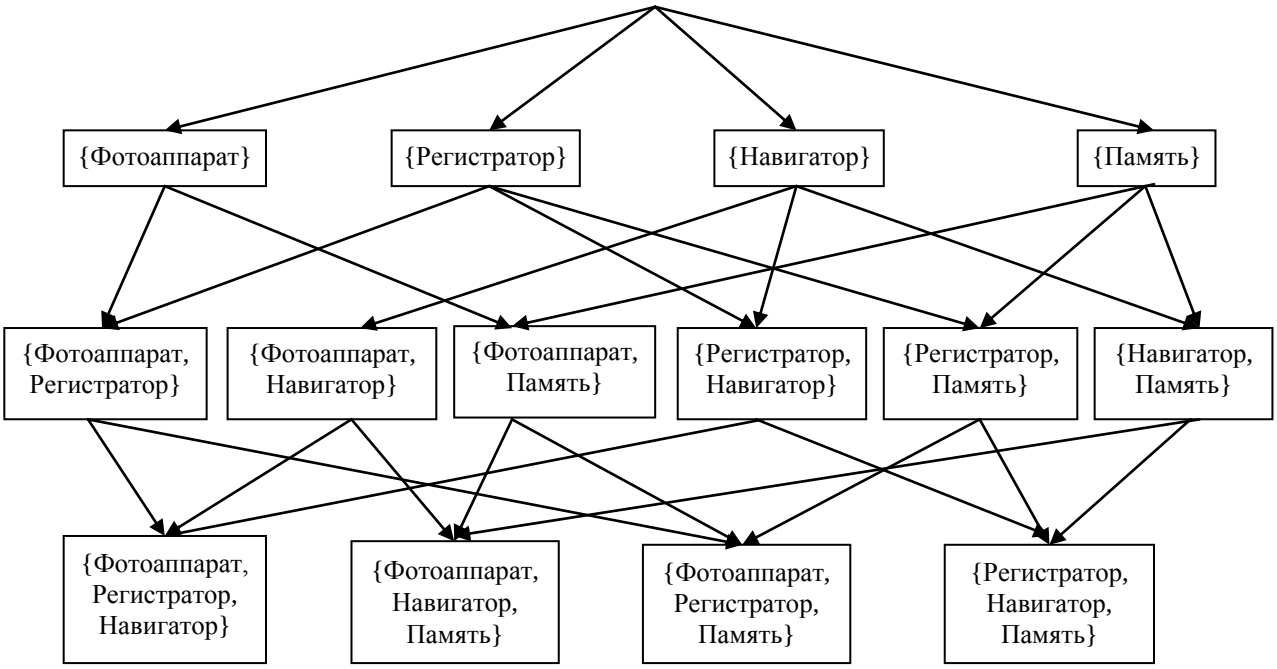

Рисунок 1.7 – Схема всех возможных комбинаций.

С помощью схемы всех возможных комбинаций можно рассчитать вероятности появления любой комбинации элементов.

Ассоциативный метод и метод пересечений широко применяются для подбора сопутствующего товара. Большое количество покупок групп товаров делает результат расчётов вероятностей более точными, так как ИП показывают огромное количество товаров, которое покупалось одновременно с просматриваемым товаром.

Однако эти методы плохо масштабируются и, как только происходит выход за рамки транзакции, результаты расчёта вероятностей портятся. Дело в том, что поисковые интересы ИП не возможно разделить на отдельные транзакции и самые популярные поисковые термины встречаются у большинства ИП в рамках периода наблюдения $\Delta t = 4$ часа. Эти методы не учитывают веса терминов.

Метод частоты терминов.

Метод частоты терминов (*TF – Term Frequency*) широко применяется для классификации обычных текстов. Этот метод можно применить и для классификации ИП по их поисковой истории и ИР по их текстовому содержанию.

Пусть исходными данными для метода являются поисковые термины ИП в период с 13 по 19 мая 2013 г. Пусть *USR* – множество наблюдаемых ИП и usr_i – *i*-ый наблюдаемый ИП. Каждый usr_i, $usr_i \in USR$, по оставленным им персональными данными может быть распределён в одну из десяти социально-демографических групп $USR_i \in \{USR_1, ..., USR_{10}\}$ таких, что $\cup USR_i = USR$ и $USR_i \cap USR_k = \varnothing$, $i \neq k$. Эти группы используются для таргетирования рекламы ИР, в которой помимо пола принимается во внимание возрастная категория Интернет-пользователя (таблица 1.7).

После построения базы данных для структуризации запросов ИП и содержания ИР применение *TF*-метода (приложение 1) не составляет никакого труда.

С учётом имеющейся в наличии вычислительной мощности, при формировании статистических таблиц (таблицы 1.8 и 1.9) было принято решение об ограничении числа терминов *nof*(*V*) в глобальном словаре терминов *V* и их длины. В [52] указана средняя длина русских слов равная 5,28 символов. Однако в поисковой истории ИП оказалось большое число терминов, длина которых равна 4, поэтому необходимо рассматривать термины, длина которых больше или равна 4.

Таблица 1.7 – Таблица соц-дем классификации

Соц-дем	**Пол**	**Возраст**
USR_1	МУЖ	от 12 до 17
USR_2	МУЖ	от 18 до 24
USR_3	МУЖ	от 25 до 34
USR_4	МУЖ	от 35 до 44
USR_5	МУЖ	от 45 до 55
USR_6	ЖЕН	от 12 до 17
USR_7	ЖЕН	от 18 до 24
USR_8	ЖЕН	от 25 до 34

USR_9	ЖЕН	от 35 до 44
USR_{10}	ЖЕН	от 45 до 55

Таким образом, каждая социально-демографическая группа USR_i может быть представлена числовым вектором $f_i = (f_{i,1}, \ldots, f_{i,j}, \ldots, f_{i,nof(Vu)})$ размером $nof(V_u)$, где $f_{i,j}$ – вес j-ого поискового термина из глобального словаря терминов V_u. Числовые координаты $f_{i,j}$, $1 \le j \le nof(V_u)$ расположены в характеристическом векторе f_i, в том же порядке, что и термины в глобальном словаре V_u.

Переход от вербального к числовому представлению результатов исследования отдельно взятой социально-демографической группы USR_i в виде характеристического вектора происходит за счет позиционного кодирования терминов словаря, подсчёта числа их вхождений в запросы ИП группы в течение всей поисковой истории и расчета частоты употребления этих терминов (*TF*-значений) в группе.

Для расчета *TF*-значений применяется следующая формула:

$$f_{ij} = \frac{nof(v_{i,j})}{\sum_{j=1}^{nof(Vu)} nof(v_{i,j})}, \qquad (1.1)$$

где $nof(v_{i,j})$, $v_{i,j} \in V_u$, – число вхождений термина v_j в запросы пользователей i-ой соц-дем группы в течение всей поисковой истории.

Для сравнения результатов необходимо построить глобальный вектор $fg = (fg_1, \ldots, fg_j, \ldots, fg_{nof(Vu)})$, где для расчёта координат fg_j в числителе и знаменателе формулы 1.1 будет соответственно число вхождений всех терминов v_j для всех USR_i, $1 \le j \le 10$.

Таблица 1.8 – Таблица весов искомых слов для мужчин разного возраста

Word	fg	f_1	f_2	f_3	f_4	f_5
наличие	0,110336	0,013158	0,211583	0,182885	0,09277	0,159288
онлайн	0,100315	0,013158	0,11484	0,113593	0,117544	0,067463
скачать	0,091481	0,197368	0,000329	0,090875	0,102195	0,125556
бесплатно	0,066189	0,013158	0,039816	0,073836	0,074054	0,063247
смотреть	0,064272	0,013158	0,05561	0,06134	0,076612	0,029984
купить	0,059397	0,013158	0,071405	0,105642	0,056954	0,1026
Игры	0,050918	0,013158	0,040803	0,000379	0,086711	0,000234
Сайт	0,037042	0,171053	0,046397	0,040515	0,042144	0,038416

отзывы	0,036855	0,013158	0,051662	0,000379	0,042682	0,059733
Фото	0,032542	0,013158	0,059559	0,059447	0,000135	0,04029
Порно	0,031521	0,328947	0,118789	0,000379	0,035142	0,000234
контакте	0,029917	0,013158	0,000329	0,000379	0,000135	0,046849
Видео	0,02923	0,013158	0,050346	0,000379	0,033661	0,033497
Москве	0,027396	0,013158	0,000329	0,045816	0,028948	0,06231
Цена	0,025855	0,013158	0,042119	0,039	0,018715	0,08714
официальный	0,022355	0,013158	0,000329	0,034457	0,031911	0,000234
магазин	0,021917	0,013158	0,000329	0,038243	0,000135	0,02811
Москва	0,0215	0,013158	0,000329	0,045816	0,02087	0,030452
одноклассники	0,019938	0,013158	0,000329	0,000379	0,063283	0,000234
торрент	0,01873	0,013158	0,05232	0,000379	0,022889	0,000234
России	0,017605	0,013158	0,000329	0,031427	0,000135	0,000234
САНКТ-ПЕТЕРБУРГ	0,017271	0,013158	0,000329	0,032942	0,000135	0,000234
Фильм	0,01723	0,013158	0,000329	0,000379	0,030295	0,000234
интернет	0,016834	0,013158	0,000329	0,000379	0,000135	0,000234
Игра	0,01673	0,013158	0,040803	0,000379	0,000135	0,000234
Карта	0,016625	0,013158	0,000329	0,000379	0,021678	0,022722

Таблица 1.9 — Таблица весов искомых слов для женщин разного возраста

Word	***Fg***	f_6	f_7	f_8	f_9	f_{10}
наличие	0,110336	0,123077	0,067952	0,103787	0,152243	0,099602
онлайн	0,100315	0,148178	0,16483	0,131092	0,130391	0,09041
скачать	0,091481	0,149798	0,113407	0,091456	0,119034	0,090822
бесплатно	0,066189	0,045344	0,072084	0,097915	0,09747	0,06599
смотреть	0,064272	0,071255	0,082185	0,105843	0,094163	0,056661
купить	0,059397	0,00081	0,000459	0,048738	0,049741	0,105776
Игры	0,050918	0,038866	0,249311	0,046242	0,050029	0,041569
Сайт	0,037042	0,080162	0,000459	0,026571	0,022283	0,079023
отзывы	0,036855	0,00081	0,000459	0,03347	0,000144	0,040472
Фото	0,032542	0,00081	0,039486	0,03303	0,043128	0,039649
Порно	0,031521	0,00081	0,045914	0,070757	0,000144	0,000137
контакте	0,029917	0,02753	0,052801	0,000147	0,107821	0,022225
Видео	0,02923	0,00081	0,059688	0,033617	0,040253	0,000137
Москве	0,027396	0,00081	0,000459	0,000147	0,000144	0,042118
Цена	0,025855	0,00081	0,000459	0,000147	0,039247	0,000137
официальный	0,022355	0,0583	0,000459	0,000147	0,000144	0,038551
магазин	0,021917	0,00081	0,000459	0,027305	0,000144	0,037591
Москва	0,0215	0,00081	0,000459	0,021873	0,000144	0,038688
одноклассники	0,019938	0,00081	0,024793	0,027158	0,000144	0,000137
торрент	0,01873	0,116599	0,02112	0,000147	0,000144	0,000137
России	0,017605	0,00081	0,000459	0,01879	0,000144	0,000137
САНКТ-	0,017271	0,02834	0,000459	0,000147	0,000144	0,055426

ПЕТЕРБУРГ						
Фильм	0,01723	0,05749	0,000459	0,000147	0,029183	0,000137
интернет	0,016834	0,00081	0,000459	0,023341	0,000144	0,03032
Игра	0,01673	0,044534	0,000459	0,038315	0,023289	0,000137
Карта	0,016625	0,00081	0,000459	0,019671	0,000144	0,024009

Статистика получена, веса слов рассчитаны, можно приступать к формированию графиков весов (рисунок 1.8). На графиках рисунка 1.8 четко видно, что каждая соц-дем группа имеет свои интересы и своё поведение в сети Интернет. Каждый вектор f_i имеет своё отклонение относительно глобального вектора *fg*.

Рассматриваемый метод *TF* напрямую зависит от качества формирования словаря V_u и от персональных данных ИП. Метод частоты терминов *TF* может быть применён для классификации ИП, исключительно с целью подбора рекламы, однако если его применить для ИР с динамическими элементами, то можно столкнуться с проблемой изменений этих показателей с каждой загрузкой ИР.

Представленные выше методы классификации могут быть использованы для первичной сегментации Интернет-объектов с ограниченным множеством характеристик. Это может быть классификация ИП по соц-дем признакам или ИР по структуре. Для персонализации поиска необходимо применять более сложные методы с возможностью формирования групп объектов со сложными характеристиками.

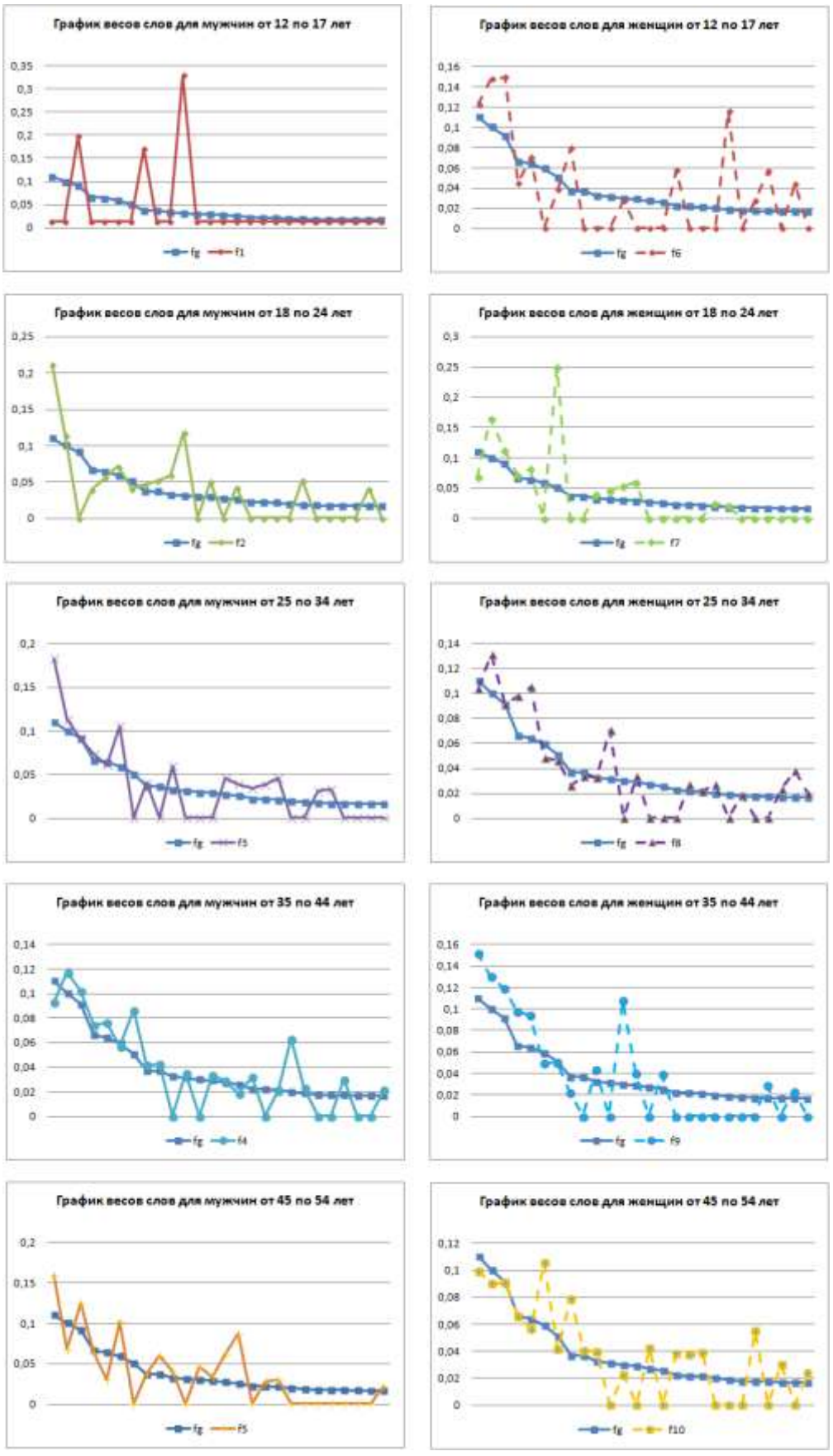

Рисунок 1.8 – Графики весов слов для ИП, разделённых по соц-дем признакам

1.3. Кластерные методы классификации Интернет-пользователей и Интернет-ресурсов

Кластеризация – это автоматическое разбиение элементов некого множества на группы (кластеры) в зависимости от показателей их схожести. Элементами множества может быть, что угодно: объекты с определённым набором данных или вектора характеристик. Большинство исследователей считают, что прародителем кластерного анализа является Роберт Коати Трион (*Robert Choat Tryon*) – американский исследователь поведения животных, который предложил систематизировать методы анализа влияния окружающей среды (экология, социальный уровень и т.д.) на поведение субъектов исследования (животных, людей) и предложил группировать субъекты исследования в кластеры. Предложенный им метод позволил с большой точностью определять причины и возможные последствия поведения человека в стрессовых ситуациях, исходя из его социального окружения.

У кластеризации существует большое количество практических применений. Кластеризация позволяет, например, провести анализ данных, поиск информации или группировку объектов по признакам и свойствам. Так же кластеризация сама по себе является важной формой абстракции данных, и в этой области был получен ряд интересных научных результатов.

Говоря о кластеризации Интернет-объектов, необходимо определить следующие базовые понятия.

Объект – элементарная единица, которая может быть представлена с помощью набора числовых характеристик, и с которой оперируют алгоритмы кластеризации. Каждому объекту x_i , $i \in I$, сопоставляется вектор числовых характеристик $z_i = (z_{i,1}, \ldots, z_{i,j}, \ldots, z_{i,n})$. Кардинальность вектора n определяет размерность пространства характеристик. Расстояние $\rho(x_i,x_k)$ между объектами x_i и x_k – результат применения выбранной метрики в пространстве характеристик. В настоящее время, существует большое количество метрик для оценки расстояния между векторами одного и того же векторного пространства. К простым метрикам можно отнести евклидово расстояние или его квадрат, манхэттенское расстояние,

степенное расстояние и другие [3, 30]. К сложным метрикам можно отнести расстояние между центрами тяжести, групповое среднее расстояние, расстояние Чебышева и другие. С подробным описанием различных метрик можно познакомиться в приложении 2.

Перенесём математическое описание абстрактных объектов в область исследования поисковых запросов ИП. Пусть USR – множество наблюдаемых ИП, usr_i – i-ый наблюдаемый ИП, $usr_i \in USR$. В произвольный момент времени $t_k \in T$, указанный ИП можно представить характеристическим вектором следующего вида:

$$u_i(t_k) = (u_{i,1}(t_k), \ldots, u_{i,j}(t_k), \ldots, u_{i,nof(Vu)}(t_k)), \qquad (1.2)$$

где

- $u_{i,j}(t_k)$ – вес j-ого поискового термина из глобального словаря терминов V_u в момент времени t_k, равный числу вхождений этого термина в запросы в поисковой истории i-го ИП, в течение наблюдаемого временного окна Δt;

- $nof(V)$ – размер вектора i-го ИП, равный числу слов в глобальном словаре терминов V_u.

Числовые координаты $u_{i,j}(t_k)$, $1 \le j \le nof(V_u)$ расположены в характеристическом векторе в порядке, соответствующем лексикографическому порядку следования соответствующих терминов в словаре V_u. Переход от вербального к числовому представлению результатов происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений в запросы поисковой истории ИП.

В наши дни методы кластерного анализа широко применяются для решения широкого спектра задач в Интернете. Кластерные методы со сложными алгоритмами оптимизации применяются в поисковых системах, Интернет-магазинах, системах анализа контента сайтов, системах проверки подлинности текстов диссертаций и ещё во многих сферах. Методы кластерного анализа разнообразны. Их можно разбить на множество групп:

- по способу обработки данных: иерархические (агломеративные методы и дивизивные методы); неиерархические методы (итеративные);

- по способу анализа данных: чёткие и нечёткие;

- по числу применений алгоритмов кластеризации: с одноэтапной кластеризацией и с многоэтапной кластеризацией;

- по возможности расширения объёма обрабатываемых данных: масштабируемые и не масштабируемые;

- по времени выполнения кластеризации: потоковые (в режиме реального времени) и непотоковые (по накоплению информации).

Существует ключевая разница между понятием кластеризация и понятием классификация. Кластеризация позволяет разбить множество объектов на группы (кластеры), а классификация – относит каждый объект к одной из заранее определённых групп.

В процессе решения задач кластеризации-классификации можно выделить четыре группы задач:

- выделение характеристик объектов;

- определение метрики – для кластеризации объектов применяется метрика близости объектов;

- разбиение объектов на группы с применением методов кластерного анализа;

- классификация вновь появившегося объекта.

В книге «Прикладная статистика» [3] предложена 8-ми этапная схема решения задач классификации. Каждый этап этой схемы представляет собой полноценный процесс с входными и выходными потоками, возможна и обратная связь.

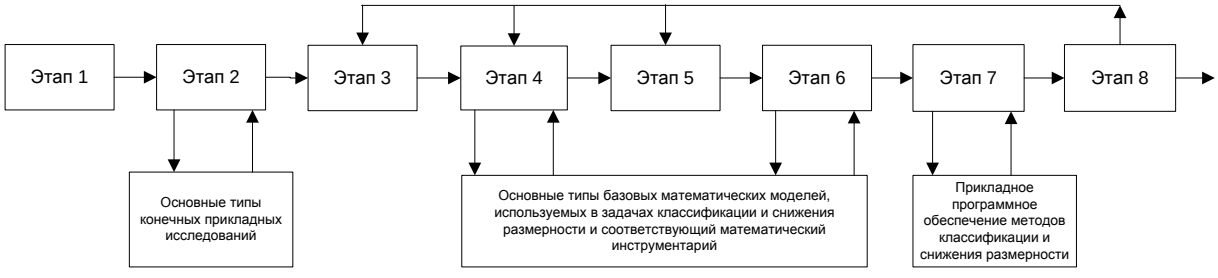

Рисунок 1.9 – Схема поэтапного процесса решения задач классификации

Этап 1. Установочный – на этом этапе должна быть сформулирована постановка задачи, включающая в себя характер научных или практических выводов, которые требуется получить на выходе.

Этап 2. Постановочный – на этом этапе необходимо сформулировать цели предметно-содержательной установки на этапе 1 в терминах основных типов прикладных задач, рассматриваемых в теории статистических методов классификации.

Этап 3. Информационный – состоит в выработке и реализации плана сбора исходной статистической информации.

Этап 4. Априорно математико-постановочный – на основании выводов и информации, полученных в результате реализации этапов 1-3, требуется осуществить предварительный выбор базовых математических моделей, которые целесообразно использовать в математической постановке данной конкретной задачи. При этом факторами, от которых решающим образом зависит выбор, являются характер конечных прикладных целей исследования, природа и форма исходных статистических данных.

Этап 5. Разведочный анализ – этот этап составляют всевозможные методы предварительной статистической обработки, пропущенных исходных данных с целью выявления специфики их вероятностной и геометрической природы. На выходе этапа должны быть уточненные сведения о физическом механизме генерирования наших исходных данных, а значит, о базовой математической модели этого механизма.

Этап 6. Апостериорный математико-постановочный – на этом этапе уточняется математическая постановка решаемой задачи с учётом выводов.

Этап 7. Вычислительный – производится вычислительная реализация намеченного к использованию, выбранного на предыдущем этапе, математического инструментария решения задачи.

Этап 8. Итоговый – анализируются и интерпретируются результаты проделанной работы. В зависимости от результатов этого анализа, либо формируются окончательные научные или прикладные выводы, либо даются

уточнения и дополнения к заданию и происходит возврат к одному из предыдущих этапов (обычно к этапу 3, 4 или 5). На последнем этапе следует ожидать положительного результата, удовлетворяющего поставленным задачам и выбранным математическим моделям, в противном случае необходимо вернутся к одному из предыдущих этапов для доводки принятых решений.

1.4. Математические модели кластерных методов – иерархические и итерационные алгоритмы кластеризации

Иерархические алгоритмы можно отнести к нечётким алгоритмам кластеризации, т.к. число кластеров заранее не известно. В чистом виде иерархическая кластеризация является одноэтапной, немасштабируемой и непотоковой, т.к. применяется единственный алгоритм кластеризации по принципу отдалённости (ближайшего или дальнего) соседа на основе статистики конкретных исследуемых объектов.

В качестве исходных данных выступают поисковые запросы ИП. Широко применяемые ассоциативные и статические методы могут быть использованы для предварительной сортировки объектов. В частности, эти методы могут быть использованы для определения продолжительности поискового интереса в сети или для подбора похожих товаров в Интернет-магазинах. Для работы со сложными объектами такими, как ИП и ИР могут применяться иерархические алгоритмы кластеризации, т.к. агломеративные и дивизивные иерархические алгоритмы отлично справляются с задачей группировки объектов с большим количеством признаков – свойств или характеристик. Тем самым, если исследовать поисковую деятельность ИП за некий период времени Δt, то в конце периода наблюдения каждому пользователю можно будет сопоставить характеризующий вектор, координаты которого будут использованы для построения иерархий [3].

Исследование структуры иерархий удобно вести в терминах теории графов [7]. В графе иерархии вершина может быть концом нескольких стрелок, но она является началом только одной стрелки. Иерархия является бинарной, тогда и

только тогда, когда в её графе каждая вершина, соответствующая множеству, содержащему более одного элемента, является концом двух стрелок.

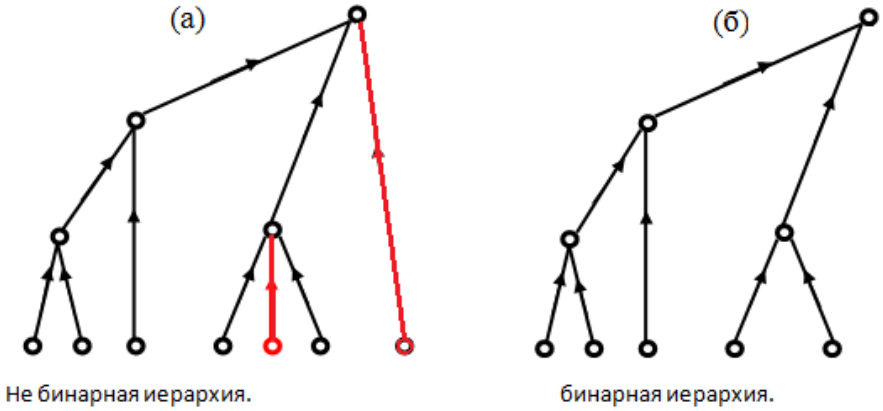

Рисунок 1.10 – Бинарная (б) и не бинарная (а) иерархии

Иерархической классификацией множества объектов $X = \{X_1,\ldots, X_{nof(X)}\}$ называется построение иерархии на X, отражающей наличие однородных, в определённом смысле, классов X и взаимосвязи между классами. Алгоритмы иерархической классификации бывают дивизивные, в которых начальное множество X постепенно разделяется на всё более мелкие подмножества, и агломеративные – в которых точки множества X постепенно объединяются во всё более крупные подмножества. Полученные графы иерархий при помощи этих алгоритмов называются, соответственно, дивизивными и агломеративными.

В отличие от иерархических (агломеративных и дивизивных) методов кластеризации, итерационные методы, например метод k-средних, который использует в качестве входного параметра число классов, на которые проводится разбиение множества X, или метод Форель, для которого необходимо указать радиус шаров, которыми покрывается выборка X. Учитывая сказанное, итерационные методы следует отнести к точным методам кластеризации.

Обзор существующих методов кластеризации и обсуждение проблемы выбора рационального метода кластеризации ИП и ИР представлены в приложении 3.

1.5. Основные результаты и выводы по первой главе

1. Показано, что статические методы классификации, основанные на персональной информации ИП (пол, возраст, место проживания и др.), широко применяется в социальных сетях для целевого подбора Интернет-рекламы, предоставления ИП другой адресной информации, например, гороскопов. Некластерные методы классификации (ассоциативный метод, метод пересечений и др.) широко применяются в Интернете для подбора «похожего» или «сопутствующего» товара в Интернет-магазинах. Несмотря на простоту и распространённость статических и некластерных методов, они не могут быть использованы для персонализации поиска в широком понимании этого слова: эти методы могут быть направлены лишь на достижение локальной цели – получение коммерческой выгоды.

2. Методом частоты терминов *TF* доказано, что каждая социально-демографическая группа ИП может быть охарактеризована личным графиком весов слов, отражающим интересы пользователей, входящих в группу. Как оказалось, каждая соц-дем группа уникальна в своём поисковом поведении, отсюда следует, что ИП могут быть выделены в отдельные группы по половой и возрастной принадлежности и это автоматически приведёт к разделению по интересам, соответствующим этим группам. Несмотря на индивидуальность каждого ИП, их можно и нужно группировать, как с помощью статической информации (пол, возраст и т.д.), так и на основе анализа их поисковой деятельности в Интернете.

3. Исследованы основные методы классического кластерного анализа (иерархические и итерационные), когда объекты исследования представлены с помощью числовых характеристических векторов, после предварительного формирования глобального словаря терминов и применения метода позиционного кодирования.

4. Уточнена разница между терминами «кластеризация» и «классификация». Кластеризация позволяет разбить множество объектов на группы (кластеры), а классификация – относит каждый объект к одной из заранее

определённых групп. Для выполнения задач кластеризации и классификация была выбрана схема решения соответствующих задач, состоящая из восьми ключевых этапов. Наличие такой схемы должно сделать настоящее исследование более структурированным.

2. ЛИНГВИСТИЧЕСКИЙ АНАЛИЗ ЗАПРОСОВ ИНТЕРНЕТ-ПОЛЬЗОВАТЕЛЕЙ И ТЕКСТОВ ИНТЕРНЕТ-РЕСУРСОВ

Прежде чем приступить к изложению основного материала и результатов диссертационной работы, необходимо остановиться на одной очень важной, хотя и носящей вспомогательный характер теме – лингвистической обработке текстов запросов Интернет-пользователей и Интернет-ресурсов, предваряющей их кластерный анализ.

2.1. Методы анализа содержания текста

Лингвистический анализ – метод исследования текста, который может быть охарактеризован как лингвосмысловой анализ. Он представляет собой изучение методов, позволяющих автоматизировано «понимать» текст, т.е. уметь извлекать из него нужную информацию и отвечать на вопросы, заданные относительно текста. Лингвистический анализ применим, в частности, к извлечению информации, машинному переводу, а также ко многим областям искусственного интеллекта, относящимся к общению с пользователем.

Существует множество подходов к лингвистическому анализу. Среди них можно выделить статистический анализ, анализ признаков, семантический анализ и комбинированный подход.

Статистический анализ подразумевает изучение последовательности слов в предложениях текста, а также вывод определенных закономерностей на основании проведённого изучения. Для этого производится подсчёт частоты встречи слов в тексте, а также вероятность появления слов друг за другом. С помощью статистических методов анализа текста решается проблема классификации текстов [18]. Соответственно, анализируя содержание слов, можно получать вероятности встречи *N*-граммы или частоту терминов (*TF* или *TF-IDF*) в художественном или в научном тексте. После этого найденные вероятности переводятся в веса и складываются. Текст будет относиться к тому

классу, вес которого окажется больше.

Анализ признаков заключается в изучении морфемных, морфологических и синтаксических признаков слов и предложений в тексте. Это изучение необходимо для того, чтобы дальше можно было строить модель (структуру) предложений и на её основании извлекать требуемую информацию.

Для построения такой структуры необходимо произвести несколько операций. Первая заключается в определении грамматических признаков всех слов. Для этого в языках, в которых существуют падежи, спряжения, склонения и времена, устанавливаются все эти признаки. Также определяется лемма каждого слова – его словарная форма. Для определения леммы слова используется алгоритм-лемматизатор, который либо с помощью сравнения со словарём, либо посредством последовательного отсечения окончаний и аффиксов (префиксов, суффиксов и постфиксов) и добавления нормализованного окончания выделяет основу слова. Вторая операция сводится к построению модели предложений в тексте. Посредством анализа грамматических признаков, найденных ранее, составляется зависимость слов друг от друга – сначала в пределах предложения, а затем и в пределах текста, если необходим более широкий анализ. Третья операция сводится к поиску искомой информации. Пользовательский запрос проходит лемматизацию, после чего происходит поиск необходимых лемм по всех предложениям. Однако то, что не представляет большого труда для лингвиста и просто человека, является большой проблемой для вычислительной системы. Описываемый выше анализ сложно чётко сформулировать, т.к., например, слово «истории» можно рассматривать не только как дательный падеж, но и как именительный/винительный падеж множественного числа слова «история». Отдельной препоной являются омонимы («ягуар» – это автомобильный бренд, энергетический напиток или хищник?), т.е. созвучные слова; если в тексте встречается несколько слов с одинаковыми леммами, нет гарантии того, что это – одно и то же слово. Для выяснения этого необходимо провести более сложный кластерный анализ для определения содержания текстов ИР в целом после получения характеристических векторов. Без кластерного

анализа (или без применения методов классификации текстов) значений слов такие неоднозначности довольно сложно разрешить. Ещё одной значительной проблемой является разбор отсутствующих в словаре слов, когда нужно обращаться к лингвистическому эксперту, который в свою очередь вносит изменения в словаре лемм.

Чётко заданных правил для такого анализа не существует; их можно вывести экспериментальным путём, однако всё равно будут существовать исключения, отработать которые по правилам будет невозможно.

Семантический анализ занимается разбором текста относительно значения слов внутри него. Обычно этот вид анализа применяется после проведения грамматического анализа и дополняет его своими выводами. В частности, семантический анализ позволяет выявлять несвязность слов и предложений внутри текста, хотя они и могут быть согласованы грамматически. Также семантический анализ позволяет определять метафоры, перен*о*сные значения, истинный смысл созвучных слов в зависимости от контекста, и т.д. К сожалению, серьёзных успехов в этой области пока достичь не удалось, т.к. этот вид анализа является наиболее сложным и наименее формализованным, хотя и самым востребованным. Простые способы семантического анализа позволяют классифицировать текст, выделять эмоциональную окраску текста (с помощью выявления определённых слов и анализа словосочетаний на предмет метафор и иносказаний) и его тему (по синтаксическим признакам и количеству повторяющихся слов в предложениях). В частности, с помощью семантического анализа происходит выдача контекстной рекламы на многих сайтах и в поисковых системах [10]. Страница, выдаваемая пользователю, исследуется на предмет наличия повторяющихся ключевых слов, после чего автоматизированный генератор рекламы выдаёт связанную с найденными ключевыми словами выборку.

Комбинированный подход подразумевает использование нескольких из вышеописанных подходов в связке, последовательной или параллельной, для повышения точности анализа. Чаще всего для сложного анализа текста

применяют анализ признаков, совмещённый со статистическим анализом для ранжирования результата поиска и разрешения неоднозначностей; реже используются вкрапления семантического анализа в любой из вышеописанных методов. В частности, такой подход используется в текстовых редакторах для выявления сложных ошибок (несогласованность текста, рекомендации по разбиению текста на абзацы, и т.д.).

В рамках данной работы, обработка текста проводиться с помощью комбинированного подхода на основании статистического метода и метода анализа признаков.

2.2. Лингвистическая обработка запросов Интернет-пользователей и текстов Интернет-ресурсов

Для применения методов лингвистического анализа необходимы леммы всех слов текста (анализ признаков) и частота встречи этих лемм в тексте (статистический анализ). Следует отметить, что анализ моделей связей между словами внутри предложений и предложениями внутри текста в данной работе не рассматривается.

Первым шагом к решению поставленной проблемы анализа содержания, как ИР, так и запросов ИП является разбиение текста Интернет-страниц и Интернет-запросов на отдельные слова (термины), т.к. текст в формальном определении является просто набором слов. Здесь уже возможны проблемы и неоднозначные трактовки: что считать словом, как относиться к сложным знакам пунктуации и т.д. Введём набор простых правил, описывающих большинство случаев, которые могут встретиться в априорно правильном тексте.

Словом или **термином** назовем последовательность символов букв, ограниченная с обеих сторон пробелами либо знаками препинания, в которой могут присутствовать цифры, в том числе и на первой позиции. Все знаки препинания и специальные символы («+», «-», «/», «=» и т.д.) заменяются пробелами, тем самым, непрерывная последовательность символов превращается в слова, отдельные друг от друга пробелами.

Леммой или **корнем слова** будем считать урезанную последовательность символов термина, получаемую помощью специально разработанного алгоритма отсечения окончаний с учетом двухуровневого словаря, включающего глобальный словарь терминов и словарь лемм, который может заполняться с помощью существующих открытых словарей [52] и динамически пополняться при появлении новых терминов. Очевидно, что нескольким терминам может соответствовать одна лемма.

Из сформулированных правил следует, любые тексты действительно подойдут под описания, приведённые выше. «Правильными» текстами в данном случае будут считаться тексты, как на русском, так и на английском языках, находящиеся в открытом доступе в сети Интернет и допускающие компьютерную обработку. Априори будем считать, что ИР не содержать недопустимые для заданного языка символы либо тексты с синтаксическими опечатками. Все специальные символы фильтруются. Выявление синтаксических опечаток – отдельная серьёзная задача, которая может быть отведена лингвистическому эксперту, который, в свою очередь, может внести исправления в словарь лемм. Пользуясь справочниками [52], удалось написать алгоритм заполнения словаря терминов и лемм, соответствующих приведенным выше правилам.

В диссертации, как уже было сказано, применяться комбинированный подход, основанный на статистическом методе, методе анализа признаков и особенностях *DOM*-моделей ИР. Семантический анализ требует дополнительного исследования, и не будет рассматриваться в рамках текущей работы.

На рисунке 2.1 представлены этапы процесса обработки содержания запросов ИП и текстов ИР от первичного «грязного» текста до лемм. Результатом выполнения этого процесса является формирование статистики лемм и наращивания словаря с помощью лингвистического эксперта. Схема алгоритма лингвистической обработки терминов (слов) приведена на рисунке 2.2.

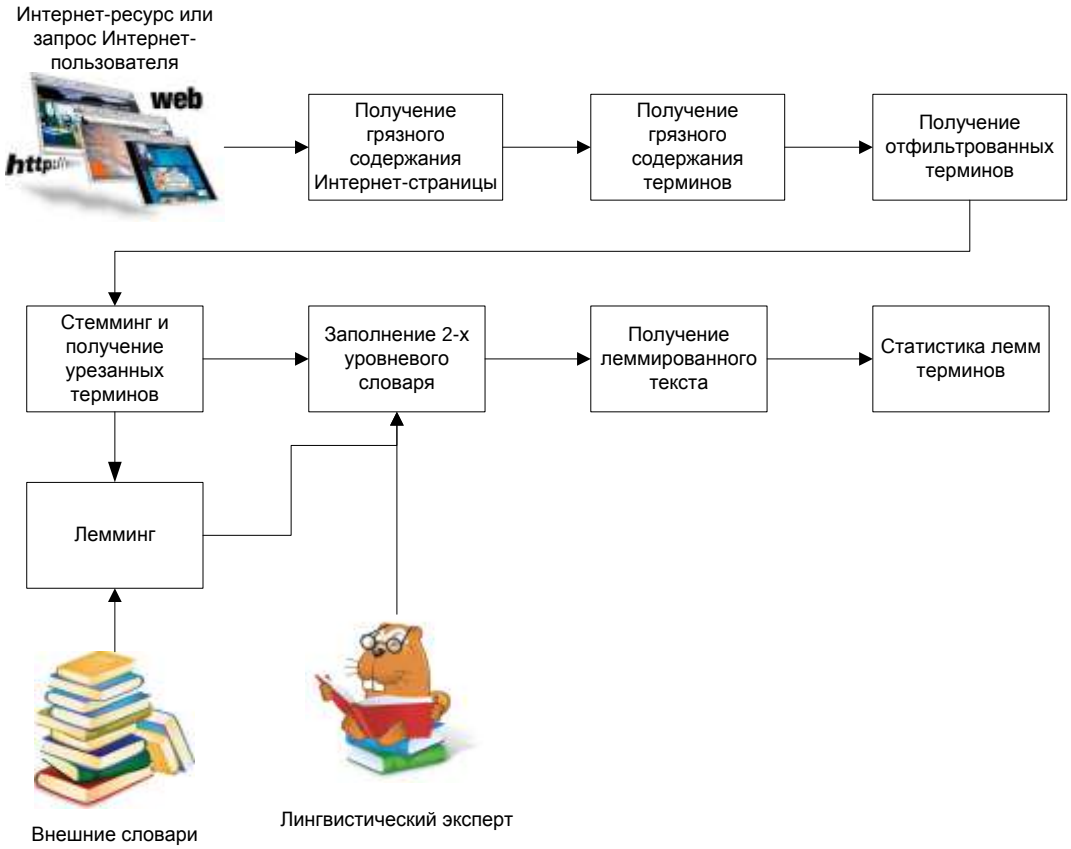

Рисунок 2.1 – Процесс лингвистической обработки запросов ИП и текстов ИР.

Так как на этапе работы рассматриваемого алгоритма происходит полный перебор слов из текста ИР или поискового запроса ИП, целесообразно проводить здесь же подсчёт встречаемости слов в тексте, применяя известные статистические методы. Затем, на основании полученной информации будут сформированы характеристические вектора классифицируемых Интернет-объектов и глобальный словарь терминов, с которыми будут дальше работать алгоритмы кластерного анализа.

В алгоритме рисунка 2.2, можно воспользоваться особенностями *DOM*-модели ИР, позволяющей локализировать позиции терминов и тем самым выделять особо важные термины (например, заголовки и наименование Интернет-страниц), с целью повышения значений числовых характеристик у ключевых терминов или даже исключения динамических элементов, «просочившихся» при чтении содержания текстов ИР.

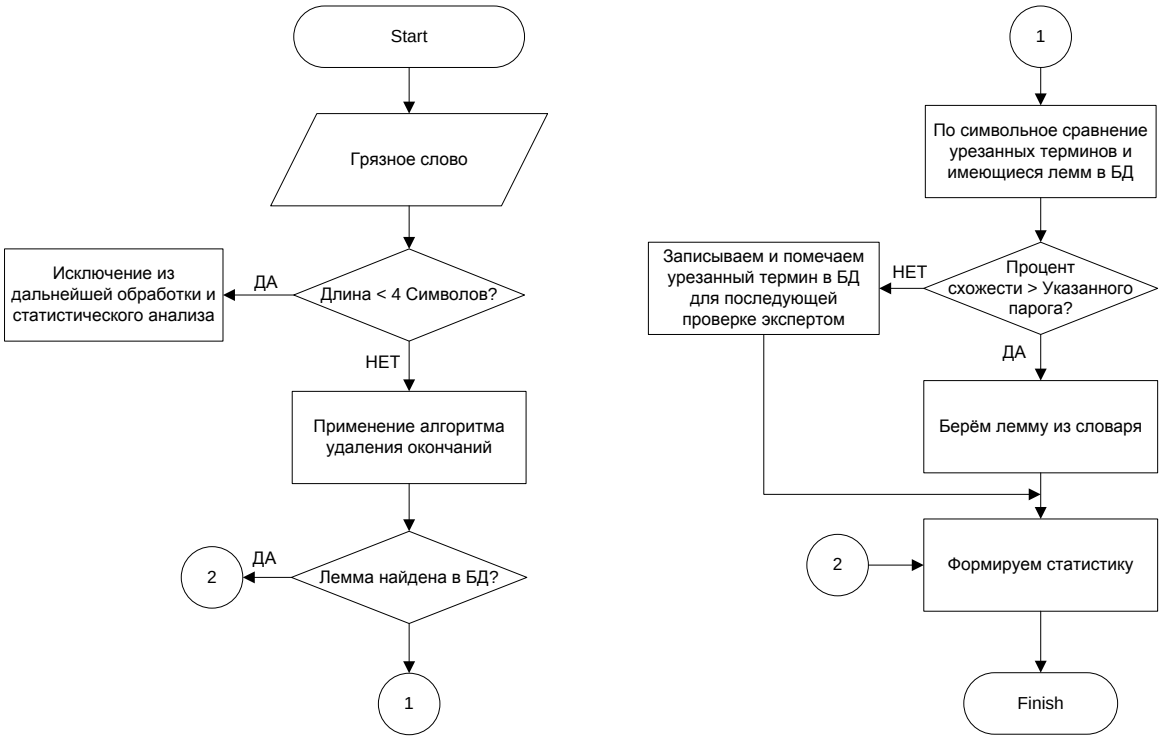

Рисунок 2.2 – Схема алгоритма лингвистической обработки терминов.

С помощью описанного выше алгоритма, текст ИР разбивается на термины, которые уже можно обрабатывать. Необходимо обратить внимание на процесс обработки терминов, а точнее на стемминг (*stemming*) и лемматизацию терминов для приведения слова к начальной форме и тем самым сгруппировать их, что, в свою очередь, сделает статистику терминов более достоверной.

Начальной стадией работы почти всех лемматизаторов является стемминг. Этот процесс подразумевает выделение корня слова, тогда как лемматизатор на основании корня подбирает базовую форму, подходящую для него. Одними из признанных фаворитов среди стеммеров является программы, реализующие алгоритмы усечения окончаний [72]. Алгоритмы усечения окончаний могут быть использованы как для русских, так и для английских терминов, а полученные впоследствии усеченные термины могут быть применены в качестве лемм для формирования статистики первоначальных слов из считанного текста. В добавок алгоритм работает с множеством окончаний, разбитым на несколько подмножеств: окончания герундия («в», «вши», «вшись» – некоторые окончания деепричастий для русского языка или «*ing*» для английского языка), окончания прилагательных («ыми», «ми», «ие»), причастные суффиксы («ющ», «вш»,

«ивш»), возвратные постфиксы («сь», «ся»), глагольные окончания («ла», «ли», «ем»), префиксы («до», «за», «над») и окончания существительных.

Процесс стемминга легко реализуется с использованием БД, для чего достаточно воспользоваться специальным словарём масок (маски на *T-SQL* представляются набором символов, в которых должны присутствовать символы «%» и/или «_»), который может быть совмещён со списком специальных символов для фильтрации. Получившаяся в результате обработки усечённая форма терминов, проверяется по БД с помощью словаря лемм. Если лемма найдена, то оригинальный термин привязывается к найденной леммы (рисунок 2.2). Если урезанному термину невозможно было сопоставить лемму из словаря, то он посимвольно сравнивает с леммами из словаря. На последних шагах алгоритма в словарь лемм включаются урезанные термины, которым невозможно было сопоставить уже имеющиеся леммы. Они помечаются, чтобы лингвистический эксперт смог провести проверку и, при необходимости, поправить или назначить новые леммы. Такой алгоритм сам по себе универсален и весьма точен, однако и ошибки, как показала практика, случаются достаточно часто, например, при наличии у слов приставок, что может привести к получению несловарных корней либо к чрезмерному усечению слов. Однако эту проблему всегда может решить квалифицированный лингвистический эксперт.

В предложенном алгоритме достаточно воспользоваться двух уровневым словарём и для статистической обработки текстов и для формирования характеристических векторов. Если для обработки терминов потребуется другие промежуточные результаты стемминга, то алгоритм нужно будет перенастроить на систему словарей более высокого уровня (например, 3-х или 4-х уровневых словарей), добавляя в него блоки обработки дополнительного словаря.

Резюмируя всё сказанное выше, получим следующую последовательность действий, по преобразованию исходного запроса ИП/текста ИР в представление, пригодное для дальнейшей обработки алгоритмами кластерного анализа:

а) выделение всех терминов из запроса ИП/текста ИР;

б) стемминг и получение урезанных терминов после удаления окончаний;

в) проверка урезанных терминов по словарю лемм БД. Если лемма найдена, переход к пункту д).

г) сохранение помеченных урезанных терминов, для которых не определена лемма из БД. При необходимости лингвистический эксперт может подтвердить или изменить помеченные урезанные термины, превращая их в новые леммы;

д) формирование статистики терминов и характеристических векторов;

После выполнения перечисленных пунктов, осуществляется переход к кластерному анализу лингвистически подготовленных запросов ИП и текстов ИР.

2.3. Основные результаты и выводы по второй главе

1. Исследование основных подходов к лингвистическому анализу текста (статистический анализ, анализ признаков, семантический анализ и т.д.) позволило выбрать комбинированный метод обработки запросов ИП и текстов ИР. Для представления первичного «грязного» текста в виде последовательности лемм предложено использовать комбинацию статистического метода и метода анализа признаков, учитывающего особенности *DOM*-моделей ИР. Модели связи между словами в предложениях и между предложениями в данной работе не рассматриваются.

2. Предложено использовать двухуровневый словарь, включающий глобальный словарь терминов и словарь лемм, который может заполняться с помощью существующих открытых словарей и динамически пополняться при появлении новых терминов. Корректировку словаря может проводить лингвистический эксперт.

3. Разработан и опробован алгоритм лингвистической обработки первичных терминов с помощью усечений окончаний с применением стемминга и лемматизации терминов, что превращает последовательность терминов (слов) теста в последовательность урезанных терминов и затем в последовательность лемм, которая в конечном итоге делает статистику терминов более достоверной.

3. РАЗРАБОТКА МЕТОДОВ КЛАСТЕРИЗАЦИИ ИНТЕРНЕТ-ОБЪЕКТОВ С ДИНАМИЧЕСКИМИ КОМПОНЕНТАМИ

3.1. Динамические изменения в кластерной структуре Интернет-объектов

Современные Интернет-ресурсы являются динамичными объектами. Если бы они содержали исключительно статические компоненты, то расчёт центров соответствующих кластеров можно было бы проводить в дискретные моменты времени, в моменты появления новых ИР. Кластеризация Интернет-пользователей также носит динамический характер, так как человеческое поведение являются динамичным процессом и это отражается в поисковых историях ИП. В задачах кластеризации ИП нужно учитывать не только текущие характеристики кластеризуемых объектов, но и их временны*е*, то есть динамические изменения по факту появления новых поисковых запросов. В этой главе исследуются динамические изменения кластерной структуры Интернет-объектов.

Пусть X множество всех наблюдаемых объектов $x_i \in X$, $1 \le i \le nof(X)$, отнесенных к одному из кластеров $X_l \subseteq X$, $1 \le l \le nof(K)$, где $K = \{X_1, \ldots, X_l, \ldots, X_{nof(K)}\}$ – множество всех сформированных кластеров. В разные моменты времени $t_k \in T$, $k = 0,1,2, \ldots$ проводим наблюдение за изменением состояния кластерной структуры в зависимости от характеристик объектов x_i, при этом состояние каждого i-го объекта в произвольный момент времени t_k отображается характеристическим вектором $z_i(t_k)$. Здесь необходимо говорить о временно*й* составляющей в качестве дополнительного параметра для всех элементов вектора, характеризующего объект. Если объект исследования при иерархической кластеризации представлен вектором $z_i = (z_{i,1}, \ldots, z_{i,j}, \ldots, z_{i,n})$, который не зависит от времени, то в динамической системе кластеризации необходимо говорить о векторе $z_i(t_k) = (z_{i,1}(t_k), \ldots, z_{i,j}(t_k), \ldots, z_{i,n}(t_k))$, координаты которого привязаны к моментам времени t_k. В любой фиксированный момент времени t_k (или интервал

времени Δt_k) можно выделить несколько кластеров, внутри которых объекты обладают общими характеристиками. Изменение характеристик объекта $u_i \in U$ в момент времени t_k может привести к глобальным изменениям на уровне всей кластерной структуры и тем самым через период времени Δt_k будет необходимо провести новую кластеризацию всех объектов из U. Если после истечения времени Δt_k число кластеров, их содержание, размеры и положение их центров не изменяются, то речь может идти о так называемой статической кластерной структуре. Однако совсем иначе обстоит дело в ситуациях, когда с течением времени кластерная структура изменяется, когда объекты с течением времени начинают обладать некоторыми новыми характеристиками и образуют группу объектов, функционирование которых находится на границе кластеров или даже за его пределами. В таком случае кластерная структура претерпевает временны*ы*е изменения и становится динамической.

Следует посмотреть на кластеризацию, как на задачу мониторинга совокупности Интернет-объектов с n-мерным характеристическим вектором числовых признаков $z_i(t_k)$, где индекс i соответствует номеру объекта. Измерение характеристик данных объектов осуществляется в дискретные, не обязательно равноотстоящие моменты времени t_k. Через интервал времени Δt_k проводится проверка стабильности кластерной структуры и при необходимости, т.е. при наличии динамических изменений, ее коррекция. Например, для кластеризации ИП, с целью исключения влияния слишком старых наблюдений, мониторинг состояния кластеров целесообразно организовать по принципу временного окна, т.е. в $nof(V_u)$-мерном пространстве при анализе кластерной структуры должны учитываются только объекты, зафиксированные в последнем временном окне Δt_k. Если говорить о временном окне для учёта новых Интернет-объектов можно предположить следующие динамические изменения в структуре кластеров: образование новых кластеров, слияние кластеров, расщепление или дробление кластеров, исчезновение кластеров, перемещение центров кластеров [15]. Всё ли так в действительности?

Изменения в структуре кластеров: образование новых кластеров.

Для множества характеристических векторов ИП $U(t_{k+1})$ в момент времени $t_{k+1} > t_k$ образование новых кластеров может быть связано с появлением новых объектов или с резким устойчивым изменением поисковой активности существующих объектов.

Для первого случая, каждый новый объект исследования $u_{nof(U(t_k))+p}(t_{k+1}) \in U(t_{k+1})$, $p \geq 1$, представляет собой новый, одиночный изолированный кластер, для которого проводится расчёт меры близости (евклидового расстояния) до всех остальных объектов $U(t_{k+1}) \backslash \{u_{nof(U(t_k))+p}(t_{k+1})\}$. По результатам расчёта евклидового расстояния определяется самый близкий к $u_{nof(U(t_k))+p}(t_{k+1})$ объект $u_{near}(t_{k+1})$:

$$\rho(u_{nof(U(t_k))+p}(t_{k+1}), u_{near}(t_{k+1})) = \min_{1 \leq i \leq nof\ (U(t_k))} \rho\left(u_{nof(U(t_k))+p}(t_{k+1}), u_i(t_{k+1})\right).$$

Определение ближайшего соседа $u_{near}(t_{k+1})$ позволит инициализировать место положения нового объекта в новой кластерной структуре.

Продолжаем наблюдать за Интернет-пользователем u_i в период времени Δt_k. С появлением новых объектов ИП увеличивается глобальный словарь терминов V_u. Если на момент времени t_{k+1} (в предшествующий интервал времени Δt_k) ИП $u_i(t_{k+1})$ не проводил никакую поисковую деятельность, то увеличение словаря терминов V_u никак не отражается на его характеристическом векторе, а лишь приводит к появлению новых нулевых координат: если в момент времени t_k характеристический вектор ИП имел такой вид

$$u_i(t_k) = \left(u_{i,1}(t_k), u_{i,2}(t_k), \ldots, u_{i,nof(V_u)}(t_k)\right),$$

то в момент времени t_{k+1} он преобразуется к виду:

$$u_i(t_{k+1}) = \left(u_{i,1}(t_{k+1}), u_{i,2}(t_{k+1}), \ldots, u_{i,nof(V)}(t_k), 0, 0, \ldots, 0\right).$$

На рисунке 3.1 представлена иллюстрация первоначальной кластерной структуры в момент времени t_k.

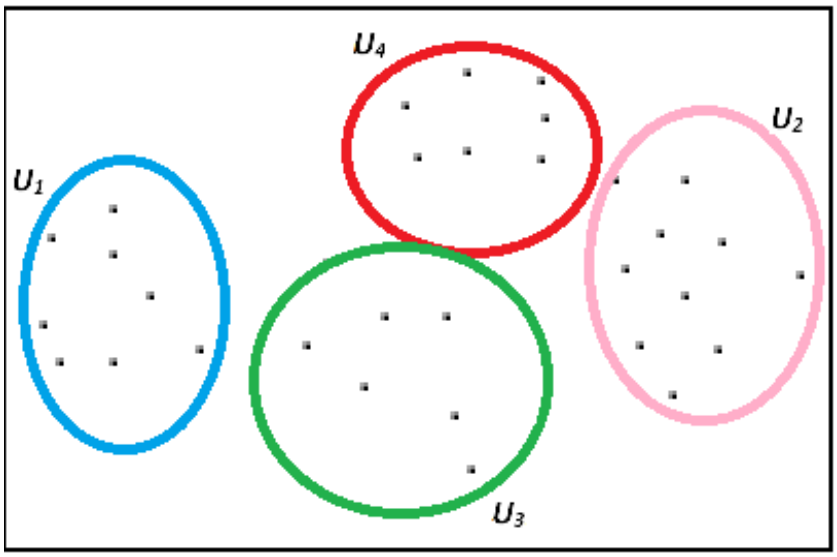

Рисунок 3.1 – Иллюстрация распределения объектов по кластерам в момент времени t_k

Если экспериментальным путём в интервале времени Δt_k появились новые объекты $u_{nof(U(t_k))+p}(t_{k+1}) \in U(t_{k+1})$, $p \geq 1$, то после проведения повторной кластеризации появляются новые кластеры (рисунок 3.2), сформированные с помощью новых объектов.

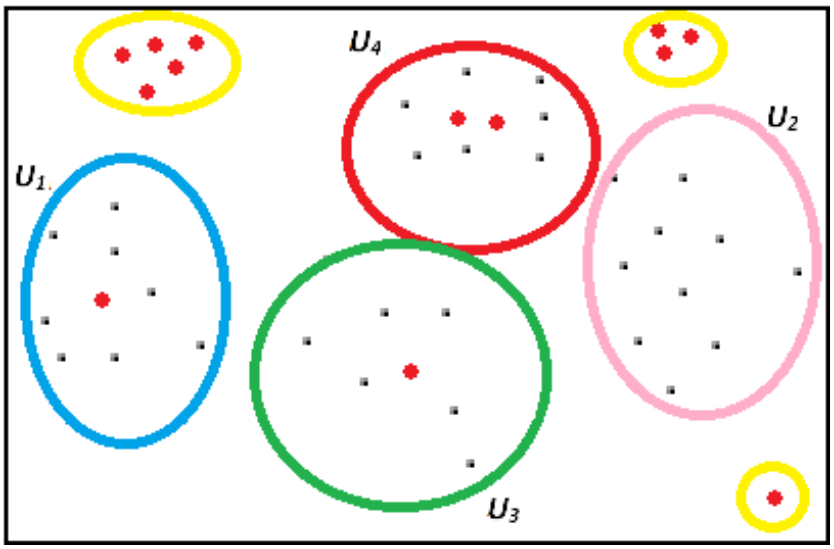

Рисунок 3.2 – Иллюстрация распределения объектов по кластерам в момент времени t_{k+1}

На рисунке 3.2 можно заметить формирование новых кластеров из вновь появившихся объектов, одиночные кластеры, кардинальное число которых равно

1, и насыщение (наращивание) уже сформированных кластеров за счёт поступления новых объектов.

Если на момент времени t_{k+2} поисковая деятельность одного или нескольких пользователей резко меняется, характеристические вектора этих объектов меняют значения одного или нескольких своих координат и, как следствие, структура кластера изменяется, что, в свою очередь, приводит к изменению значения меры близости и расстояния между объектами одного кластера. Это может привести к формированию новых кластеров (рисунок 3.3).

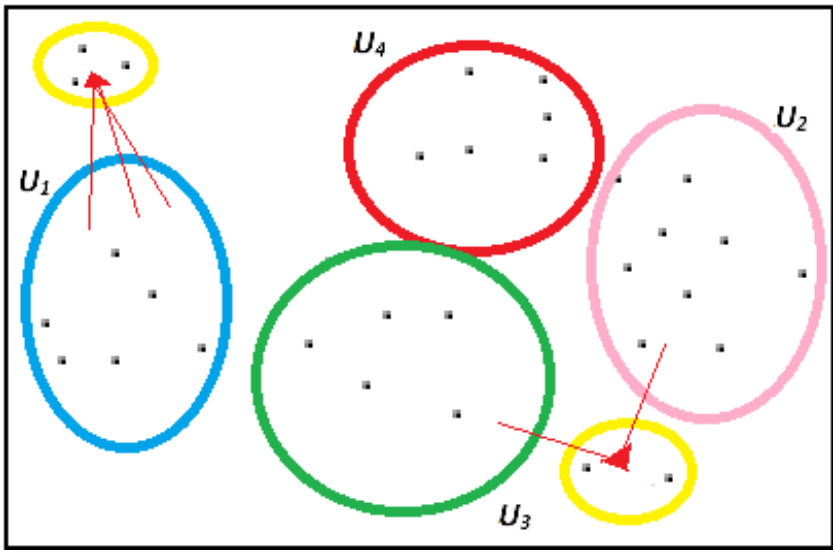

Рисунок 3.3 – Иллюстрация перераспределения объектов в момент времени t_{k+2}

Для решения задачи появления новых кластеров можно воспользоваться коэффициентом принадлежности i-ого объекта к m-ому кластеру.

Коэффициент принадлежности $b_{i,m}(t_{k+2})$ объекта $u_i(t_{k+2})$ к кластеру $U_m(t_{k+2})$ из кластерной структуры $K(t_{k+2})$ ($U_m(t_{k+2}) \in K(t_{k+2})$) в произвольный момент времени t_{k+2}:

$$b_{i,m}(t_{k+2}) = \frac{1}{\sum_{l=1}^{nof(K(t_{k+2}))} \left(\frac{(\rho_m(t_{k+2}))^2}{(\rho_l(t_{k+2}))^2} \right)} \quad \text{и} \quad \sum_{m=1}^{nof(K(t_{k+2}))} b_{i,m}(t_{k+2}) = 1, \qquad (3.1)$$

где

- $nof(K(t_{k+2}))$ – число кластеров в кластерной структуре $K(t_{k+2})$;

- $\rho_m(t_{k+2}) = \sqrt{\sum_{j=1}^{nof(V_u(t_{k+2}))} \left(e_{m,j}(t_{k+2}) - u_{i,j}(t_{k+2})\right)^2}$ – евклидово расстояние между объектом u_i и центром e_m m-ого кластера кластерной структуры ИП.

Выявление множества новых кластеров K^{new} начинается с выявления множества новых объектов U^{free} с малыми степенями принадлежности ко всем существующим кластерам. Если число таких свободных объектов $nof(U^{free})$ сопоставимо с размерами кластеров, они формируют компактную группу объектов с общими свойствами. Компактность объектов, определенная ниже (формула 3.6), является признаком появления новых кластеров. Возможное число новых кластеров $nof\left(K^{new}\right)$ в момент времени t_k определяется соотношением:

$$nof\left(K^{new}\right) = int\left(\frac{nof\left(U^{free}\right)}{d_1 \times N_{min}}\right), \qquad (3.2)$$

где

- $int(...)$ – целая часть аргумента;

- $nof(U^{free})$ – число свободных объектов, которые не привязаны ни к одному из кластеров;

- d_1 – заданная пороговая величина в интервале [0,1];

- $N_{min} = \min\left(nof\left(U_1^*\right), ..., nof\left(U_l^*\right), ..., nof\left(U_{nof(K)}^*\right)\right)$ – минимальный размер кластера, при вычислении которого учитываются только «хорошие» объекты $U_l^* \subseteq U_l$ с достаточно большими значениями степеней принадлежности;

- $nof\left(U_l^*\right)$– число хороших объектов l-ого кластера, для которых значение степени принадлежности к указанному кластеру $b_{i,l} \geq d_2$;

- d_2 – заданная пороговая величина в интервале [0,1].

Изменения в структуре кластеров: слияние кластеров.

По определению, слияние кластеров – это формирование нового разбиения, когда $nof(K')$-ое количество кластеров превращаются в $nof(K'')$-ое количество кластеров, причем $nof(K'') < nof(K')$.

Пусть слияние кластеров происходит в момент времени t_{k+3}. С изменением поискового вектора некоторые кластеры могут приблизиться друг к другу и слиться в единую группу (рисунок 3.4).

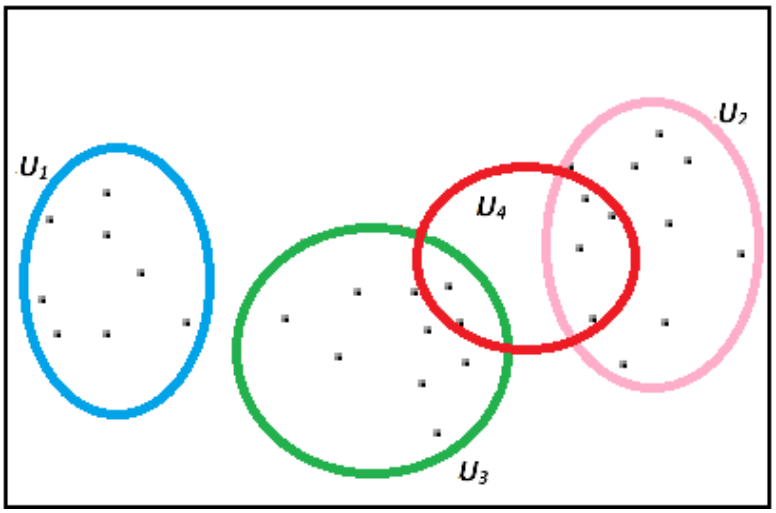

Рисунок 3.4 – Иллюстрация слияния кластера U_4 и перераспределение его объектов и центра между кластерами U_3 и U_2 в t_{k+3}

Выявление сливающихся кластеров начинается с выделения объектов u_i, имеющих высокие степени принадлежности (формула 3.1) одновременно для двух кластеров U_l и U_m : $b_{i,l} \approx b_{i,m} \rightarrow 1$. Если число таких сливающихся объектов достаточно большое, то это и является признаком слияния кластеров. Для итеративных методов кластеризации (метода k-средних или метода Форель) наблюдается сближение центров кластеров между сливающимися кластерами. Количественным критерием для объявления двух кластеров сливающимися кластерами может служить мера их сходства:

$$I_{l,m} = \frac{\sum_{i=1}^{nof(U)} \min\left(b_{i,l}, b_{i,m}\right)}{\sum_{i=1}^{nof(U)} b_{i,l}}, \qquad (3.3)$$

где $b_{i,l}$ и $b_{i,m}$ – степени принадлежности i-го объекта к кластерам U_l и U_m, соответственно. Однако мера сходства $I_{l,m}$ не является симметричной, так как $I_{l,m} \neq I_{m,l}$, поэтому для выявления указанного сходства предпочтительней использовать меру

$$Mc_{l,m} = \max\left(I_{l,m}, I_{m,l}\right), \qquad (3.4)$$

в соответствии с которой кластеры будут считаться сливающимися, если значение $Mc_{l,m}$ превышает некоторый порог h (при $h = 0$ все кластеры будут считаться слившимися, при $h = 1$ слившихся кластеров никогда не будет).

Если говорить об иерархической кластеризации, то наблюдать за картиной слияния кластеров лучше всего на ранних стадиях их формирования, т.к. на более поздних этапах происходит увеличение количества объектов в кластерах и как следствие вероятность полного слияния крупных кластеров снижается. Тогда можно будет говорить о расщеплении или дроблении кластеров. Слияние кластеров в итерационных методах кластеризации имеет определенные недостатки – не учитывается форма кластеров и близость их центров.

Изменения в структуре кластеров: расщепление или дробление кластеров.

Расщепление или дробление кластеров можно наблюдать, например, в момент времени t_{k+4}, когда некоторые кластеры увеличиваются в размерах из-за большого числа новых объектов, что может привести к неоднородности их внутренней структуры (рисунок 3.5).

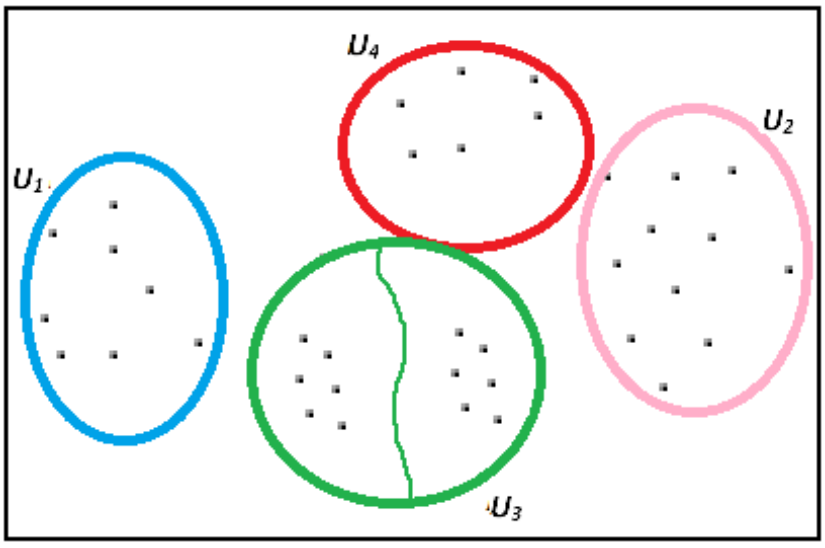

Рисунок 3.5 – Иллюстрация расщепления кластера U_3 и формирования внутри него двух разделённых сгустков в t_{k+4}

В качестве критерия для такого расщепления можно применить анализ

многоэкстремальности гистограмм признаков. В [13] предложено рассматривать кластер, как расщеплённый, если

$$R = \frac{f_{\max} - f_{\min}}{f_{\max}} \geq g\,, \qquad (3.5)$$

где f_{max} и f_{min} – значения глобальных максимума и минимума гистограммы [13] хотя бы для одного из компонентов характеристического вектора, а $g \in [\mathbf{0},\mathbf{1}]$ – пороговое значение.

В иерархической кластеризации расщепление или дробление кластеров можно выявить на более поздних этапах, т.к. на более ранних стадиях количество объектов и размерность кластеров не позволит наблюдать над процессом изменения однородности малых кластеров. Это связано не только с размерами кластеров, состав которых может измениться с течением времени, но и с характером самого алгоритма агломеративной кластеризации. На более ранних стадиях целесообразнее говорить о слиянии кластеров в иерархической системе.

Изменения в структуре кластеров: исчезновение кластеров.

Исчезновение кластеров напрямую связано с исчезновением объектов этих кластеров или с превращением их характеристических векторов в нулевые вектора.

Исчезновение кластера происходит, когда наблюдается полный переход его объектов в состав другого (других) кластера (кластеров). Если в момент времени t_{k+4} кластер $U_l(t_{k+4})$ содержал $nof(U_l(t_{k+4}))$ элементов, а кластер $U_m(t_{k+4})$ содержал $nof(U_m(t_{k+4}))$ элементов и $U_l(t_{k+4}) \cap U_m(t_{k+4}) = \varnothing$, то исчезновение кластера U_m в момент времени t_{k+5} будет объявлено, если $U_l(t_{k+5}) = U_l(t_{k+4}) \cup U_m(t_{k+4})$ и $U_m(t_{k+5}) = \varnothing$.

Другая причина исчезновения кластера связана с превращением характеристических векторов его объектов в нулевые вектора. Пусть в момент времени t_{k+4} $U_m(t_{k+4}) = \{u_{m^1}, ..., u_{m^i}, ..., u_{m^{nof(Um(tk+3))}}\}$, где $u_{m^i} \neq (0, ..., 0)$. Если в момент времени t_{k+5} $U_m(t_{k+5}) = \{\vec{0}_1, \vec{0}_2, ..., \overrightarrow{0_{nof(U_m(t_{k+3}))}}\}$, то кластер U_m перестаёт существовать и можно сказать, что произошло его исчезновение.

Изменения в структуре кластеров: перемещение центров или дрейф кластеров.

В иерархических алгоритмах кластеризации нет понятия «центр кластера». В частности, в агломеративном алгоритме на более высоких уровнях иерархии принадлежность к кластеру определяется минимальным расстоянием от любого элемента этого кластера: расчёт центров кластеров отсутствует. В методе *k*-средних на каждой итерации проводится перерасчёт координат центров и в этих случаях скачкообразные перемещения центров кластеров приводят к глобальным изменениям на уровне кластеров в целом и на уровне объектов в частности. С течением времени поведение ИП изменяется, их поисковые запросы, которые формируют поисковые векторы, могут вызывать медленные изменения положений центров кластеров – дрейф центров кластеров (рисунок 3.6).

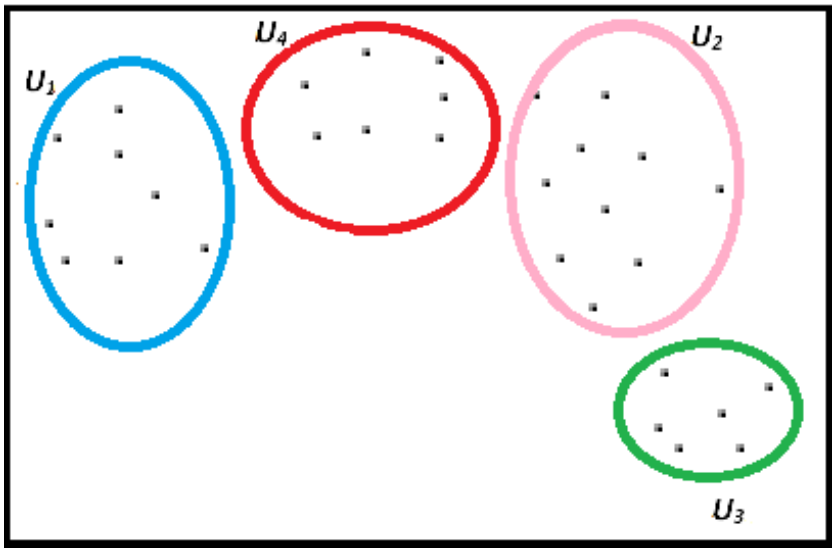

Рисунок 3.6 – Иллюстрация дрейфа кластеров в момент времени t_{k+6}

Такие изменения могут быть достаточно незначительными и носить непрерывный характер, но их целесообразно выявлять и отслеживать, поскольку они, в конце концов, могут стать причиной скачкообразных изменений кластерной структуры. Выявить степень дрейфа кластеров [50] можно с использованием меры компактности P_l кластера U_l по отношению к отнесенным к нему объектам:

$$P_l = \frac{\sum_{i=1}^{nof(U_l^*)} b_{i,l}^2 \times \rho(u_i^*, e_l)^2}{\sum_{i=1}^{nof(U_l^*)} b_{i,l}}, \qquad (3.6)$$

где

- $nof(U_l^*)$– число хороших объектов l-ого кластера, для которых значение степени принадлежности к указанному кластеру $0 \le b_{i,l} \le 1$;

- $b_{i,l}$ – коэффициент принадлежности объекта u_i кластеру U_l , $0 \le b_{i,l} \le 1$;

- $\rho(u_i*,e_l)$ – евклидово расстояние между хорошим объектом $u_i* \in U^*_l$ и центром e_l кластера U_l;

Дрейф имеет место, если компактность кластера, рассчитанная для двух последних временных окон, уменьшается, т.е. если справедливо неравенство

$$P_l(t_{k+6}) = P_l(t_{k+6}) - P_l(t_{k+5}) < q, \qquad (3.7)$$

где $q < 0$ – заданная пороговая величина.

Если условие (3.7) выполняется, то необходимо проводить перерасчёт координат центра кластера U_l.

3.2. Переход от динамической к статической кластеризации с применением числовых коэффициентов усиления

Интернет-пользователей по их поведению никак нельзя отнести к статическим объектам – в течение суток они характеризуются разной поисковой активностью в сети. Динамическая активность ИП связана, в первую очередь, с его географическим местом, полом, возрастом, социальным положением и, конечно, с его распорядком дня. Тем не менее, различают статическую и динамическую кластеризацию. Статическая кластеризация имеет место, если число кластеров, их размеры и центры с течением времени не изменяются. Однако если с течением времени кластерная структура меняется, объекты кластеризации начинают менять свои свойства, образуя новые кластеры, или переходят из одного кластера в другой, то в этом случае требуется динамическая кластеризация. Статические методы кластеризации — иерархические или

итерационные (см. приложение 3) хорошо известны и широко применяются для классификации различного рода объектов. Методы динамической кластеризации применяются реже, так как требуют существенно б*о*льших вычислительных затрат, обусловленных периодическим наблюдением за множеством классифицируемых объектов, обработкой полученных данных и формированием кластерных структур.

Очевидно, что персонализация Интернет-поиска должна базироваться на кластеризации, как пользователей, так и информационных ресурсов. Необходимо отслеживать поисковую историю пользователей, объединяя их в кластеры по интересам и содержание ресурсов, объединяя их в кластеры по темам. Вопрос заключается в том, можно ли применять статические методы кластерного анализа к динамическим объектам? Как можно снизить влияние динамики исследуемых объектов на качество кластеризации, а, следовательно, и на их классификацию?

Метод экспериментального исследования динамики кластерных структур Интернет-пользователей и Интернет-ресурсов.

Исследование динамики кластерных структур предполагает параллельное наблюдение, как за пользователями, так и за ресурсами. В дискретные моменты времени $t_k \in T$, $k = 0,1,\ldots$, осуществляется формирование числовых векторов, характеризующих текущую поисковую активность пользователей и текущее содержание ресурсов. Эксперимент начинается с построения указанных векторов в момент времени t_0, расчёта евклидового расстояния между объектами, формирования кластеров пользователей и ресурсов. Через интервалы времени $\Delta t = t_{k+1} - t_k$ осуществляется проверка стабильности ранее сформированной кластерной структуры.

Наблюдение проводится в течение 24 часов с интервалом Δt равным 1 часу. При анализе кластерной структуры учитываются только результаты поисковой активности за последние 4 часа (период в 4 часа подтверждается результатами экспериментов таблицы 3.1 – это максимальный период времени в течение которого пользователь может интересоваться конкретной темой), включая последнюю выборку в момент наблюдения. Это позволяет организовать своего

рода скользящее временное окно наблюдения и исключить влияние слишком старых наблюдений – данные старых наблюдений удаляются. На основании полученной информации и математического анализа динамики кластерной структуры делается вывод о целесообразности применений статической или динамической кластеризации, как для пользователей, так и для ресурсов.

Анализ поисковой активности пользователей в зависимости от географии и времени суток позволяет на неформальном уровне сделать вывод о необходимости применения динамической кластеризации для их классификации. Динамическая кластеризация позволяет проводить классификацию пользователей в любой момент времени и на любой территории. А какой метод кластеризации лучше применить для ресурсов? На первый взгляд динамическая кластеризация должна быть применена и для них, так как содержание ресурсов, включающих достаточно большое число динамических компонентов, постоянно меняется. Но так ли это?

Пусть U – множество наблюдаемых пользователей, u_i – i-ый наблюдаемый пользователь, $u_i \in U$. Если производится наблюдение за поисковой активностью пользователя u_i, то в произвольный момент времени $t_k \in T$ можно сформировать характеристический (поисковый) вектор i-го пользователя, имеющий следующий вид:

$$u_i(t_k) = (u_{i,1}(t_k), \ldots, u_{i,j}(t_k), \ldots, u_{i,nof(Vu)}(t_k)),$$

где

- $u_{i,j}(t_k)$ – числовые координаты j-го поискового термина из глобального словаря терминов V_u в момент времени t_k, равный числу вхождений этого термина в запросы i-го пользователя, выполненные в течение соответствующего временного окна;

- $nof(V_u)$ – размер поискового вектора i-го пользователя, равный числу слов в глобальном словаре терминов.

Числовые координаты $u_{i,j}(t_k)$, $1 \le j \le nof(V_u)$, расположены в характеристическом векторе в том же лексикографическом порядке, что и термины в словаре V_u. Переход от вербального к числовому представлению

результатов поисковой активности происходит за счет позиционного кодировании терминов и подсчёта числа их вхождений в запросы пользователя. Другой, известный способ представления вербальной информации, использующий показатель *TF-IDF* был исключен из рассмотрения, так как общее число слов, применяемое для расчётов, не является постоянным. Кроме того, предлагаемый способ хорошо согласуется с методами реляционного представления и обработки информации, основанными на применении *SQL*-языка.

Пусть W – множество наблюдаемых ресурсов, w_i – наблюдаемый i-ый ресурс, $w_i \in W$. В произвольный момент времени $t_k \in T$ можно представить i-й ресурс характеристическим вектором следующего вида:

$$w_i(t_k) = (w_{i,1}(t_k), \ldots, w_{i,j}(t_k), \ldots, w_{i,nof(Vw)}(t_k)),$$

где

- $w_{i,j}(t_k)$ – числовые координаты j-го поискового термина из глобального словаря терминов V_w в момент времени t_k, равный числу вхождений этого термина в текст i-го ресурса, в течение наблюдаемого временного окна.

Для работы с текстовым содержанием ресурсов используем *DOM*- модели их страниц. *Document Object Model* (*DOM*) – объектная модель Интернет-документа, позволяющей получить доступ к текстовому содержимому, как статических, так и динамических (содержащих динамические компоненты) страниц наблюдаемых ресурсов. Следует отметить, что даже при статичном *HTML*-коде страниц, их объектные модели могут иметь динамический характер. Поэтому кластеризация ресурсов, основанная на прямом анализе *HTML*-кода страниц, не является целесообразной.

Проблема влияния динамических компонентов на поведение ресурсов в кластерных структурах была выявлена после длительного наблюдения за содержанием их *DOM*-моделей – ресурсы могут все время перемещаться между кластерами. Более того, наблюдения показывают, что для *DOM*-моделей ресурсов характерна периодичность, то есть через $n \times t_k$ моментов времени текстовое содержание этих ресурсов повторяется.

В научно-технической литературе рассмотрены различные математические

методы для определения состояния кластеров в кластерных структурах. В одних источниках предлагается использовать степень принадлежности объекта к кластеру на основании матрицы разбиения, в других – рекомендуется рассчитывать меру компактности объектов внутри кластера. Имеется множество иных численных показателей для анализа состояния кластеров, таких как, например, предполагаемое число новых кластеров и степень дрейфа объектов внутри них. В рамках данной главы используется формула расчёта степени принадлежности объектов к разным кластерам (формула 3.1)

Результаты исследования динамики кластерных структур в выбранных фокус-группах.

Для исследования динамических эффектов в кластерных структурах, были выбраны 30 пользователей в возрасте от 18 до 44 лет, проживающих в Москве (самая активная группа по данным исследовательской компании *TNS* [82]) и 100 информационных ресурсов, принадлежащих группе новостных сайтов с различной тематикой. Таким образом, применена предварительная группировка объектов по статической информации: пол, возраст и место проживания для ИП и тематика для ИР.

С помощью формулы (3.1) экспериментально были определены показатели принадлежности, отражающие состояния кластерной структуры для случайно выбранного ИП из выбранной группы (таблица 3.1). Полученные результаты графически интерпретированы на рисунке 3.7. Графики подтверждают динамический характер поисковой активности пользователей: в зависимости от времени изменяется поисковый интерес пользователя и, как следствие, его принадлежность к кластерам.

Таблица 3.1 – Коэффициенты принадлежности пользователей к кластерам в разные моменты времени

Кластер / **Момент времени (t_k)**	U_1	U_2	U_3	U_4	**Кластер** / **Момент времени (t_k)**	U_1	U_2	U_3	U_4
t_0=8	0	0	0	0	**20**	0	0	0	0
9	0	0	0	0	**21**	0	0	0	0

10	0,5104	0,2713	0,0597	0,1586	**22**	0,2007	0,5915	0,1928	0,0150
11	0,6061	0,1870	0,1559	0,0510	**23**	0,4092	0,4497	0,1294	0,0117
12	0,1775	0,1297	0,2011	0,4917	**24**	0,3907	0,3057	0,0937	0,2099
13	0,1896	0,1582	0,3959	0,2563	**1**	0	0	0	0
14	0,0436	0,1192	0,5991	0,2381	**2**	0	0	0	0
15	0,0574	0,0857	0,5513	0,3056	**3**	0	0	0	0
16	0,1601	0,2294	0,3491	0,2614	**4**	0	0	0	0
17	0,2151	0,1697	0,2967	0,3185	**5**	0	0	0	0
18	0,0653	0,1196	0,4892	0,3259	**6**	0	0	0	0
19	0,2448	0,0674	0,4691	0,2187	**7**	0	0	0	0

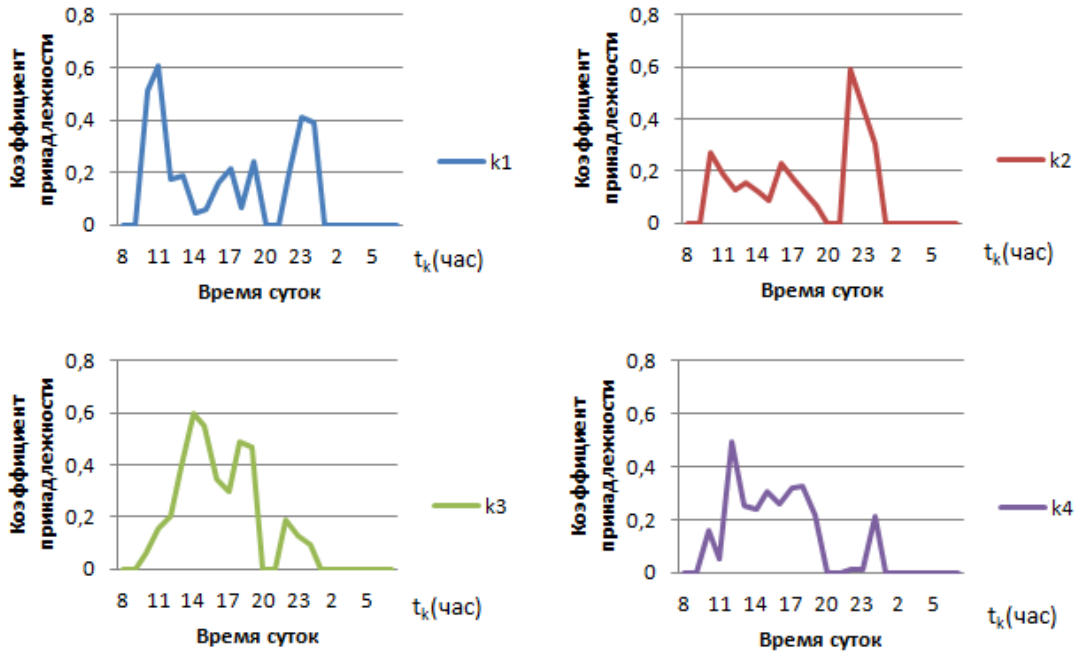

Рисунок 3.7 – Графики изменения коэффициента принадлежности пользователя для разных кластеров в разные моменты времени.

Результаты расчёта коэффициентов принадлежности ресурса к кластерам (таблица 3.2) подтверждают динамичность характеристик ресурсов. На рисунке 3.8 построенные по данным таблицы 3.2 графики скрещиваются, то есть в разные моменты времени наблюдаемый Интернет-ресурс принадлежит разным кластерам.

Для устранения этого эффекта предлагается использовать *DOM*-модель ресурсов с целью их «очистки» от динамических компонентов: информации партнёрских сетей, различного рода сообщений, рекламы и т.д. На ряду с этим *DOM*-модель может быть использована для расчета весовых коэффициентов

характеристического вектора ресурса.

Таблица 3.2 – Коэффициенты принадлежности ресурса к кластерам в разные моменты времени без применения весовых коэффициентов усиления

Кластер / Момент времени (t_k)	W_1	W_2	W_3	Кластер / Момент времени (t_k)	W_1	W_2	W_3
t_0=8	0,483	0,208	0,31	**20**	0,303	0,339	0,358
9	0,382	0,397	0,221	**21**	0,411	0,289	0,300
10	0,419	0,307	0,274	**22**	0,419	0,3	0,281
11	0,362	0,306	0,332	**23**	0,308	0,352	0,34
12	0,233	0,338	0,429	**24**	0,15	0,46	0,389
13	0,31	0,363	0,327	**1**	0,31	0,275	0,416
14	0,309	0,441	0,25	**2**	0,331	0,343	0,326
15	0,393	0,257	0,351	**3**	0,377	0,282	0,341
16	0,45	0,313	0,238	**4**	0,358	0,267	0,375
17	0,401	0,173	0,426	**5**	0,279	0,451	0,270
18	0,311	0,386	0,303	**6**	0,352	0,308	0,341
19	0,157	0,449	0,394	**7**	0,345	0,297	0,358

Учитывая особенности *DOM*-модели Интернет-страниц ресурса, каждому элементу вектора $w_i(t_k)$ может быть сопоставлен весовой коэффициент, рассчитанный по формуле

$$\frac{w_p \times k_1}{nW} \times k_2 \times 100, \qquad (3.8)$$

где nW – общее число слов в *DOM*-модели страницы; w_p – число вхождений слова на p-ой позиции в конкретном тэге *DOM*-модели страницы; k_1 – коэффициент усиления ($k_1 > 1$), значение которого рассчитывается, исходя из наименования тэга, определяющего контекст слова на странице; k_2 – коэффициент усиления ($k_2 \geq 1$), значение которого рассчитывается по формуле отношений площадей, занимаемых словами на странице:

$$k_2 = \frac{S_p / cnt_p}{S_{total} / nW}, \qquad (3.9)$$

где S_p – площадь области на p-ой позиции; cnt_p – число слов на p-ой позиции; S_{total} – площадь информационного текста; nW – общее число слов в *DOM*-модели страницы.

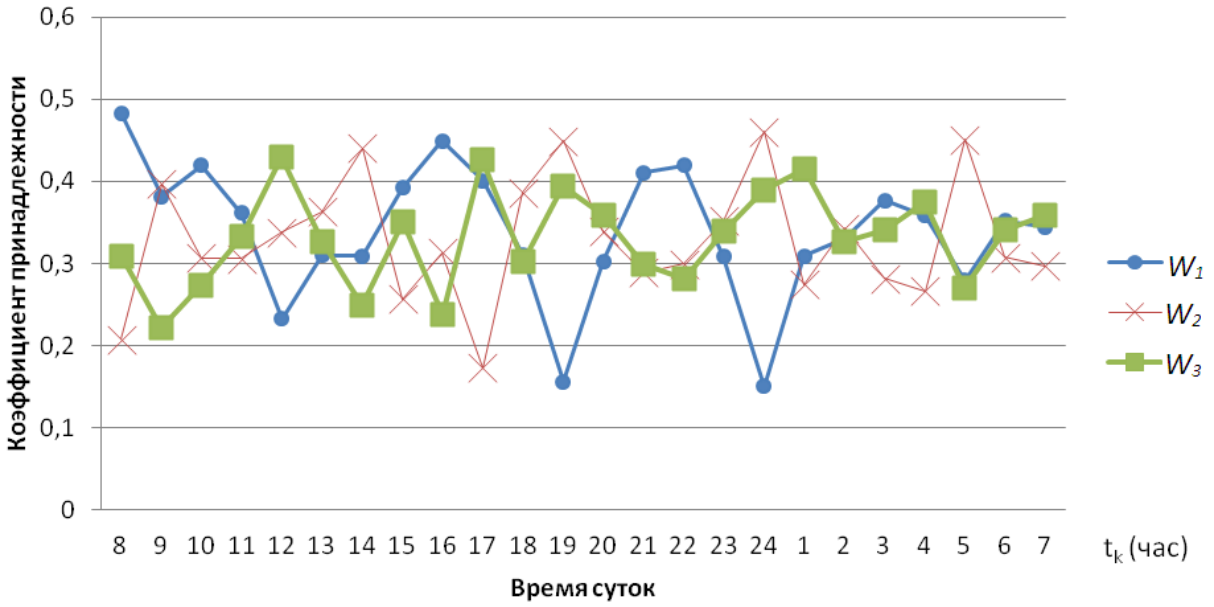

Рисунок 3.8 – Графики коэффициентов принадлежности ресурса для разных кластеров в разные моменты времени без использования весовых коэффициентов усиления.

Проведём повторный расчёт степеней принадлежности наблюдаемого ресурса (таблица 3.3) с применением весовых коэффициентов, рассчитанных по формуле 3.8 и получим новые графики (рисунок 3.9).

Таблица 3.3 – Принадлежность ресурса к кластерам в разные моменты времени с применением весовых коэффициентов усиления

Кластер / Момент времени (t_k)	W_1	W_2	W_3	Кластер / Момент времени (t_k)	W_1	W_2	W_3
t_0=8	0,207	0,533	0,260	**20**	0,089	0,559	0,352
9	0,167	0,602	0,231	**21**	0,350	0,592	0,059
10	0,270	0,481	0,249	**22**	0,071	0,586	0,343
11	0,307	0,560	0,134	**23**	0,255	0,487	0,259
12	0,329	0,624	0,047	**24**	0,214	0,628	0,158
13	0,13	0,613	0,257	**1**	0,190	0,605	0,204
14	0,222	0,581	0,197	**2**	0,316	0,530	0,154
15	0,320	0,547	0,133	**3**	0,222	0,481	0,298
16	0,219	0,575	0,206	**4**	0,264	0,528	0,208
17	0,119	0,500	0,381	**5**	0,360	0,604	0,035
18	0,249	0,635	0,116	**6**	0,239	0,502	0,259
19	0,298	0,567	0,135	**7**	0,087	0,528	0,385

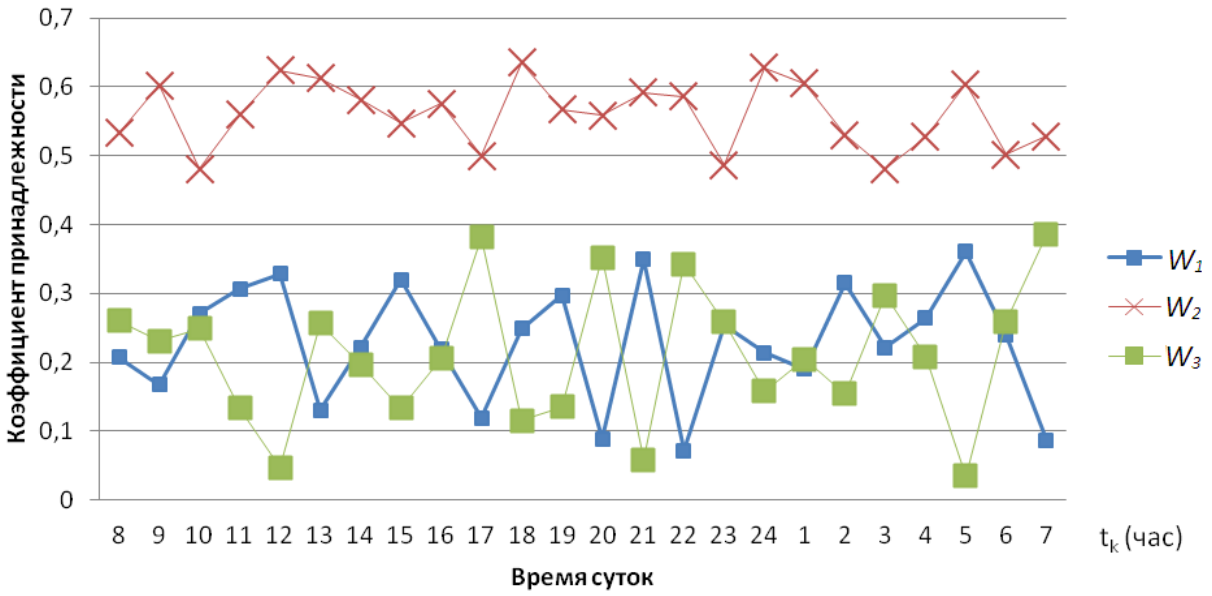

Рисунок 3.9 – Графики коэффициентов принадлежности ресурса для разных кластеров в разное время суток с применением весовых коэффициентов.

Использование коэффициентов усиления, построенных на основании *DOM*-модели, приводит к сдвигу расчетного коэффициента принадлежности в сторону одного из кластеров (график W_2). Остальные графики смещаются вниз. До применения коэффициентов усиления средние значения коэффициентов принадлежности к кластерам W_1 и W_3 равны 0.340 и 0.331, а после их применения снизились до 0.229 и 0.211, соответственно.

С целью персонализации Интернет-поиска был проведён кластерный анализ пользователей и ресурсов. Если для кластеризации пользователей целесообразно применять динамический подход, то динамическая кластеризации ресурсов приводит к неопределённостям: в разные, но близкие моменты времени информационные ресурсы могут принадлежать разным кластерам. Причиной этому является наличие интерактивных динамических компонентов, которые периодически обновляют свое содержание. На рисунке 3.8 видно, что при применении динамической кластеризации коэффициенты принадлежности одного и того же ИР к кластерам находятся в постоянной вариации, следовательно, подтвердить его принадлежность к конкретному кластеру невозможно. Использование коэффициентов усиления, построенных на основании *DOM*-

модели, приводит к сдвигу расчетного коэффициента принадлежности в сторону одного из кластеров (рисунок 3.9, график W_2). Таким образом, ИР, которые на первый взгляд были динамическими, утрачивают это свойство. Предложенный метод кластеризации ресурсов с применением коэффициентов усиления можно применять для любых современных ИР. Это позволяет отказаться от динамической кластеризации ресурсов, сохраняя на высоком уровне степень их принадлежности тем или иным кластерам и, как следствие, обеспечить стабильность кластерной структуры в целом.

3.3. Трёхтактная кластеризация Интернет-ресурсов с применением *DOM*- фильтрации

Интернет-ресурсы – это сайты или отдельные страницы сайтов, содержащие текстовую информацию, представляющую интерес для ИП. С помощью специальных методов ИР индексируются и ссылки на них выдаются как результаты поисковых запросов ИП. К сожалению, фактом является то, что большинство найденных ИР почти не содержит информации, отвечающей поисковым интересам пользователей. Одним из подходов к разрешению этой проблемы является классификация ИР с применением методов кластерного анализа [42].

В последнее время произошли значительные изменения в средствах реализации Интернет-ресурсов: стали применяться языковые кодировки содержимого текста (*Unicode Transformation Format UTF*-8 и *UTF*-16, *Windows* 1250 и 1251 и др.), *Java*-скрипты (*JavaScript – JS*), использующие технологию асинхронных интерактивных интерфейсов (*Asynchronous JavaScript and XML – AJAX*) и, конечно, каскадные таблицы стилей (*Cascading Style Sheets – CSS*). Внедрение современных технологий сделали ИР более адаптивными и интерактивными, а значит и более динамичными. Главная страница любого новостного сайта (например, сайта *news.mail.ru*) является ярким примером динамичного ИР. В разные моменты времени б*о*льшая часть компонентов *DOM*-

модели ИР [84] меняет свое текстовое содержание. *HTML*-теги могут представлять всего лишь каркасную основу для браузеров, так как их текстовое содержание инкапсулировано в *DOM*-модели и может динамически изменяться с каждой новой загрузкой в браузер или после наступления определенных событий.

С точки зрения кластерного анализа, появление динамических компонентов ИР – *JS*-скриптов и *CSS* – привело к серьезным изменениям в поведении кластерных структур ИР. Для статического *HTML*-кода достаточно единожды провести кластеризацию заданного множества ресурсов и полученная структура долгое время останется неизменной. *HTML*-страницы, содержащие исключительно статические компоненты, легко поддаются классификации методом частоты слов (*Term Frequency – TF*) [74], так как они могут быть рассмотрены, как обычные текстовые документы [18]. Для динамических ИР всё зависит от текстового содержания динамических компонентов *DOM*-модели – кластеры перестают быть неподвижными. Они могут меняться, как качественно (перемещение центров кластеров), так и количественно (изменение числа и размеров кластеров). Как можно добиться хорошей степени кластеризации ИР при наличии динамических компонентов? Какие меры необходимо принять для снижения (или устранения) динамических эффектов в кластерной структуре при кластеризации ИР?

В данном параграфе решается задача кластеризации ИР с динамическими компонентами методом, предполагающим использование особенностей их *DOM*-моделей. Предлагаемый метод, в отличие от представленных в [3, 57], позволяет автоматически определять и устранять влияние динамических компонентов ИР на их кластерную структуру.

Метод экспериментального исследования динамики кластерных структур.

Исследование динамики кластерных структур основывается на наблюдении за ИР.

Пусть $T = \{t_0, t_1, \ldots, t_k, \ldots\}$ – упорядоченное по возрастанию множество дискретных моментов времени. Наблюдение за ИР начинается в момент времени

t_0, а в моменты времени t_k производится обработка результатов наблюдения, полученных в интервалах (t_{k-1}, t_k), $k \geq 1$.

Наблюдение осуществляется за ИР, образующими множество $W = \{w_1, \ldots, w_i, \ldots, w_{nof(W)}\}$, где nof – функция, возвращающая в качестве значения число элементов в конечном множестве-аргументе.

Рассмотрим произвольный момент времени t_k, $k \geq 1$. В ходе наблюдения за ИР $w_i \in W$ в момент времени t_k формируется словарь терминов ресурса $Vw_i(t_k)$, включающий $nof(Vw_i(t_k))$ терминов. Этот словарь по определенным правилам получается из словаря $Vw_i(t_{k-1})$ и новых уникальных терминов – результатов наблюдения за ИР w_i в интервале (t_{k-1}, t_k).

Из словарей терминов ресурсов $Vw_i(t_k)$ формируется глобальный словарь терминов $V(t_k) = \{v_1(t_k),\ldots,v_j(t_k),\ldots,v_{nof(V(t_k))}(t_k)\}$ такой, что

$$V(t_k) = \bigcup_{i=1}^{nof(W)} Vw_i(t_k).$$

Следует отметить, что $nof(Vw_i(t_0)) = 0$ и для любого $t_k \in T$ $nof(Vw_i(t_k)) \geq nof(Vw_i(t_{k-1}))$. Аналогично, для глобального словаря терминов – $nof(V(t_0)) = 0$ и для любого $t_k \in T$ $nof(V(t_k)) \geq nof(V(t_{k-1}))$.

В момент времени t_k для каждого ИР $w_i \in W$ строится характеристический вектор $w_i(t_k) = (w_{i,1}(t_k),\ldots,w_{i,j}(t_k),\ldots,w_{i,nof(V(t_k))}(t_k)))$, где $w_{i,j}(t_k)$ – число вхождений термина $v_j(t_k) \in V(t_k)$ в текстовый контент ИР w_i в интервале наблюдения (t_{k-1}, t_k).

На основе характеристических векторов ресурсов из W, с помощью какого-либо известного алгоритма кластеризации строится кластерная структура ИР, соответствующая моменту времени t_k, – $C(t_k) = \{c_1(t_k), \ldots, c_m(t_k), \ldots, c_{nof(C(t_k))}(t_k)\}$. Кластер $c_m(t_k)$ может быть представлен парой

$c_m(t_k) = \langle W_m(t_k), z_m(t_k)\rangle$, где $W_m(t_k) \subseteq W$ – множество ИР, отнесенных к m-му кластеру в момент времени t_k, $z_m(t_k)$ – центр кластера $c_m(t_k)$.

Кластерную структуру $C(t_k)$ будем называть стабильной структурой, если для любого момента времени $t_{k*} > t_k$ $V(t_{k*}) = V(t_k)$ и при этом $nof(C(t_{k*})) = nof(C(t_k))$ и для любого кластера $c_m(t_{k*}) \in C(t_{k*})$ выполняются следующие соотношения:

$W_m(t_{k*}) = W_m(t_k)$, $\rho(z_m(t_{k*}), z_m(t_k)) \leq \varepsilon$, где $\rho(z_m(t_{k*}), z_m(t_k))$ – евклидово расстояние между центрами m-го кластера в моменты времени t_{k*} и t_k, а ε – произвольное число, много меньшее радиуса кластера в момент времени t_k.

Множество ИР W назовем статичным, если в процессе наблюдения за ним в некий момент времени $t_k \in T$ будет построена стабильная кластерная структура $C(t_k)$. В противном случае множество W будем называть динамичным множеством ИР.

Интерес представляют последствия расширения исходного множества наблюдаемых ИР за счет добавления к ним новых ИР. В этом плане можно рассмотреть следующую задачу.

Пусть множество ИР W расширяется за счет нового, не принадлежащего W ресурса. Повлечет ли это значительные изменения в кластерной структуре ИР или прежняя структура сохраниться, а новый ИР просто будет добавлен в один из имеющихся кластеров.

Пусть W – множество ИР, наблюдаемых до момента времени t_0 включительно, w – новый, добавляемый ИР. В произвольный момент времени $t_k \in T$, $k \geq 1$, указанный ИР можно представить характеристическим вектором

$$w(t_k) = (w_1(t_k),..., w_j(t_k),..., w_{nof(V(t_k))}(t_k)),$$

где

- $w_j(t_k)$ – вес j-ого поискового термина из глобального словаря терминов $V(t_k)$, равный числу вхождений этого термина в текст ресурса w в течение временного интервала (t_{k-1}, t_k);

- $nof(V(t_k))$ – размер характеристического вектора ИР, равный числу слов в глобальном словаре терминов в момент времени t_k.

Числовые координаты $w_j(t_k)$, $1 \leq j \leq nof(V(t_k))$, расположены в характеристическом векторе в том же порядке, что и термины в словаре $V(t_k)$. Переход от вербального к числовому представлению результатов происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений в текст ИР.

Для оценки стабильности кластерной структуры используем следующие метрики.

1. Степень принадлежности $b_m(t_k)$ (формула 3.1).

2. Приращение кардинальности характеристического вектора ИР:

$$\Delta nof(V(t_k)) = nof(V(t_k)) - nof(V(t_{k-1})),$$

где $nof(V(t_k)) \geq nof(V(t_{k-1}))$ для двух непосредственно следующих друг за другом моментов дискретного времени.

Разработанная система анализа тестового содержания ИР показала, что многократный анализ содержания одного и того же ИР в разные моменты времени может привести к формированию совершенно разных характеристических векторов. Сформированные в разные моменты времени $t_{k'} \neq t_{k''}$ характеристические вектора $w(t_{k'})$ и $w(t_{k''})$ могут отличаться, как качественно (изменяется число вхождений терминов в текст), так и количественно (изменяется значение кардинальности характеристических векторов).

Результаты анализа текстового содержания одной из новостных страниц сайта *news.mail.ru* приведены в таблице 3.4.

Таблица 3.4 – Число вхождений терминов в текст ИР

	v_1	v_2	v_3	…	v_j	…	$v_{nof(V(t_0))}$	$v_{nof(V(t_0))+1}$	$v_{nof(V(t_0))+2}$	…	$v_{nof(V(t_k))}$	$nof(V(t_k))$
t_0	1	2	1		1		3	-	-		-	276
t_1	1	2	1		2		3	1	1		-	331
t_2	1	2	1		0		3	0	1		-	376
…												
t_9	1	2	1		0		3	1	0		2	388
t_{10}	1	2	1		0		3	1	0		2	388

Здесь в заголовках столбцов стоят термины из глобального словаря, в строках – моменты дискретного времени t_k, в ячейках – числа вхождений терминов в текст наблюдаемого ИР $w_j(t_k)$. Параметр $nof(V(t_k))$ указывает размер глобального словаря терминов или, что то же самое, кардинальность характеристического вектора $w(t_k)$.

Данные таблицы 3.4 были использованы для расчёта частоты употребления терминов в тексте ИР. Результаты зафиксированы в таблице 3.5, в ячейках которой представлены значения частоты употребления терминов $f_j(t_k)$ в

соответствующие моменты времени.

Таблица 3.5 – Частота употребления терминов×10^4

	v_1	v_2	v_3	..	v_j	…	$v_{nof(V(t_0))}$	$v_{nof(V(t_0))+1}$	$v_{nof(V(t_0))+2}$	…	$v_{nof(V(t_k))}$	$nof(Vw(t_k))$
t_0	33	65	33		33		98	0	-		-	276
t_1	32	64	32		64		96	32	33		-	331
t_2	34	68	34		0		100	0	34		-	376
…												
t_9	33	66	33		0		99	33	0		66	421
t_{10}	34	67	34		0		100	34	0		67	421

Через $nof(Vw(t_k))$ в таблице 3.5 обозначено общее число терминов в тексте ИР.

Анализ данных, приведённых в таблицах 3.4 и 3.5, позволяет сделать следующие выводы.

1. С увеличением числа наблюдений кардинальность вектора $w(t_k)$ возрастает и стремится к некому предельному значению. Использование классических методов кластеризации для динамических векторов $w(t_k)$ приводит к снижению качества результатов кластерного анализа, так как с каждым наблюдением появляются лишние координаты – увеличивается размерность пространства, в котором проводится эксперимент, усложняются расчёты.

2. Снижается значение частоты употребления слов, так как $f_j(t_k)$ обратно пропорционально общему числу терминов $nof(Vw(t_k))$, содержащему, как статические термины основного теста ИР, так и динамические термины.

3. Формируются три множества терминов, из которых состоит исследуемый объект:

- множество терминов с постоянным числом вхождений $V_A = \{v_j \mid w_j(t_k) = const$ и $w_j(t_k) \neq 0$ для любого момента времени $t_k \in T\}$. Термины, принадлежащие к этому множеству, можно отнести к статическим компонентам ИР;

- множество терминов с переменным числом вхождений, появившихся хотя бы один раз $V_B = \{v_j \mid w_j(t_k) \neq const$ и $w_j \neq 0$ для любого момента времени $t_k \in T\}$. Термины, принадлежащие к этому множеству, относятся, как к статическим, так и к динамическим компонентам ИР;

- множество терминов, имеющих хотя бы одно нулевое число вхождений V_C

= $\{v_j \mid$ существует $t_k \in T$, для которого $w_j = 0\}$. Термины, принадлежащие к этому множеству относятся к динамическим компонентам ИР.

Указанные множества позволяют определить статические и динамические компоненты ИР. После формирования множеств V_A, V_B и V_C можно вычислить иерархический путь этих компонентов в *DOM*-модели, а затем их автоматически исключить из всех страниц исследуемого ИР.

По данным в таблице 3.4 (столбец $nof(V(t_k))$) можно построить график приращения кардинальности характеристического вектора $\Delta nof(V(t_k))$, показанный на рисунке 3.10.

Рисунок 3.10 – График приращения кардинальности вектора характеристик

Из рисунка 3.10 видно, что система достигает стабильного состояния за конечное число наблюдений $N \approx 5$. Эта величина может быть использована в качестве признака насыщения словаря. Для разных страниц одного и того же сайта N можно считать константой.

Варьирование элементов и кардинальностей характеристических векторов напрямую влияет на степень принадлежности исследуемых объектов к кластерам. Если поставить задачу определения принадлежности исследуемого объекта в разные моменты времени $t_k \in T$, то можно будет наблюдать за его перемещением между сформированными кластерами: в целом меняется кластерная структура, а в частности – принадлежность исследуемых объектов к конкретным кластерам. Из

этого следует, что классические методы кластеризации текстовых документов [3], не могут быть применены к динамическим объектам. Поэтому необходим новый, более совершенный механизм кластеризации ИР с динамическими компонентами.

Метод трехтактной кластеризации динамичных ИР с обратной связью.

Для кластеризации ИР с динамическими компонентами предлагается использовать метод трёхтактной кластеризации ИР, схема реализации которого показана на рисунке 3.11.

На вход схемы поступают *URL*-адреса Интернет-страниц – объектов кластеризации. Задачей первого блока является *DOM*-фильтрация – выявление компонентов *DOM*-модели страницы с динамическими признаками и их удаление. Второй блок отвечает за формирование характеристического вектора на основе анализа исключительно статических компонентов *DOM*-модели Интернет-страницы. Третий блок непосредственно осуществляет кластеризацию объекта.

Рисунок 3.11 – Схема трехтактной кластеризации динамических ИР

Предложенная схема кластеризации имеет ряд преимуществ по сравнению с классическими методами кластеризации [3]. Во-первых, снижаются вычислительные затраты, что связано с уменьшением размеров характеристических векторов $w(t_k)$. Во-вторых, повышается точность результата кластеризации, так как сразу выделяются «мусорные» динамические компоненты страниц, которые перестают участвовать в кластеризации. Снижается уровень шума, повышается стабильность кластерной структуры.

При первом поступлении в систему, в момент времени t_1, исследуемый ИР «расщепляется» на отдельные *DOM*-компоненты и формируется характеристический вектор $w(t_1)$. При повторном наблюдении, в момент времени t_2, после формирования вектора $w(t_2)$ и его сравнения с вектором $w(t_1)$, выделяются основные динамические компоненты. Например, для новостных

страниц сайта *news.mail.ru* после повторного наблюдения выделяются более 60% всех динамических компонентов, а после третьего наблюдения процент их обнаружения вырастает до 80%. Для более сложных сайтов число наблюдений растет (например, для сайта *rbc.ru* число необходимых наблюдений может достичь 10-ти и более итераций). Учитывая сказанное, в блоке фильтрации предусмотрена обратная связь, которая срабатывает до тех пор, пока приращение кардинальности характеристического вектора $\Delta nof(V(t_k))$ не становится близким к нулю.

Прогонка Интернет-страниц через *DOM*-фильтр, исследование их содержания без и с его применением позволили получить следующие зависимости для характеристического вектора ИР (таблица 3.6, рисунок 3.12).

Таблица 3.6 – Кардинальность вектора $w(t_k)$ без и с применением *DOM*-фильтрации

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
Без применения *DOM*-фильтрации	276	331	376	395	404	411	411	416	421	421
С применением *DOM*-фильтрации	283	200	184	180	179	179	175	175	175	172

Рисунок 3.12 – Графики зависимости кардинальности вектора характеристик от числа наблюдений без и с применением *DOM*-фильтрации.

Из рисунка 3.12 следует, что без применения *DOM*-фильтрации число

элементов в характеристическом векторе возрастает по мере увеличения числа наблюдений, а в случае его применения – оно уменьшается.

Исследуемый ИР покидает блок *DOM*-фильтрации, обладая высокой степенью статичности, и попадает на следующий блок, задача которого состоит в формировании вектора $w(t_k)$. На вход блока формирования поступает характеристический вектор ИР с минимальным числом динамических компонентов. Результаты сравнения векторов, построенных без и с применением *DOM*-фильтрации могут сильно отличаться, в зависимости от степени статичности ИР (см. таблицу 3.6). На последнем этапе исследуемый объект попадает в блок кластеризации, функция которого состоит в кластерном анализе и формировании кластерной структуры поступающих объектов одним из известных методов [15].

Применение *DOM*-фильтрации для кластеризации ИР с динамическими компонентами улучшает значения их степеней принадлежности к релевантным кластерам (рисунок 3.13).

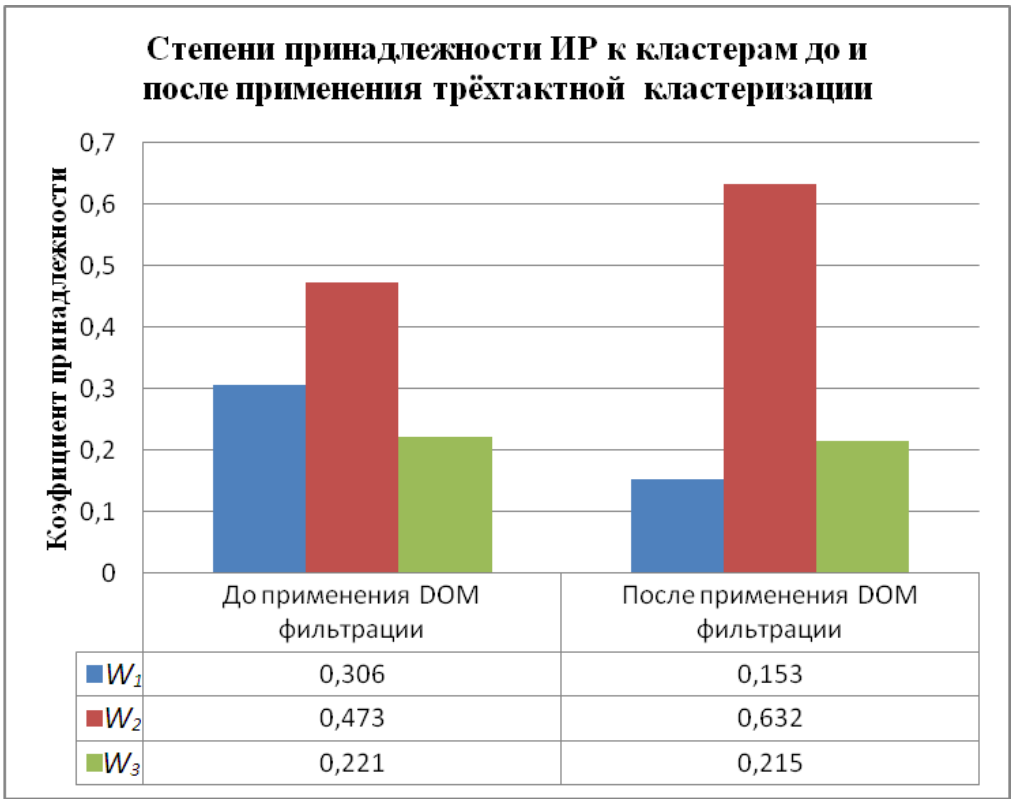

Рисунок 3.13 – Степени принадлежности ИР кластерам до и после применения трёхтактной кластеризации.

Применение трёхтактной схемы кластеризации приближает исследуемый ИР к кластеру W_2, повышает значение степени принадлежности ресурса к релевантному кластеру примерно на 25%. Степень принадлежности ресурса к другим, менее релевантным кластерам кластерной структуры, уменьшается.

Экспериментальный анализ динамичных ИР, в качестве которых выступили новостные сайты, выявил низкую степень их принадлежности к кластерам в соответствующих кластерных структурах, что делает эти кластерные структуры нестабильными. Причиной этому является наличие интерактивных динамических компонентов ИР, которые периодически обновляют свое содержание.

Для увеличения стабильности кластерной структуры предложено использовать *DOM*-фильтр, повышающий степень принадлежности ИР к кластерам, снижающий кардинальность их характеристических векторов.

Основанный на *DOM*-фильтрации метод трехтактной кластеризации ИР обеспечивает стабильность формируемых кластерных структур, его можно применять для анализа любых современных Интернет-ресурсов.

3.4. Выбор методов кластеризации Интернет-пользователей и Интернет-ресурсов, прошедших *DOM*-фильтрацию

Исследуемые Интернет-объекты характеризуются высокой степенью динамичности: поведение ИП в течение всего дня является мало предсказуемым, а ИР содержат огромное число динамических элементов в своих *DOM*-представлениях.

С целью применения классических методов кластерного анализа к Интернет-объектам необходимо сначала избавиться от динамических явлений в их кластерной структуре – исчезновения, расщепления, слияния, и дрейфа кластеров. Для этого целесообразно организовать скользящее окно для исследования поведения пользователей в Интернете. (Понятие скользящего окна было рассмотрено выше в п.3.2. разделе «Метод экспериментального исследования динамики кластерных структур Интернет-пользователей и Интернет-ресурсов»). Для снижения влияния динамических компонентов ИР

можно применять числовые коэффициенты усиления и/или метод трёхтактной кластеризации с *DOM*-фильтрацией, что существенно повысит стабильность кластерной структуры.

В приложении 3 подробно рассмотрены классические методы иерархического и итерационного кластерного анализа. Эксперименты не выявили особых проблем по дивизивной и агломеративной кластеризации Интернет-пользователей. Хотя эти методы просты в применении, они относятся к неточным методам, так как число кластеров заранее неизвестно и необходимо применять некие критерии однородности, что существенно усложняют реализацию и тестирование результатов. В приложении 3 подробно описаны и реализованы два широко применяемых итерационных метода – метод *k*-средних и метод Форель. Для первого метода входным параметром является число кластеров *k*, которое остаётся неизменным в течение всего процесса кластерного анализа. Для второго метода, входным параметром является диаметр шара, внутри которого будут находиться объекты кластера. В результате приведённых экспериментов с кластеризацией по методу Форель (см. П.3) были получены объекты, не вошедших ни в одном из сформированных кластеров, что приводит к многократному повторению эксперимента с разным значением радиуса шара до полной кластеризации всех объектов исследования.

Качественный анализ экспериментальных данных, приведенных в приложении 3, показал, что для кластеризации ИП наиболее целесообразно использовать иерархические методы, причем агломеративный метод имеет небольшое преимущество перед дивизивным методом. Дело в том, что для ИП можно применять принцип классификации по десяти соц-дем группам для проверки однородности кластеров, а при кластеризации ИР сложно заранее качественно определить подходящий критерий однородности. По этой причине для кластеризации ИР после применения коэффициентов усиления и/или *DOM*-фильтрации динамических компонентов, использовались итерационные методы, преимущественно метод *k*-средних. Если рассматривать ИП и ИР, как обобщённые объекты исследования, то для их кластеризации целесообразно

применять метод *k*-средних. Это связано с тем, что число рассматриваемых ИР будет в сотни или даже тысячи раз превышать число ИП и в этом случае говорить о каком-то реальном критерии однородности для иерархических методов будет неуместно.

3.5. Основные результаты и выводы по третьей главе

1. Представлены и подробно описаны основные динамические явления, которые могут возникнуть при регулярной кластеризации Интернет-объектов. В кластерных структурах Интернет-объектов, динамические явления могут проявиться в образовании новых кластеров, расщеплении, дроблении или исчезновении. Также возможны случаи дрейфа кластеров. Все указанные динамические явления, могут быть детектированы с использованием специальных критериев таких, как степень принадлежности объектов к кластерам, мера компактности кластеров и т.д.

2. Предложена методика преодоления динамических явлений в кластерных структурах ИП. Для Интернет-пользователей оптимальным решением оказалось применение временного окна, связанного с их поисковой активностью и жизненной деятельность.

3. Для Интернет-ресурсов экспериментально доказана эффективность применения числовых коэффициентов усиления, использование которых уменьшает влияние динамических компонентов *DOM*-моделей, улучшает значения показателя степени принадлежности и тем самым стабилизирует кластерную структуру ИР.

4. Для ИР предложен метод трёхтактная кластеризации с применением *DOM*-фильтрации. Суть метода состоит в выявлении динамических компонентов в *DOM*-модели ИР, с последующим их исключением, осуществляемым до формирования характеристического вектора ресурса. Устранение самой причины возникновения динамических изменений в кластерной структуре, обеспечивает необходимый уровень её стабильности.

4. ОБОБЩЁННОЕ МАТЕМАТИЧЕСКОЕ ОПИСАНИЕ ИНТЕРНЕТ-ОБЪЕКТОВ И ЕГО ПРИМЕНЕНИЕ В КЛАСТЕРНОМ АНАЛИЗЕ ДЛЯ ПЕРСОНАЛИЗАЦИИ ПОИСКА

Возвращаясь к схеме поэтапного процесса классификации Интернет-объектов (рисунок 1.9), обратим внимание на этапы 4 – 6, определяющие основные виды базовых математических моделей, которые могут быть использованы в качестве математического обоснования для применения понятия обобщённого объекта при кластеризации ИП и ИР. В этой главе вводится новое понятие «обобщённый характеристический вектор» для обобщённого объекта исследования. С использованием графовой модели проводится сравнительный анализ комбинированного метода кластеризации, когда объекты исследования ИП и ИР подвергаются раздельной кластеризации, и обобщённого метода кластеризации, когда объекты исследования ИП и ИР представляется единым обобщённым вектором характеристик и участвуют в едином процессе кластеризации.

4.1. Метод экспериментального исследования графовой модели для комбинированной кластеризация

Пусть, как и раньше, U – множество наблюдаемых пользователей, u_i – i-ый наблюдаемый пользователь, $u_i \in U$. В произвольный момент времени $t_k \in T$ можно сформировать характеристический (поисковый) вектор i-го пользователя, имеющий вид:

$$u_i(t_k) = (u_{i,1}(t_k), \ldots, u_{i,j}(t_k), \ldots, u_{i,nof(Vu(tk))}(t_k)),$$

где

- $u_{i,j}(t_k)$ – вес j-го поискового термина из глобального словаря терминов V_u в момент времени t_k, равный числу вхождений этого термина в запросы i-го пользователя, выполненные в течение соответствующего временного окна;

- $nof(V_u(t_k))$ – размер поискового вектора i-го пользователя, равный числу слов в глобальном словаре терминов V_u в момент времени t_k.

Пусть W – множество наблюдаемых ресурсов, w_i – наблюдаемый i-ый ресурс, $w_i \in W$. В произвольный момент времени $t_k \in T$ можно также представить i-й ресурс характеристическим вектором

$$w_i(t_k) = (w_{i,1}(t_k), \ldots, w_{i,j}(t_k), \ldots, w_{i,nof(Vw(tk))}(t_k)),$$

где

- $w_{i,j'}(t_k)$ – вес j'-ого поискового термина из локального словаря терминов V_w в момент времени t_k, равный числу вхождений этого термина в текст i-го ресурса, в течение наблюдаемого временного окна.

- $nof(V_w(t_k))$ – размер поискового вектора i-го ресурса, равный числу уникальных слов в локальном словаре терминов V_w в момент времени t_k.

Числовые координаты $u_{i,j}(t_k)$, $1 \le j \le nof(V_u(t_k))$, и $w_{i,j}(t_k)$, $1 \le j \le nof(V_w(t_k))$, расположены в характеристическом векторе, в том же лексикографическом порядке, что и термины в словарях $V_u(t_k)$ и $V_w(t_k)$, соответственно. Переход от вербального к числовому представлению происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений в запросы пользователей и в статические тексты ресурсов.

Пусть X – множество объектов, для произвольных пар которых x', $x'' \in X$ евклидово расстояние $\rho(x',x'')$ взято в качестве меры близости. Графовая модель множества X с мерой близости ρ строится по следующим правилам: вершины графа представляют элементы множества X, а веса ребер графа – расстояния между объектами из X, соответствующими вершинам, ассоциированным с данным ребром.

В комбинированной модели кластеризации рассматриваются два множества объектов U и W, которые обрабатываются и подвергаются кластерному анализу раздельно. Как следствие формируются два словаря терминов V_u и V_w. Учитывая это, для комбинированной кластеризации получаем два полносвязных неориентированных графа $G^u = \langle U, E^u \rangle$ и $G^w = \langle W, E^w \rangle$, в которых U и W – множества вершин, представляющие ИП и ИР, а E^u и E^w – множества ребер между вершинами соответствующих графов (рисунок 4.1). Для каждого ребра $(u_{i'}, u_{i''}) = e^u_{i',i''} \in E^u$ задан неотрицательный вес $\rho(e^u_{i',i''})$ равный евклидову расстоянию между

вершинами $u_{i'}$ и $u_{i''}$. Аналогично, для каждого ребра $(w_{i'}, w_{i''}) = e^{w}_{i',i''} \in E^{w}$ задан неотрицательный вес $\rho(e^{w}_{i',i''})$ равный евклидову расстоянию между вершинами $w_{i'}$ и $w_{i''}$. Матрицы весов ребер графов G^{u} и G^{w} обозначим через A и B, соответственно.

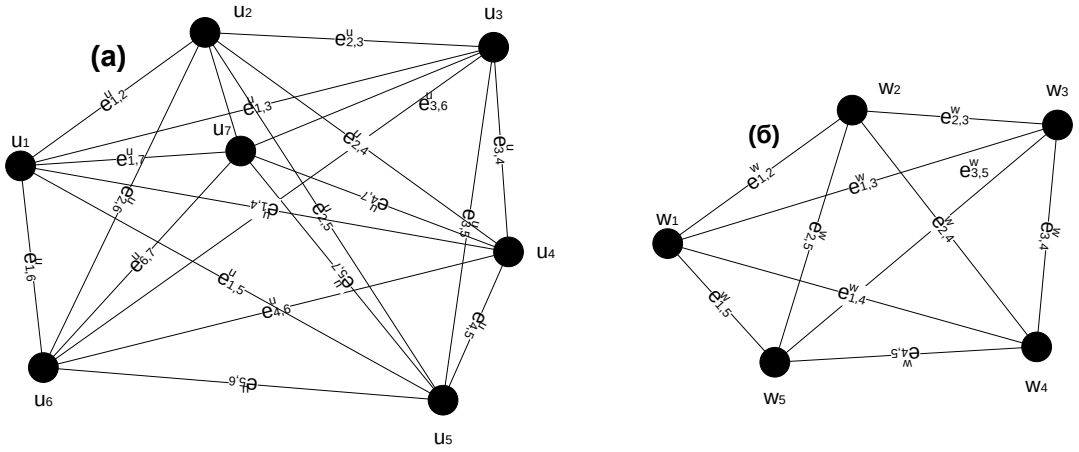

Рисунок 4.1 – Иллюстрация представления ИП (а) и ИР (б) с использованием графовой модели

В комбинированной модели для каждой вершины графа ИП G^{u}, представляющей ИП $u_i \in U$, имеется свой собственный набор вершин $W_i \subseteq W$ графа G^{w}, соответствующий тем ИР, которые посетил ИП в течение актуального временного окна. Построение минимального остова на вершинах W_i и подсчет суммы весов вошедших в остов ребер графа G^{w} позволяет оценить вес $Q(u_i)$ для каждой вершины u_i из U.

Алгоритм, решающий задачу нахождения остова минимального веса во взвешенном графе $G_i^{w} = \langle W_i, E_i^{w} \rangle$, где $E_i^{w} \subseteq E^{w}$, заключается в следующем.

1. Строим граф $G_i^{w1} = \langle W_i, E_i^{w1} \rangle$, состоящий из множества вершин W_i и единственного ребра, имеющего минимальный вес.

2. Если граф $G_i^{wk} = \langle W_i, E_i^{wk} \rangle$ уже построен и $k < nof(W_i) - c$, где c – число компонентов связности графа G_i^{wk}, то строим граф $G_i^{w(k+1)}$, добавляя к множеству рёбер графа G_i^{wk} ребро, имеющее минимальный вес среди рёбер, не входящих в E_i^{wk} и не составляющее циклов с рёбрами из E_i^{wk}.

Этот алгоритм будет применяться для расчёта $Q(u_i)$ – веса каждой вершины

u_i в графе $G^u = \langle U, E^u \rangle$ ИП (рисунок 4.2).

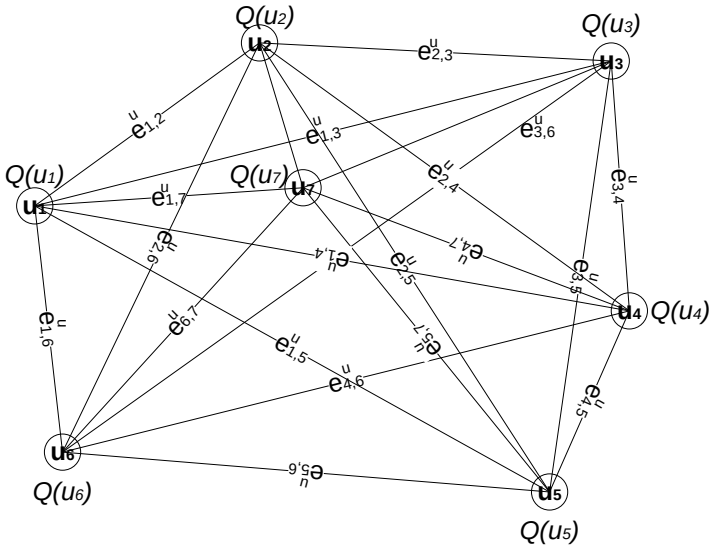

Рисунок 4.2 – Граф ИП после расчёта весов вершин $Q(u_i)$

После вычисления весов вершин можно приступить к расчёту длины кратчайшего пути от точки u_i, взяв за основу алгоритм Дейкстры [1]. Ближайшим соседом точки u_i является такая точка $u_r \in U$, что для любой другой точки $u_k \in U$ выполняется неравенство $\rho(u_i, u_r) \le \rho(u_i, u_k)$. До расчёта $Q(u_i)$ матрица весов ребер графа G^u (таблица 4.1) была симметричной, то есть $A(i,i) = 0$ и $A(i,k) = A(k,j)$. Соответствующий граф был не ориентированным.

Таблица 4.1 – Симметричная матрица весов для графа ИП до расчёта весов вершин

	u_1	u_2	u_3	u_4	u_5	u_6	u_7
u_1	0	A(1,2)	A(1,3)	A(1,4)	A(1,5)	A(1,6)	A(1,7)
u_2	A(2,1)	0	A(2,3)	A(2,4)	A(2,5)	A(2,6)	A(2,7)
u_3	A(3,1)	A(3,2)	0	A(3,4)	A(3,5)	A(3,6)	A(3,7)
u_4	A(4,1)	A(4,2)	A(4,3)	0	A(4,5)	A(4,6)	A(4,7)
u_5	A(5,1)	A(5,2)	A(5,3)	A(5,4)	0	A(5,6)	A(5,7)
u_6	A(6,1)	A(6,2)	A(6,3)	A(6,4)	A(6,5)	0	A(6,7)
u_7	A(7,1)	A(7,2)	A(7,3)	A(7,4)	A(7,5)	A(7,6)	0

После расчёта весов $Q(u_i)$ построим несимметричную матрицу A^* такую, что $A^*(i,i) = 0$ и $A^*(i,k) = A(i,k) + Q(u_i)$ (таблица 4.2).

Таблица 4.2 – Несимметричная матрица весов рёбер графа ИП после добавления весов вершин $Q(u_i)$

	u_1	u_2	u_3	u_4	u_5	u_6	u_7
u_1	0	$A(1,2)+Q(u_1)$	$A(1,3)+Q(u_1)$	$A(1,4)+Q(u_1)$	$A(1,5)+Q(u_1)$	$A(1,6)+Q(u_1)$	$A(1,7)+Q(u_1)$
u_2	$A(2,1)+Q(u_2)$	0	$A(2,3)+Q(u_2)$	$A(2,4)+Q(u_2)$	$A(2,5)+Q(u_2)$	$A(2,6)+Q(u_2)$	$A(2,7)+Q(u_2)$
u_3	$A(3,1)+Q(u_3)$	$A(3,2)+Q(u_3)$	0	$A(3,4)+Q(u_3)$	$A(3,5)+Q(u_3)$	$A(3,6)+Q(u_3)$	$A(3,7)+Q(u_3)$
u_4	$A(4,1)+Q(u_4)$	$A(4,2)+Q(u_4)$	$A(4,3)+Q(u_4)$	0	$A(4,5)+Q(u_4)$	$A(4,6)+Q(u_4)$	$A(4,7)+Q(u_4)$
u_5	$A(5,1)+Q(u_5)$	$A(5,2)+Q(u_5)$	$A(5,3)+Q(u_5)$	$A(5,4)+Q(u_5)$	0	$A(5,6)+Q(u_5)$	$A(5,7)+Q(u_5)$
u_6	$A(6,1)+Q(u_6)$	$A(6,2)+Q(u_6)$	$A(6,3)+Q(u_6)$	$A(6,4)+Q(u_6)$	$A(6,5)+Q(u_6)$	0	$A(6,7)+Q(u_6)$
u_7	$A(7,1)+Q(u_7)$	$A(7,2)+Q(u_7)$	$A(7,3)+Q(u_7)$	$A(7,4)+Q(u_7)$	$A(7,5)+Q(u_7)$	$A(7,6)+Q(u_7)$	0

Из таблицы 4.2. видим, что вес ребра $A^* = A(i,j)+Q(u_i) \neq A(j,i)+ Q(u_j) = A^*$ так как в общем случае $Q_i(u_i) \neq Q(u_j)$. Теперь, для графического представления матрицы A^*, воспользуемся дуальным орграфом $G^u* = \langle U^*, E^u* \rangle$, в котором для любой дуги $e^u*_{i',i''} \in E^u*$ существует дуга $e^u*_{i'',i'} \in E^u*$, при условии, что $i' \neq i''$ (рисунок 4.3).

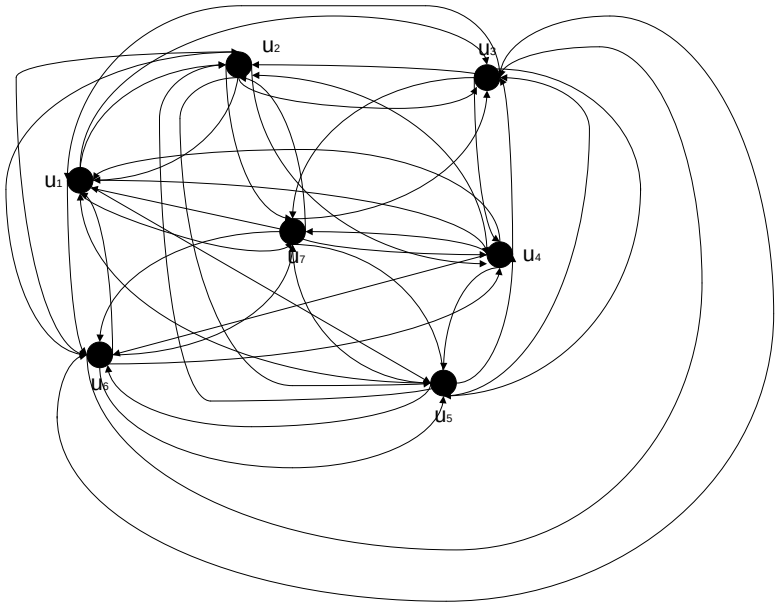

Рисунок 4.3 – Орграф ИП после расчёта весов $A^*(i,k)$

Алгоритм расчета длины кратчайшего пути на основе алгоритма Дейкстры, написанный на псевдокоде, имеет вид:

```
-- Инициализация переменных
```

-- множество вершин графа
$U^* = \{u_1, u_2, \ldots, u_{nof(U)}\}$
-- вес ребра $e^{u*}_{i,j}$ соединяющий вершину u_i с вершиной u_j
$A(i,j)+Q(u_i)$
-- множество посещённых вершин, расчёт будет проводиться относительно первой вершины графа (это не принципиально: можно выбрать любую другую вершину в качестве начальной)
$U^{*\prime}$
-- массив расстояний от первоначальной точки до любой вершины с индексом p.
$\rho_{min}[1..nof(u)]$, где $\rho_{min}[p=1] = 0$;

Для всех $u_i \in U^*$
    $\rho_{min}[i] \leftarrow \infty$
$\rho_{min}[1] \leftarrow 0$
$U^{*\prime} \leftarrow \varnothing$
пока $\exists u_j \notin U^{*\prime}$
    пусть $u_j \notin U^{*\prime}$: $\rho_{min}[j]_{минимальный}$
    Для всех $u_i \notin U^{*\prime}$
    если $\rho_{\min}[i] > \rho_{\min}[j] + \min\left((A(i,j)+Q(u_i)),(A(j,i)+Q(u_j))\right)$ то
        $\rho_{\min}[i] > \rho_{\min}[j] + \min\left((A(i,j)+Q(u_i)),(A(j,i)+Q(u_j))\right)$
    $U^{*\prime} \leftarrow U^{*\prime} \cup \{u_j\}$

Из псевдокода алгоритма видно, что для комбинированной кластеризации в расчёте кратчайшего пути будут всегда учитываться две составляющие: расстояние между вершинами $\rho(u_i,u_j) = A(i, j)$ и смещение $Q(u_i)$.

4.2. Метод экспериментального исследования графовой модели для обобщённой кластеризация

Для обобщённой модели кластеризации, исследуемые объекты (ИП и ИР) рассматриваются как единое множество. Для них формируется обобщённый характеристический вектор X, получаемый путём объединения поисковых запросов ИП с текстовым содержанием ИР и формирования обобщённого словаря терминов $V = V_u \cup V_w$.

Пусть X – множество наблюдаемых объектов, x_i – i-ый наблюдаемый объект, $x_i \in X$. В произвольный момент времени $t_k \in T$ можно сформировать обобщённый характеристический вектор i-го объекта, имеющий следующий вид: $x_i(t_k) = (x_{i,1}(t_k), \ldots, x_{i,j}(t_k), \ldots, x_{i,nof(V)}(t_k))$,

где

- $x_{i,j}(t_k)$ – вес j-го термина из глобального словаря терминов V в момент времени t_k, равный числу вхождений этого термина в запросы i-го пользователя или в текстовое содержание i-ого ресурса, выполненные в течение соответствующего временного окна;

- $nof(V)$ – размер характеристического вектора i-го объекта, равный числу слов в глобальном словаре терминов V.

Числовые координаты $x_{i,j}(t_k)$, $1 \le j \le nof(V)$, расположены в обобщённом характеристическом векторе в том же порядке, что и термины в глобальном словаре V. Переход от вербального к числовому представлению происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений. Для обобщённой модели кластеризации формируется множества обобщённых объектов $X = U \cup W$ и глобальный словарь терминов $V = V_u \cup V_w$. Теоретически размерность $nof(V)$ обобщённого вектора X не будет превосходить сумму ($nof(V_u)$ + $nof(V_w)$), где $nof(V_u)$ и $nof(V_w)$ размерности характеристических векторов u_i ИП и w_i ИР соответственно. Однако на практике $nof(V) = nof(V_w)$, т.к. исследуемые ИР были получены в результате поисковой деятельности самых ИП и посещаемые ими ИР в той или иной мере содержат термины из словаря ИП V_u. Уже на этапе определения размерности обобщённого вектора проявляется первое преимущество обобщённой кластеризации – один общий вектор для объектов любой природы.

Каждый обобщённый объект может быть представлен $nof(V_w)$-мерным обобщённым характеристическим вектором. Следует отметить, что для i-го ИП характеристический вектор u_i является $nof(V_u)$-мерной проекцией $nof(V_w)$-мерного обобщённого характеристического вектора $x_i(t_k)$, добавленные координаты которого будут нулевыми. Отсюда, для объектов u_i в обобщенном случае получаем те же расчетные расстояния между объектами, что и для комбинированного метода.

Пусть в $nof(V)$-мерном пространстве задана функция расстояния между

точками этого пространства и конечное множество точек X, которые могут рассматриваться, как вершины графа. Отрезки, соединяющие точки из X могут рассматриваться, как ребра, а расстояние между этими точками – весами соответствующих ребер. Получаем один полносвязный неориентированный граф $G^x = \langle X, E^x \rangle$, в котором X – множество вершин, E^x – множество рёбер. Для каждого ребра графа ребра $(x_{i'}, x_{i''}) = e^x_{i',i''} \in E^x$ задан неотрицательный вес $\rho(e^x_{i',i''})$ равный евклидову расстоянию между вершинами $x_{i'}$ и $x_{i''}$. Воспользуемся графами, показанными на рисунке 4.1, и добавим недостающие ребра для получения полносвязного обобщенного графа G^x (рисунок 4.4). Чтобы не загромождать рисунок 4.4, на нем представлены дополнительные рёбра только для одной вершины. Для остальных вершин нужно дорисовать недостающие рёбра, для получения полносвязного графа.

В обобщённой модели нет необходимости рассчитывать веса вершин, как это было для комбинированной модели. Определяются только веса рёбер.

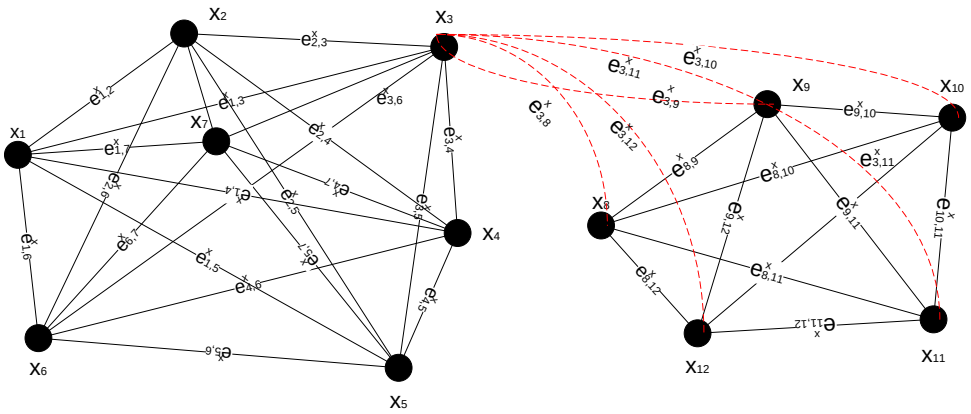

Рисунок 4.4 – Представление объектов исследования с помощью графовой модели после применения обобщённого характеристического вектора

После построения полносвязного графа, необходимо сформировать соответствующую матрицу весов рёбер (назовем ее C), размер которой будет зависеть от числа всех объектов (ИП и ИР) участвующих в исследовании (таблица 4.3).

Таблица 4.3 – Симметричная матрица весов рёбер для обобщённого графа

	x_1	x_2		x_7	x_8		x_{12}
x_1	0	C(1,2)		C(1,7)	C(1,8)		C(1,12)
x_2	C(2,1)	0		A(2,7)	A(2,8)		C(2,12)
x_7	C(7,1)	C(7,2)		0	C(7,8)		C(4,7)
x_8	C(8,1)	C(8,2)		C(8,7)	0		C(8,12)
x_{12}	C(12,1)	C(12,2)		C(12,7)	C(12,8)		0

По аналогии с комбинированной кластеризацией проведем расчёт длины кратчайшего пути по алгоритму Дейкстры от точки x_i. Алгоритм расчёта длины кратчайшего пути при этом не изменится, изменится только исходное множество объектов.

Общая протяженность пути будет зависеть исключительно от меры близости (евклидова расстояния) между объектами т.к. у вершин нет весов. Таким образом, для обобщённого вектора получаем более простой алгоритм по сравнению с комбинированным методом, учитывающий лишь одну составляющую – расстояние между вершинами $\rho(u_i,u_j) = C(i, j)$.

4.3. Результаты экспериментального сравнения комбинированной и обобщённой кластеризации

Проведем сравнительный аналитический анализ для определения наилучшего метода кластеризации, выбирая из двух методов, отличающихся по способу формирования характеристических векторов – комбинированной или обобщённой кластеризации. Для этой цели рассмотрим экспериментально полученные данные о поисковой активности ИП в период с 18 по 24 ноября 2013 года. Сформируем соответствующие характеристические вектора и рассчитаем веса рёбер и длину кратчайшего пути. Варьировать будем только размерность поискового вектора, т.к. поисковая активность ИП характеризуется динамичностью, а ИР после применения весовых коэффициентов усиления и трёхтактной кластеризации с обратной связью, полученных с помощью *DOM*-

модели, делают их статическими (п. 3.2, 3.3).

Первый эксперимент с комбинированной кластеризацией объектов.

Наблюдаем за поисковой активностью случайно выбранных двух ИП (для б*о*льшего числа ИП увеличится вычислительная нагрузка, а принцип останется тем же) и формируем их характеристические вектора методом позиционного кодирования (реализовано в среде *MS SQL Server* 2012, приложение 4).

Случайным образом выбираем двух ИП – u_{8263} и u_{12401}. Граф ИП содержит всего две вершины и одно ребро $e^{u}_{8263,\ 12401}$ (рисунок 4.5). Матрица весов рёбер графа ИП, состоящая всего из четырех элементов, показана в таблице 4.4.

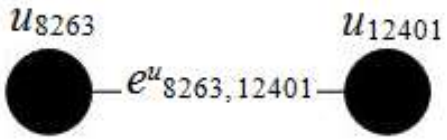

Рисунок 4.5 – Пример неориентированного графа G^u для двух ИП

Таблица 4.4 – Симметричная матрица весов рёбер графа ИП

	u_{8263}	u_{12401}
u_{8263}	0	28
u_{12401}	28	0

Переходим к Интернет-ресурсам. Наблюдаем за содержанием трех случайно выбранных ИР w_8, w_{48} и w_{98}, на которые осуществляли заходы ИП u_{8263} и u_{12401} (два из трех ИР посещались каждым пользователем) и формируем характеристические вектора методом позиционного кодирования.

Граф ИР содержит три вершины и три ребра $e^{w}_{48,8}$, $e^{w}_{48,98}$ и $e^{w}_{98,8}$, (рисунок 4.6). Матрица весов ребер графа ИП, состоящая из девяти элементов, показана в таблице 4.5.

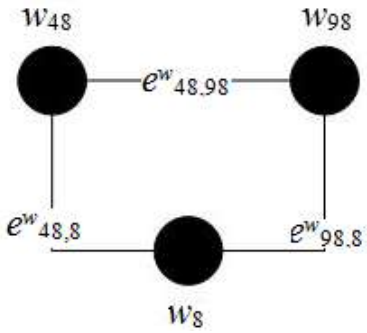

Рисунок 4.6 – Пример неориентированного графа G^w для трех ИР

Таблица 4.5 – Симметричная матрица весов рёбер графа ИР

	w_{48}	w_{98}	w_8
w_{48}	0	63	116
w_{98}	63	0	121
w_8	116	121	0

Проведём расчёт весов вершин $Q(u_{8263})$ и $Q(u_{12401})$. Поскольку пользователь u_{8263} посещал страницы w_{48} и w_{98}, его вес $Q(u_{8263}) = B(48,98) = 63$. Пользователь u_{12401} посещал страницы w_{98} и w_8, вес $Q(u_{12401}) = B(98,8) = 121$. После расчёта весов $Q(u_i)$ строим матрицу A^* (таблица 4.6).

Таблица 4.6 – Несимметричная матрица весов графа ИП

	u_{8263}	u_{12401}
u_{8263}	0	28+63=91
u_{12401}	28+121=149	0

Ориентированный граф ИП, соответствующий матрице A^* таблица 4.6, показан на рисунок 4.7. $e^{u}_{12401,8263}+ Q(u_{12401})$ $e^{u}_{8263,12401}+ Q(u_{8263})$

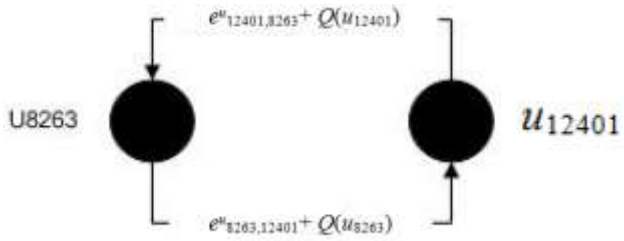

Рисунок 4.7 – Орграф ИП с двумя вершинами после расчёта весов вершин

Первый эксперимент с обобщённой кластеризацией объектов.

Для сравнения результатов воспользуемся обобщённым

характеристическим вектором для представления ИП u_{8263} и u_{12401} и ИР w_8, w_{48} и w_{98}, как еденного объекта исследований (рисунок 4.8).

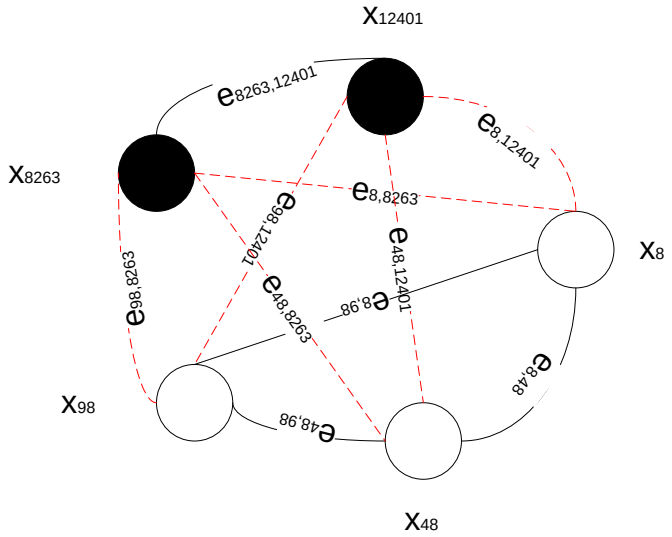

Рисунок 4.8 – Пример неориентированного графа обобщённых объектов G^x с двумя ИП и тремя ИР

После формирования глобального словаря V и обобщённых характеристических векторов $x_i(t_k)$ можно приступить к расчёту евклидового расстояния для представления рёбер и матрицы весов (таблица 4.7).

Таблица 4.7 – Симметричная матрица весов для обобщённого случая

	x_8	x_{48}	x_{98}	x_{8263}	x_{12401}
x_8	0	116	121	93	93
x_{48}	116	0	63	165	165
x_{98}	121	63	0	169	169
x_{8263}	93	165	169	0	28
x_{12401}	93	165	169	28	0

Обсуждение первых экспериментов.

Расстояние между объектами ИП при обобщённой кластеризации оказалось фиксированным и не зависящим от направления дуг между соответствующими вершинами графа. Минимальное расстояние между объектами ИП осталось прежним, равно 28. Минимальное расстояние между объектами ИП и ИР увеличилось с 91 до 93. Максимальное расстояние тоже увеличилось с 149 до 169. Минимальное и максимальное расстояние между объектами ИР осталось

прежним, 63 и 121, соответственно.

Полученный результат не позволяет выявить преимущество и недостатки обобщённой и комбинированной кластеризации.

Продолжение экспериментов по комбинированной и обобщённой кластеризации объектов.

Продолжим эксперименты с разными входными параметрами:

- увеличивая число ИП при фиксированном числе ИР;

- увеличивая число ИР при фиксированном числе ИП.

Будем варьировать число ИП, выбирая значения *nof*(*U*) из ряда 2, 3, 5, 10, 15, 20, и число ИР, выбирая значения *nof*(*W*) из ряда 3, 5, 10, 15, 20, для получения разных комбинаций числа объектов в кластерной структуре. Для каждого значения входных параметров эксперимент будем повторять 10 раз (всего 6×5×10 = 300 экспериментов), причем каждый эксперимент будет характеризоваться уникальным набором объектов. После десятикратного повторения эксперимента, для каждого из наборов входных параметров и сравнения ниже указанных числовых показателей при применении комбинированной и обобщённой кластеризации, определяем следующие показатели:

- *min_U* и *max_U* – среднее минимальное и среднее максимальное расстояние между двумя объектами ИП, у которых хотя бы один общий ИР;

- *min_R* и *max_R* – среднее минимальное и среднее максимальное расстояние между двумя объектами ИР:

- *min_UR* и *max_UR* – среднее минимальное и среднее максимальное расстояние между двумя объектами разной природы (ИП – ИР) в кластерной структуре.

Экспериментальная работа была реализована с учётом историчности и с применением функции *NEWID* (в среде *MS SQL Server* 2012) для случайного выбора, т.е. для увеличения числа объектов новые «случайно» подобранные объекты добавлялись к уже существующим объектам. Таким образом, с одной стороны, будет соблюдена случайность выборки, а, с другой стороны, – принцип формирования и наращивания кластерной структуры.

При расчётах евклидовых расстояний между объектами матрица весов достигала размерности (20+20)×(20+20) = 40×40 = 1600 элементов. Размер словаря (а следовательно и координальность характеристических векторов) достигла 5699 слов. Для простоты и удобства работы был сформирован урезанный словарь терминов из уникальных слов длиной не меньше 5 символов (без какой-либо лингвистической обработки). Полная таблица с результатами расчётов представлена в приложении 5.

В ходе экспериментов получены следующие интересные результаты.

1) В результате применения обобщённых характеристических векторов число объектов-кластеров или, другими словами, «выпавших» не кластеризованных объектов ИП u_i в кластерной структуре уменьшилось (см. столбец *OUT* в приложении 5).

2) В результате применения обобщённых характеристических векторов минимальное расстояние между объектами ИП значительно уменьшается, когда $nof(U) \leq 5$. Как только $nof(U)$ начинает увеличиваться, минимальное расстояние между объектами ИП на участке $nof(U) = 15$ может слегка увеличиться (рисунок 4.9).

3) В результате применения обобщённых характеристических векторов максимальное расстояние между ИП значительно уменьшается, примерно в 3 раза (рисунок 4.10).

4) В результате применения обобщённых характеристических векторов среднее минимальное расстояние между ИР уменьшается в незначительной степени (рисунок 4.11). Однако, как только число $nof(U)$ превышает 20, а $nof(W)$ больше 5, график смещается в худшую сторону. Это заметно в конце эксперимента, когда график, соответствующий обобщённой кластеризации поднимается выше графика, полученного для комбинированной кластеризации.

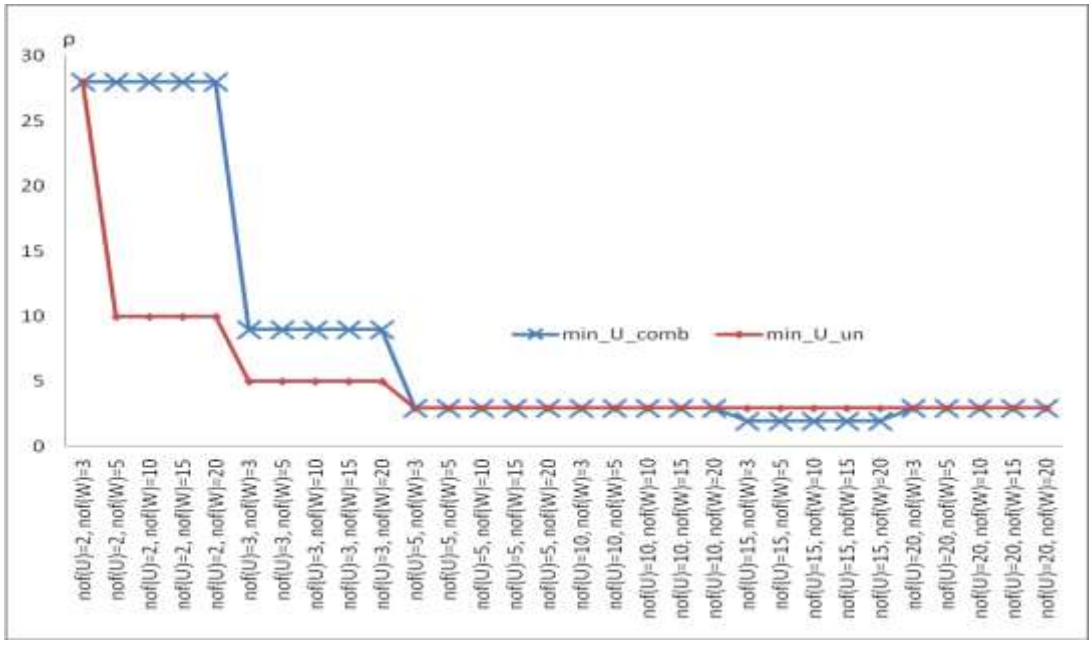

Рисунок 4.9 – Графики минимальных расстояний между объектами ИП для комбинированной (*min_U_comb*) и обобщённой (*min_U_un*) кластеризации

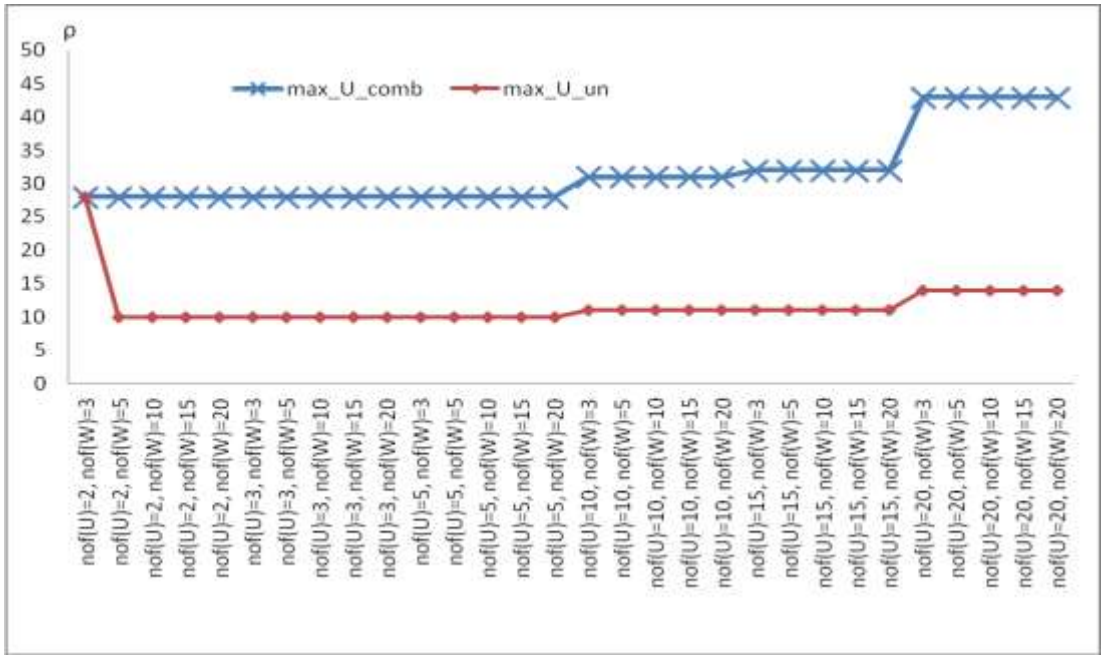

Рисунок 4.10 – Графики максимальных расстояний между объектами ИП для комбинированной (*max_U_comb*) и обобщённой (*max_U_un*) кластеризации

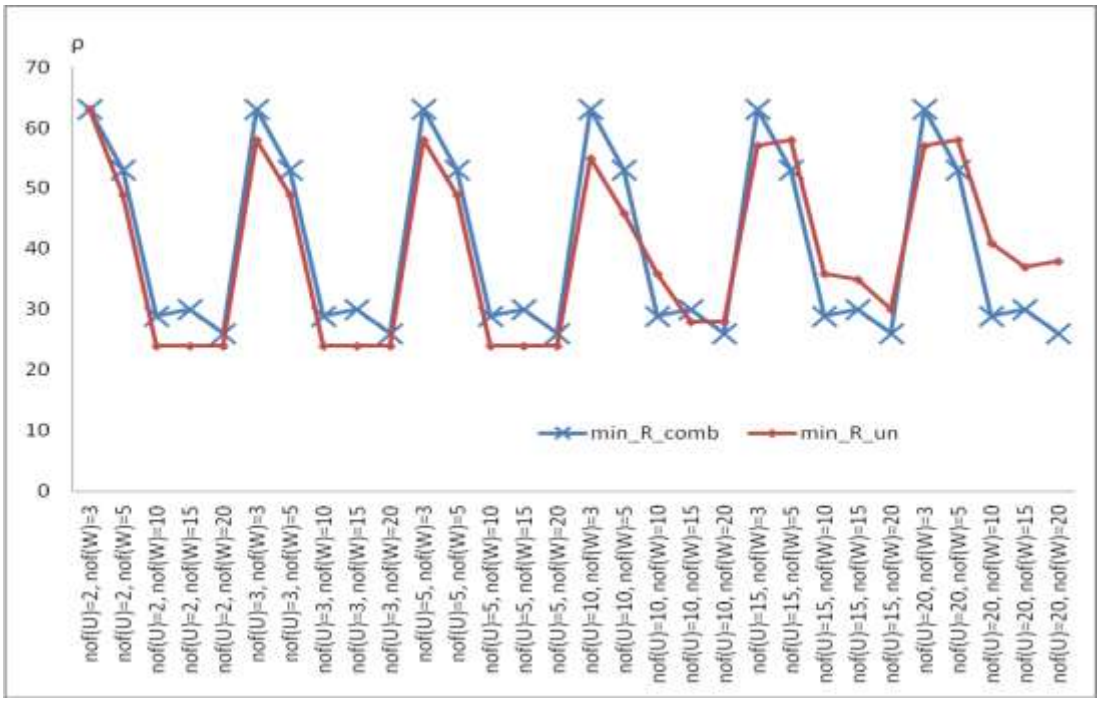

Рисунок 4.11 – Графики минимальных расстояний между объектами ИР для комбинированной (*min_R_comb*) и обобщённой (*min_R_un*) кластеризации

5) В результате применения обобщённых характеристических векторов максимальное расстояние между ИР в значительной степени уменьшается (рисунок 4.12).

6) В результате применения обобщённых характеристических векторов минимальное расстояние между объектами ИП и объектами ИР уменьшается (рисунок 4.13). Это особенно заметно на первых стадиях эксперимента, когда $nof(U) \leq 3$.

7) В результате применения обобщённых характеристических векторов максимальное расстояние между объектами ИП и объектами ИР в значительной степени уменьшается (рисунок 4.14). Исключением являются лишь результаты в начале эксперимента, когда число $nof(U) = 2$ и $nof(W) = 3$.

На первый взгляд, на текущем этапе экспериментальных исследований, наблюдается небольшое преимущество применения обобщённых характеристических векторов для решения задачи кластерного анализа. Однако, после выполнения расчётов и сравнения основных показателей, можно получить

более точные результаты, которые напрямую отражают состояние кластерной структуры и позволят понять, что с ней происходит. Дело в том, что для формирования кластеров был выбран итерационный метод *k*-средних, который напрямую зависит от входного параметра – числа кластеров *k*. Но число кластеров зависит от первоначального состояния объектов в кластерной структуре.

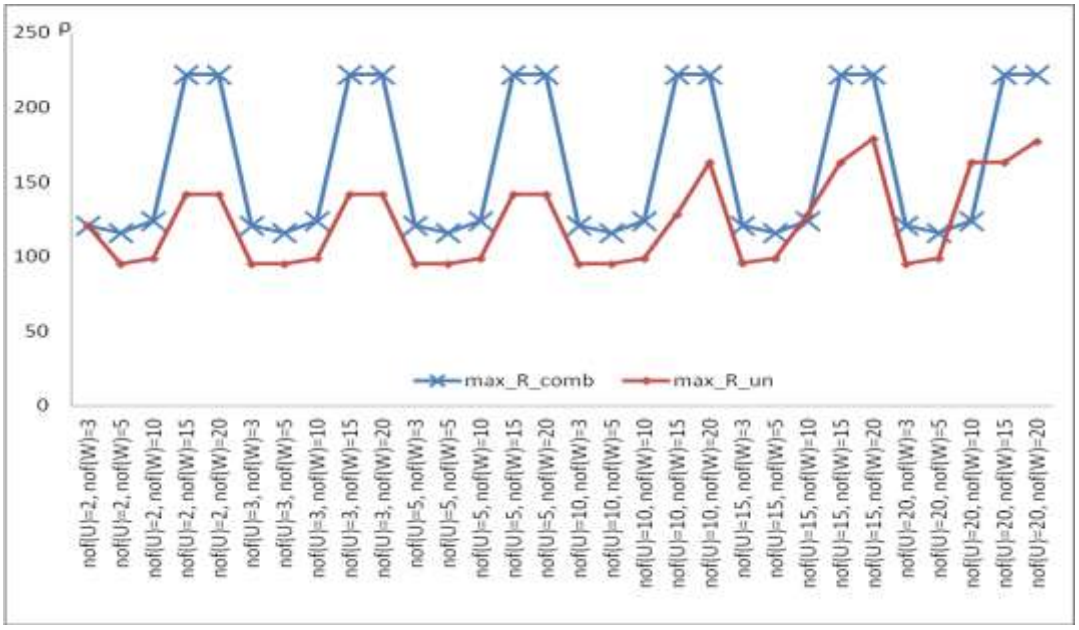

Рисунок 4.12 – Графики максимальных расстояний между объектами ИР для комбинированной (*max_R_comb*) и обобщённой (*max_R_un*) кластеризации

Из полученных расчётов можно извлечь более интересные показатели, которые косвенно могут повлиять на меру компактности (формула 3.6) – чем больше разница между минимальными и максимальными расстояниями Δ, тем кластер будет менее «сгущённым» и концентрация объектов в единице объёма пространства уменьшается и, как следствие, кластерная структура станет менее стабильной. Расчётные таблицы, подтверждающие этот факт, представлены в приложении 5.

8) В результате применения обобщённых характеристических векторов пространство между объектами ИП в значительной степени сокращается (рисунок 4.15).

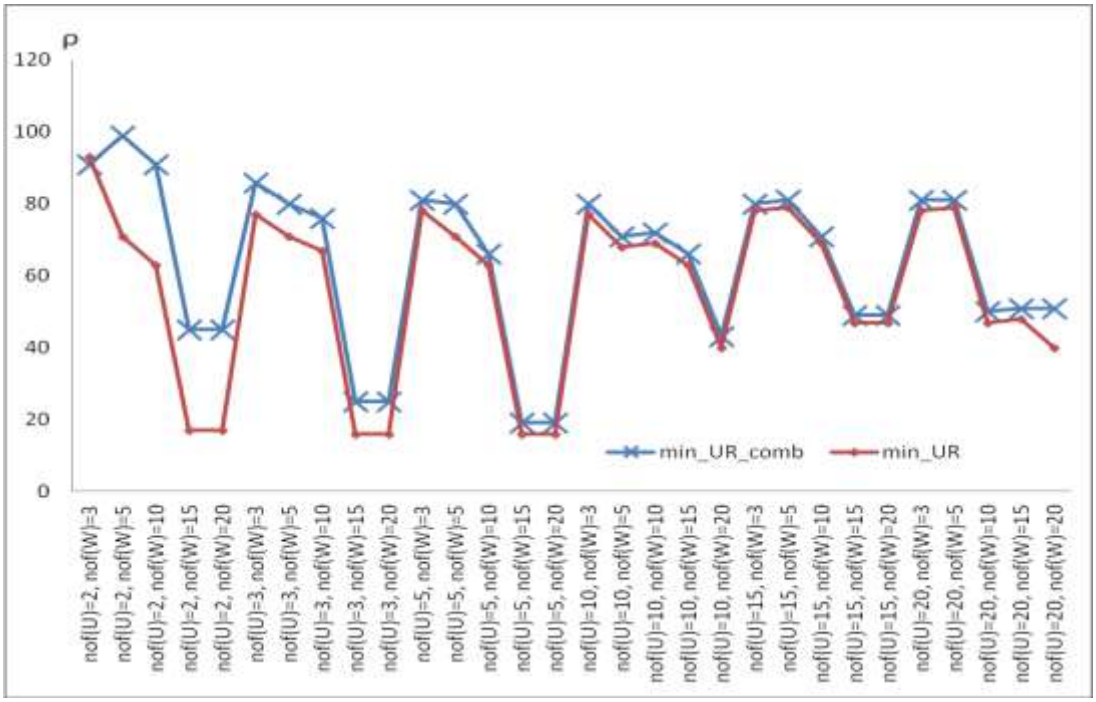

Рисунок 4.13 – Графики минимальных расстояний между объектами ИП и объектами ИР для комбинированной (*min_UR_comb*) и обобщённой (*min_UR*) кластеризации

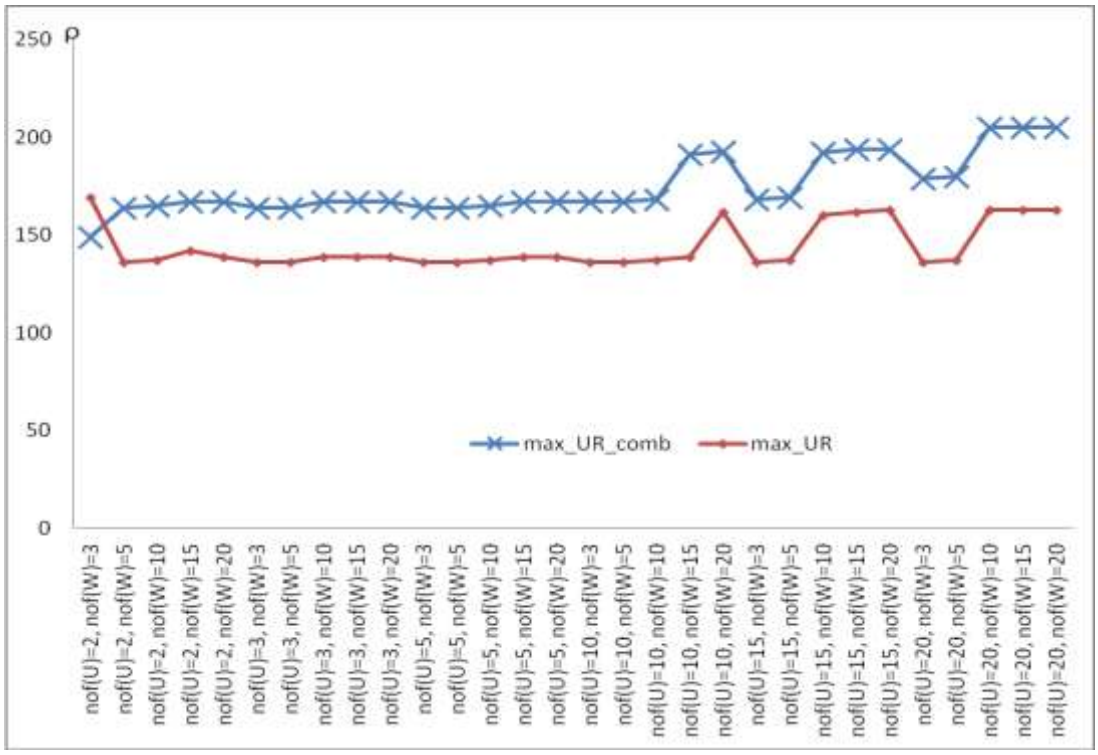

Рисунок 4.14 – Графики максимальных расстояний между объектами ИП и объектами ИР для комбинированной (*max_UR_comb*) и обобщённой (*max_UR*) кластеризации

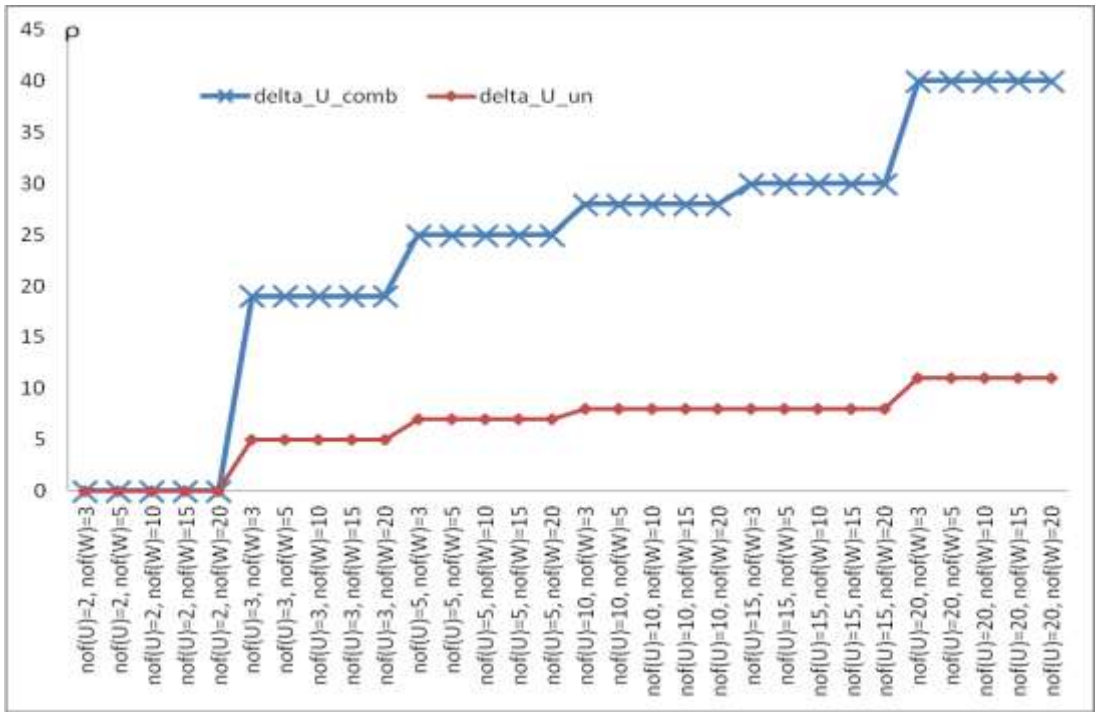

Рисунок 4.15 – Графики Δ расстояний между объектами ИП для комбинированной (*delta_U_comb*) и обобщённой (*delta_U_un*) кластеризации

На рисунке 4.15 видно, что с увеличением *nof*(*U*) экспансия пространства для комбинированной кластеризации увеличивается примерно в 4 раза по сравнению с его экспансией для обобщённой кластеризации.

9) В результате применения обобщённых характеристических векторов пространство, где размещаются объекты ИР, в значительной степени сокращается (рисунок 4.16) – локальные максимумы (когда *nof*(*W*) увеличивается) примерно в 2 раза; локальные минимумы (когда *nof*(*W*) минимально) примерно в 2 раза.

На рисунок 4.16 видно, что на притяжении всего эксперимента с увеличением *nof*(*U*) экспансия пространства для комбинированной кластеризации увеличивается до 2 раз по сравнению с его же экспансией для обобщённой кластеризации. Не смотря на то, что на рисунке 4.11 в конце эксперимента обобщённая кластеризация уступала комбинированной, после расчёта Δ (рисунок 4.16) заметно превосходство обобщённой кластеризации над комбинированной кластеризацией.

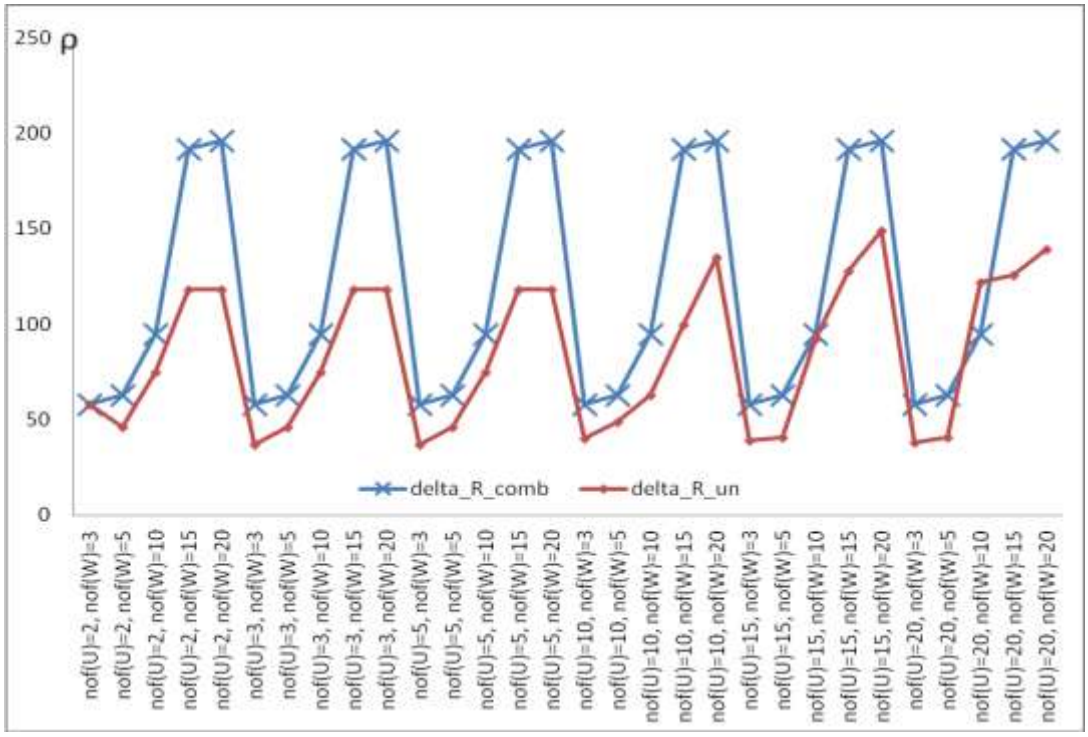

Рисунок 4.16 – Графики Δ расстояний между объектами ИР для комбинированной (*delta_R_comb*) и обобщённой (*delta_R_un*) кластеризации

10) В результате применения обобщённых характеристических векторов, пространство, где размещаются объекты ИП и объекты ИР сокращается в интервале от 15% до 60% (рисунок 4.17).

Анализируя рисунок 4.17 также можно заметить преимущество обобщённой кластеризации по сравнению с комбинированной: глобальное пространство объектов *X* сокращается.

4.4. Основные результаты и выводы по пятой главе

1. Предложена новая математическая модель представления объектов исследования (ИП и ИР) как обобщённых характеристических векторов.

2. Реализована унификация объектов исследования с точки зрения их математического описания, что позволяет проводить кластеризацию ИП и ИР одновременно и без какого-либо разделения объектов по видам.

3. Показано, что комбинированная кластеризация, по сути, основывается на

выполнении отдельно кластерного анализа ИП и отдельно кластерного анализа ИР, тем самым объем вычислений удваивается. В этом случае подготавливаются разные словари терминов, формируются разные характеристические вектора. При обобщённой кластеризации формируется единый словарь терминов и обобщённые характеристические вектора, одинаково устроенные, как для ИП, так и для ИР. Это позволяет выполнять кластерный анализ для всех объектов единовременно.

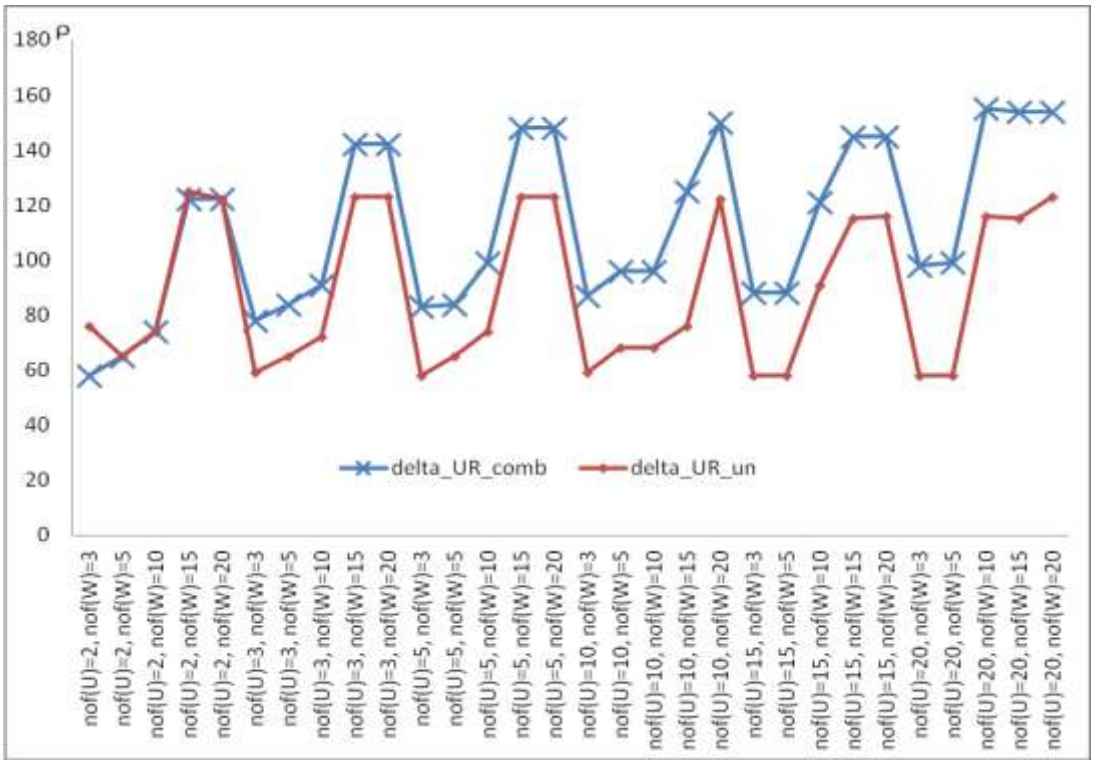

Рисунок 4.17 – Графики Δ расстояний между объектами ИП и объектами ИР для комбинированной (*delta_UR_comb*) и обобщённой (*delta_UR_un*) кластеризации

4. Доказано, что в результате применения обобщённых характеристических векторов пространство, где размещаются объекты исследования, может сокращаться до 2-х раз.

5. РЕАЛИЗАЦИЯ МЕТОДОВ КЛАСТЕРИЗАЦИИ ИНТЕРНЕТ-ПОЛЬЗОВАТЕЛЕЙ И ИНТЕРНЕТ-РЕСУРСОВ В СИСТЕМАХ ПЕРСОНАЛИЗАЦИИ ПОИСКА

Ещё раз вернёмся к рисунку 1.9, остановив внимание на этапах 7 и 8, определяющих реализацию предложенных в ходе выполнения проекта методов. По техническим условиям методы персонализации Интернет-поиска, предложенные в настоящей работе (главы 3,4), могут быть реализованы в рамках корпоративной информационно-вычислительной системы предприятия. В этой главе речь пойдёт о структурной и функциональной организации корпоративной системы персонализации поиска (КСПП) на предприятии, обслуживающей множество однородных групп пользователей, имеющих схожие поисковые интересы.

5.1. Концепция построения корпоративной системы персонализации Интернет-поиска

Экспериментальные исследования позволили определить временны*е* и вычислительные затраты на выполнение кластерного анализа реальных массивов ИП и ИР, что позволило, в свою очередь, оценить, какие программные модули КСПП должны быть установлены на персональные компьютеры (ПК) пользователей, а какие – на серверы предприятия. Указанные исследования проводились на персональном компьютере с центральным процессором *AMD FX*-6100 *Six-Core Processor* с тактовой чистотой 3.30 ГГц, имеющим оперативную память емкостью 16 ГБ. На машине в разное время был установлен браузер *Internet Explorer* 10-ой и 11-ой версии. Максимальная пропускная способность Интернет-канала составляла 5 Мб/с.

Уже было сказано, что поисковые запросы ИП можно собирать одномоментно, а именно в те моменты времени, когда начинается их поисковая деятельность. Для этой цели был создан программный модуль *internet_res_search*

(приложение 6), который позволяет не только отслеживать поисковые запросы ИП, но и определять глубину поиска, тем самым, предоставляя информацию о результатах поисковой отдачи Яндекса.

Кроме поисковой деятельности, психологический портрет ИП также формируется в результате выполнения заходов и посещений им ИР. В связи с этим был реализован программный модуль *ie_analyzer* (приложение 7), обеспечивающий автоматическое слежение и формирование *Log*-файла с хронологическим списком посещаемых ИП страниц.

Обработка содержания ИР оказалась достаточно затратной по времени. Современные ИР содержат огромное число динамических компонентов в *DOM*-модели, которые могут постоянно изменять свое содержание, кроме того, наличие больших медиафайлов (видео или картинки высокого разрешения) может сильно увеличить время полной загрузки и чтения содержания *DOM*-модели ресурса. Некоторые ресурсы могут содержать ссылки на несуществующие элементы, находящиеся на сервере самого ИР, и браузер будет пытаться загрузить их до момента *timeout*. В процессе экспериментальных исследований среднее время сканирования *DOM*-модели одной страницы ИР составляло 6-7 секунд. Если Яндекс, как результат поискового запроса, выдаёт всего лишь 10 гиперссылок на найденные ресурсы, то для чтения содержания их *DOM*-моделей потребуется более одной минуты, что является недопустимым временем для любого пользователя – лишние временные затраты на ожидание приводят к серьёзным психическим издержкам, могут сильно раздражать человека.

Процесс кластерного анализа ИП и ИР, основанный на выполнении разработанных методов, содержит много этапов (временные окна для ИП, применение числовых коэффициентов усиления из *DOM*-модели ИР, борьба с динамическими элементами ИР и применение обобщённого характеристического вектора на основе глобального словаря терминов), при этом соответствующие процедуры обработки данных весьма времязатратны. Так как кластерный анализ результатов 4-х часового периода наблюдений занимает примерно 20 минут времени (что недопустимо много для любого ИП), целесообразно выделить

специальную серверную вычислительную систему (сервер) под кластерный анализ Интернет-объектов.

С учётом сказанного на рисунке 5.1 представлена обобщённая структура КСПП, отображающая структурную организацию системы. Программные модули непосредственного наблюдения *internet_res_search* и *ie_analyzer* должны быть установлены на клиентских ПК. Программный модуль *internet_res_search* реализован и подогнан к работе с поисковой системой Яндекса. Дело в том, что поисковая система Яндекса выгружает результаты поиска в *HTML*-коде самой страницы, поэтому нет никакой необходимости инспектировать *DOM*-модель каждой страницы, что снижает в разы вычислительные затраты на чтение и анализ страниц, полученных в результате поиска. Все кластерные расчёты и вся аналитика должны быть реализованы программными модулями, установленными на корпоративных серверах. Для этой цели были разработаны специальные расчётные процедуры и фильтрующие функции, а также спроектирована структура БД *InternetDB*, поддерживаемой системой управления *MS SQL Server* 2012.

Таким образом, для персонализации Интернет-поиска работников предприятия, в рамках его корпоративной информационно-вычислительной системы должна быть создана трехзвенная КСПП. На рисинке 5.1 представлены все три взаимосвязанных звена корпоративной системы персонализации поиска: первое звено – множество ИП, второе звено – сервер кластерного анализа и третье звено – сервер БД. К сети Интернет система подключается через сервер кластерного анализа.

К сожалению, в процессе выполнения диссертационной работы реальную систему, структура которой отвечала бы обобщенной структуре рисунка 5.1, создать не удалось. Задачи реализации и тестирования предлагаемых методов кластерного анализа решались на одной мощной вычислительной установке, способной поддерживать множество виртуальных машин. Структура виртуальной КСПП в полной мере соответствовала обобщенной структуре, представленной на рисунке 5.1.

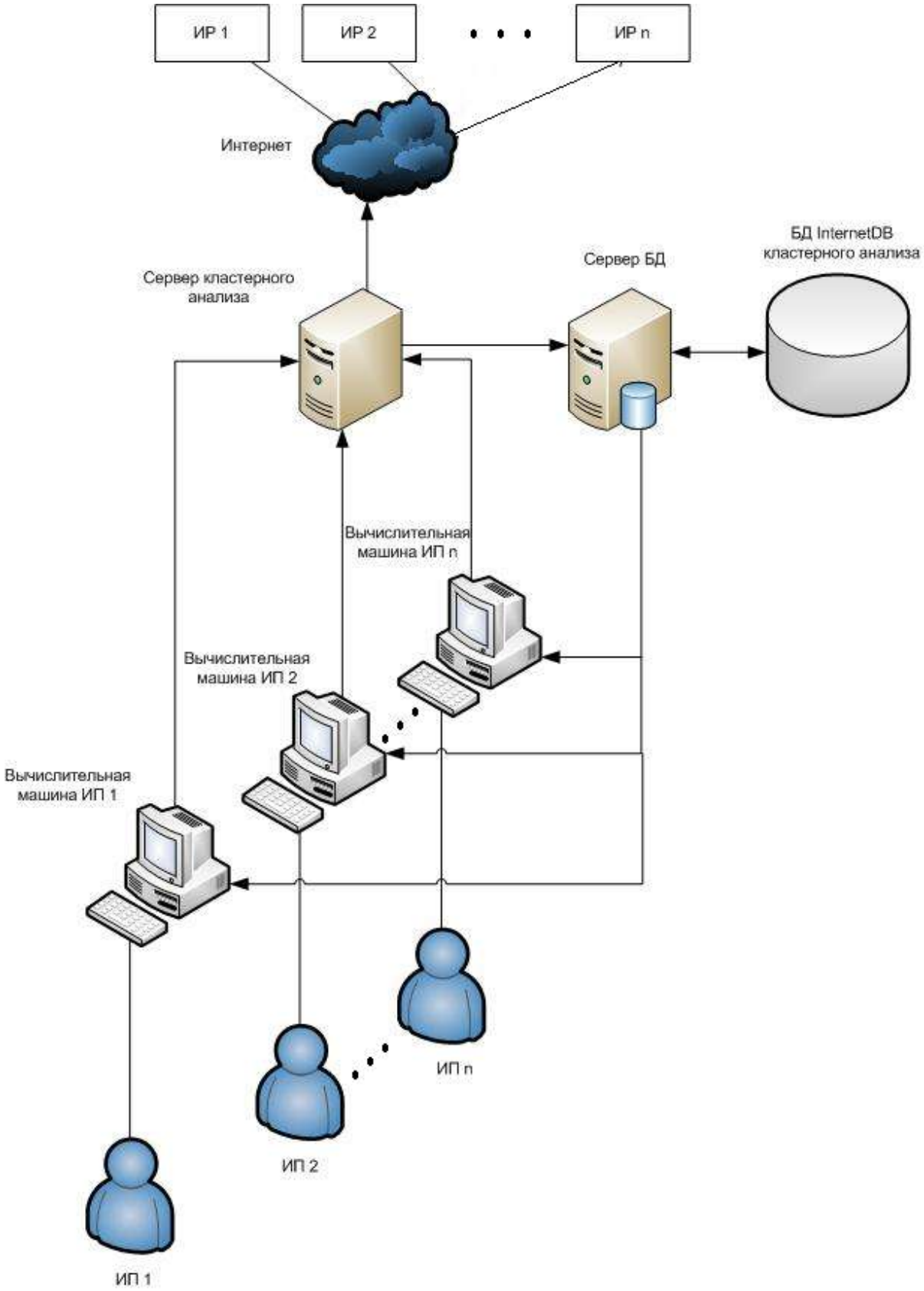

Рисунок 5.1 – Обобщенная структура корпоративной системы персонализации поиска

5.2. Структуризация данных о поисковой активности Интернет-пользователей

Крупные поисковые системы такие, как Яндекс или *Google*, пользуются файлами *cookie* в браузерах и личной информацией, оставленной ИП для персонализации поиска. Поисковые системы хорошо работают с региональными запросами при поиске магазинов и потребительских товаров. Применение региональной и соц-дем информации на стороне сайтов является статическим методом персонализации Интернет контента, т.е. пользователь (его профиль и/или *IP*-адрес) помещается в специальную базу данных, откуда и начинается персонализация содержания сайтов. Несмотря на то, что современные поисковые технологии в полной мере используют статическую информацию об ИП, результаты поиска оставляют желать лучшего.

Каждый ИП имеет свой личный психологический портрет и в течение дня посещает определенный, часто фиксированный набор *web*-страниц. Любой ИП интуитивно формирует собственную систему классификации и отбора ИР для удовлетворения своих потребностей в информации. Несмотря на то, что ИР проводят свою систему анализа поведения ИП, поисковая система не способна проводить классификацию ИР для каждого ИП в отдельности. Пользователи заинтересованы в том, чтобы поисковые системы анализировали их поисковую активность, а не только статическую информацию о поле, возрасте, местонахождении и т.д.

Для исследования поисковой активности Интернет-пользователей в рамках данной работы необходимо спроектировать БД, хранящую информацию о заходах ИП на ИР, а также словарь терминов, содержащихся в соответствующих поисковых запросах.

Структура БД заходов Интернет-пользователей (*InternetDB*).

Поисковая активность ИП отслеживается с помощью специального программного модуля (приложение 8), который записывает данные о его действиях в *Log*-файл. Как только ИП выполняет заход на ИР или покидает ИР, его действия автоматически фиксируется и сохраняется в специальном *Log*-файле,

формат и пример содержимого которого демонстрируются таблицей 5.1.

Таблица 5.1 – Формат файла заходов ИП

resp_id	comp_id	D	url
1	1	2012-08-12T17:19:57.297	http://vk.com/feed
1	1	2012-08-12T17:19:59.733	http://vk.com/audio
2	2	2012-08-12T00:03:23.470	http://vk.com/id4102895
2	2	2012-08-12T00:03:58.513	http://vk.com/id89738988
2	3	2012-08-12T00:05:56.807	http://mail.ru/
2	3	2012-08-12T00:05:56.983	http://mail.ru/

Так как формат файла заходов ИП известен, можно преступить к проектированию БД для обработки и анализа содержащихся в *Log*-файле данных. В конечном итоге, накопленные в *Log*-файлах данные трассировки заходов ИП должны быть загружены в БД для дальнейшей кластеризации ИП по их поведению в Интернете.

Начинаем с определения сущностей. На первом шаге (рисунок 5.2) можно выделить две сущности, характеризующие ИП: собственно «Интернет-пользователь» (*az_resps*) и «город» (*az_cities*).

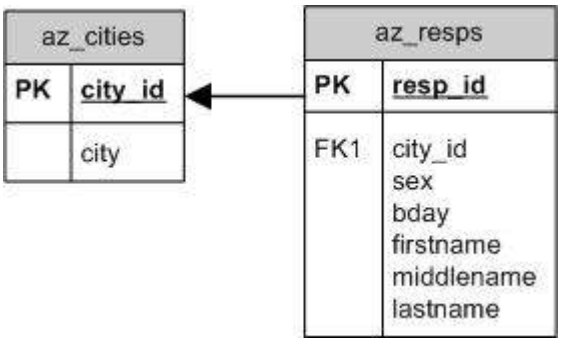

Рисунок 5.2 – Сущности *az_resps*, *az_cities* и логическая связь между ними

Загрузив данные в таблицы *az_resps* и *az_cities*, можно преступить к созданию сущности «заход» (*az_visits*), показанной на рисунке 5.3.

Исходные данные для таблицы заходов отображают заходы ИП на 200 крупнейших Интернет-сайтов, в том числе на порталы *yandex.ru*, *mail.ru*, *rambler.ru*, новостные сайты *rbc.ru*, *gazeta.ru*, *sovsport.ru* и в социальные сети Мой Мир, Одноклассники, вКонтакте и т.п.

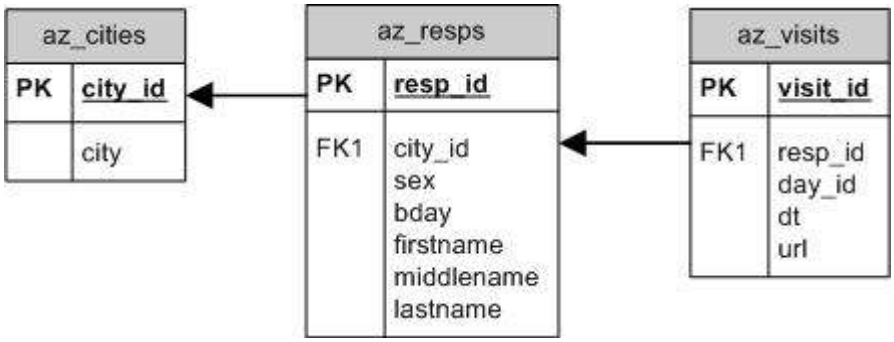

Рисунок 5.3 – Добавление сущности *az_visits*

Ежедневно ИП могут посещать более 2000 страниц, из них более 300 страниц поисковые – предложенные ИП поисковыми системами. Это обусловливает возможность сразу анализировать, как посещаемость сайтов, так и поисковую историю ИП. Таким образом, БД заходов ИП позволит выявить их поведение, интересы и, как результат, позволит сгруппировать в зависимости от интересов и поисковой направленности.

Как уже было сказано, ИР проводят персонализацию Интернет контента на основе статической информации, оставленной посетителями сайтов: пол, дата рождения, возраст и местоположение (легко определяется по *IP*-адресу). Маркетологи, применяя специальные подходы, в зависимости от пола и возраста разбивают ИП на социальные группы [10]. Поэтому в структуру БД необходимо добавить сущность (рисунок 5.4.) «социальная группа» *az_sd*. Для того чтобы связать сущности «Интернет-пользователь» и «социальная группа», создаём дополнительную ассоциативную сущность (рисунок 5.4) *az_resp_sd*. Применение данного подхода позволит распределить обработку заходов по разным базам данных или серверам.

Алгоритм анализа и преобразования поисковых строк, сформированных поисковыми системами, можно разбить на последовательность шагов (рисунок 5.5): получение доменного имени, декодирование, выявление масок, определение ключевых поисковых слов и разделение поискового выражения на отдельные слова.

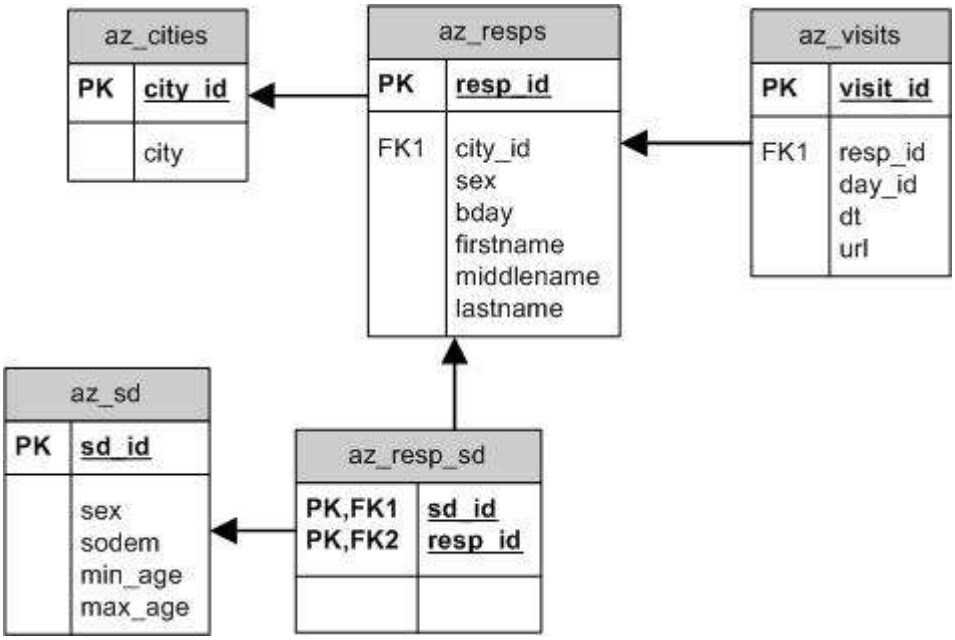

Рисунок 5.4 – Добавление сущности *az_resp_sd*

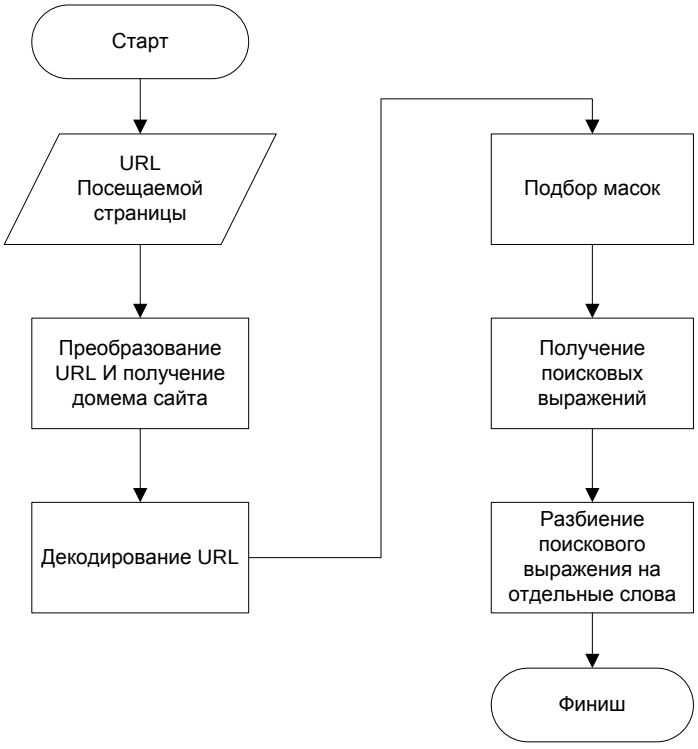

Рисунок 5.5 – Схема алгоритма получения конечных терминов из поисковых строк

На рисунке 5.5 показаны пять основных шагов обработки поисковой *URL*-строки для получения конечных терминов из поисковых строк. Для каждого шага в отдельности рассмотрим, каких изменений он потребует в структуре БД

InternetDB.

Преобразование *URL* и получение домена сайта.

Начнём с преобразования данных заходов с обработки *URL*-страниц. Для этой цели проведём своего рода реорганизацию полученных *URL*. С одной стороны, разные ИП могут посещать одну и ту же страницу, с другой стороны, *URL*, на которые ИП выполняют заходы, имеют разные структуры и длины. Поэтому есть смысл сначала определить уникальные *URL*, затем найти их доменные имена и сгруппировать по сайтам. Здесь нужно разработать специальную функцию *fnGetDomainFromUrl* (исходный код представлен в приложении 9) для получения доменного имени второго и третьего уровня.

Для сохранения уникальных *URL*, необходимо создать новую сущность «страница» (*az_pages*), а для сохранения доменов – сущность «домен» (*az_domain*). Вносим соответствующие изменения в структуру БД (рисунок 5.6).

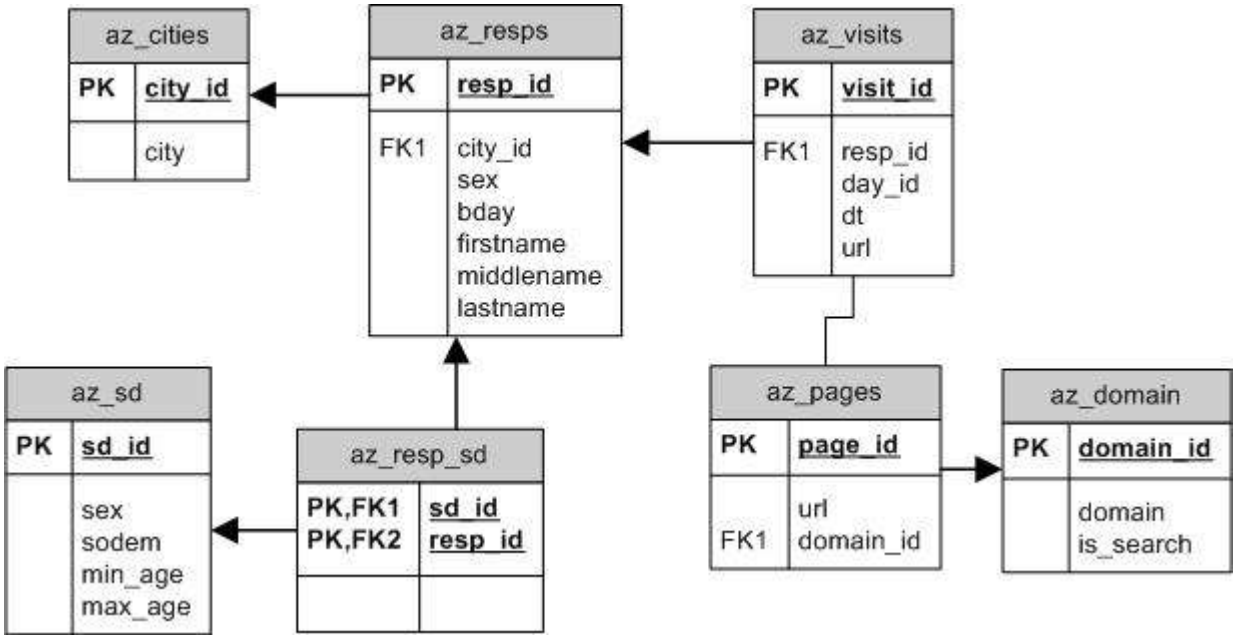

Рисунок 5.6 – Добавление сущностей *az_pages* и *az_domain*

Декодирование *URL* в ядре сервера БД.

При работе с *URL* ресурсов, есть необходимость выделить поисковые запросы, которые предоставляют насыщенную базу для исследования интересов ИП. Поисковые запросы передаются на сервер поисковой системе (*yandex.ru, mail.ru, rambler.ru*) путём инкапсуляции поискового текста в самой *URL*-

страницы, с применением специальной кодировки. Остановимся на способе кодирования поисковой строки для основных поисковых систем в рунете – *yandex.ru, mail.ru* и *rambler.ru*. Наиболее распространёнными кодировками являются *UTF-8* и *ASCII* и *Windows*-1251. Декодирование поисковых строк можно проводить с помощью *web*-сервиса «Универсальный декодер – конвертор кирилицы», который доступен по ссылке *http://2cyr.com/decode/?lang=ru* (01.12.2014 г.).

Для Яндекса инкапсулированная поисковая строка кодированная с помощью *Windows*-1251 имеет вид:

```
http://yandex.ru/yandsearch?text=%D0%BF%D0%BE%D0%B4%D1%80%D1%83%D0%B6%D0%BA%D0%B0+%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD&lr=216&oprnd=1376222967
```

Если скопировать эту ссылку и пройти по ней с помощью *web*-браузера, то получаем результат поиска и перекодируемую *URL*:

```
http://yandex.ru/yandsearch?text=подружка+магазин&lr=216&oprnd=1376222967
```

т.е. поисковые термины «подружка+магазин» были закодированы как `%D0%BF%D0%BE%D0%B4%D1%80%D1%83%D0%B6%D0%BA%D0%B0+%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD` .

Переходим к поисковой строке Мейла, где также применяется кодировка *Windows*-1251. Имеем:

```
http://go.mail.ru/search?q=%D0%B5%D1%80%D1%88%D0%BE%D0%B2%D0%B0%20%D0%BE%D0%BB%D1%8C%D0%B3%D0%B0%20%D0%B0%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%BD%D0%B4%D1%80%D0%BE%D0%B2%D0%BD%D0%B0&where=any&num=10&rch=l&sf=60
```

Копируем эту ссылку и переходим к *web*-браузеру для получения декодированной поисковой строки:

```
http://go.mail.ru/search?q=ершова%20ольга%20александровна&where=any&num=10&rch=l&sf=60
```

Снова обнаруживаем, что искомые слова «ершова ольга александровна» в

поисковой строке кодируются как:

```
%D0%B5%D1%80%D1%88%D0%BE%D0%B2%D0%B0%20%D0%BE%D0%BB%D1%8C%D0%B3%D0%B
0%20%D0%B0%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%BD%D0%B4%D1%80%D0%BE%D0%
B2%D0%BD%D0%B0
```

У Рамблера система кодировки похожа, применяется тот же *Windows*-1251:

```
http://nova.rambler.ru/srch?query=%D1%80%D0%B0%D0%B4%D0%B8%D0%BE%D1%
81%D0%BF%D0%B5%D0%BA%D1%82%D0%B0%D0%BA%D0%BB%D0%B8%20%D1%81%D0%BA%D0
%B0%D1%87%D0%B0%D1%82%D1%8C%20%D0%B1%D0%B5%D1%81%D0%BF%D0%BB%D0%B0%D
1%82%D0%BD%D0%BE
```

В *web*-браузере видим:

```
http://nova.rambler.ru/srch?query=радиоспектакли%20скачать%20бесплат
но
```

и снова поисковые строки «радиоспектакли скачать бесплатно» кодируются как:

```
%D1%80%D0%B0%D0%B4%D0%B8%D0%BE%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%B0%D
0%BA%D0%BB%D0%B8%20%D1%81%D0%BA%D0%B0%D1%87%D0%B0%D1%82%D1%8C%20%D0%
B1%D0%B5%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D0%BD%D0%BE.
```

Как оказалось, основные поисковые системы рунета применяют кодировку «*Windows*-1251» для поисковых выражений. Кодировка всех букв русского алфавита к *Windows*-1251 показана в приложении 10.

Для автоматизации процесса декодирования поисковых строк, необходимо написать и затем подключить специальную библиотеку динамической компоновки *HttpUtility.dll* (приложение 9). Новая библиотека динамической компоновки должна быть интегрирована в ядро сервера *MS SQL Server* 2012 для возможности дальнейшего применения функций кодирования (*Encode()*) и декодирования (*Decode()*). С появлением возможности декодирования поисковой строки, необходимо внести изменеия в структуру таблицы *az_pages* (рисунок 5.7) – добавляется атрибут декодированной поисковой строки *decoded_url*.

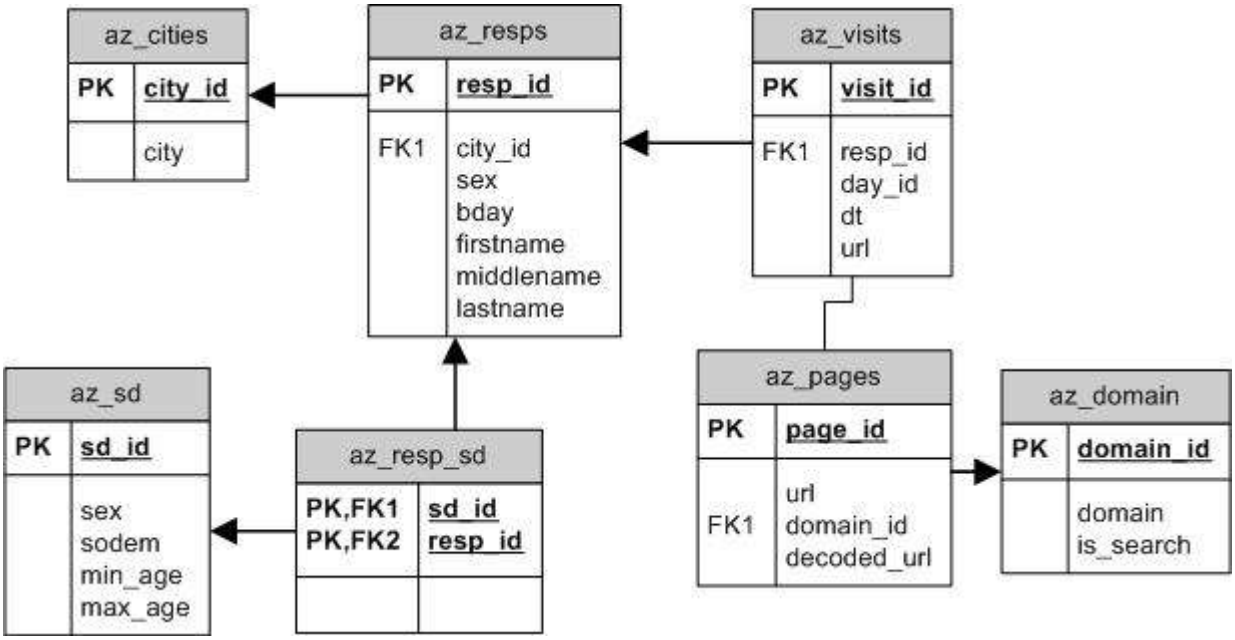

Рисунок 5.7 – Добавление атрибута *decoded_url* в сущность *az_pages*

Подбор масок.

Получив декодированную поисковую строку, обращаем внимание на то, что в дополнении к поисковым терминам появляются вспомогательные «мусорные» термины, поэтому необходимо формирование масок поисковых-*URL* с целью устранения «мусора».

Проведём статистический анализ строковых фрагментов поисковой строки. Для Яндекса выделим следующие маски: «*http://yandex.ru/yandsearch?*», «*&lr=*», «*&clid=*», «*&from*», «*&oprnd=*» с соответствующими процентами попадания 99.98%, 97.51%, 30.92%, 2.14%, 0.53%. Для Мейла выделим маски «*http://go.mail.ru/search?*», «*mailru=*», «*srch?*», «*iewtf=*», «*&us=*», «*ussp=*» с соответствующими процентами попадания 100%, 28.06%, 8.39%, 9.71%, 25.03%, 2.49%. Маски для Рамблера: «*http://nova.rambler.ru/*», «*search?*», «*srch?*» с соответствующими процентами попадания 100%, 63.48%, 30.06%.

После проведённого статистического анализа поисковых строк можно сформировать обобщённые маски для рассматриваемых поисковых сайтов.

Таблица 5.2 – Обобщённые маски поисковых сайтов

Сайт	**Маска**
yandex.ru	`http://yandex.ru/yandsearch?text= <  > &lr=<`

	>&oprnd=< >
mail.ru	http://go.mail.ru/search?q=< >%20< >%20< >&where=any&num=< >&rch=< >&sf=< >
rambler.ru	http://nova.rambler.ru/srch?query=< >%20< >%20< >

Маски поисковых сайтов сформированы и появляется необходимость добавления двух новых сущностей – «маска» (*az_mask*) и «маска-домен» (*az_domain_mask*) в структуру БД (рисунок 5.8).

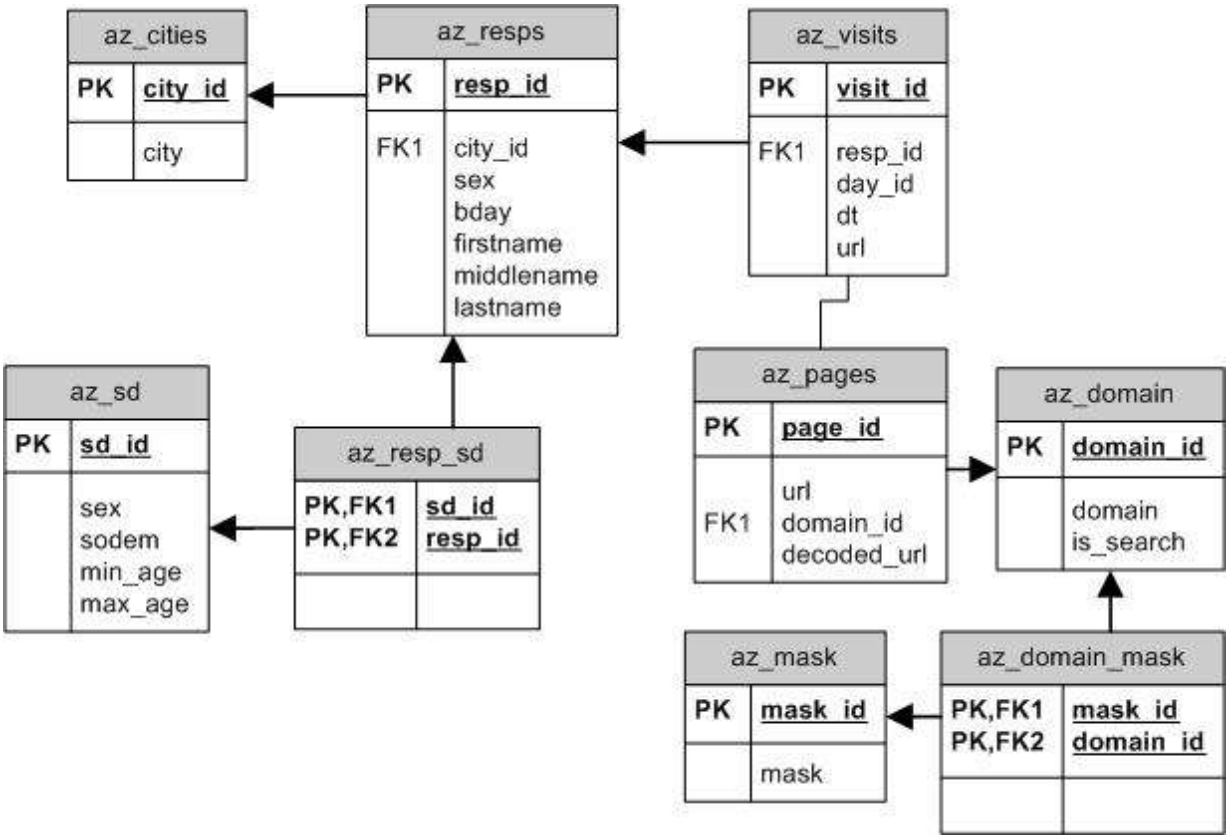

Рисунок 5.8 – Добавление сущностей *az_mask* и *az_domain_mask*

Получение ключевых поисковых выражений.

Поисковые маски выявлены и загружены. Можем приступать к формированию статистики упомянутых в запросах терминов. Для этого необходимо перейти к обработке поисковых *URL*, выделению поисковых терминов, их лемматизации и затем к расчету статистики встречаемости терминов. При рассмотрении *URL*, определились ключевые выражения, после которых и располагаются искомые термины пользователей. Для каждого сайта

своё ключевое выражение: для сайта *mail.ru* ключевое выражение – «*q*=»; для *rambler.ru* ключивое выражение – «*query*=»; для *yandex.ru* ключивое выражение – «*text*=».

Учитывая сказанное, необходимо добавлить сущность «ключевое выражение» (*az_key_word*) в структуру БД (рисунок 5.9).

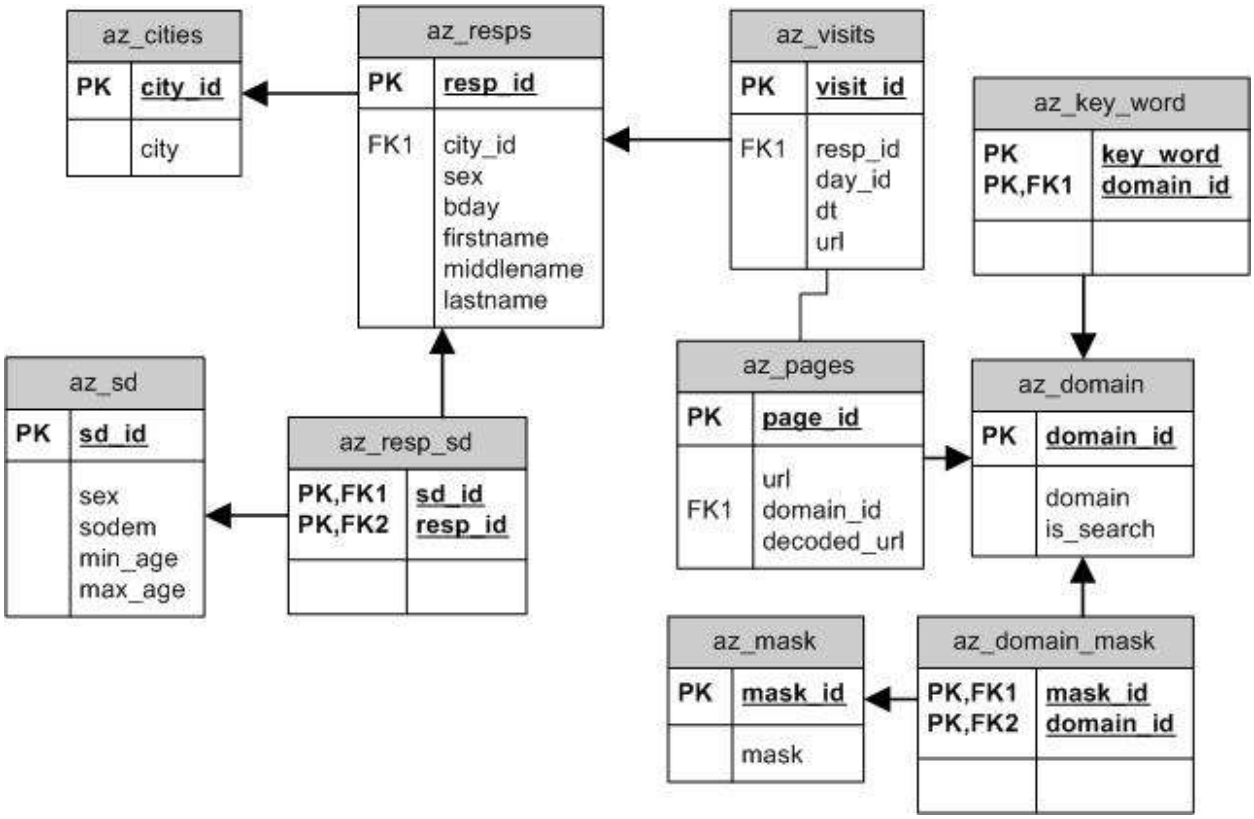

Рисунок 5.9 – Добавление сущности *az_key_word*

Разбиение поискового выражения на отдельные слова.

На этом этапе обработки поисковой строки, значение столбца *decoded_url* таблицы *az_pages* будет состоять из специальных знаков или пробелов и набора слов в поисковой строке (#пробел<слово1> . <слово2> ; <слово3> ,#). Для получения отдельных слов из декодированной и преобразованной *URL*-строки (*decoded_url*) требуется специальная функция *az_split()* (исходный код в приложении 9), которая разбивает длинную строку на отделные слова, если верно указан разделитель между словами. После того, как поисковые слова выявлены, приступим к формированию таблицы статистистики слов. Для этого в БД добавим две новые сущности: «слово» (*az_words*) и ассоциативнуя сущность «слово-поисковая строка» (*az_pages_words*) (рисунок 5.10). Отсутвие уникальных ключей

в таблице *az_pages_words* связано с возможностью появления одного и того же слова несколько раз в поисковой строке.

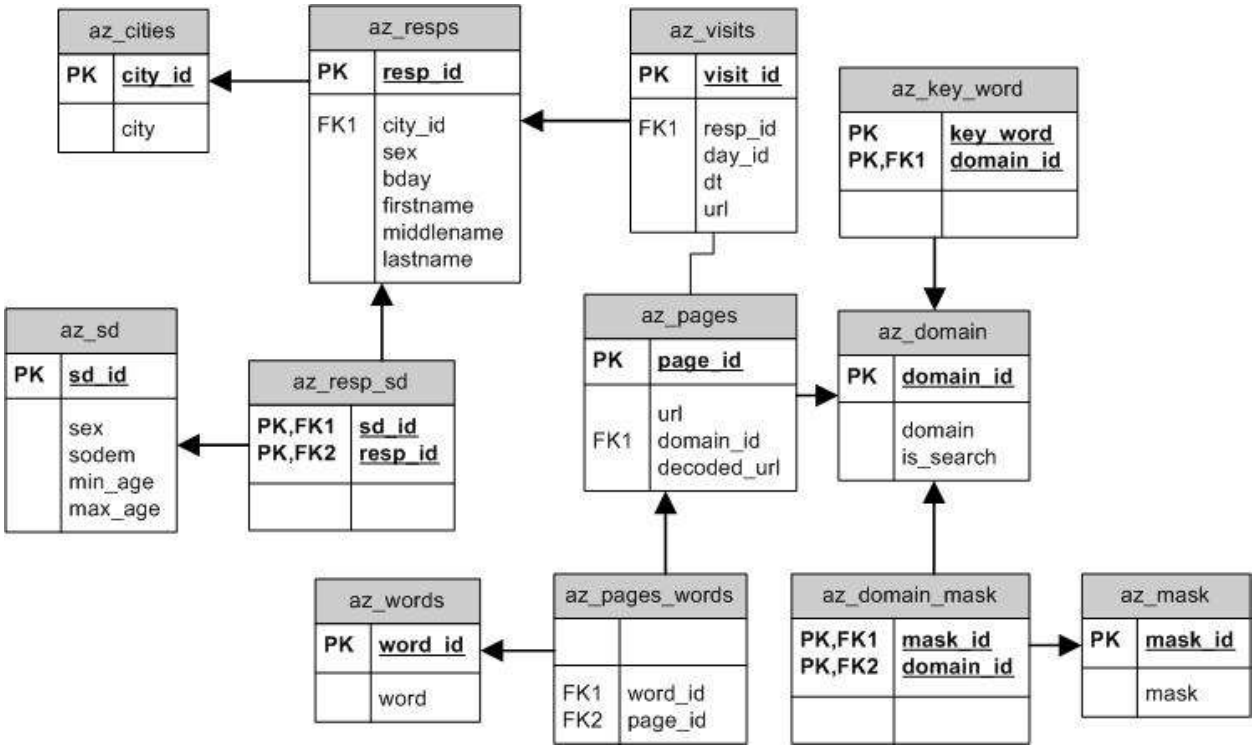

Рисунок 5.10 – Добавление сущностей *az_words* и *az_pages_words*

На рисунке 5.10 представленна полная структура БД для обработки заходов ИП *InternetDB*. БД поисковой активности ИП содержит 12 таблиц, необходимые для структуризации получаемых в процессе наблюдения данных и последующего проведения кластерного анализа ИП по истории поиска.

5.3. Структуризация данных о содержании Интернет-ресурсов

ИР по своей структуре является либо отдельно взятой *web*-страницей, либо набором *web*-страниц с одинаковым доменным именем и связанных между собой гиперссылками. Для визуальной интерпретации ИР необходимо использовать *web*-браузер, который преобразует тэги *DOM*-модели в соответствующие визуальные образы. Например, тэги <*p*> и </*p*> указывают на начало и конец параграфа, <*table*> и </*table*> указывают на таблицу, а <*title*> и </*title*> – на заголовок ИР. В свою очередь каждый тэг может содержать атрибуты, дополняющие их визуальные характеристики.

Для ИР *HTML* является стандартным языком разметки *web*-страниц. В 2000 году был опубликован расширенный язык разметки *web*-страниц *XHTML*. Со временем ИР становятся все более интерактивными и динамичными благодаря применению динамических компонентов, инкапсулированных в их *DOM*-модели.

Структура БД для хранения и обработки данных о содержании ИР.

Web-разработчики не ограничиваются классическим *HTML*-кодом, используя при создании ИР различные технологии, – *CSS, JavaScript, Ajax,* приводящие к появлению динамических компонентов. В связи с этим необходимо проводить инспекцию ИР исходя из *DOM*-моделей *web*-страниц. Использование *DOM*-модели позволяет получить доступ к любым элементам ИР и их атрибутам. Это даёт возможность манипулировать *web*-документами, как объектами (*object*), со всеми их компонентами, их атрибутами и свойствами. *DOM*-модель позволяет представить ИР в виде дерева, каждый узел которого может быть одновременно, как родительским (*parent*), так и дочерним (*child*) узлом по отношению к другому узлу дерева (рисунок 5.11). Тэг *HTML* является начальным звеном *DOM*-модели, корнем начиная с которого «растает дерево» ИР.

Для получения доступа к конкретному тэгу *HTML*-документа, необходимо пройти путь от корневого узла (*HTML*-тэга) до целевого узла и затем прочитать значения конкретных атрибутов. С помощью *DOM*-дерева *HTML*-документа можно разбить содержание ИР на параграфы, списки, разделы, гиперссылки и другие компоненты из структуры страницы.

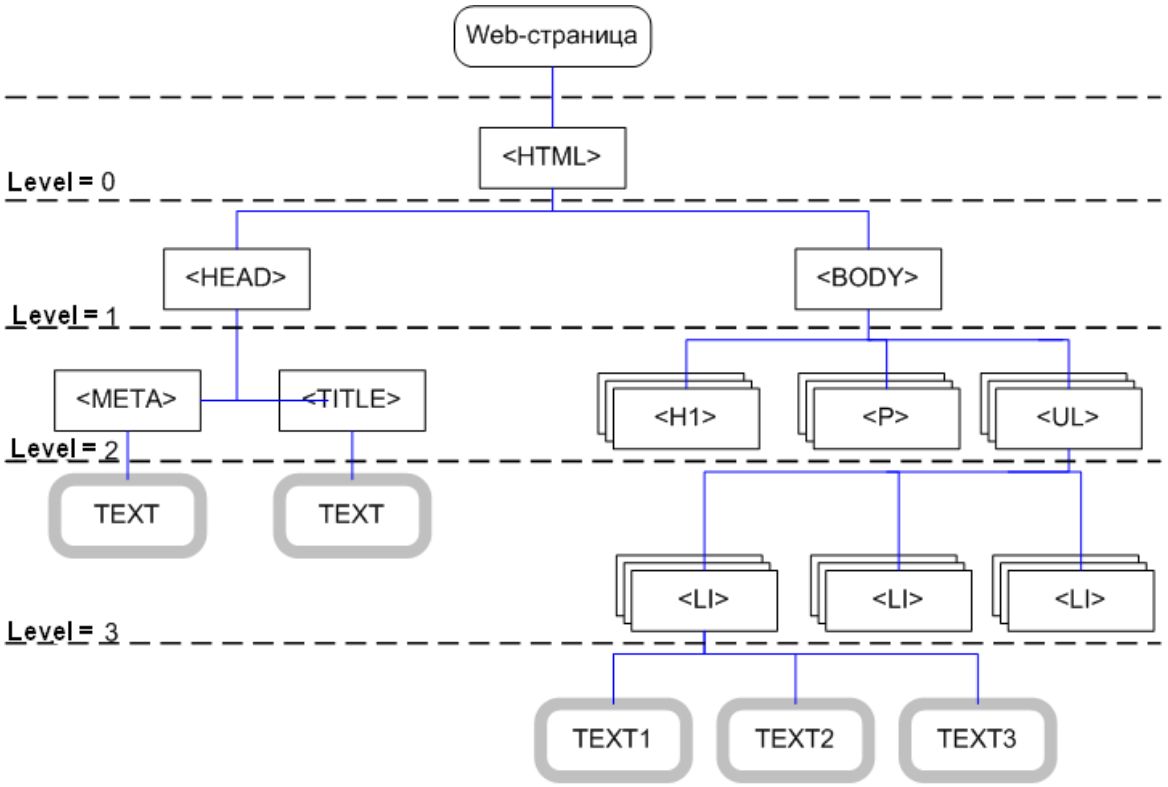

Рисунок 5.11 – Пример *DOM*-дерева *web*-страницы

Для доступа к элементам *DOM*-модели имеются два наиболее распространенных метода:

- прямой доступ к *HTML*-элементу (*htmlElement*) *DOM*-модели по уникальному идентификатору. В данном случае, необходимо наличие уникального идентификатора требуемого *HTML*-элемента. Например, на главной странице *mail.ru* имеем:

```
<div class="portal-headline__projects" id="portal-headline__box">;
```

- доступ к *HTML*-элементу по названию тэга. В этом случае необходимо сначала отобрать набор (*htmlElementCollection*) с конкретным названием тэга, а затем произвести поиск нужного тэга по значениям атрибутов. Например, на главной странице *mail.ru*, чтобы найти этот *HTML*-элемент, необходимо сначала найти набор элементов, у которых тэг называется «*a*», а затем проводить поиск по атрибуту «*name* = "*clb*598679"»:

```
<a name="clb598679" href="http://my.mail.ru"
class="social__title__link"><i class="social__title__link__icon icon
icon_social icon_social_big icon_social_my"></i><span
```

```
class="social__title__link__text">Мой Мир</span></a>
```

В среде *Microsoft Visual Studio* 2010 достаточно воспользоваться объектом *System.Windows.Forms.WebBrowser* для доступа к объектам *DOM*-модели, загруженных *URL*. Основными функциями для работы с *HTML*-элементами являются:

- *getElementById* – функция, возвращающая ссылку на узел документа, которую можно использовать для чтения и редактирования свойств и обращения к методам узла;

- *getElementByTagName* – функция, возвращающая массив из элементов, имеющих конкретный тэг;

- *getAttribute* – функция, возвращающая значение конкретного атрибута *HTML*-элемента.

В алгоритме доступа к *DOM*-элементам (рисунок 5.12) *getElementById* имеет более высокий приоритет, чем *getElementByTagName*. Это актуально, особенно для ИР с одинаковой структурой, когда разные *URL* одного и того же ИР имеют идентичную *DOM*-модель.

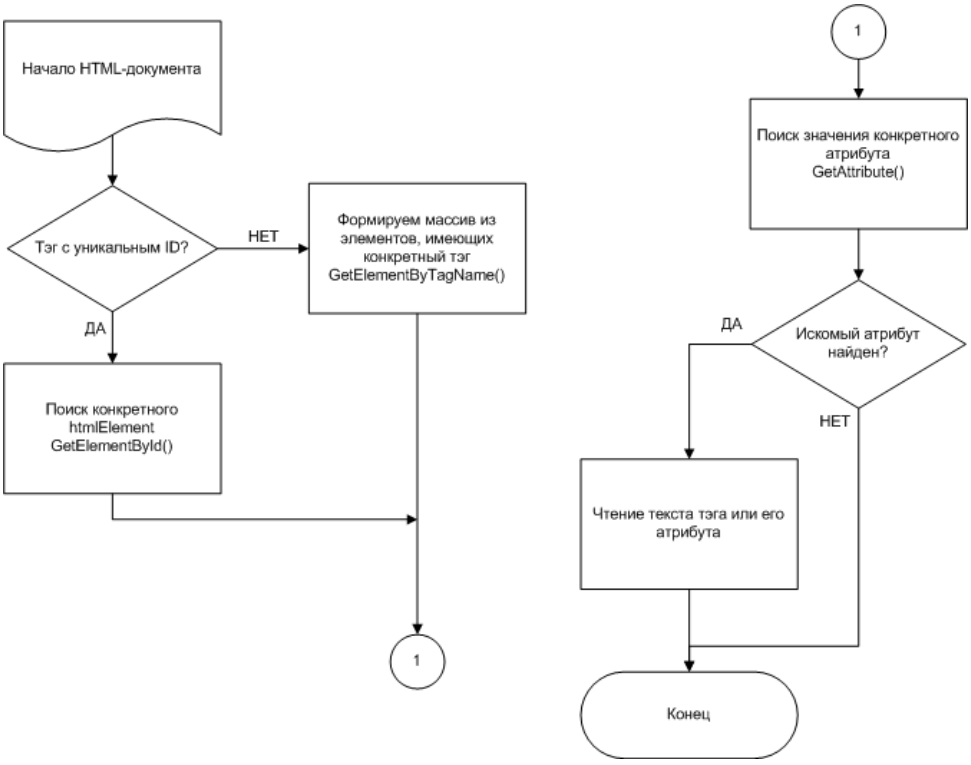

Рисунок 5.12 – Схема алгоритма доступа к *DOM*-элементам

Для структуризации содержания ИР по тэгам достаточно дополнить ранее разработанную структуру БД для ИП (рисунок 5.10) еще 3-мя сущностями. Для начала необходимо выделить отдельную сущность для всех *URL* исследуемых ИР, создав таблицу *Pages*. Весь справочник *HTML* тэгов [47] *DOM*-модели разместим в словарной таблице *HTML_element*. Результаты чтения тэгов расположим в ассоциативной сущности *HTML_value*. На рисунке 5.13 показана структура из трёх таблиц для хранения данных о тэгах и их значениях.

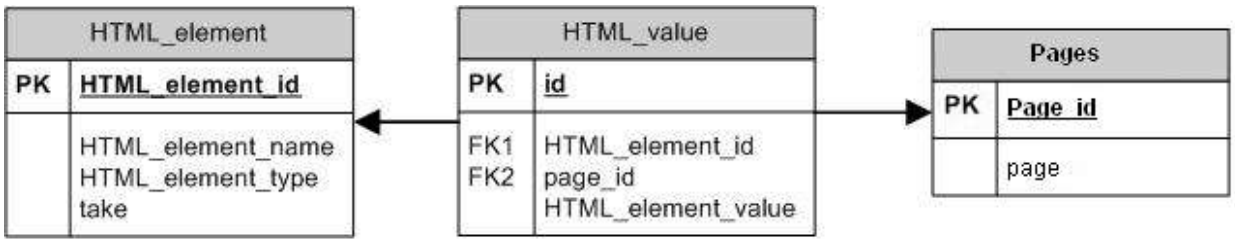

Рисунок 5.13 – Структура БД для хранения данных о тэгах и их значениях

Используя экспериментальные данные о заходах ИП, полученные в период с 12 по 18 ноября 2012 года, можем определить топ-100 наиболее популярных *URL*. За указанную неделю было всего выполнено 6848059 заходов из них 116674 заходов на новостные разделы основных Интернет-порталов – *news.yandex.ru*, *news.mal.ru* и *news.rambler.ru*. Заходы выполнялись пользователями разного пола, возраста и места жительства на территории РФ. По результатам эксперимента стандартным способом группировки по *URL*-страницы сформирована таблица 5.3 топ-100 популярных новостных страниц рунета.

Таблица 5.3 – Список наиболее популярных новостных страниц

page_id	Page	Cnt
1	http://news.yandex.ru/	1281
2	http://news.rambler.ru/	1109
3	http://news.mail.ru/	824
4	http://news.rambler.ru/16368909/	822
5	http://news.rambler.ru/16414323/	698
6	http://news.rambler.ru/16334722/	665
7	http://news.rambler.ru/16320803/	589
8	http://news.rambler.ru/16413374/	549
9	http://news.rambler.ru/16341657/	491
10	http://news.mail.ru/inregions/st_petersburg/91/society/	490

	0907518/?frommail=1	
…	…	…
98	http://news.rambler.ru/16370673/	166
99	http://news.mail.ru/politics/10940888/?frommail=1	163
100	http://news.rambler.ru/16352490/	162

По списку *URL*, представленному в таблице 5.3, можно запустить специально разработанного программного робота *HTMLDocDom* (исходный код программы в приложении 11) для считывания содержимого *DOM*-модели.

Структуризация объектов исследования – ИП и ИР – является первым этапом решения задачи персонализации поиска. Со структурированными данными можно проводить исследования, подтверждать или опровергать эффективность различных методов кластеризации и классификации.

5.4. Описание программных модулей *internet_res_search* и *ie_analyzer*

С учётом предложенной выше структуры корпоративной системы персонализации поиска, с использованием принципов объектно-ориентированного программирования было разработано два программных модуля *internet_res_search* и *ie_analyzer*, имеющих графический пользовательский интерфейс. В качестве операционных систем, для которых было разработано программное обеспечение КСПП, выбраны операционные системы ряда *MS Windows*, поскольку на сегодняшний день они имеет наибольшее распространение. В качестве инструментальной системы выбрана *MS Visual Studio* 2010, обеспечивающая большие возможности для разработки и отладки объектно-ориентированных приложений под *MS Windows*, а также имеющая богатый набор визуальных компонентов для построения интуитивно понятного и удобного графического интерфейса пользователя. Преимуществом среда разработки *MS Visual Studio* 2010 является также возможность программирования на таких языках высокого уровня, как *Visual Basic*, *Visual C#* и *Visual C++*.

В приложениях 6 и 7 приводится полный исходный код программ. В данном же параграфе кратко рассмотрим их особенности и ключевые моменты.

Интерфейс программы и логика работы *internet_res_search.*

Графический интерфейс пользователя программного модуля *internet_res_search* достаточно прост (рисунок 5.14).

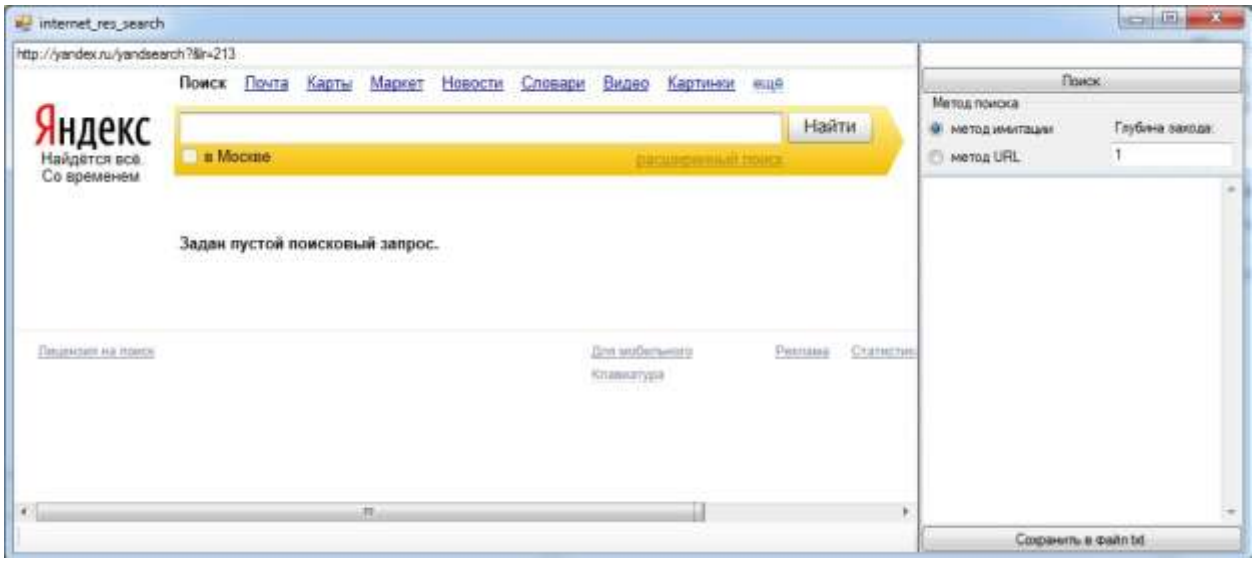

Рисунок 5.14 – Графический интерфейс программы *internet_res_search*

В верхнем левом углу выгружается *URL*-адрес текущий загруженной страницы поисковой системы Яндекс. Сразу под *URL*-адресом располагается сам браузер (компонент *WebBrowser*). В правом верхнем углу располагается специальное поле для поисковых слов, а под ним кнопка «Поиск». Интерфейс предусматривает два метода автоматической навигации: «метод имитации» и «метод *URL*». Каждый из указанных методов выполняет автоматическую поисковую навигацию в зависимости от указанного уровня глубины захода. Каждый из методов имеет свои особенности. Метод имитации – работает на уровне *DOM*-модели страницы, когда проводится поиск конкретных элементов страницы, а затем осуществляется имитация нажатия кнопки мыши. Например, для имитации нажатия кнопки «Поиск» на главной странице Яндекса, необходимо сначала найти элемент <*input class*="*b-form-button__input*" *type*="*submit*" *value*="" *tabindex*="2" *hidefocus*="*true*"/>, а затем вызвать событье "*click*": *htmlElement.InvokeMember*("*click*"). Метод *URL* – отлично подходит для навигации поисковой системы Яндекса. Как уже было сказано раннее, в поисковой строке может быть инкапсулировано множество управляющих команд, позволяющих

управлять поисковой отдачей удалённых серверов. Например, для перехода к 5-ой странице поисковых результатов нужно указать номер страницы уменьшенный на 1 (*p*=4) в самой поисковой *URL*: *http*://*yandex.ru/yandsearch*?*p*=4...и т.д. После нажатия кнопки «Поиск» происходит поисковая навигация и как только страница загружается, осуществляется сканирование *HTML*-кода и отбор гиперссылок в том же порядке, в каком их выгружает сама поисковая система. Список гиперссылок выгружается в специальное поле, расположенное в правой части окна графического интерфейса пользователя, с возможностью его сохранения в текстовый файл для дальнейшей обработки.

Формат выходного файла.

Выходным файлом является файл с расширением *txt*. Каждая строка этого файла имеет следующий формат: <*url*>*n*, где *url – URL* гиперссылки, выданной поисковой системой. Исходный код программы и примеры выходных файлов находятся в приложении 6.

Интерфейс программы и логика работы *ie_analyzer*.

Интерфейс программы *ie_analyzer* состоит из ограниченного числа визуальных компонентов (рисунок 5.15):

- полей «Ваш логин» (на почте *mail.ru*) и «Пароль». При желании пользователь может ввести свой личный логин и пароль. Если в корпоративной сети предприятия имеется почтовый сервер, то можно будет привязать к каждому пользователю его личный почтовый адрес, откуда будет происходить рассылка *Log*-файла со списком посещаемых ИР на сервер кластерного анализа (см. рисунок 5.1);

- кнопки «Запуск с *Windows*» и «Удалить из реестра», предназначенные для включения и отключения автоматического запуска программы при загрузке *Windows*, так как имеется возможность установки программы с постоянным запуском при загрузки ОС и тем самым нет никакой необходимости перезапускать её заново.

- кнопка «Отправка *e-mail*» для отправки *Log*_файла *ie_analyzer* сразу на сервер кластерного анализа или на корпоративный почтовый сервер, а затем на

сервер кластеризации.

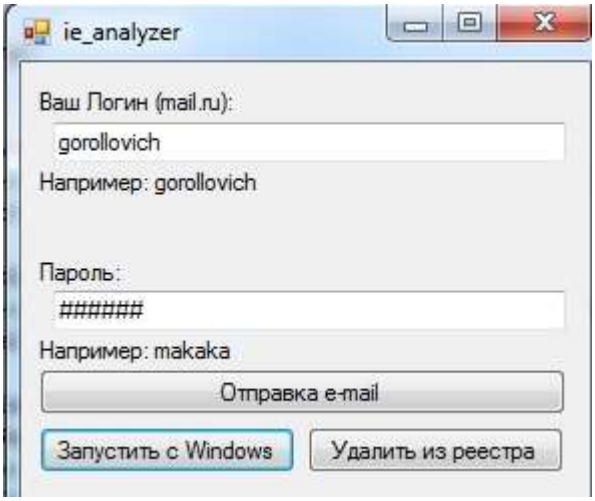

Рисунок 5.15 – Графический интерфейс программы *ie_analyzer*

Напомним, что рассматриваемый программный модуль позволяет отслеживать посещаемость Интернет-ресурсов ИП. Данные о всех посещённых ИР будут отправлены для дальнейшей обработки и кластеризации в КСПП. Достаточно будет установить программу *ie_analyzer* один раз и добавить её в реестр *Windows* нажатием кнопки «Запуск с *Windows*» для постоянного мониторинга активности ИП.

Формат выходного файла.

Выходным файлом является файл с расширением *txt*. Каждая строка этого файла имеет следующий формат: *<a>\T<b>\T<c>\T<d>\T<e>*, где *a* – название браузера (например, *InternetExplorer*); *b* – уникальный идентификатор объекта (*handle*) окна браузера; *c* – *URL* браузера (адрес Интернет-ресурса); *d* – время посещения Интернет-ресурса; *e* – состояние *URL* (активная *URL*-страница или закрытая *URL*-страница). Исходный код программы и примеры выходных файлов находятся в приложении 7.

Алгоритмы работы программных модулей на стороне ИП.

Первым, основным источником исходной информации о поисковой деятельности ИП являются *Log*-файлы, сформированные программой *ie_analyzer*. Установив эту программу на ПК, ИП автоматически становится объектом кластерного анализа КСПП. Вся поисковая история и история заходов

автоматически становится доступной для кластерного анализа. В программном коде *ie_analyzer* есть возможность установки временн*о*го счётчика (*Timer*) для автоматизации отправления *Log*-файлов по истечению определённых промежутков времени на сервер. Вторым источником информации для анализа со стороны ИП может служить программа *internet_res_search*, которая позволяет учитывать глубину поиска и формировать поисковую историю. Программа *internet_res_search* может быть установлена на стороне сервера кластерного анализа для сравнения результатов поиска до кластеризации, и после её применения.

В разработанной программе *ie_analyzer* происходит мониторинг запущенных приложений в оболочке ОС *Windows ShellWindows*. С момента запуска программы осуществляется постоянное слежение за процессами (задачами) внутри оболочки *ShellWindows*, т.е. проводится постоянные наблюдения за процессами, которые можно найти в диспетчере задач *Windows*. Как только запускается приложение *iexplorer*, получаем его *handle* и начинаем следить за его активность – любые события фиксируются в *Log*-файле. Алгоритм работы программного модуля представлен на рисунке 5.16.

Разработанная программа *internet_res_search* выполняет две основные задачи: имитацию выполнения поиска и сбор гиперссылок со страниц поисковой системы. Как уже было сказано, эта программа может быть установлена, как на стороне ИП, так и на стороне сервера кластерного анализа. Данная программа отслеживает состояние встроенного браузера и выделает список гиперссылок, который формирует поисковая система. Программа реализует два метода – метод имитации и метод *URL*. Алгоритм работы программного модуля представлен на рисунке 5.17. и рисунке 5.18.

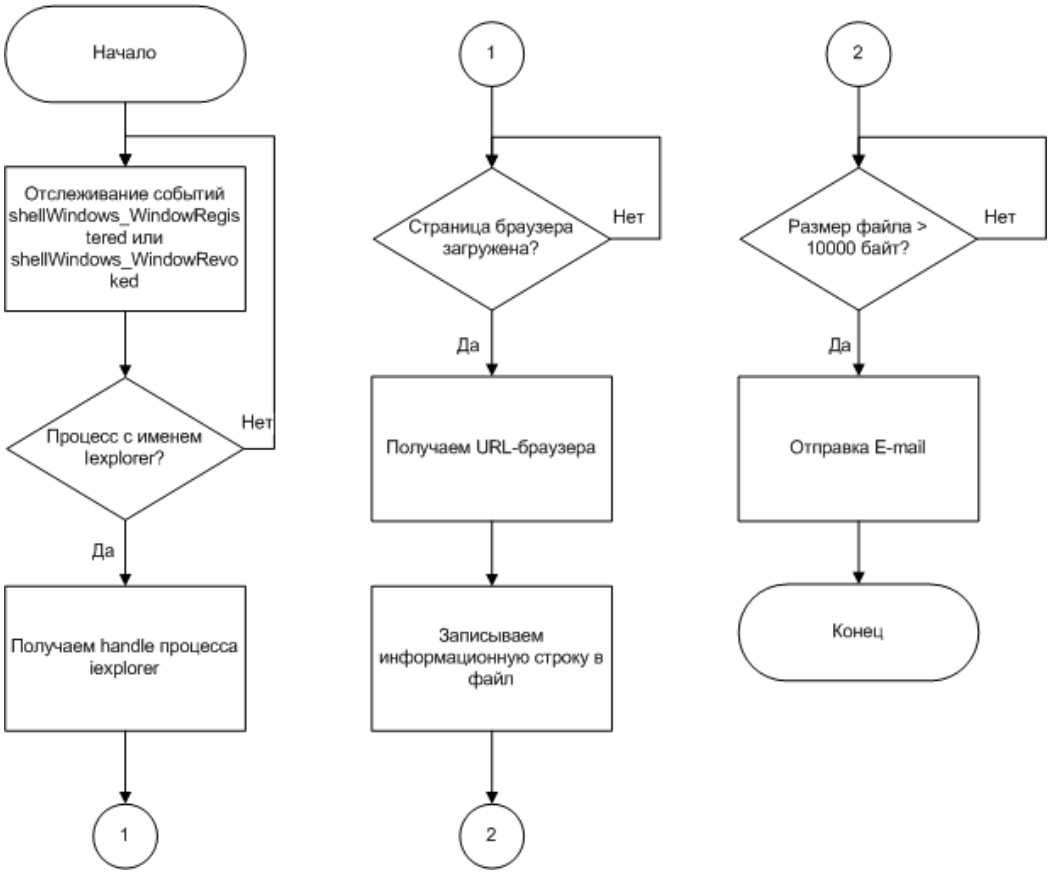

Рисунок 5.16 – Схема алгоритма работы программного модуля *ie_analyzer*

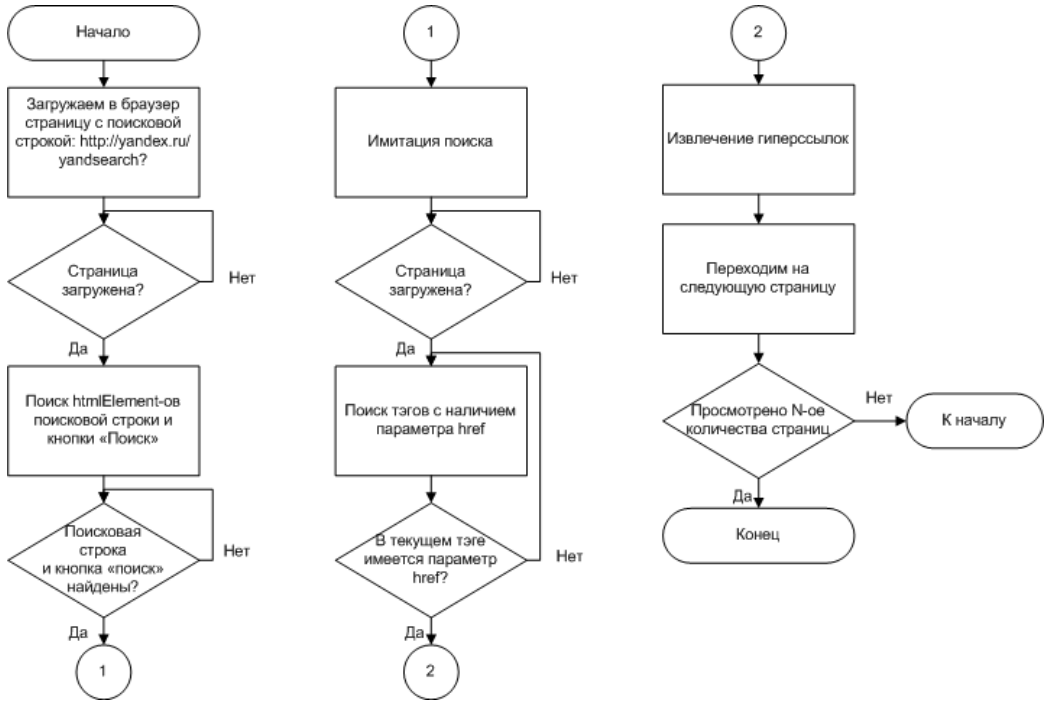

Рисунок 5.17 – Схема алгоритма работы программного модуля *internet_res_search* в режиме имитации выполнения поиска

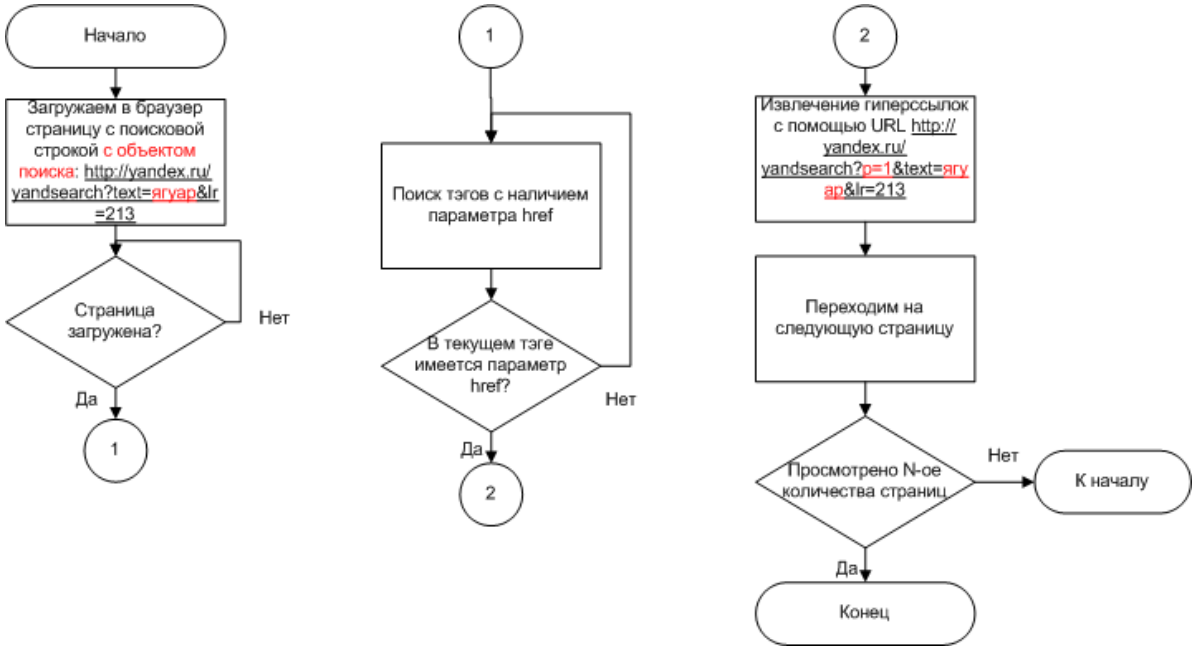

Рисунок 5.18. – Схема алгоритма работы программного модуля *internet_res_search* в режиме применения *URL*

Алгоритм начинает выполняться после захода на страницу *http://yandex.ru/yandsearch?* с помощью встроенного браузера. Ожидается момент полной загрузки страницы, проводится поиск *htmlElement*-ов поисковой строки и кнопки «Найти». Затем имитируется запись «что ищем» в поисковой строке и нажатие кнопки «Найти». После ожидания полной загрузки страницы извлекаются все гиперссылки. Число итераций зависит от указанного уровня «Глубина захода».

В целом оба метода (рисунок 5.17 и рисунок 5.18) достигают одного и того же результата, но ввиду того, что на сайтах крупных поисковых систем постоянно могут возникать изменения в коде страницы и в её *DOM*-модели, необходимо поддерживать работоспособность модуля в двух режимах.

5.5. Описание программного модуля *HTMLDocDom*

Программный модуль *HTMLDocDom* и его графический интерфейс разработаны в среде *MS Visual Studio* 2010. Он работает под управлением ОС *MS Windows*. Модуль *HTMLDocDom* должен быть установлен на сервере кластерного анализа.

В приложении 11 приводится исходный код программы, в данном же параграфе кратко рассматриваются особенности и ключевые моменты программы.

Интерфейс пользователя и логика работы *HTMLDocDom*

Программа *HTMLDocDom* имеет достаточно простой интерфейс (рисунок 5.19).

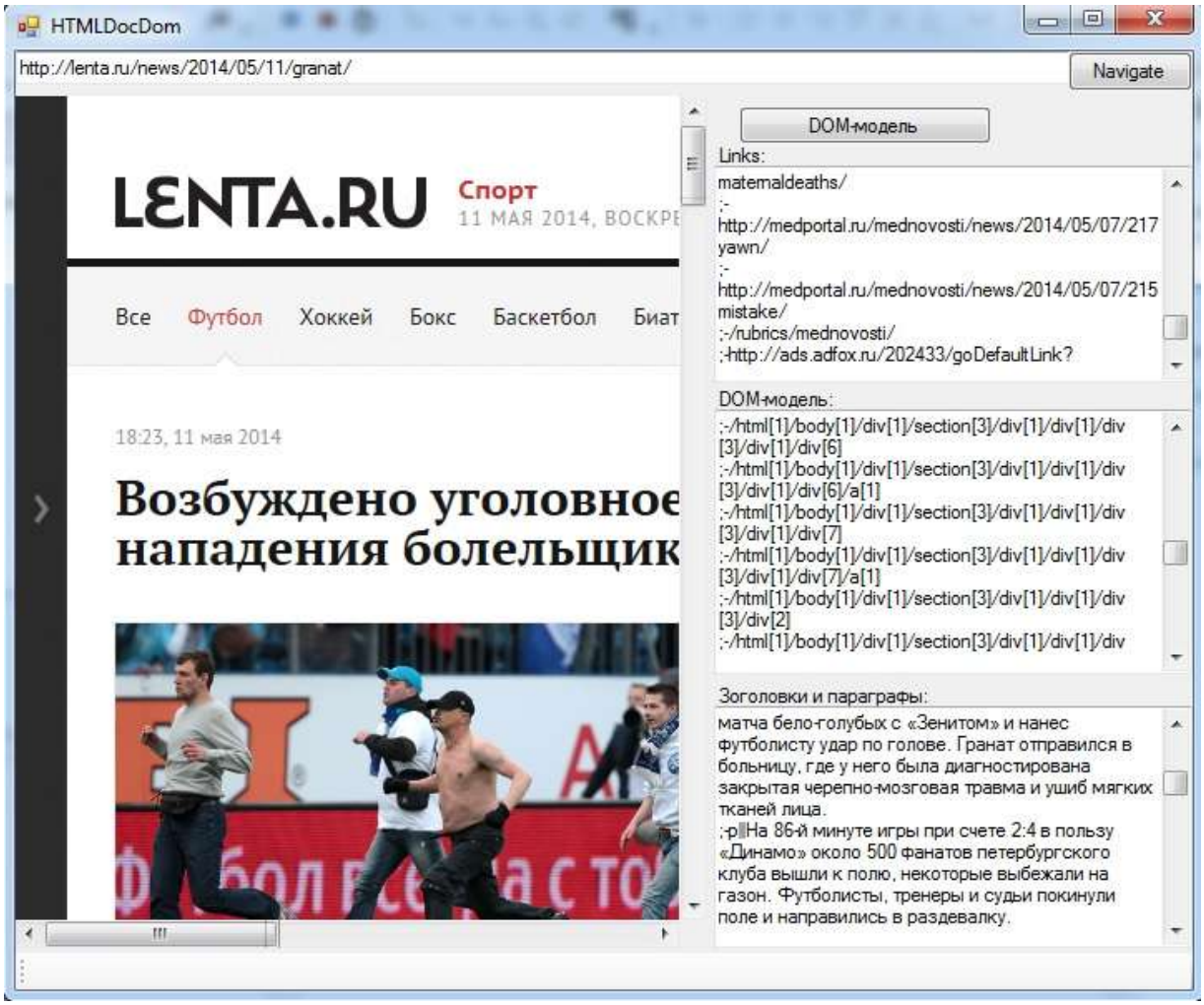

Рисунок 5.19 − Графический интерфейс программы *HTMLDocDom*

Главным модулем этой программы является библиотека *HtmlAgilityPack* [8], которая позволяет извлекать из *DOM*-дерева и тем самым применить числовые коэффициенты усиления (п. 3.2) и трёхтактную кластеризацию ИР с обратной связью (п. 3.3).

Формат выходного файла.

В текущей версии программы *HTMLDocDom* предусмотрено всего три

выходных файла:

- файл гиперссылок – <*http*-гиперссылка>;

- файл *DOM*-модели – <*xpath*>;

- файл заголовок и параграфов – <текст>.

Алгоритм работы программного модуля *HTMLDocDom.*

Алгоритм доступа к *HTML*-элементам и *DOM*-модели представлен на рисунке 5.20.

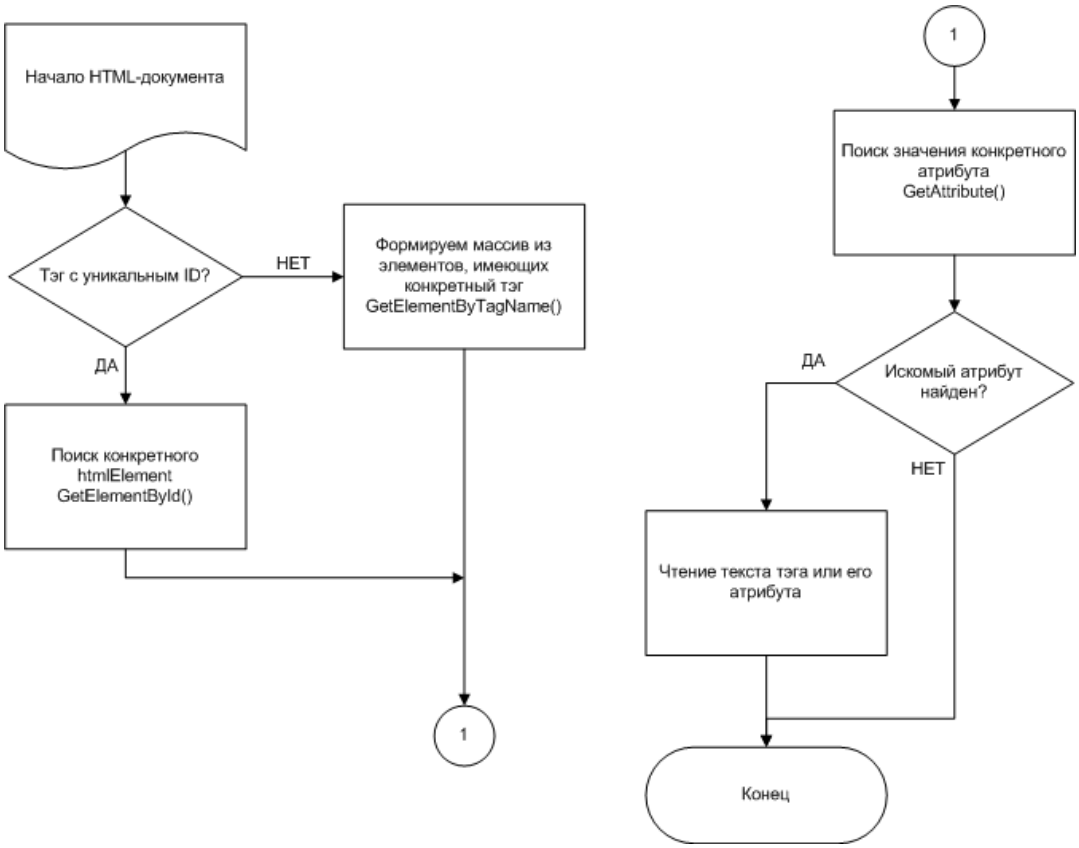

Рисунок 5.20 – Схема алгоритма работы программного модуля *HTMLDocDom*

Доступ к *HTML*-элементам и *DOM*-модели в среде *Microsoft Visual Studio* 2010 возможен с помощью следующих функций (*.NET Framework* 4 и более поздние версии): *getElementById()*, *getElementByTagName()*, *getAttribute()*. Функция *getElementById()* возвращает ссылку на узел документа, которую можно использовать для чтения и редактирования свойств, а также для обращения к методам узла. Функция *getElementByTagName()* возвращает массив из элементов, имеющих конкретный тэг. Функция *getAttribute()* возвращает значение

конкретного атрибута *HTML*-элемента.

5.6. Подсистема кластерного анализа и классификации Интернет-пользователей и Интернет-ресурсов

Как уже было сказано, в соответствии с концепцией корпоративной системы персонализации поиска все программы кластерного анализа должны выполняться на сервере, работая с СУБД *MS SQL Server* 2012.

Для стабильной работы системы (включая процессы тестирования, исследования и интерпретации результатов) необходимо выделить более 10 ГБ памяти на жёстком диске, исключительно под БД *InternetDB*. Это связано с объёмом реально сохраняемых и обрабатываемых данных.

В первоначальную структуру БД *InternetDB* (рисунок 5.10) на этапе реализации были добавлены вспомогательные сущности: *az_small_words* – таблица урезанных терминов и *az_pages_words_power* – таблица привязки ИР к терминам с учётом их весов, в зависимости от типа терминов (0 – для поисковых терминов ИП, 1 – для заголовков, 2 – для краткого описания или вступления и 3 – само содержание ИР).

Алгоритм кластерного анализа и классификации объектов реализован в среде *MS SQL Server* 2012. Объекты исследования – ИП и ИР – были представлены с помощью характеристических векторов и размещены в таблицах БД. Для реализации кластерного анализа применялся метод *k*-средних. Процесс полностью автоматизирован и не требует никакого стороннего вмешательства. Эксперт может лишь варьировать входные параметры системы, если кластеризация не приводит к априори ожидаемым результатам.

Структура подсистемы кластерного анализа, который состоит из главного модуля (запуска и управления) и четырех расчётных процедур, показанных на рисунке 5.21. Эксперт – аналитик, получающий результаты кластерного анализа и имеющий возможность контролировать такие параметры, как число кластеров в кластерной структуре, размер временны*х* интервалов наблюдения за активностью ИП и мера близости объектов кластеризации.

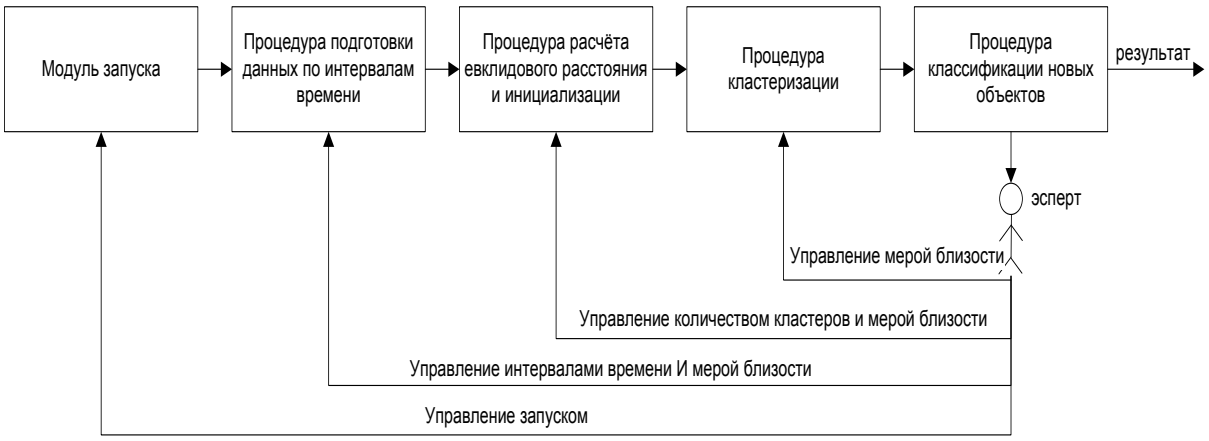

Рисунок 5.21 – Структура подсистемы кластерного анализа

Модуль запуска и управления.

Модуль запуска и управления реализован как *SQL*-скрипт, контролирующий ход выполнения всего алгоритма подсистемой кластерного анализа. В этом модуле управления имеются все необходимые переменные для управления кластерным анализом:

```
-- Список переменных и их инициализация для первой процедуры подготовки данных по интервалам времени.
declare @step1_node1_coef smallint -- числовой коэффициент усиления для заголовка.
SET @step1_node1_coef = 10
declare @step1_node2_coef smallint -- числовой коэффициент усиления для краткого описания.
SET @step1_node2_coef = 5
declare @step1_nodesearch_coef smallint -- числовой коэффициент для поисковых терминов ИР.
SET @step1_nodesearch_coef = 50
declare @step1_hour_step smallint; -- числовой коэффициент поискового интервала ИП.
SET @step1_hour_step = 4; -- переменная для интервала времени.
declare @step1_day_id int; -- переменная для первого дня недели.
SET @step1_day_id = 4494; -- код дня, соответствует 2014.04.21
declare @step1_recount bit; -- переменная признак 0 - не повторяем подготовку данных; 1 - иначе.
SET @step1_recount = 1; -- подготавливаем данные.
declare @step1_online_hours_day smallint; -- переменная количества часов онлайн для ИП
```

```
SET @step1_online_hours_day = 8

declare @step1_online_days_week smallint; -- переменная количества дней онлайн для ИП

SET @step1_online_days_week = 7

declare @step1_word_len smallint; -- переменная длины терминов.

SET @step1_word_len = 4

-- Запуск первой процедуры подготовки данных по интервалам времени.

EXEC az_clustering_step1 @day_id_begin = @step1_day_id, @node1_coef = @step1_node1_coef, @node2_coef = step1_node2_coef, @nodesearch_coef = step1_nodesearch_coef, @hour_step = @step1_hour_step, @online_hours_day = @step1_online_hours_day, @online_days_week = @step1_online_days_week, @word_len = @step1_word_len, @recount = @step1_recount;

GO

-- Список переменных и их инициализация для второй процедуры расчета меры близости и первоначального распределения объектов по кластерам.

declare @step2_proc_day_id int -- переменная для конкретного дня недели.

declare @step2_proc_hour_start smallint; -- переменная начала интервала времени.

declare @step2_clust_num smallint; -- переменная предпочитаемого количества кластеров

declare @dist_type tinyint; -- идентификатор меры близости.

--1 - Евклидово расстояние; 2 - квадрат евклидового расстояния; 3 - Менхэтенское расстояние;

SET @step2_proc_day_id = 4494;

SET @step2_proc_hour_start = 0;

SET @step2_clust_num = 4;

SET @dist_type = 1;

-- Запуск второй процедуры расчета меры близости и первоначального распределения объектов по кластерам.

IF OBJECT_ID('tempdb..#GlobalObjectClusterTableMAIN') IS NOT NULL DROP TABLE #GlobalObjectClusterTableMAIN

CREATE TABLE #GlobalObjectClusterTableMAIN

([object_id] int NOT NULL, [word] varchar(50) NOT NULL, [power] float NOT NULL, [cluster] int NOT NULL, PRIMARY KEY ([object_id], [word]))

INSERT INTO #GlobalObjectClusterTableMAIN ([object_id], [word], [power], [cluster])

EXEC az_clustering_step2 @day_id = @step2_proc_day_id, @hour_start = @step2_proc_hour_start, @clust_num = @step2_clust_num, @dist_type = @dist_type
```

```
GO
-- Список переменных и их инициализация для третьей процедуры кластеризации.
declare @dist_type tinyint; -- идентификатор меры близости.
--1 - Евклидово расстояние; 2 - квадрат евклидового расстояния; 3 - Менхэтенское расстояние;
SET @dist_type = 1
-- Запуск третьей процедуры кластеризации
IF OBJECT_ID('tempdb..#GlobalObjectClusterTableMAINFINAL') IS NOT NULL DROP TABLE #GlobalObjectClusterTableMAINFINAL
CREATE TABLE #GlobalObjectClusterTableMAINFINAL
([object_id] int NOT NULL, [word] varchar(50) NOT NULL, [power] float NOT NULL, [cluster] int NOT NULL, PRIMARY KEY ([object_id], [word]))
INSERT INTO #GlobalObjectClusterTableMAINFINAL ([object_id], [word], [power], [cluster])
exec az_clustering_step3 @dist_type = @dist_type
GO
-- Список переменных и их инициализация для четвёртой процедуры классификации новых объектов.
declare @step4_day_id int; -- переменная для конкретного дня недели.
declare @step4_hour_start smallint; -- переменная для интервала времени.
declare @dist_type tinyint; -- идентификатор меры близости.
--1 - Евклидово расстояние; 2 - квадрат евклидового расстояния; 3 - Менхэтенское расстояние;
set @step4_day_id = 4494;
set @step4_hour_start = 0;
set @dist_type = 1;
-- Запуск четвертой процедуры классификации новых объектов.
EXEC az_clustering_step4 @day_id = @step4_day_id, @hour_start = @step4_hour_start, @dist_type = @dist_type
GO
```

Исходный код всех процедур *az_clustering_step*1, *az_clustering_step2*, *az_clustering_step*3 *и az_clustering_step*4 приведен в приложении 12 . После выполнения процедуры *az_clustering_step*4 выводится главный качественный показатель – процент попадания в целевую группу.

Процедура подготовки данных по интервалам времени.

Основная задача этой процедуры – преобразование данных о поисковой активности ИП и содержания ИР в удобную для обработки форму. В этой процедуре сначала выполняется разбиение данных в зависимости от указанного интервала времени с помощью переменой *@step*1*_hour_step*, затем формируется глобальный словарь терминов с интервальным разбиением для динамического применения. В конце процедуры для всех объектов исследования формируются характеристические вектора в зависимости от состояния соответствующего словаря в конкретном интервале времени. Алгоритм работы процедуры представлен на рисунке 5.22.

Из рисунка 5.22 видно, что алгоритм подготовки данных для кластерного анализа структурно прост. Он включает 16 шагов и, при этом, в нем отсутствуют какие либо уловные переходы и циклы. Выполнение второго шага позволяет выделить самую активную группы ИП – с 8-ми часовой активностью в киберпространстве за сутки и в течение 7-ми дней в неделю. Этот шаг позволяет устранить из кластерного анализа случайные объекты, активность которых может только испортить глобальную статистику терминов и качество сформированного обобщённого глобального словаря. С одной стороны шаги 3–7 нужны для формирования динамических характеристических векторов, а с другой – для применения метода трёхтактной кластеризации ИР с обратной связью (п. 3.3). Шаг 8-ой обеспечивает воплощение идеи обобщённого объекта исследования, обобщённого словаря терминов и обобщённого характеристического вектора. На 9-ом шаге реализована методика применения числовых коэффициентов усиления на основание *DOM*-модели. Отбор терминов нужной длины (шаг 11) позволит отфильтровать «мусорные» термины и подготавливает их для лемматизации. Выполнение шага 11 влечёт за собой формирование обобщённого словаря лемм (шаг 12), на основе которого можно будет применять метод позиционного кодирования при переходе от словарного вектора к числовому вектору (шаг 13). Результатами выполнения шагов 14 и 15 является формирование числовых характеристических векторов для ИП и ИР и их сохранение в таблицах БД для

дальнейшей обработки. Полный *SQL*-скрипт процедуры представлен в приложении 12.

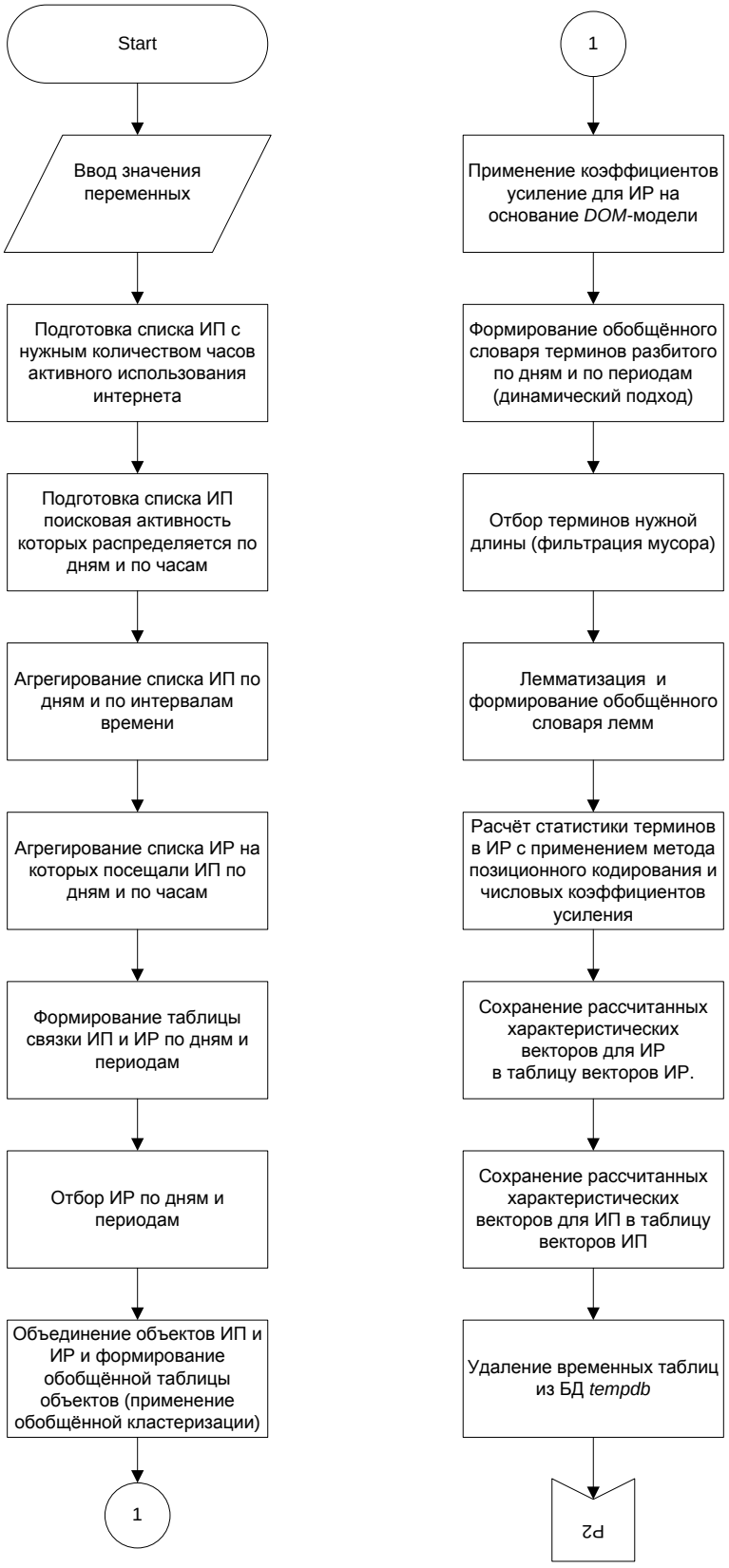

Рисунок 5.22 – Схема алгоритма подготовки данных для кластерного анализа

Процедура инициализации объектов и их первоначального распределения по кластерам.

Собственно с этой процедуры начинается кластерный анализ. Применяются специальные подходы, удобные для реализации в среде *MS SQL Server* 2012 и обеспечивающие экономию памяти на жёстком диске. Например, применяется отрицательное кодирование для отличия Интернет-объектов – идентификаторы ИП становятся отрицательными, а у ИР они остаются положительными.

Рассматриваемую процедуру можно в будущем доработать для предварительной инициализации объектов для кластеризации методом Форель. Для этого нужно будет внедрить переменную радиуса кластера.

Схема алгоритма работы процедуры представлен на рисунке 5.23.

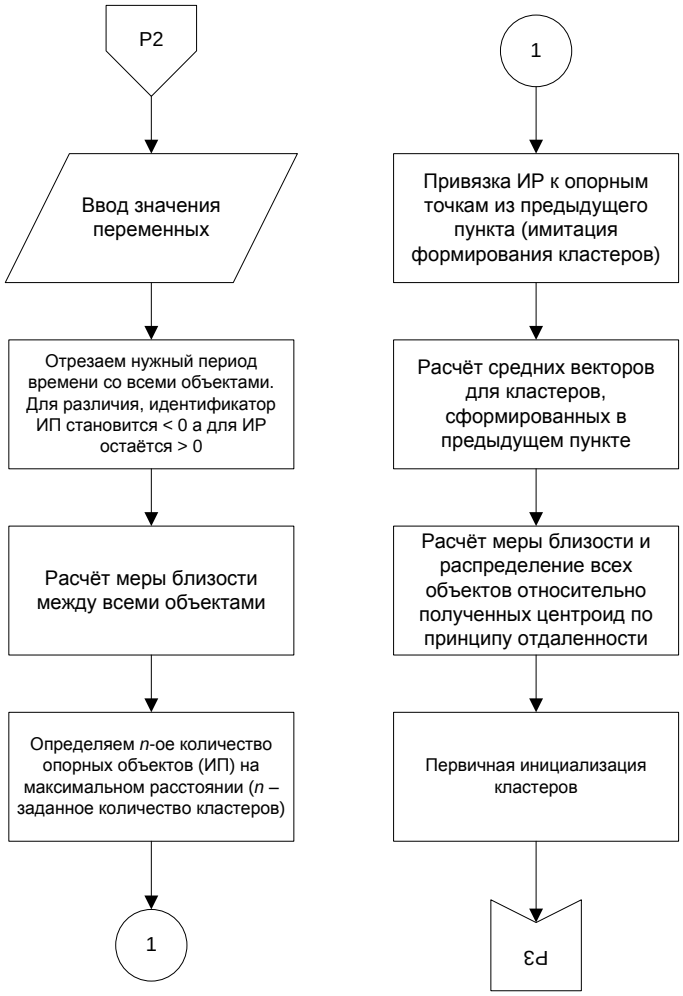

Рисунок 5.23 – Схема алгоритма инициализации объектов и их первоначального распределения по кластерам

Алгоритм процедуры состоит из 8-ми шагов. С целью экономии дискового

пространства и исключения необходимости добавления новых столбцов была осуществлена кодировка идентификаторов ИП отрицательными значениями (шаг 2). На шаге 3 эксперт-аналитик выбирает метрику: евклидово расстояние, его квадрат, мэнхетенское расстояние и др. Для расчёта меры близости объектов исследования используются выбранная метрика и характеристические вектора объектов, которые были ранее загружены в таблицу векторов БД. С применением таблиц БД, задача расчёта среднего вектора сводится к расчёту среднего числового значения для каждой из координат объектов, при помощи функции *AVG()* и группировки по кластерам. На шаге 4 по указанному входному значению требуемого числа кластеров *n* происходит подбор ИП, находящихся на максимальном расстоянии друг от друга (выбор опорных точек). Как только опорные точки определены можно приступить к формированию первых *n*-кластеров с помощью привязки ИР соответствующим опорным точкам (шаг 5). Когда кластеры сформированы, для каждого кластера по стандартной схеме рассчитываются средний вектор (шаг 6), а затем расстояния от всех объектов кластеров до средних (шаг 7). Результатом выполнения этой процедуры является первичная инициализация объектов – первичное распределение их по кластерам. Стоит отметить важность первичной инициализации объектов, ее влияние на дальнейший процесс кластерного анализа, качество которого напрямую от неё зависит. Полный *SQL*-скрипт процедуры представлен в приложении 12.

Процедура кластеризации.

Третья процедура предназначена исключительно для выполнения кластеризации – от момента получения первичного распределения объектов по кластерам до достижения стабильной кластерной структуры с неподвижными центрами. Стабильная кластерная структура достигается с помощью итерационного цикла, который завершается, как только объекты перестают перемещаться между кластерами (рисунок 5.24).

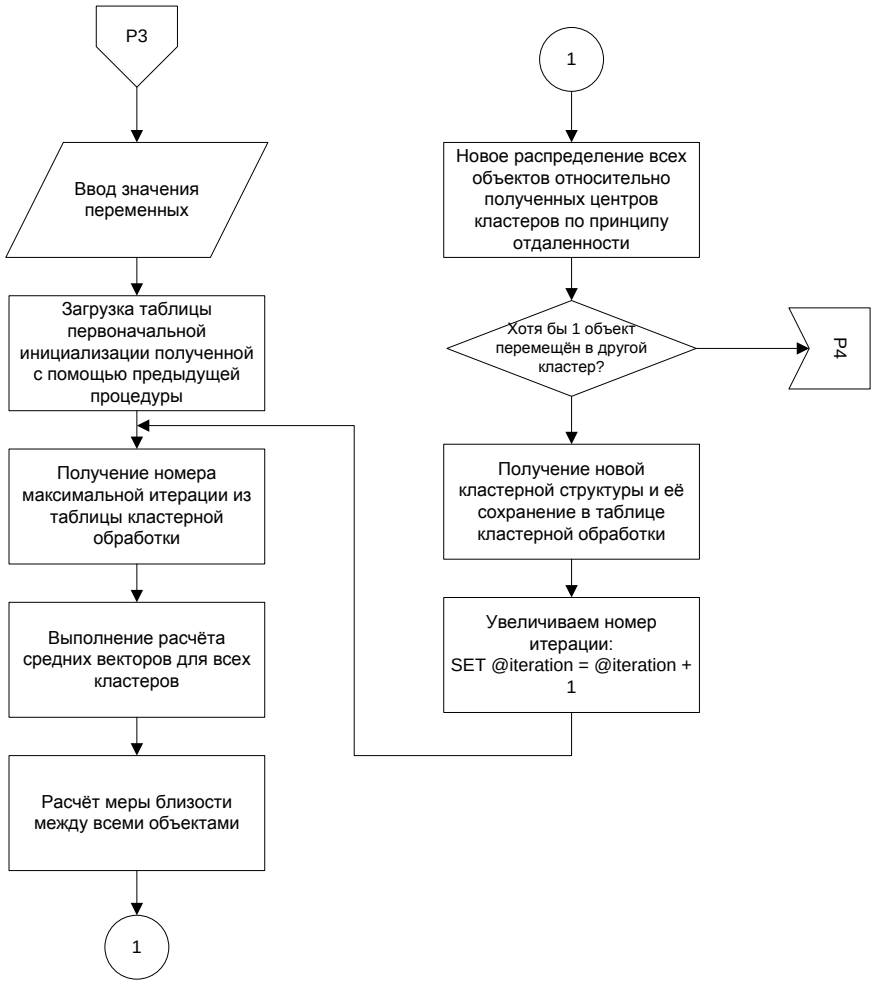

Рисунок 5.24 – Схема алгоритма кластеризации Интернет-объектов

На рисунке 5.24 стоит обратить внимание на итерационный цикл кластеризации. При каждой итерации обрабатывается последнее состояние кластерной структуры по значению столбца *iteration* (шаг 3). Затем, по стандартной схеме проводится расчёт среднего вектора (шаг 4) и меры близости между всеми объектами (шаг 5). Результат 4 шага может привести к перераспределению объектов в кластерной структуре. Как только объекты перестают мигрировать между кластерами, цикл завершается, и текущее состояние кластерной структуры выдается как результат. Полный *SQL*-скрипт процедуры представлен в приложении 12.

Процедура классификации новых объектов (персонализация ИР).

Четвёртая процедура предназначена для оценки прогнозирования или расчёта показателя попадания в целевую группу (рисунок 5.25). В результате

выполнения предыдущей процедуры, получаем стабильную кластерную структуру, в состав которой входят, как ИП, так и ИР. Оценить качество сформированных кластеров можно путём внедрения в неё новых объектов. Кластеризуем ИР, наблюдаемые в следующем временном окне, и затем проверяем посещали ли их ИП, находящиеся в одном и том же кластере.

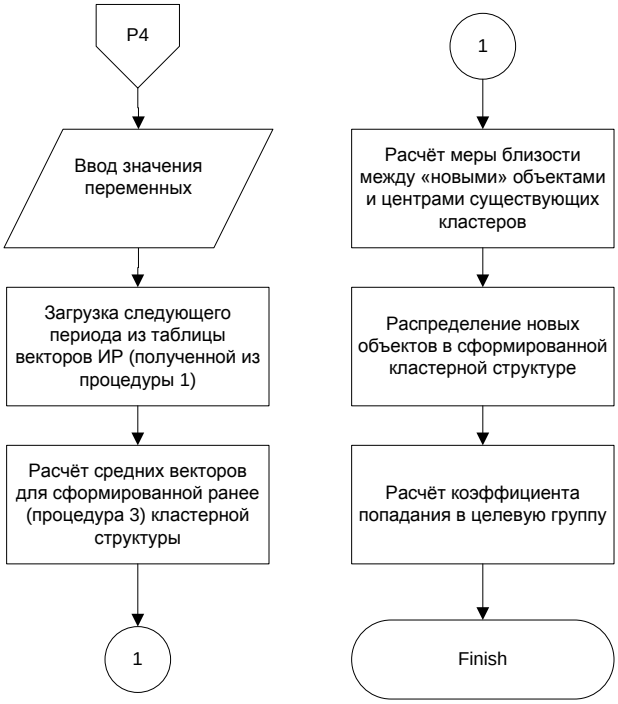

Рисунок 5.25 – Схема алгоритма классификации новых объектов

На рисунок 5.25 следует обратить внимание на то, что алгоритм классификации новых объектов, в большинстве своих шагов, повторяет шаги алгоритма инициализации объектов и их первоначального распределения по кластерам (рисунок 5.23). Это своего рода одна очередная итерация процесса кластерного анализа. По завершению выполнения этой процедуры получаем процент попадания в целевую группу, то есть отношение числа ИП, которые на самом деле посетили хотя бы один из новокластеризованых объектов к числу ИП, участвующих в кластеризации.

5.7. Экспериментальные исследования и оценка результатов

Основные вычислительные задачи, связанные с персонализацией поиска распределяются между сервером кластерного анализа и сервером БД. На сервере

кластерного анализа происходит формирование и обработка *L*og-файлов и сохраняется последнее состояние кластерной структуры, которое рассчитывается на сервере БД. Между сервером кластерного анализа и сервером БД существует постоянная синхронизация и обновление.

В виду ограниченности объема главы, нет возможности рассмотреть все эксперименты и расчеты возможных показателей оценки степени персонализации поиска, проведенные в ходе подготовки диссертации. Тем не менее, рассмотрим несколько ключевых показателей в рамках задачи обобщённой кластеризации с применением временного окна для ИП, числовых коэффициентов усиления из *DOM*-модели и трёхтактной кластеризации с обратной связи для ИР.

Эксперименты проводились с использованием разработанных программных модулей *internet_res_search*, *ie_analyzer* и *HTMLDocDom* на персональном компьютере с центральным процессором *AMD FX*-6100 *Six-Core Processor*, имеющим тактовую чистоту 3.30 ГГц, и оперативной памятью ёмкостью 16 ГБ. Все этапы кластеризации проводились раздельно. Формирование кластеров и расчёты проводился в среде *MS SQL Server* 2012.

Эксперимент по оценке попадания в целевую группу.

Эксперимент проводился с данными об активности ИП, полученными в период с 21 по 27 апреля 2014 года. В эксперименте участвовали 13454 ИП разного пола и возраста, проживающие в 419 городах России и выполнившие более 12000000 заходов на различные ИР. Из всего списка были выделены 335 наиболее активных ИП, выполнявших заходы на ИР в течение 8 и более часов в день, все 7 дней в неделю. Из всех выполненных заходов львиная доля (более 50%) принадлежит заходам на сайты социальных сетей. В связи с тем, что в диссертации решается задача персонализации поиска, круг рассматриваемых ИР был сужен, и в эксперименте обрабатывались поисковые запросы ИП к сайтам *yandex.ru*, *mail.ru* и *rambler.ru*, а также заходы на новостные сайты *news.mail.ru*, *news.rambler.ru* и *news.yandex.ru*. С одной стороны, новостные сайты могут отражать определенные особенности и интересы ИП, выполняющих заходы на их страницы, с другой стороны, они содержат большое число динамических

элементов, меняющих текстовое содержание ИР, что позволяет применять числовые коэффициенты усиления и трёхтактную кластеризацию с обратной связью.

Объекты исследования – ИП и ИР – объединялись, и затем формировался обобщённый глобальный словарь урезанных терминов и обобщённый характеристический вектор. Эксперимент проводился круглосуточно (с 0 до 23 часов) с интервалом времени (окном наблюдения) 4 часа. Следует отметить, что при необходимости величину окна наблюдения может менять (изменяя значения переменной *@step1_hour_step*) с целью повышения показателей персонализации. В качестве показателей персонализации можно использовать коэффициент попадания в целевую группу, точность попадания, полноту выборки, выпадение и др. В глобальном словаре терминов накапливались термины, длина которых равна или превышает 4 символа. Чтение содержания ИР и фильтрация динамических объектов выполнялась с помощью разработанной программы *HTMLDocDom*, а уже на сервере БД применялись числовые коэффициенты усиления и обобщённая кластеризация объектов.

Подготовка обобщённых словарей терминов и характеристических векторов.

Для выполнения эксперимента необходимо разбить весь массив полученных за неделю данных о ИП и ИР на подмассивы, отнесенные к 4 часовым окнам наблюдения, начиная с нуля часов. Для каждого подмассива необходимо сформировать:

- динамическую таблицу, содержащую глобальные словари терминов и лемм;

- динамическую таблицу, содержащую характеристические вектора ИП;

- динамическую таблицу, содержащую характеристические вектора ИР;

Построенные таблицы используются для выполнения кластерного анализа и расчёта процента попадания в целевую группу.

В процессе формирования указанных таблиц можно наблюдать за графиками (рисунок 5.26) количества объектов для каждого периода наблюдения.

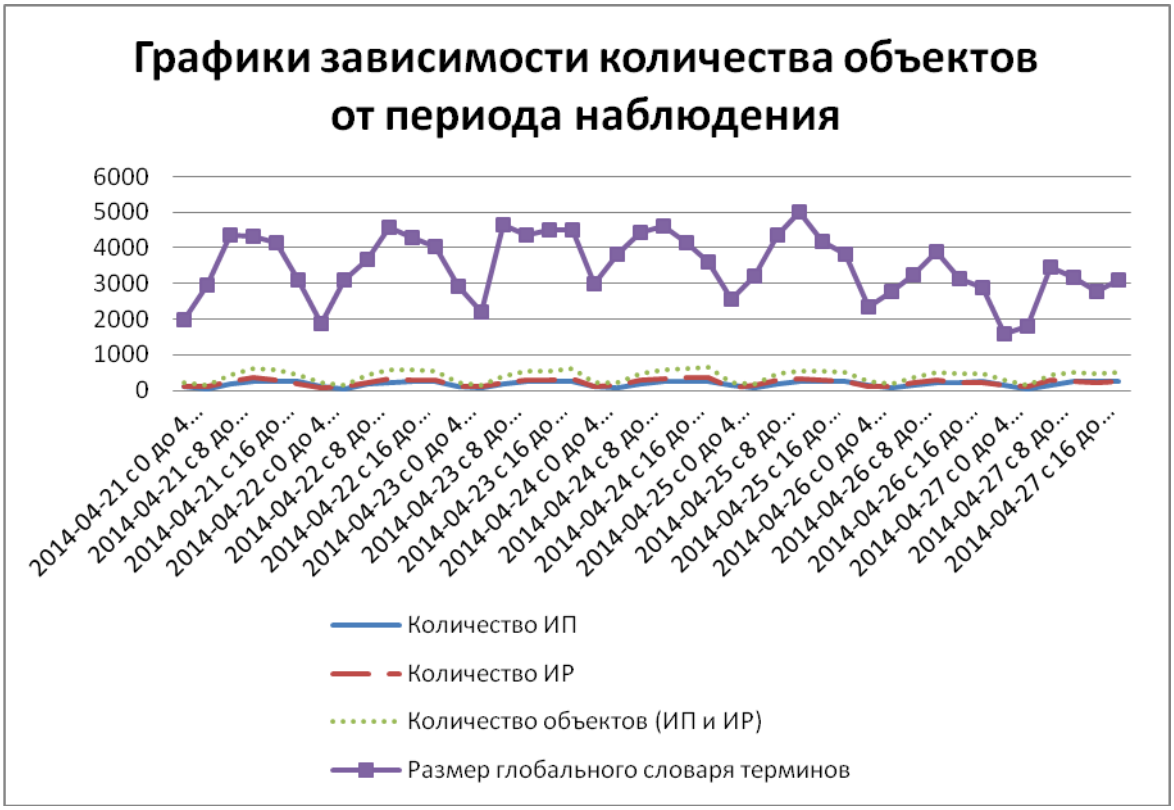

Рисунок 5.26 – Графики зависимости количества объектов от периода наблюдения

На рисунке 5.26 стоит обратить внимание на два важных момента.

1) Наблюдается четкая периодичность всех представленных графиков – графики достигают своих локальных максимумов в промежутке времени с 12 до 16 часов. Локальные минимумы появляются также периодически в промежутке между 0 и 4 часами. Это объясняется суточным циклом поведения ИП – время активной жизни сменяется временем отдыха и сна.

2) График размера глобального словаря терминов показывает на серьёзные объемы вычислений, необходимый для расчетов средних векторов и евклидовых расстояний для всех пара объектов. Для пика вычислений (с 12 до 16 часов 25 апреля 2014) объем вычислений характеризуется величиной $\frac{558!}{2\times(558-2)!}\times 5005 = 777792015$ расчётов.

Как только все таблицы заполнены, можно приступать к кластеризации объектов. Для оценки качества целесообразно применять показатель *Ta* –процент попадания в целевую группу:

$$Ta = \frac{CNT\_U_{Targeted}(t_{k+1})}{CNT\_U_{All}(t_k)} \times 100 \ , \qquad (5.1)$$

где $CNT\_U_{Targeted}(t_{k+1})$ – число ИП, для которых определен хотя бы один подходящий ИР в момент времени t_{k+1}; $CNT\_U_{All}(t_k)$ – число ИП, которые участвовали в кластер-анализе в момент времени t_k.

Для расчета процента попадания в целевую группу, в числителе используем число уникальных пользователей, для которых после кластеризации удалось угадать хотя бы посещение одного ИР в следующем периоде Δt. А в знаменателе общее число наблюдаемых пользователей.

Эксперимент 1: метод *k*-средних, число кластеров 2 при Δt = 4 час.

Для первого эксперимента основными входными параметрами являются число кластеров k = 2, то есть объекты будут распределены по двум кластерам, и Δt = 4 час. (рисунок 5.27).

Рисунок 5.27 – Графики зависимости процентов попадания в целевую группу и кластеризации от периода наблюдения при k = 2 и Δt = 4 час.

На рисунке 5.27 графики имеют похожую форму. Заметно, что при повышении процента кластеризации увеличивается и процент попадания в целевую группу. Не смотря на наличие пиковых значений (порядка 60% в периоды с 4 до 8 часов 23 апреля 2014 г., с 12 до 16 часов 23 апреля 2014 г., с 8 до 12 часов 24 апреля 2014 г., с 4 до 8 часов 25 апреля 2014 г. и с 4 до 8 часов 26 апреля 2014 г.), имеются и очень низкие показатели (порядка 20% в периоды с 20 до 24 часов 24 апреля 2014 г., с 8 до 12 часов 25 апреля 2014 г., с 8 до 12 часов 26 апреля 2014 г. и с 12 до 16 часов 26 апреля 2014 г.). Среднее значение процента попадания в целевую группу равно 38,449%. Дисперсия равна 114,862.

Эксперимент 2: метод *k*-средних, число кластеров 3 при Δt = 4 час.

Входными параметрами являются $k = 3$ и $\Delta t = 4$ час. В этом эксперименте объекты будут распределены по трём кластерам (рисунок 5.28).

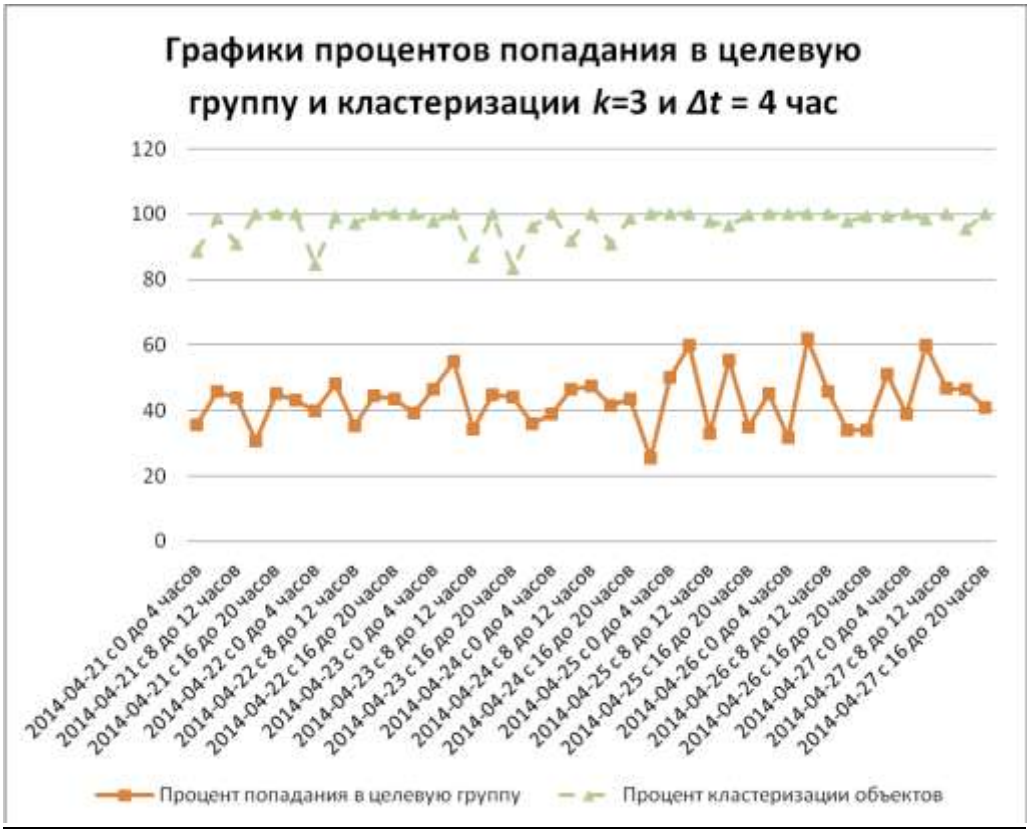

Рисунок 5.28 – Графики зависимости процентов попадания в целевую группу и кластеризации от периода наблюдения при $k = 3$ и $\Delta t = 4$ час.

На рисунке 5.28 сложно заметить подобие графиков процента попадания в целевую группу и процента кластеризации объектов. С одной стороны имеются

пять точек, где график процента попадания достигает своего максимума (≈ 60% в периоды с 4 до 8 часов 23 апреля 2014 г., с 4 до 8 часов 25 апреля 2014 г., с 12 до 16 часов 25 апреля 2014 г., с 4 до 8 часов 26 апреля 2014 г. и с 4 до 8 часов 27 апреля 2014 г.), с другой стороны, присутствует всего одно значение с низким показателем (≈ 25% в период с 20 до 24 часов 24 апреля 2014 г.). Среднее значение процента попадания в целевую группу равно 43,111%. Дисперсия равна 67,831.

Эксперимент 3: метод *k*-средних, число кластеров 4 при Δt = 4 час.

Входными параметрами являются k = 4 и Δt = 4 час. В этом эксперименте объекты будут распределены по трём кластерам (рисунок 5.29).

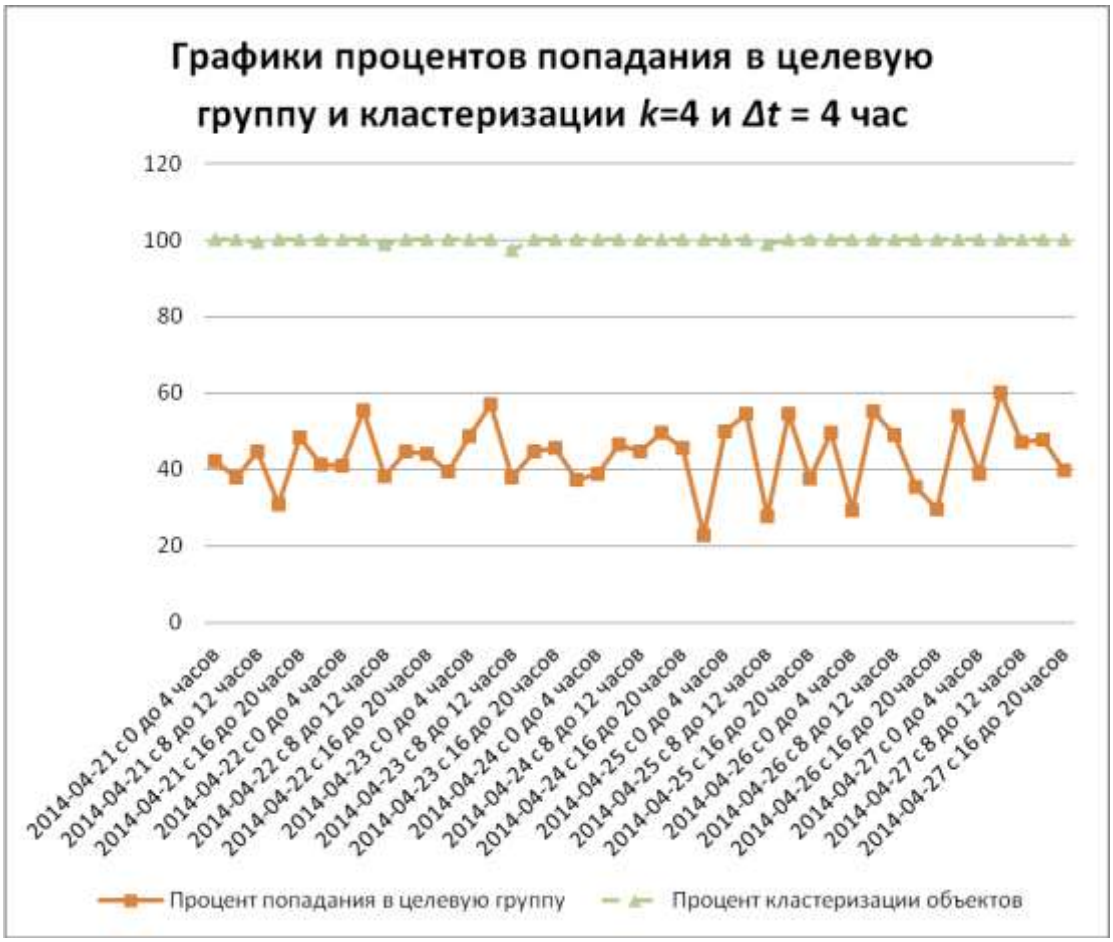

Рисунок 5.29 – Графики зависимости процентов попадания в целевую группу и кластеризации от периода наблюдения при k = 4 и Δt = 4 час.

На рисунке 5.29 процент кластеризации практически равен 100% для всех периодов наблюдения. На графике процента попадания в целевую группу заметно

увеличилось число локальных максимумов, приближающиеся к 60% (периоды с 4 до 8 часов 22 апреля 2014 г., с 4 до 8 часов 23 апреля 2014 г., с 4 до 8 часов 25 апреля 2014 г., с 12 до 16 часов 25 апреля 2014 г., с 4 до 8 часов 26 апреля 2014 г. и с 20 до 24 часов 26 апреля 2014 г.). Среднее значение процента попадания в целевую группу равно 43,617%. Дисперсия равна 72,015.

Эксперимент 4: метод *k*-средних, число кластеров 5 при Δt = 4 час.

Входными параметрами являются k = 5 и Δt = 4 час. В этом эксперименте объекты будут распределены по трём кластерам (рисунок 5.30).

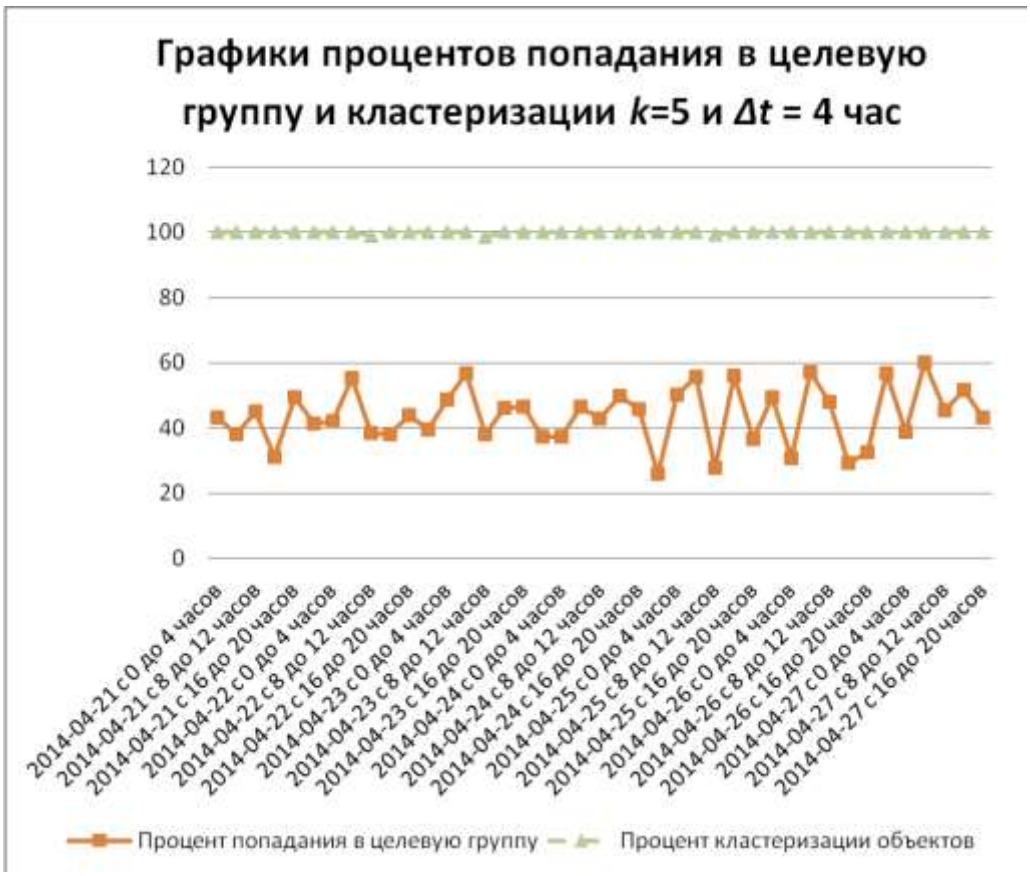

Рисунок 5.30 – Графики зависимости процентов попадания в целевую группу и кластеризации от периода наблюдения при k = 5 и Δt = 4 час.

На рисунке 5.30 график процента кластеризации является константой – в большинстве периодов все объекты исследования оказались распределенными по кластерам. На графике наблюдаем пиковые значения (≈ 60%), и минимальные значения (≈ 26%), всего для одного периода наблюдений. Среднее значение процента попадания в целевую группу равно 43,782%.

Дисперсия равна 75,822.

После выполнения этой серии экспериментов, построения графиков (рисунки 5.27 – 5.30), расчёта средних значений коэффициента попадания в целевую группу и его дисперсии, можно сделать вывод о том, что оптимальным решением для применения кластерного анализа являются условия $k = 3$ или $k = 4$, так как при этих условиях получаем хорошее значение коэффициента попадания в целевую группу с минимальной дисперсией.

Результаты экспериментов сильно привязаны к значению интервала времени $\Delta t = 4$ часам. При $\Delta t < 4$ поведение ИП становится более локализированным. Тем ни менее, уменьшается размер, как самих кластеров, так и характеристических векторов и обобщённого глобального словаря терминов, вследствие чего получаемые значения коэффициентов должны возрасти. Нужно отметить ещё одну особенность, касающуюся периодов с низкими значениями коэффициентов попадания в целевую группу – это динамические процессы внутри кластерных структур: диффузия, поглощение кластеров или перемещение объектов из одного кластера в другой (п. 3.1).

Эксперимент 5: метод *k*-средних, число кластеров 4 при $\Delta t = 1$ час.

Для пятого эксперимента входными параметрами являются число кластеров $k = 4$ и размер окна наблюдения $\Delta t = 1$ час. В этом эксперименте объекты будут распределены по четырём кластерам.

Экспериментальные исследования проводились в течение одних суток (21 апреля 2014 г.), разделённых на 24 периода по одному часу каждый.

На рисунке 5.31 график процента кластеризации колеблется от 78% до 100%, то есть не все объекты исследования оказываются распределёнными по кластерам. Однако график процента попадания в целевую группу в значительной степени смещается вверх по сравнению с графиками рисунков 5.27 –5.30. На графике рисунка 5.31 наблюдаем пиковые значения ≈ 83%, и минимальное значение равное 40%. Среднее значение процента попадания в целевую группу равно 63,799%. Дисперсия равна 94,076.

Рисунок 5.31 – Графики зависимости процентов попадания в целевую группу и кластеризации от периода наблюдения при $k = 4$ и $\Delta t = 1$ час.

Последний эксперимент отражает влияние входных параметров (числа кластеров k и периода времени Δt) на эффективность кластеризации как ИП так и ИР. Как показали эксперименты коэффициент попадания в целевую группу увеличился с 43,617% до 63,799% как только был уменьшен период времени Δt. Но средний процент кластеризации объектов снизился до 95%. Сказанное демонстрирует существенное влияние значений входных параметров на качество кластеризации ИП и ИР. Это также указывает на важную роль, которую может играть опытный эксперт-аналитик, регулирующий значения входных параметров в процессе кластеризации.

Эксперимент по оценке качества персонализации поиска.

Эксперимент проводился с целью сравнения результатов работы поисковой системы Яндекса при поиске многозначных терминов (омонимов) с результатами классификации, выполненными после кластерного анализа КСПП, на одних и тех же ссылках. Примером омонима является термин «ягуар», который имеет множество значений – это в первую очередь хищное млекопитающее, это и марка автомобилей, это и художественный фильм. У термина «ягуар» могут быть и другие значения – энергетический напиток и название песни. Сейчас, к сожалению, поисковая система Яндекса не способна определить то, в каком смысле термином «ягуар» интересуется тот или иной ИП.

Любой ИП, начиная поисковую деятельность, вводит определённый набор терминов в поле поисковой строки, а затем переходит на интересующиеся его ресурсы, тем самым, выполняя фильтрацию не нужных результатов.

26 мая 2014 года на поисковый термин «ягуар», Яндекс выдал более 3000000 гиперссылок. Первые 50 гиперссылок, полученные при поиске термина «ягуар» в Яндексе (в таблице 5.4), использованы при реализации предлагаемой методики кластерного анализа, которая состоит из нескольких последовательных этапов. На первом этапе выполнялся автоматизированный заход с помощью программы *HTMLDocDom* по всем 50 гиперссылкам и выполнялся сбор их текстового содержания. По результатам многократного анализа *DOM*-модели страниц, соответствующих указанным гиперссылкам, применялись числовые коэффициенты усиления и трёхкратная кластеризация ИР с обратной связью.

Как только определилось множество ИР, формировался глобальный словарь терминов и затем глобальный словарь лемм, на основе которого строились характеристические вектора ИР.

Таблица 5.4 – Таблица результатов первых 50 гиперссылок при поиске термина «ягуар» в Яндексе

позиция	url	позиция	url
1	http://www.jaguar.ru/index.html	26	http://www.jaguaryasenevo.ru/
2	http://www.jaguarcenter.ru/	27	http://www.zr.ru/auto/jaguar/
3	http://ru.wikipedia.org/wiki/%DF%E3%F3%E0%F0	28	http://www.antoncars.ru/katalog-avto/Jaguar/
4	http://www.jaguar.com/market-selector.html	29	http://moscow.auto.ru/cars/jaguar/
5	http://jaguar.musa-motors.ru/	30	http://www.carexpert.ru/models/jaguar/XJ/
6	http://cars.mail.ru/catalog/jaguar/	31	http://jaguar.autopassage.ru/
7	http://carsguru.net/catalog/jaguar/	32	http://traditio-ru.org/wiki/%D0%AF%D0%B3%D1%83%D0%B0%D1%80
8	http://jaguar-drink.com/	33	http://www.fast-torrent.ru/film/yaguar.html
9	http://jaguar.drom.ru/	34	http://www.zoodrug.ru/topic1400.html
10	http://jaguar.arteks.ru/	35	http://www.drive.ru/jaguar
11	http://www.zooclub.ru/wild/hish/70.shtml	36	http://www.bigcatfiles.info/jaguar/
12	http://www.livecars.ru/news/2014/05/25/jaguar_xe/	37	http://jaguar.avilon.ru/
13	http://ru.wikipedia.org/wiki/Jaguar	38	http://market.autoua.net/catalog/cars/jaguar/
14	http://quto.ru/Jaguar/	39	http://jaguar.110km.ru/xj.html
15	http://baskino.com/films/boeviki/3489-yaguar.html	40	http://www.jaguar77.ru/
16	https://lurkmore.to/%DF%E3%F3%E0%F0	41	http://mp3-life.com/song/37072/
17	http://www.major-jaguar.ru/	42	http://dic.academic.ru/dic.nsf/enc_colier/2439/%D0%AF%D0%93%D0%A3%D0%...
18	http://bigcats.ru/index.php?bcif=jaguars.shtml	43	http://www.jaguar-co.ru/
19	http://jaguar.110km.ru/	44	http://www.russlav.ru/alkogolizm/vred_napitka_yaguar.html
20	http://jagaman.ru/	45	http://animalsarea.ru/zhivotnie/koshki/yaguar/37-yaguar.html
21	http://jaguar.110km.ru/	46	http://www.ellittattoo.ru/wiki/208.phtml
22	http://jagaman.ru/	47	http://vk.com/wiki_jaguar
23	http://jaguar-fc.ru/forum/index.php	48	http://www.avtovzglyad.ru/jaguar/
24	http://www.jaguar.alea.ru/	49	http://www.autonet.ru/Auto/Ttx/Jaguar
25	http://www.kinopoisk.ru/film/14292/	50	http://autorambler.ru/catalogue/Jaguar/

Для сравнения результатов работы поисковой системы Яндекса с результатами применения предлагаемой методики кластерного анализа была смоделирована и реализована ситуация, когда четыре ИП, заинтересованные разными сущностями, обозначаемыми термином «ягуар», осуществляли Интернет-поиск. Эти ИП выполняли заходы на *n*-ое число ИР, отвечающих их интересам. Исходя из поисковой активности ИП и с учетом посещаемых ими ИР, формировались характеристические вектора ИП и затем были получены первых средние 4-х кластеров.

С помощью процедур кластерного анализа (приложение 12) была выполнена кластеризация этих объектов, конечный результат которой представлен в таблице 5.5.

Таблица 5.5. Конечный результат кластерного анализа

Кластер	**Число объектов**	**Список объектов кластера**
1	28	{2,39,26,21,6,23,35,49,31,10,27,40,13,24,29,38,14,19,12,50,17,7,37, 28,48,5,1,9}
2	3	{15,33,25}
3	4	{34,11,42,36}
4	15	{45,3,46,41,4,16,22,47,43,20,44,30,8,32,18}

Кластер 1 – кластер с объектами, которые оказались ближе к значению «ягуар» – автомобиль, кластер 2 – кластер с объектами, которые оказались ближе к значению «ягуар» – фильм; кластер 3 – кластер с объектами, которые оказались ближе к значению «ягуар» – животное; кластер 4 – кластер с оставшимися объектами из полученной выборки.

Получив конечный результат распределения объектов по кластерам, можно рассчитать важные коэффициенты, отражающие качество полученного результата. Для оценки качества могут применяться следующие показатели: точность попадания, полнота и выпадение.

Следует обратить особое внимание на тот факт, что полнота выборки и выпадение не могут быть представлены в качестве опорных показателей в

условиях обсуждаемого эксперимента, так как в нем выборка проводилась на весьма ограниченном числе ИР. Дело в том, что при большой выборке (тысячи и миллионы ИР) как только увеличивается точность результата поиска, полнота выборки снижается. Для выборки малого объема результат оценки будет искаженным. Тем ни менее, для полноты картины будут представлены все три указанные выше показателя.

Точность попадания (*precision*) – доля релевантной информации во всем объёме информации, полученной в результате поиска, – вычисляется по формуле:

$$precision = \frac{TR}{TR + FR}, \qquad (5.2)$$

где TR – множество релевантных документов, полученных при поиске; FR – множество нерелевантных документов, полученных при поиске.

Приведём расчёты точности попадания и сравним их, для поисковой системы Яндекса (первые 50 гиперссылок), с одной стороны, и, для полученной выше кластерной структуры, с другой стороны (рисунок 5.32).

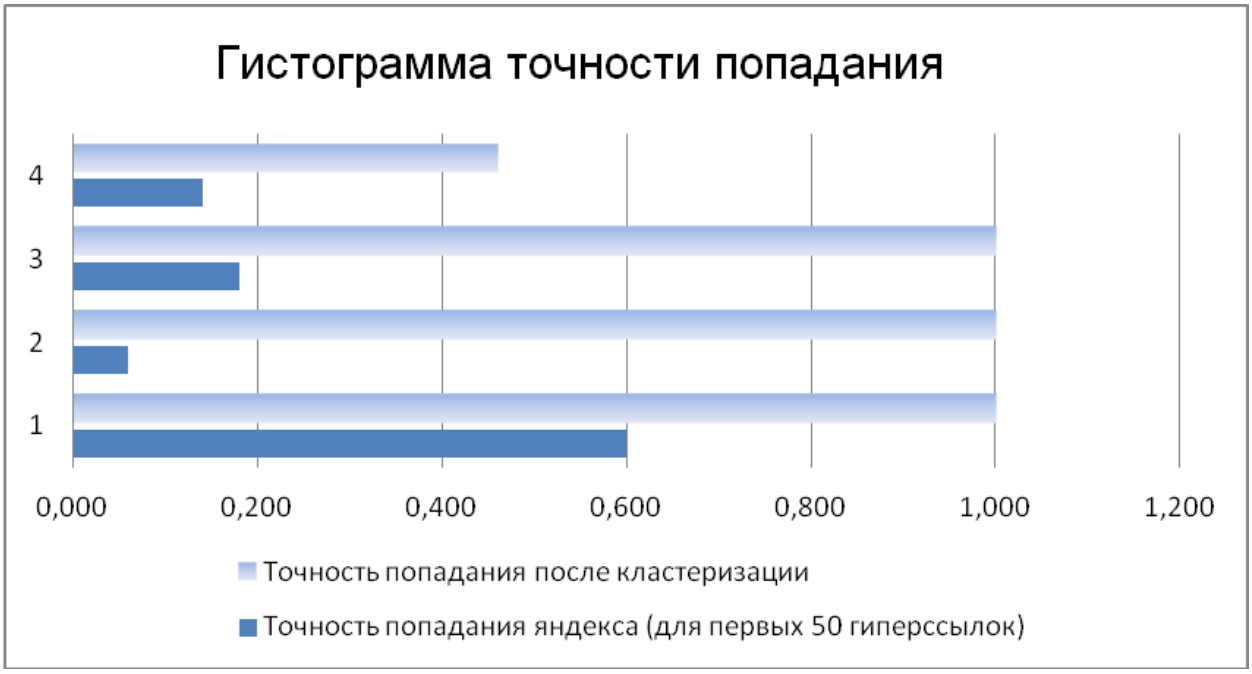

Рисунок 5.32 – Гистограмма точности попадания

На рисунке 5.32 видно преимущество применения предлагаемой методики персонализации поиска. Точности попадания для 3-х кластеров равен 1 при посещении до 4-х релевантных ИР.

Полнота выборки (*recall*) рассчитывается по следующей формуле:

$$recall = \frac{TR}{TR + FN}, \qquad (5.3)$$

TR – множество релевантных документов, полученных при поиске; *FN* – множество релевантных документов, которые не были получены при поиске, т.е. поисковая система ложно признала этих документов не релевантными.

Приведём расчёты полноты выборки и сравним их, для поисковой системы Яндекса (первые 50 гиперссылок), с одной стороны, и, для полученной выше кластерной структуры, с другой стороны (рисунок 5.33).

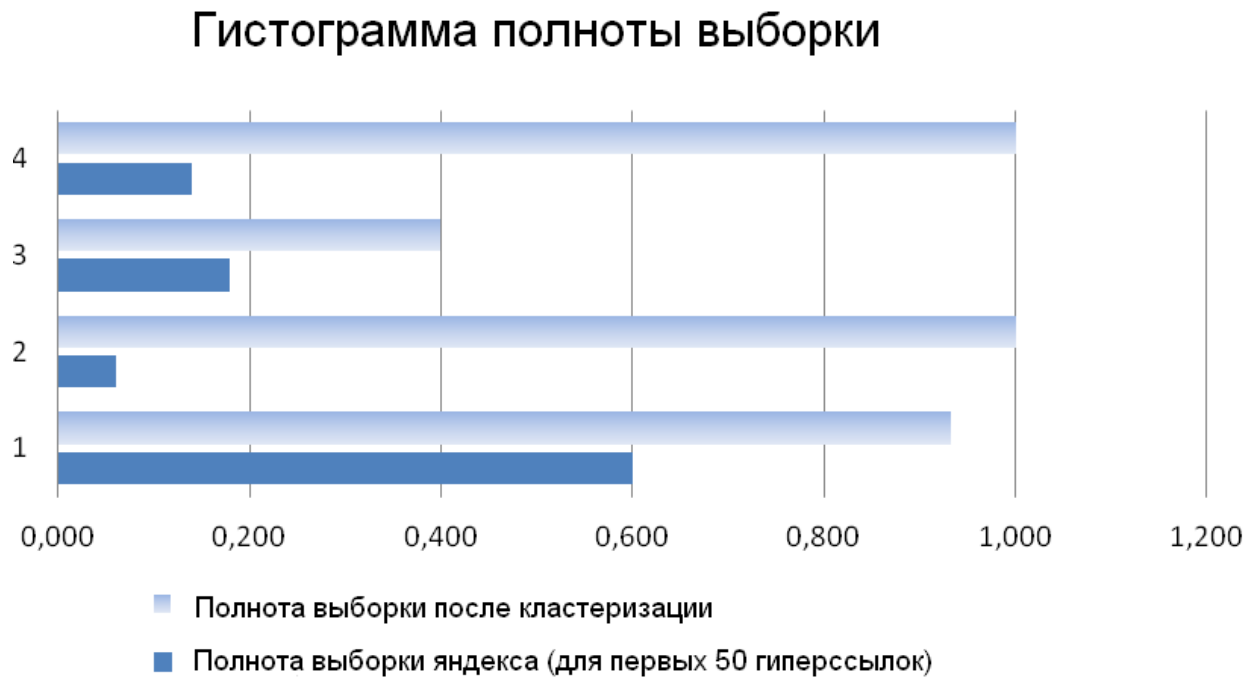

Рисунок 5.33. – Гистограмма полноты выборки

На рисунке 5.33 заметно ещё одно преимущество применения предлагаемой методики для персонализации поиска. Полнота выборки для 2-х кластеров равен 1.

Понятно, что чем выше точность и полнота, тем лучше. Но в реальной жизни максимальная точность и максимальная полнота не достижимы одновременно, и приходится искать некий баланс между ними. Поэтому, хотелось бы иметь некую метрику, которая объединяла бы в себе данные о точности и полноте кластеризации. Выпадение (*fall-out*) или *F*-мера [32] представляет собой

гармоническое среднее между точностью и полнотой. Она стремится к нулю, если точность или полнота стремится к нулю.

Выпадение рассчитывается по следующей формуле:

$$F = \frac{2 \times precision \times recall}{precision + recall}, \qquad (5.4)$$

где, F – выпадение; *precision* –точность попадания; *recall* – полнота выборки.

Выпадение F считается хорошим показателем для анализа производительности поисковой системы (рисунок 5.34).

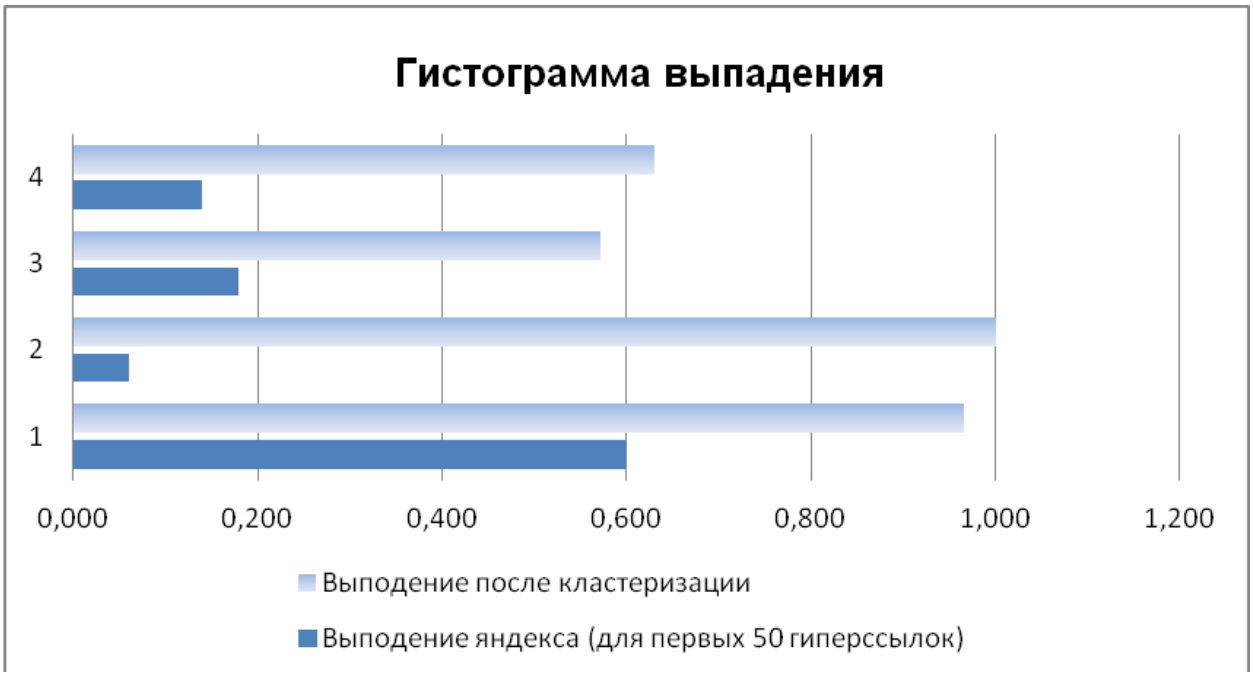

Рисунок 5.34. – Гистограмма выпадения

Полученные низкие показатели для поисковой системы Яндекса в первую очередь связаны с тем, что исследование проводилось исключительно на первых 50 гиперссылках (нет возможности кластеризовать все 3000000 ИР). Не смотря на это, в результате проведённых экспериментов, можно считать доказанным преимущество применения кластерного анализа для персонализации поиска. Дело в том, что 95% всех ИП пользуются гиперссылками, приведёнными на первых пяти страницах выдачи, полученной при поиске в Яндексе.

Если применять предлагаемую кластерную методику классификации ИР по поисковой истории и интересам самых ИП, то в значительной степени могут повыситься точность попадания и полнота выборки, также увеличится и

выпадение. Следует отметить, что значения указанных показателей будут увеличиваться всё больше и больше, стремясь к 1, при возрастании активности ИП.

При выполнении описанного выше эксперимента были обнаружены сайты плагиаты (*http://www.jaguarcenter.ru/*, *http://jaguar.musa-motors.ru/*, *http://www.major-jaguar.ru/*, *http://www.jaguar.alea.ru/*, *http://www.jaguaryasenevo.ru/*, *http://jaguar.autopassage.ru/* и др.) имеющие минимальное расстояние между собой. Тем самым обнаружено ещё одно преимущество применения кластерного анализа при классификации, а затем персонализации ИР.

5.8. Основные результаты и выводы по пятой главе

1. Обоснована структурная организация корпоративной системы персонализации поиска – КСПП, базирующаяся на сервере кластерного анализа и сервере БД.

2. Предложен способ структуризации данных о поисковой активности ИП в рамках реляционной модели данных. Добавление сущностей в модель проводится поэтапно, по мере рассмотрения задач хранения и обработки поисковых историй ИП, наблюдаемых в течение скользящего временного окна. Предложен алгоритмический преобразователь наблюдаемых неструктурированных данных о поисковой активности ИП в данные реляционной БД. Затронута кодировка поисковых строк и разработана специальная функция, извлекающая инкапсулированные и закодированные в поисковых строках термины с применением фильтрующих масок.

3. Предложен способ структуризации данных о современных динамичных ИР. Для этой цели предложено использовать *DOM*-модель ИР и структуризацию провести на основании общедоступного словаря *HTML*-тэгов.

4. Разработаны программные инструменты, применение которых в корпоративной сети предприятия позволит реализовать кластеризацию Интернет-объектов – ИП и ИР. Представлены результаты экспериментов по кластеризации

Интернет-объектов с помощью разработанного инструментария с применением виртуальной КСПП в среде *MS SQL Server* 2012. Макет реальной КСПП, к сожалению, реализовать не удалось в силу ряда организационно-технических причин.

5. Разработан человеко-ориентированный интерфейс для программных модулей *internet_res_search* и *ie_analyzer*, отвечающих за сбор информации о навигации ИП в сети Интернет и о посещающие ими ИР. Реализован интерфейс пользователя программного робота *HTMLDocDom*, отвечающего за сбор информации о текстовом содержании ИР. Эта программа используется для чтения текста из *DOM*-модели ИР и построения его дерева, что позволяет применить, с одной стороны, числовые коэффициенты усиления, а, с другой стороны, выделить динамические компоненты ИР, для их дальнейшей фильтрации.

6. Представлены структура подсистемы КСПП, непорсредственно реализующей кластеризацию и классификацию Интернет-объектов, специфицированы алгоритмы всех разработанных модулей подсистемы кластерного анализа.

7. Выполнено исследование и сравнительная оценка результатов работы КСПП. Сравнения проводились по точности попадания, полноте выборки и выпадению с поисковой системой Яндекса. Уровень персонализации поиска, достигаемый виртуальной КСПП по указанным показателям превосходит персонализацию Яндекса, по крайней мере, при рассмотрении первых 50-ти гиперссылок его выдачи.

ЗАКЛЮЧЕНИЕ

В диссертационной работе предложен комплексный подход к кластеризации ИП и ИР, используемой для их классификации в рамках мер по персонализации Интернет-поиска.

В работе приведён обзор существующих некластерных методов классификации объектов, исследована возможность их применения к Интернет-объектам – ИП и ИР. Некластерные методы могут быть примененые на первых этапах классификации объектов до начала кластерного анализа с целью предварительного распределения объектов. Был сделан вывод о том, что применение алгоритмов кластерного анализа является наиболее предпочтительным вариантом для решения поставленной задачи. Кластерные методы, учитывая их разнообразие, способны справиться с огромными объёмами данных и работать с векторами большой размерности.

В начале работы над диссертацией объекты классификации рассматривались в отдельности. Иерархические алгоритмы кластерного анализа, главным преимуществом которых является простота реализации, оказались эффективно применимыми для кластеризации ИП. Это связано с тем, что при иерархической кластеризации ИП формируются своего рода деревья с нисходящими или восходящими рёбрами и, тем самым, объекты объединяются по принципу близости. В виду огромного числа ИР, для них целесообразно применять итерационные алгоритмы, которые дают возможность работать с центрами кластеров и позволяют добавлять новые объекты к уже существующим кластерам.

На заключительном этапе диссертационной работы ИП и ИР стали рассматриваться как объекты с подобными характеристиками, что позволило ввести понятие обобщённого Интернет-объекта, отвечающего обобщённому глобальному словарю терминов и идентифицируемому обобщённым характеристическим вектором.

Наряду с адоптацией известных алгоритмов кластерного анализа были предложены новые подходы для снижения и устранения динамической составляющей самих Интернет-объектов. Так, для ИП было принято решение применять скользхящие временны*ы*е окна в их связи с ритмом дня, а для ИР предложено воспользоваться числовыми коэффициентами усиления, получаемыми на основе сканирования *DOM*-модели, и специальным *DOM*-фильтром для устранения динамических компонентов, постоянно меняющих текстовый контент ИР. Предложенные методы делают кластерные структуры более стабильными, что позволяет применять стандартные методы кластерного анализа.

В ходе диссертационной работы получены следующие основные результаты.

1. Предложена и реализована процедура лингвистическойя обработки текста, базирующаяся наиспользовании двухуровневого словарь терминов и лемм с возможностью применения открытых словарей. При необходимости предусмотрема возможность обращения к «лингвистическому эксперту» для лемматизации новых или не стандартных теминов.

2. Для достижения стабильности кластерной структуры и устранения динамического эффекта, разработан метод наблюдения за ИП, основанный на применения скольязщего временного окна. С этой же целью для наблюдения за ИР разработан метод анализа содержания (сканирования) *DOM*-модели ресурса с последующим применением числовых коэффициентов усиления.

3. С целью выявления и фильтрации динамических компонентов *DOM*-модели предложена трёхтактная схема кластеризации ИР с обратной связью. Реализация схемы позволяет превращать динамичные ИР в статические ИР и применять к последним стандартные алгоритмы кластерного анализа.

4. Разработана математическая модель представления и процедуры формирования характеристических векторов ИП и ИР, числовые координаты которых, расположены в порядке, соответствующем лексикографическому порядку следования термины в глобальном словаре. Переход от вербального к

числовому представлению координат происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений в текст поисковых запросов или текстовый контент статических компонентов *DOM*-модели ИР.

5. Предложено использовать понятие обобщённого Интернет-объекта, применение которого позволяет одновременно проводить кластерный анализ как ИП, так и ИР. Унификация объектов наблюдения делает кластерный анализ более транспарентнтным и глобальным. ИП, в зависимости от их интересов, могут храниться вместе с информационно-релевантными им ИР.

6. Разработанные методы имеет практическую ценность, поскольку в совокупности обеспечивают целостный подход к решению задачи персонализации поиска в сети Интернет на базе классификации ИР в соответствии с предпочтениями (интересами) ИП. Эти методы позволяют использовать ИР, посещаемые одними ИП, в качестве предпочтительных ИР для других ИП, находящихся в том же кластере. Можно сказать, что это своего рода социализация поиска.

Предлагаемый подход может быть использован на уровне корпоративной сети и охватывает практически все этапы решения целевой задачи:

- первичный сбор информации, в пределах заданного скользящего временного окна, о поисковой активности ИП и посещаемых ими ИР;

- многократное сканирование *DOM*-модели ИР, применение числовых коэффициентов усиления и *DOM*-фильтрации;

- структуризацию объектов с помощью специализированной БД;

- формирование глобальных словарей терминов и лемм;

- формирование характеристических векторов ИП и ИР, а также характеристических векторов обобщенных Интернет-объектов;

- расчёт меры близости между исследуемыми объектами;

- первоначальную инициализацию объектов;

- кластеризацию обобщённых объектов на основе алгоритма *k*-средних.

Использование результатов кластеризации для персонализации поиска – результаты анализа поисковой активности ИП в текущем интервале времени

могут быть применены для прогноза его информационных потребностей в следующих интервалах времени.

7. Разработан набор программных модулей для слежения за активностью ИП и получения текстового содержания ИР с учётом их *DOM*-моделей. В среде *MS SQL Server* 2012 разработаны специальные хранимые процедуры, выполняющие все необходимые расчёты – от формирования словарей терминов до конечного распределения объектов по кластерам. Указанные модули и хранимые процедуры образуют единую программную систему, которая, будучи установленной на серверы локальной сети, позволит, например, организовать на предприятии корпоротивную систему персонализации поиска.

8. Проведена оценка эффективности предложенных методов и средств, сделан вывод о возможности их внедрения в существующие поисковые системы. Это могло бы сделать посковые системы более «социальными», и со временем получить гибрид стандартной поисковой системы и социальной сети. С одной стороны, Интернет-пользователи с одинаковыми интересами имели бы более релевантный, персонолизированный результат поиска, а, с другой стороны, они могли бы делиться ссылками на релевантные ИР. Социализированные поисковые системы позволяли бы использовать ИР, посещаемые одними ИП, в качестве предпочтительных ИР для других ИП, находящихся в одних и тех же кластерах, вести целенаправленную поисковую деятельность, отвечающую устремлениям групп одинаково думающих пользователей.

Представленные в дисертации результаты позволяют сделать вывод о преспективности применения предложенных методов не только с целью персонализации поиска, но и для решения множества других интересных задач. Например, применение принципа обобщения объектов исследования может продвинуть наши представления о машинном обучении. На практике предложенные методы могут быть применены для обработки социальных анкет и обеспечить получение высокоточных оценок поведения людей.

В будущем стоит задуматься о продолжении проведенных исследований в направлении усовершенствования методов кластеризации Интернет-объектов за

счет снижения размерности характеристических векторов и внедрения системы динамической адаптации кластерного анализа.

СПИСОК СОКРАЩЕНИЙ И ТЕРМИНОВ

Assembly – управляемый модуль приложений, компонент сервера БД *MS SQL Server*.

Cookie – небольшой файл данных, отправляемый веб-сервером и хранимый на компьютере пользователя. Применяется для сохранения данных на стороне пользователя о его персональных предпочтениях и настройках.

***CSS* (*Cascading Style Sheets*)** – каскадные таблицы стилей, позволяющие гибко манипулировать стилем и внешним видом (фонтами, размером шрифтов, цветами и т.д.) элементов веб-страниц.

***DOM* (*Document Object Model*)** – объектная динамическая модель, используемая для *XML*/*HTML*-документов. По сути, это дерево компонентов веб-страницы.

***DLL* (*Dynamic Link Library*)** – библиотека динамической компоновки.

***HTML* (*HyperText Markup Language*)** – язык разметки гипертекста, разработанный британским учённым Тимом Бернерсон-Ли (*Timothy Berners-Lee*) в 1989-1991 г.г. *HTML* используется для создании веб-страниц во Всемирной паутине. Написанный на *HTML* код является платформо-независимым, т.е. *HTML*-файл может быть создан при помощи одного устройства, а затем запущен в браузере на другом устройстве (компьютере, телефоне и т.д.).

HttpUtility – класс пакета *.NET Framework* 4.5, содержащий функции кодирования и декодированбия *URL*-страниц.

***Log*-файл** – текстовый файл со специальной структурой (с фиксированным разделителем строк и столбцов), куда записывается информация в определённом порядке. Для этих файлов обычно настраивают *SQL*-процедуру *BULK INSERT* для загрузки в БД.

***nof* (*number of*)** – функция, возвращающая в качестве значения число элементов в конечном множестве-аргументе.

***RU*-нет (рунет)** – российский Интернет, сокращенно.

***Tag* (тэг)** – специальный компонент *HTML*-кода, который предаёт разным элементам веб-страницы (тексту, картинке, гиперссылке и др.) соответствующий вид. Любой *HTML*-элемент состоит из 3-х соответствующих частей: начального тэга, содержания тэга и конечного тэга. Например, на главной странице портала МЭИ имеется тэг с заголовком

```
<TITLE>
Национальный исследовательский университет "Московский энергетический институт"
</TITLE>
```

Веб-страница – структура последовательных тэгов. Для определения структуры и визуального эффекта веб-страницы используются специальные компоненты или тэги. Кроме тэгов на веб-странице могут располагаться тексты, картинки, гиперссылки, апплеты и т.д. Современные веб-страницы не ограничиваются *HTML*- или *XHTML*-тэгами. В коде этих страниц можно обнаружить динамические компоненты и сложные скрипты, реализованные средствами *CSS, JavaScript* или *Ajax*.

***URL* (*Universal Resource Locator*)** – универсальный локатор – адрес ресурса.

***Web* (или *World Wide Web – WWW*) –** обозначение Всемирной паутины. Распределённая система, предоставляющая доступ к связанным между собой документам, расположенным на одном или разных компьютерах, подключенных к Интернету.

***XHTML* (*eXtensible HyperText Markup Language*)** – расширенный язык разметки гипертекста. Язык *XHTML* можно представить, как результат слияния языков *HTML* и *XML*.

***XML* (*eXtensive Markup Language*)** – язык разметки, текстовый формат, предназначенный для хранения структурированных данных.

***XPath* (*XML Path Language*)** – *XML/HTML* имеет древовидную структуру. В документе всегда имеется корневой элемент. У элемента дерева всегда существуют потомки и предки, кроме корневого элемента, у которого предков нет, а также тупиковых элементов (листьев дерева), у которых нет потомков. Каждый элемент дерева находится на определенном уровне. У элементов на

одном уровне бывают предыдущие и следующие элементы.

Веб-браузер – специальная программа для интерпретации *HTML*-страниц, позволяющая выполнять интерактивные действия. Популярные браузеры – *Opera, Mozilla Firefox, Google Chrome, Internet Explorer* и *Safari*.

Временн*о*е окно – интервал времени $\Delta = t_{k+1} - t_k$. Принцип временного окна – метод разделения большого интервала времени на более мелкие интервалы. Для каждого интервала времени проводиться проверка состояний кластеров и при необходимости проводится повторная кластеризация.

Гиперссылка – часть веб-документа, ссылающаяся на другой элемент в самом документе, либо на элементы других документов на других серверах (в других доменах).

Интернет-контент – содержание Интернет-ресурса (текст, картинки, интерактивные элементы, встроенные динамические элементы, скрипты и др.)

Интернет-портал – сайт, который предоставляет пользователю различные интерактивные сервисы, работающие в рамках этого сайта. Интернет-портал может состоять из нескольких сайтов, если они объединены под одним доменным именем.

Интернет-холдинг – бизнес-структура, основной доход которой обусловлен предоставлением широкого спектра Интернет-услуг. Имеет головную компанию, в подчинении которой находятся дочерние предприятия, с контрольными пакетами акций, принадлежащими головной компании. Примеры: *mail.ru, yandex.ru* и *rambler.ru*.

ИП – Интернет-пользователь.

ИР – Интернет-ресурс.

Кластеризация (*cluster* – сгусток, гроздь, скопление) – способ группировки многомерных объектов, основанный на представлении результатов отдельных наблюдений точками подходящего геометрического пространства с последующим выделением групп, как сгустков этих точек.

Поисковая история – список поисковых терминов и посещённых пользователем Интернет-страниц Всемирной паутины.

Сервер БД – сервер базы данных.

Таргетинг – рекламный механизм, позволяющий выделить из всей имеющейся аудитории только ту ее часть, которая удовлетворяет заданным критериям (целевую аудиторию) и показать рекламу именно ей.

СПИСОК ЛИТЕРАТУРЫ

1. Алгоритм Дейкстры // Электронный ресурс // Викиконспекты Национального Исследовательского Университета ИТМО СПб. URL: http://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%94%D0%B5%D0%B9%D0%BA%D1%81%D1%82%D1%80%D1%8B. (01.11.2014 г.)

2. Алгоритм кластеризации *k*-means // Электронный ресурс // URL: http://robocraft.ru/blog/computervision/1061.html (1.12.2014 г.)

3. **Айвазян С. А., Бухштабер В. М., Енюков И. С., Мешалкин Л. Д.** Прикладная статистика: Классификация и снижение размерности. М.: Финансы и статистика. – 1989.

4. **Айвазян С.А., Енюков И.С., Мешалкин Л. Д.** Прикладная статистика. Основы моделирования и первичная обработка данных. М.: Финансы и статистика. – 1983.

5. **Афонин А.А., Крейнес М.Г.** Кластеризация текстовых коллекций: помощь при содержательном поиске и аналитический инструмент // Сборник научных статей «Интернет-порталы: содержание и технологии». Выпуск 4 / ФГУ ГНИИ ИТТ «Информика». – М.: Просвещение. – 2007. – С. 510-537.

6. **Барсегян А. А.** Методы и модели анализа данных: OLAP и Data mining. / А. А. Барсегян, М. С. Куприяенов, В. В. Степаненко, И. И. Холод. – СПб. : БХВ-Петербург. – 2004.

7. **Басакер Р., Саатн Т.** Конечные графы и сети. Перевод с английского. – М: Наука. – 1973.

8. Библиотека работы с DOM HTML-документов для C# // Электронный ресурс // URL: http://htmlagilitypack.codeplex.com/ (1.12.2014 г.)

9. Википедия. Яндекс // Электронный ресурс // URL: http://ru.wikipedia.org/wiki/%DF%ED%E4%E5%EA%F1 (1.12.2014 г.).

10. **Вирин Ф. Ю.** Интернет-маркетинг. Полный сборник практических инструментов. – М: Эксмо. – 2010.

11. **Воронцов К. В., Колосков А. О.** Профили компактности и выделение опорных объектов в метрических алгоритмах классификации // Искусственный интеллект. – 2006. № 2. – С. 30-33.

12. **Воронцов К. В.** Лекции по алгоритмам кластеризации и многомерного шкалирования // Электронный ресурс // URL: http://www.ccas.ru/voron/download/Clustering.pdf (1.12.2014 г.)

13. **Гайдышев И.** Анализ и обработка данных: специальный справочник. – СПб.: Питер – 2001. – 752 с.

14. **Гилл А.** Введение в теорию конечных автоматов: теоретические основы технической кибернетики.– М.: Наука. – 1966.

15. **Гимаров В. А., Дли М. И., Битюцкий С. Я.** Задачи нестационарной кластеризации состояния нефтехимического оборудования // Нефтегазовое дело. – 2004. // Электронный ресурс // URL: http://www.ogbus.ru/authors/Gimarov/Gimarov_1.pdf (1.12.2014 г.)

16. ГОСТ Р 7.0.11-2011 - Система стандартов по информации, библиотечному и издательскому делу. Диссертация и автореферат диссертации. Структура и правила оформления. М.: Стандартинформ. – 2012. // Электронный ресурс // URL: http://protect.gost.ru/document.aspx?control=7&id=179727 (01.12.2014)

17. **Гулин В.В.** Исследование и разработка методов и программных средств классификации текстовых документов // Электронный ресурс // URL: http://www.mpei.ru/LANG/RUS/Publish/InfoAcadCncl/2013/GulinVV.pdf (01.11.2014)

18. **Гулин В.В.** Сравнительный анализ методов классификации текстовых документов // Вестник МЭИ. – 2011, № 6. – М.: Изд. дом МЭИ. – С. 100-108.

19. **Голощапов А.** Microsoft Visual Studio 2010. – СПб: BHV. – 2011.

20. **Дунаев Е. В.** Автоматическая рубрикация web-страниц в интернет-каталоге с иерархической структурой / Е. В. Дунаев, А. А. Шелестов // Интернет-математика 2005. Автоматическая обработка веб-данных. – М. 2005. – С. 382-398 .

21. **Дюран Б.** Кластерный анализ – М.: Статистика. – 1977. – 128 с.

22. **Загоруйко И. Г., Елкина В., Лбов Г. С.** Алгоритмы обнару-

обнаружения эмпирических закономерностей. – Новосибирск: Наука. – 1985. – 110 с.

23. **Ицик Б.** Microsoft SQL Server 2012. Высокопроизводительный код T-SQL. Оконные функции. – М: Русская Редакция. – 2013.

24. **Ицик Б., Сарка Д., Талмейдж Р.** Microsoft SQL Server 2012. Создание запросов. Учебный курс Microsoft. – СПб: BHV. – 2014.

25. **Китова Н.П.** Реклама в социальных сетях: особенности, функциональные возможности, инструменты продвижения // Экономика и управление. 2011 // Электронный ресурс // URL: http://ecsocman.hse.ru/data/2012/05/28/1271377512/56.pdf (1.12.2014 г.)

26. **Киселева Ю. Е.** Автоматическая сегментация запросов интернет-магазинов // Программные продукты и системы. 2010. № 3 // Электронный ресурс // URL: http://swsys.ru/index.php?page=article&id=2579 (1.12.2014 г.).

27. Классификация и кластер. / Под ред. Дж. Вэи Райзина. – М.: Мир. – 1980.– 390 с.

28. **Куралёнок И. Е.** Оценка систем текстового поиска // Электронный ресурс // URL: http://www.dissercat.com/content/otsenka-sistem-tekstovogo-poiska (01.11.2014).

29. **Максаков А.** Сравнительный анализ алгоритмов классификации и способов представления Web-документов // Российский семинар по Оценке Методов Информационного Поиска (РОМИП), 2005 // Электронный ресурс // URL: http://romip.ru/romip2005/05_specs.pdf (1.12.2014 г.).

30. **Мандель И. Д.** Кластерный анализ. – М.: Финансы и статистика. – 1988.

31. **Маслов М. Ю., Пяллинг А.А., Трифонов С.И.** Автоматическая классификация веб-сайтов / Результаты исследования компании Яндекс 2008 // Электронный ресурс // URL: http://download.yandex.ru/company/experience/rcdl2008/rcdl_sites_autoclassification.pdf (1.12.2014 г.).

32. Методы оценки качества классификации текста // Электронный ресурс // URL:http://datamin.ubbcluj.ro/wiki/index.php/Evaluation_methods_in_text_categorizat

ion (1.12.2014 г.)

33. **Миркин Б. Г.** Методы кластер-анализа для поддержки принятия решений: обзор / Б. Г. Миркин Национальный исследовательский университет «Высшая школа экономики». – М.: Изд. дом НИУ «Высшая школа экономики». – 2011. – 39 с.

34. Навигатор веб-мастера. Динамический HTML // Электронный ресурс // URL: http://www.webnav.ru/books/html4/dhtml/ (1.12.2014 г.)

35. **Нейман Ю.** Вводный курс теории вероятностей и математической статистики. – М.: Наука. – 1968.

36. Онлайн исследования в России: тенденции и перспективы / Под редакцией Шашкина А. В. и Поздняковой М. Е. – М.: Издательство Института социологии РАН. – 2006.

37. **Ночевнов Д.** Методы и средства сегментации web-сайтов // XVth International Conference “Knowledge-Dialogue-Solution” KDS-2. 2009 // Электронный ресурс // URL: http://www.foibg.com/ibs_isc/ibs-15/ibs-15-p13.pdf (1.12.2014 г.)

38. **Паутов К.Г., Попов Ф.А.** Тематическая классификация веб-страниц в системах фильтрации Интернет-трафика. // Электронный архив Уральского Федерального Университета 2005. // URL: http://elar.urfu.ru/bitstream/10995/1419/1/IMAT_2005_20.pdf (1.12.2014 г.)

39. **Плахов А.** Поисковая технология спектр / доклады Yet another Conference 2010 // Электронный ресурс // URL: http://yac2011.yandex.ru/archive2011/video1/ (1.12.2014 г.)

40. **Петцольд Ч.** Программирование с использованием Microsoft Windows Forms. Перевод с английского. – СПб: Русская Редакция. – 2006.

41. Поисковая технология «Матрикснет» // Электронный ресурс // URL: http://company.yandex.ru/technologies/matrixnet/ (1.12.2014 г.)

42. Разделяй и властвуй: кластерные поисковики // Электронный ресурс // UPGRADE твой компьютерный еженедельник: сетевой журнал 2008. URL: http://www.upweek.ru/razdelyaj-i-vlastvuj-klasternye-poiskoviki.html. (10.02.2014)

43. Руководство по поисковой оптимизации для начинающих // Электронный ресурс // URL: http://static.googleusercontent.com/media/www.google.ru/ru/ru/intl/ru/webmasters/docs/search-engine-optimization-starter-guide-ru.pdf (1.12.2014 г.)

44. **Сарка Д., Лах М., Йеркович Г.** Microsoft SQL Server 2012. Реализация хранилищ данных. Учебный курс Microsoft. – М: Русская Редакция. – 2014.

45. **Сегаран. Т.** Программируем коллективный разум. / Пер. с англ. – СПб: Символ-Плюс. – 2008.

46. **Спинеллис Д., Гусиои Г.** Идеальная архитектура. Ведущие специалисты о красоте программных архитектур. Перевод с английского. – СПб: Символ-Плюс. – 2013.

47. Справочник HTML // Электронный ресурс // URL: http://htmlbook.ru/html (1.12.2014 г.)

48. **Сухов К.** HTML5 путеводитель по технологии. – М.: ДМК Пресс, – 2013.

49. **Троелсен Э.** Язык программирования C# 2010 и платформа .NET 4.0, 5-е изд. Перевод с английского. – М. : ООО «И.Д. Вильямс». – 2011.

50. **Ту Дж., Гонсалес Р.** Принципы распознавания образов. – М.: Мир. – 1978. – 401 с.

51. **Чубукова И. А.** Data Mining : Учебное пособие / И. А. Чубукова. М. Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний. – 2006. – 382 с.

52. **Шаров С.А.** Частотный словарь русского языка // Электронный ресурс // URL: http://www.artint.ru/projects/frqlist.asp (1.12.2014 г.).

53. **Шаграев А.Г.** Модификация, разработка и реализация методов классификации новостных текстов // Электронный ресурс // http://www.mpei.ru/LANG/RUS/Publish/InfoAcadCncl/2014/ShagraevAG_diss.pdf (01.11.2014)

54. Яндекс. Статистика. Аудитория сервисов Яндекса // Электронный ресурс // URL: http://stat.yandex.ru/stats.xml?ReportID=-225&ProjectID=1 (1.12.2014 г.).

55. **A. K. Jain, R. C. Dubes** Algorithms for Clustering Data – Englewood Cliffs. – N.Y.: Prentice Hall. –1988. – 334 p.

56. **Bien J., Tibshirani R.** Hierarchical Clustering With Prototypes via Minimax Linkage // Journal of the American Statistical Association. 2011 // Электронный ресурс // URL: http://faculty.bscb.cornell.edu/~bien/papers/jasa2011minimax.pdf (1.12.2014 г.)

57. **Chakarbarti S.** Mining the web: discovering knowledge from hypertext data. – San Francisco: Morgan Kaufmann Publishers. – 2003.

58. Easily parse HTML Documents in C# // Электронный ресурс // URL: http://olussier.net/2010/03/30/easily-parse-html-documents-in-csharp/ (1.12.2014 г.)

59. **Eirinaki M., Vazirgiannis M.** Web Mining for Web Personalization // ACM Transactions on Internet Technology, 2003. vol. 3, No, 1 // Электронный ресурс // URL: http://doi.acm.org/10.1145/643477.643478 (1.12.2014 г.)

60. **El-Hamduchi A., Willet P.** Comparison of hierarchic agglomerative clustering methods of document retrieval // The Computer Journal. 1989. vol.32 №3 // Электронный ресурс // URL: http://comjnl.oxfordjournals.org/content/32/3/220.full.pdf (1.12.2014 г.)

61. **Florek K., Lukaszewicz J., Perkal J., Steinhaus H., Zubrzycki S.** Sur la liaison et la division des points d'un ensemble fini // Colloqium Math. 1951 № 2 // Электронный ресурс // URL: http://matwbn.icm.edu.pl/ksiazki/cm/cm2/cm2145.pdf(1.12.2014 г.) .

62. **Gondse P., Raut A.** Main Content Extraction From Web Page Using Dom // International Journal of Advanced Research in Computer and Communication Engineering. 2014. № 3 // Электронный ресурс // URL: http://www.ijarcce.com/upload/2014/march/IJARCCE5H%20%20a%20pranjali%20%20MAIN%20CONTENT%20EXTRACTION.pdf (1.12.2014 г.)

63. **Guandong X., Yanchun Z., Lin L.** Web Mining and Social Networking techniques and applications. – N. Y.: Springer. – 2011.

64. **Gupta S., Kaiser G., Grimm P., Chiang M., Starren J.** Automating Content Extraction of HTML Documents. Dordrecht: Kluwer Academic Publishers. 2004. // Электронный ресурс // URL: https://york.cs.columbia.edu/crunch/WWWJ.pdf

(1.12.2014 г.)

65. Internet 2011 in numbers // Электронный ресурс // URL: http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/ (1.12.2014 г.)

66. **Jain A., Murty M., Flynn P.** Data Clustering // ACM Computing Surveys. 1999. Vol. 31 № 3 pp. 264-323 // Электронный ресурс // URL: http://www.cs.tau.ac.il/~fiat/DataMine05/p264-jain.pdf. (1.12.2014 г.)

67. **Jardine N., Sibson R.** The construction of hierarchic and non-hierarchic classifications // The Computer Journal Oxford. 2011. pp. 177-184 // Электронный ресурс // URL: http://biocomparison.ucoz.ru/_ld/0/60_jardine_constru.pdf (1.12.2014 г.)

68. **Keith J., Sambells J.** DOM Scripting. – N. Y.: Apress. – 2010.

69. **Kogan J.** Introduction to Clustering Large and High-Dimensional data. – N. Y. : Cambridge University Press. – 2006.

70. **Lewis D.** Naive (bayes) at forty: The independence assumption in information retrieval. Springer Verlag, 1998. // Электронный ресурс // URL: http://www.cs.iastate.edu/~honavar/bayes-lewis.pdf (1.12.2014 г.)

71. **Louvan S.** Extracting the main content from web documents / Louvan S.; Eindhoven University of Technology. 2009 // Электронный ресурс // URL: http://www.win.tue.nl/~mpechen/projects/pdfs/Louvan2009.pdf (1.12.2014 г.)

72. **Porter M.** The Porter Stemming Algorithm // Электронный ресурс // URL: Сайт. – Режим доступа: snowball.tartarus.org (20.03.2014 г.)

73. **Rajalingam N., Ranjini K.** Hierarchical clustering algorithm - A Comparative Study // International Journal of Computer Applications. 2011. vol.19, №3 // Электронный ресурс // URL: http://www.ijcaonline.org/volume19/number3/pxc3873052.pdf (1.12.2014 г.)

74. **Robertson S.** Understanding Inverse Document Frequency: on theoretical arguments for IDF // Journal of Documentation, 2004, №5. – P. 503-520

75. **Sculley D.** Web-Scale K-Means Clustering // Конференция WWW 2010 // Электронный ресурс // URL: http://www.ra.ethz.ch/cdstore/www2010/www/p1177.pdf (1.12.2014 г.)

76. **Shamir R., Sharan R., Tsur D.** Cluster graph modication problems //

Discrete Applied Mathematics. 2004, vol.144, № 2 // Электронный ресурс // URL: http://www.cs.bgu.ac.il/~dekelts/publications/cmod.pdf (1.12.2014 г.)

77. **Shelly G., Woods D.** HTML, XHTML, AND CSS. – Boston: Course Technology, Cengage Learning. – 2011.

78. **Singh A.** Web Content Extraction to Facilitate Web Mining // International Journal of Electronics and Computer Science Engineering. 2012. № 1 // Электронный ресурс // URL: http://www.ijecse.org/wp-content/uploads/2012/06/Volume-1Number-3PP-1292-1299.pdf (1.12.2014 г.).

79. **Soumen C.** Mining the Web Discovering Knowledge from Hypertext Data. – San Francisco: Morgan Kauffman Publishers. – 2003.

80. **Sundar G., Narmadha D., Haran A.** Combinational Scheme for Efficient Content Extraction from Web Pages // Australian Journal of Basic and Applied Sciences. 2014. №1 // Электронный ресурс // URL:http://www.academia.edu/6387488/Combinational_Scheme_for_Efficient_Content_Extraction_from_Web_Pages (1.12.2014 г.).

81. **Sun F., Song D., Liao L.** DOM Based Content Extraction via Text Density // Lab of High Volume language Information Processing & Cloud Computing Beijing Lab of Intelligent Information Technology, Beijing Institute of Technology. 2011 // Электронный ресурс // URL: http://disnet.cs.bit.edu.cn/DOM%20Based%20Content%20Extraction%20via%20Text%20Density.pdf (1.12.2014 г.)

82. TNS The sixth sense of business // Электронный ресурс // URL: http://www.tns-global.ru/rus/data/ratings/index/ (1.12.2014 г.).

83. **Witten I., Frank E.** Data mining. – San Francisco, CA: Morgan Kaufmann Publishers. – 2005.

84. XML Document Object Model (DOM) // Электронный ресурс // Microsoft developer network: библиотека разрабочика. URL: http://msdn.microsoft.com/ru-ru/library/hf9hbf87(v=vs.110).aspx. (10.02.2014 г.)

СПИСОК ИЛЛЮСТРАТИВНОГО МАТЕРИАЛА

ПРИЛОЖЕНИЕ 1. ИСХОДНЫЙ *SQL*-КОД КЛАСТЕРИЗАЦИИ МЕТОДАМИ *TF* и *TF-DOM*

```
--SQL-код кластеризации методом TF
-- Случайным образом выбираем 10 URL
SELECT TOP(10) page_id, page, cnt into #pages
FROM [HTML].[dbo].[Pages]
ORDER BY NEWID()

-- Находим количество слов для 10 URL
SELECT [page_id], count([Item]) as cnt_words into #words_in_page
FROM [HTML].[dbo].[az_words_count_in_pages]
WHERE page_id in
(
      select page_id FROM #pages
)
group by [page_id]
order by cnt_words desc
--select * from #words_in_page

-- Находим количество кокретних слов на одну из тем для 10 URL.
SELECT A.[tema], A.[item], B.page_id, count(A.item) as cnt_word
INTO #exact_word_cnt
FROM
[HTML].[dbo].[new_tema] A
INNER JOIN
[HTML].[dbo].[az_words_count_in_pages] B
ON A.item = B.Item
WHERE B.page_id in
(
      SELECT page_id FROM #pages
)
GROUP BY A.[tema], A.[item], B.page_id
--select * from #exact_word_cnt

-- Сводим результат.
/*
SELECT A.page_id, B.tema, A.cnt_words, sum(B.cnt_word) as tema_cnt
FROM
#words_in_page A
INNER JOIN
#exact_word_cnt B
ON A.page_id = B.page_id
GROUP BY A.page_id, B.tema, A.cnt_words
ORDER BY A.page_id, B.tema, A.cnt_words, tema_cnt DESC
*/

SELECT page_id,
COALESCE(PivotTable.[президент РФ],0) as [президент РФ],
COALESCE(PivotTable.[разоблачения и коррупция в минобороне],0) as [разоблачения и
коррупция в минобороне],
COALESCE(PivotTable.[Pussy Riot],0) as [Pussy Riot],
COALESCE(PivotTable.[война в Сирии],0) as [война в Сирии],
COALESCE(PivotTable.[заблудившийся рыбаки в тайге],0) as [заблудившийся рыбаки в
тайге],
COALESCE(PivotTable.[пробка на трассе],0) as [пробка на трассе]
INTO #page_tema_pivot
FROM
(SELECT page_id, tema,cnt_word
    FROM #exact_word_cnt) AS MainTable
PIVOT
```

```
(
SUM(cnt_word)
FOR tema IN ([президент РФ],
[разоблачения и коррупция в минобороне], [Pussy Riot],
[война в Сирии], [заблудившийся рыбаки в тайге],
[пробка на трассе])
) AS PivotTable;

select A.page_id, A.[президент РФ]*100/B.cnt_words as [президент РФ],
A.[разоблачения и коррупция в минобороне]*100/B.cnt_words as [разоблачения и коррупция в минобороне],
A.[Pussy Riot] as [Pussy Riot],
A.[война в Сирии]*100/B.cnt_words as [Pussy Riot],
A.[заблудившийся рыбаки в тайге]*100/B.cnt_words as [заблудившийся рыбаки в тайге],
A.[пробка на трассе]*100/B.cnt_words as [заблудившийся рыбаки в тайге] FROM #page_tema_pivot A
inner join #words_in_page B
ON A.page_id = B.page_id

DROP TABLE #page_tema_pivot
DROP TABLE #exact_word_cnt;
DROP TABLE #words_in_page;
DROP TABLE #pages;



--SQL-код кластеризации методом TF-DOM
-- выбираем 10 URL, из предыдущего эксперимента.
SELECT page_id, page, cnt into #pages
FROM [HTML].[dbo].[Pages]
WHERE page_id in
(4,5,17,32,46,54,65,68,77,79)

-- Находим количество слов для 10 URL
-- берём нужные нам тэги (h1, p и title)
SELECT [page_id], count([Item]) as cnt_words into #words_in_page
FROM [HTML].[dbo].[az_words_count_in_pages]
WHERE page_id in
(
      select page_id FROM #pages
)
AND html_element_id in (122, 52, 91)
group by [page_id]
order by cnt_words desc
--select * from #words_in_page

-- Находим количество кокретних слов на одну из тем для 10 URL.
SELECT A.[tema], A.[item], B.page_id, B.HTML_element_id, count(A.item) as cnt_word,
k2 = case B.HTML_element_id when 122 then 10 when 52 then 5 else 1 end
INTO #exact_word_cnt
FROM
[HTML].[dbo].[new_tema] A
INNER JOIN
[HTML].[dbo].[az_words_count_in_pages] B
ON A.item = B.Item
WHERE B.page_id in
(
```

```
      SELECT page_id FROM #pages
)
AND html_element_id in (122, 52, 91)
GROUP BY A.[tema], A.[item], B.page_id, B.HTML_element_id
ORDER BY B.Page_id

-- Сводим результат.
SELECT * FROM #exact_word_cnt

SELECT tema, page_id, SUM(cnt_word*k2) as wt FROM #exact_word_cnt
GROUP BY tema, page_id
ORDER BY page_id

DROP TABLE #exact_word_cnt;
DROP TABLE #words_in_page;
DROP TABLE #pages;
```

ПРИЛОЖЕНИЕ 2. МЕРЫ БЛИЗОСТИ

Евклидово расстояние.

Наиболее распространенная функция расстояния представляет собой геометрическое расстояние между точками (векторами) в многомерном пространстве:

$$\rho\left(\overrightarrow{X_i}, \overrightarrow{X_j}\right) = \sqrt{\sum_{k=1}^{n} (x_k^{(i)} - x_k^{(j)})^2} ,$$

где $x_k^{(i)}$ и $x_k^{(j)}$ — *k*-ая координата векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$ соответственно. n — размерность векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$.

Квадрат евклидова расстояния.

Применяется для придания большего веса более отдаленным друг от друга объектам. Это расстояние вычисляется следующим образом:

$$\rho\left(\overrightarrow{X_i}, \overrightarrow{X_j}\right) = \sum_{k=1}^{n} (x_k^{(i)} - x_k^{(j)})^2 ,$$

где $x_k^{(i)}$ и $x_k^{(j)}$ — *k*-ая координата векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$ соответственно. n — размерность векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$.

Расстояние городских кварталов (манхэттенское расстояние).

Это расстояние является средним разностей по координатам. В большинстве случаев эта мера расстояния приводит к таким же результатам, как и для обычного расстояния Евклида. Однако для этой меры влияние отдельных больших разностей (выбросов) уменьшается (т.к. они не возводятся в квадрат):

$$\rho\left(\overrightarrow{X_i}, \overrightarrow{X_j}\right) = \sum_{k=1}^{n} | x_k^{(i)} - x_k^{(j)} |$$

где $x_k^{(i)}$ и $x_k^{(j)}$ — *k*-ая координата векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$ соответственно. *n* — размерность векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$.

Степенное расстояние.

Применяется в случае, когда необходимо увеличить или уменьшить вес, относящийся к размерности, для которой соответствующие объекты сильно отличаются:

$$\rho\left(\overrightarrow{X_i}, \overrightarrow{X_j}\right) = \sqrt[r]{\sum_{k=1}^{n} (x_k^{(i)} - x_k^{(j)})^p}$$

где $x_k^{(i)}$ и $x_k^{(j)}$ — k-ая координата векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$ соответственно; n — размерность векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$; r и p – параметры, определяемые пользователем. Параметр p ответственен за постепенное взвешивание разностей по отдельным координатам, параметр r ответственен за прогрессивное взвешивание больших расстояний между объектами. Если оба параметра – r и p — равны двум, то это расстояние совпадает с расстоянием Евклида.

Расстояние Чебышева.

Этот метод применяется, когда необходимо различать расстояние между векторами по какой-либо одной координате:

$$\rho\left(\overrightarrow{X_i}, \overrightarrow{X_j}\right) = \max_{k=1..n} \left| x_k^{(i)} - x_k^{(j)} \right|,$$

где $x_k^{(i)}$ и $x_k^{(j)}$ — k-ая координата векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$ соответственно; n — размерность векторов $\overrightarrow{X_i}$ и $\overrightarrow{X_j}$.

Расстояние между центрами тяжести групп.

Применяется в случае, когда необходимо получить грубую оценку состояния кластерной структуры после расчёта координат центров тяжести. В частности это позволяет определить степень конденсации или распада кластеров.

$$d(\overrightarrow{X_i}, \overrightarrow{X_j}) = \rho(c_i, c_j),$$

Где c_i и c_j — центры тяжести векторов X_i и X_j соответственно.

Групповое среднее расстояние.

Применяется для сравнения всех элементов двух кластеров между собой.

$$d(K_1, K_2) = \frac{1}{|K_1| \times |K_2|} \times \sum_{X_i \in K_1} \sum_{X_j \in K_2} \rho(X_i, X_j),$$

где K_1 и K_2 — 2 сформированных кластера содержащих элементы X_i и X_j соответственно $X_i \in K_1$ и $X_j \in K_2$. Выбор метрики полностью лежит на исследователе, поскольку результаты кластеризации могут существенно отличаться при использовании разных мер. Существуют более сложные формулы для оценки расстояния между объектами, например формула расстояния по Колмогорову [3].

ПРИЛОЖЕНИЕ 3. АНАЛИЗ МЕТОДОВ КЛАСТЕРИЗАЦИИ ИНТЕРНЕТ-ПОЛЬЗОВАТЕЛЕЙ И ИНТЕРНЕТ-РЕСУРСОВ

П.1. Математическое описание Интернет-пользователей и их дивизивная кластеризация

Исследование метода дивизивной кластеризации Интернет-пользователей проводилось в период с 4 по 10 марта 2013 г. В эксперименте участвовали 214 ИП, которые выполняли хотя бы один поисковый запрос в сутки.

Исследование предполагало, что каждому ИП $usr_i \in USR = \{usr_1, \ldots, usr_i, \ldots, usr_{214}\}$ сопоставляется характеристический (поисковый) вектор

$$u_i = (u_{i,1}, \ldots, u_{i,j}, \ldots, u_{i,nof(Vu)}) \in U, \qquad (П.1)$$

где $u_{i,j}$ – вес j-го поискового термина из глобального словаря терминов V_u, равный числу вхождений этого термина в запросы i-го пользователя; $nof(V_u)$ – размер поискового вектора i-го пользователя, равный числу слов в глобальном словаре терминов.

В эксперименте в качестве V_u используется латинский алфавит, поэтому $nof(V_u) = 26$ – по числу букв в алфавите. Числовые координаты $u_{i,j}$, $1 \le j \le nof(V_u)$, расположены в характеристическом векторе, в том же порядке, что и термины в словаре V_u. Переход от вербального к числовому представлению результатов поисковой активности происходит за счет позиционного кодировании терминов и подсчёта числа их вхождений в запросы i-го пользователя. После формирования характеристических векторов для всех пользователей, результат представляется матрицей координат размером 214×26 = 5564 элементов. Условное изображение множества исследуемых ИП показано на рисунке П.1.

В среде *MS SQL Server* 2012 характеристические вектора сохраняются в таблице размером 214×27 ячеек (первый столбец таблицы выделяется под индекс пользователя i, $1 \le i \le 214$). На основании значений, представленных в этой таблице, производится расчёт расстояний между всеми парами u_i и u_j.

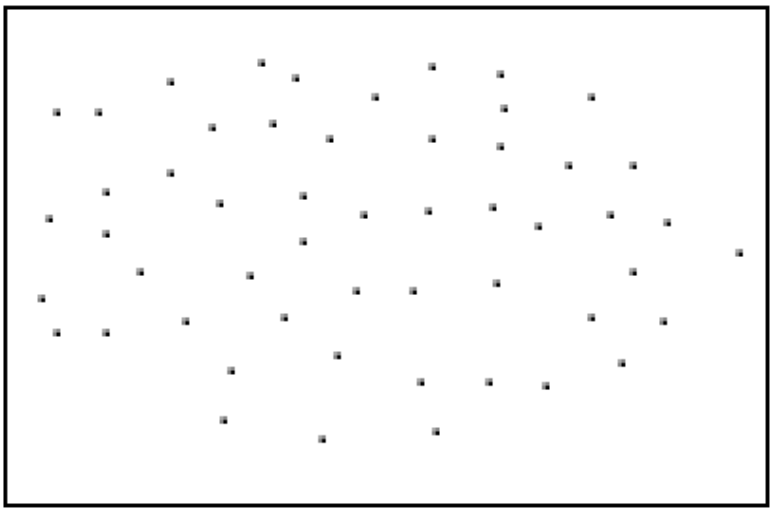

Рисунок П.1 – Иллюстрация исходного множества исследуемых объектов

Расчёт евклидового расстояния реализован в среде *MS SQL Server* 2012 с использованием формулы П.2

$$\rho(u_i,u_j)=\sqrt{\sum_{k=1}^{nof(V_u)}\left(x_{i,k}-x_{j,k}\right)^2}=\sqrt{\left(x_{i,1}-x_{j,1}\right)^2+\ldots+\left(x_{i,26}-x_{j,26}\right)^2}. \qquad (П.2)$$

После расчёта евклидовых расстояний между объектами, определяются две точки u_m и u_n, расположенные на максимальном удалении друг от друга (рисунок П.2)

$$\rho_{max}=\rho(u_m,u_n)=\max_{i,j,i\neq j}\left(\rho(u_i,u_j)\right). \qquad (П.3)$$

Для определения пары объектов u_m и u_n, принадлежащих исходному множеству, евклидово расстояние, между которыми максимально, применяется принцип последовательного перебора. Число необходимых сравнений для определения пары u_m и u_n равно

$$C^k_{nof(USR)}=\frac{nof(USR)!}{k!\times(nof(USR)-k)!}, \qquad (П.4)$$

где число наблюдаемых объектов $nof(USR)$ = 214, а k = 2 так, как разбиение бинарное. Применив формулу (П.4) получаем 22791 сравнение.

После того, как определены две максимально разнесенные точки u_m и u_n (рисунок П.2), начинается формирование кластеров, относящихся к этим точкам, на основе расчёта расстояний $\rho(u_k, u_m)$ и $\rho(u_k, u_n)$ для всех $u_k \in U\backslash\{u_m, u_n\}$ (рисунок П.3).

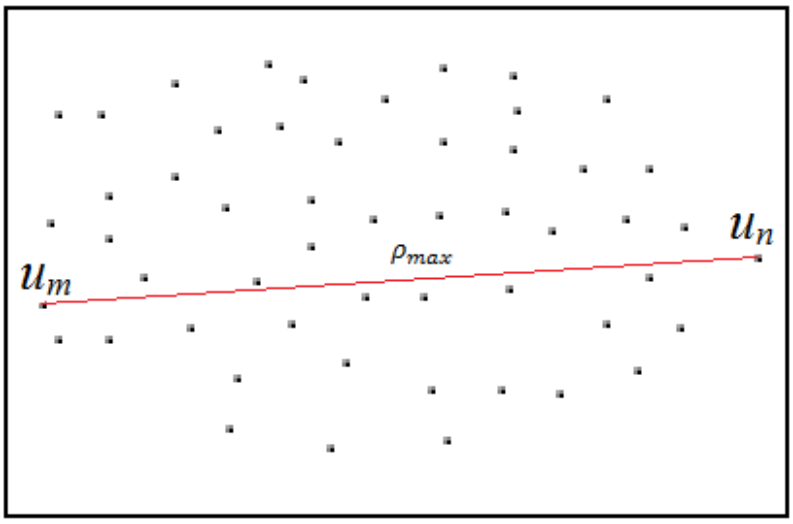

Рисунок П.2 – Определение максимально удаленных объектов

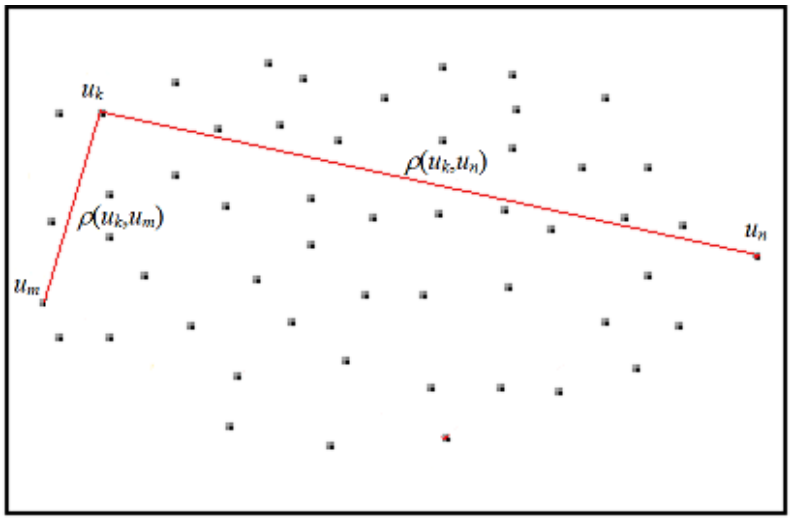

Рисунок П.3 – Распределение точек по кластерам

Если $\rho(u_k, u_m) \le \rho(u_k, u_n)$, то точку u_k следует привязать к точке u_m, если $\rho(u_k, u_n) \le \rho(u_k, u_m)$, то точку u_k нужно привязать к точке u_n. Если провести этот расчет для всех $u_k \in U\backslash\{u_m, u_n\}$, получим в итоге два кластера $U^{(1)}{}_1$ и $U^{(1)}{}_2$ таких, что $U^{(1)}{}_1 \cup U^{(1)}{}_2 = U$ и $U^{(1)}{}_1 \cap U^{(1)}{}_2 = \varnothing$ (рисунок П.4).

После второй итерации число кластеров будет равно четырем – $U^{(1.1)}{}_1$, $U^{(1.1)}{}_2$, $U^{(1.2)}{}_1$, $U^{(1.2)}{}_2$ (рисунок П.5), после третьей итерации – $U^{(1.1.1)}{}_1$, $U^{(1.1.1)}{}_2$, $U^{(1.1.2)}{}_1$, $U^{(1.1.2)}{}_2$, $U^{(1.2.1)}{}_1$, $U^{(1.2.1)}{}_2$, $U^{(1.2.2)}{}_1$, $U^{(1.2.2)}{}_2$. и так далее. Полное дробление исходного множества при помощи дивизивного алгоритма может быть достигнуто уже через $nof(USR) - 4 = 210$ шагов, когда получим одинарные группы точек.

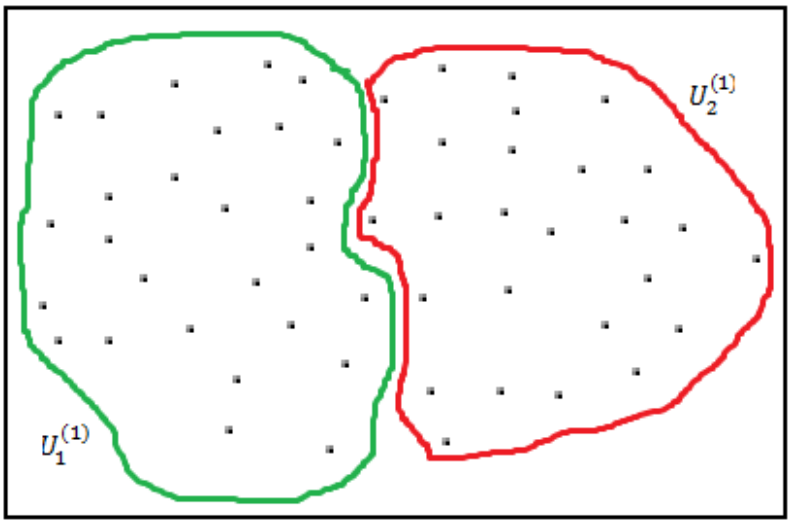

Рисунок П.4 – Два кластера после первой итерации

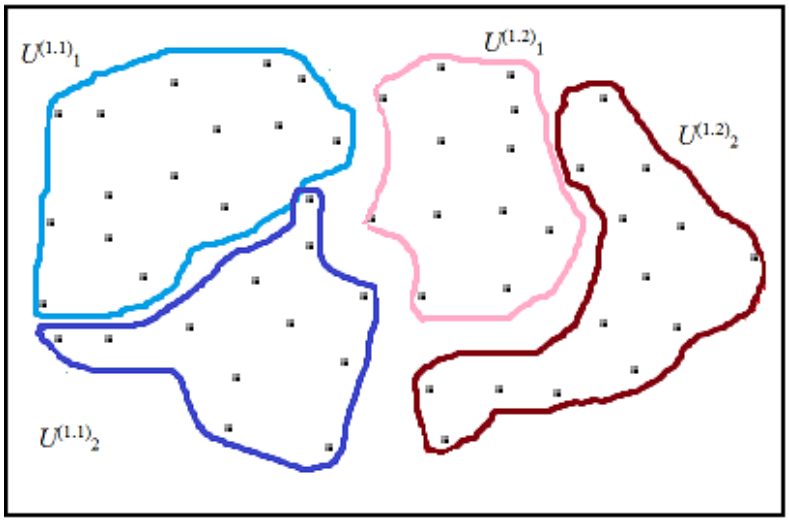

Рисунок П.5 – Кластерная структура после второй итерации.

Следует отметить, что в процессе проведения расчётов может получиться несколько пар максимально удалённых точек (с одним и тем же расстоянием). Для получения более точного результата, в этом случае, можно прибегнуть к расчёту дополнительной метрики, например, манхэттенского расстояния, группового среднего расстояния или расстояния Чебышева (приложение 12).

Сравнение классификации по статическим данным и классификации с помощью дивизивного алгоритма.

На основании персональной информации о поле и возрасте, можно построить 4-х уровневую иерархию, изображенную на рисунке П.6. На нулевом (начальном) уровне исследуемые объекты находятся в одном множестве

$U = U^{(0)} = \{u_1, u_2, ..., u_{214}\}$. На первом уровне иерархии получаем разбиение на подмножества $U^{(1)}_{муж}$ и $U^{(1)}_{жен}$ на основании пола. На втором уровне получаем разбиение из 10 групп на основании возрастных категорий. Третий уровень иерархии является последним, когда каждый объект исследования представляет с собой одинарную группу $U^{(3)}_{expression}$: $card\left(U^{(3)}_{expression}\right)=1$.

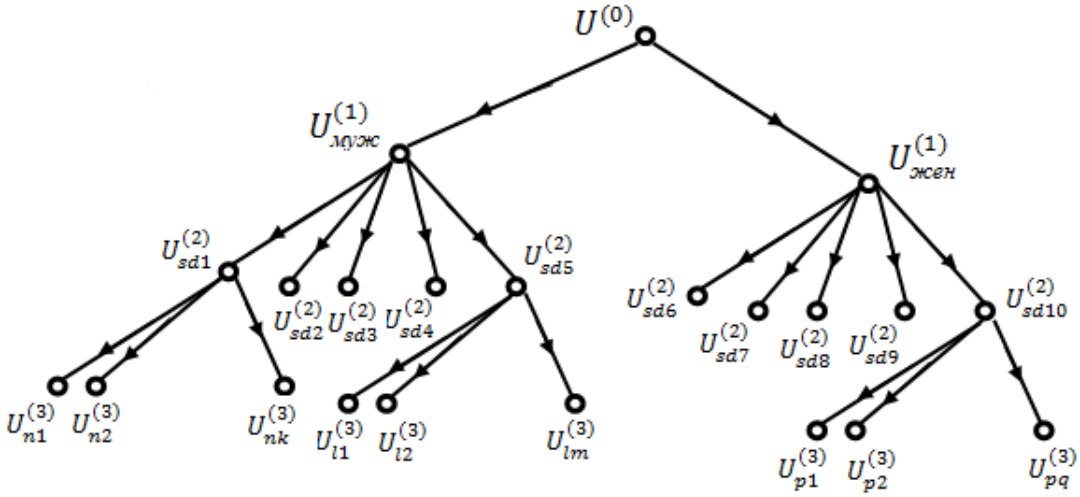

Рисунок П.6 – Граф классификации ИП с применением статической информации о поле и возрасте

Данная иерархия (уровни от 0 до 3) обладает большим разбросом (мерой близости) между объектами одного класса. Экспериментальным путём были получены результаты расчёта евклидового расстояния между объектами, когда в одной социально-демографической группе ИП находятся на максимальном расстоянии.

Ограниченное число уровней иерархии (рисунок П.6) позволяет провести классификацию максимально быстро, но качество классификации по интересам или по поисковой истории оставляет желать лучшего. С помощью дивизивного алгоритма, получается многоуровневая иерархия. Так как здесь применяется алгоритм бинарного разбиения, окончательное число уровней иерархии можно рассчитать по формуле $]log_2 214[= 8$ («вывернутыми» скобками обозначено наименьшее целое число большее или равное двоичному логарифму).

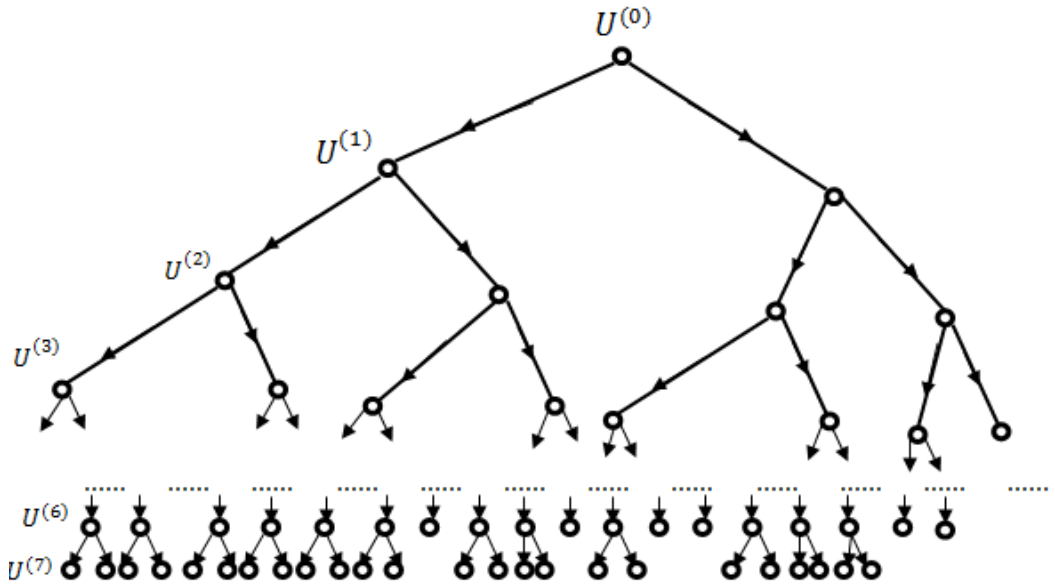

Рисунок П.7 – Граф кластеризации ИП с применением дивизивного алгоритма с бинарным базисом

Как было сказано, иерархические методы кластеризации относятся к неточным методам так, как число кластеров заранее не известно. Если на *k*-ом шаге получается разбиение, все классы которого удовлетворяют выбранному критерию однородности, то алгоритм обычно останавливается [3].

Экспериментальные исследования показали, что дивизивный метод кластеризации имеет ряд важных преимуществ по сравнению со статической кластеризацией:

- на каждом этапе кластеризации получаются группы, объекты которых ближе по поисковой истории;

- можно остановить алгоритм, если степень однородности кластерных групп удовлетворяет заданному критерию. Главным недостатком дивизивного метода выявляется большое число итераций разбиения.

П.2. Агломеративная кластеризация Интернет-пользователей

После того, как определено математическое описание объектов и выбран способ расчета евклидового расстояния в качестве меры близости, можно приступить к первой агломеративной итерации. Для этого, применив метод последовательного перебора пар элементов множества $U = \{u_1, u_2, ..., u_{214}\}$, находим

два точки u_m и u_n, для которых мера $\rho(u_m, u_n)$ будет минимальной

$$\rho_{min} = \rho(u_m, u_n) = \min_{i,j,i \neq j} \left(\rho(u_i, u_j)\right). \quad (П.5)$$

После вычисления евклидового расстояния ρ_{min}, строится кластер, состоящий из двух самых близких объектов $U_1^{(1)} = \{u_m, u_n\}$, где нижний индекс является индексом кластера, а верхний – номером итерации, на которой это кластер был построен (рисунок П.8).

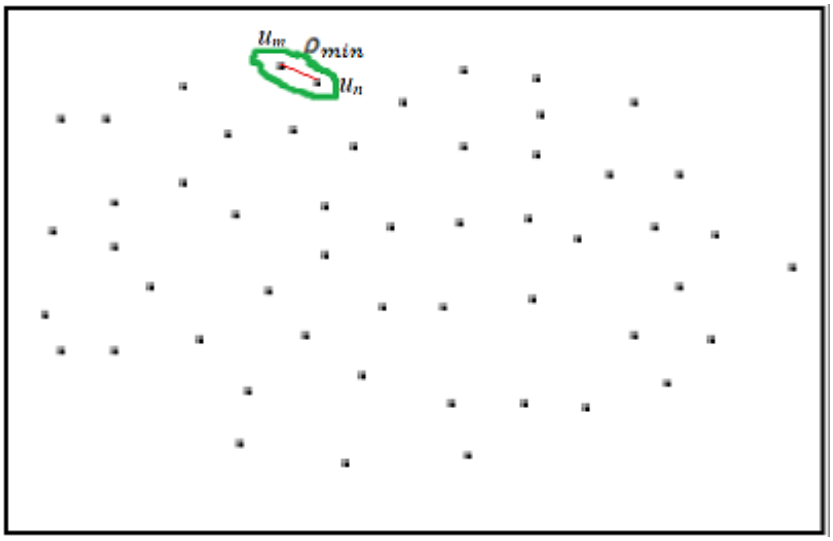

Рисунок П.8 – Формирование первого кластера из двух объектов с минимальной мерой близости

Продолжая процесс агломеризации для реальных данных, по результатам расчёта мер близости на следующей итерации происходит формирование двух новых кластеров $U_2^{(2)}$ и $U_3^{(2)}$. Это связано с одинаковым значением меры близости для двух разных пар точек (рисунок П.9).

Здесь хотелось бы указать на преимущество агломеративного метода над дивизивным: в пределах одной итерации возможно формирование нескольких кластеров одновременно. Всё зависит от расчёта меры близости: если минимальная мера близости определена для нескольких объектов единовременно, то возможно формирование нескольких кластеров за одну итерацию. Для более точного расчета необходимо применение дополнительной меры близости из ранее упомянутого списка мер.

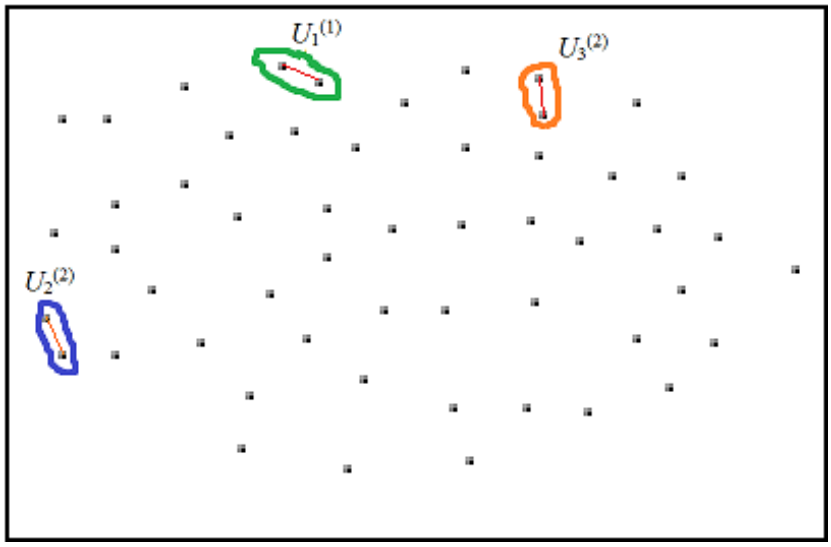

Рисунок П.9 – Формирования нескольких кластеров в ходе одной итерации

Продолжая кластеризацию согласно агломеративному алгоритму, на третьем этапе по имеющимся значениям характеристик объектов происходит наращивание уже сформированных кластеров – подмножество $U_3^{(2)}$, построенное на второй итерации, превращается после третьей итерации в новое подмножество $U_3^{(3)}$, состоящее из трех точек (рисунок П.10).

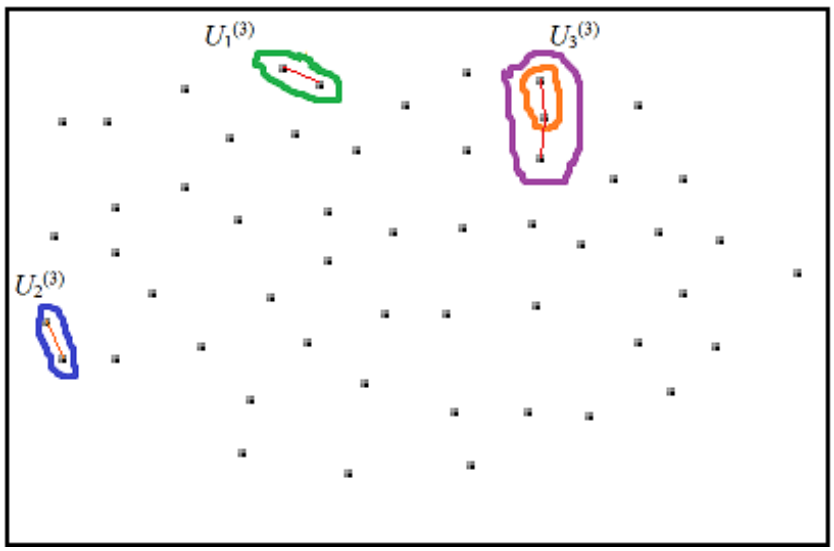

Рисунок П.10 – Иллюстрация наращивания числа объектов в кластерах

Полное завершение процесса агломерации будет достигнуто через число шагов не превосходящее *nof*(*USR*) – 1. После завершения агломеративного алгоритма получаем небинарный граф так, как в рамках одной итерации несколько объектов могут иметь одинаково минимальное расстояние от других

объектов, содержащиеся в уже сформированных кластерах (рисунок П.11).

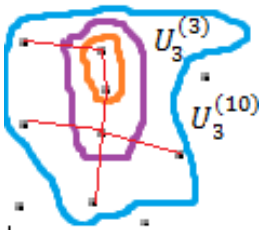

Рисунок П.11 – Формирования кластера с несколькими новыми объектами в ходе одной итерации

Здесь можно отметить ещё одно преимущество агломеративного алгоритма над дивизивным: для агломеративного метода возможно наращивания нескольких объектов в рамках одной итерации. Это связано с тем, что в основе дивизивного метода применяется бинарное разбиение множеств – каждое множество делится на два подмножества и в результате получается бинарное дерево с нисходящим направлением дуг. В агломеративном методе допускается одновременное присоединение нескольких объектов к уже сформированному кластеру, в результате чего может быть получен небинарный граф с восходящим направлением дуг.

Сравнение полученных графов – результатов дивизивного и агломеративного алгоритмов кластеризации.

Экспериментальные исследования показали, что по структуре и содержанию агломеративный метод кластеризации имеет ряд важных преимуществ по сравнению с дивизивным методом:

- существует возможность формирования нескольких кластеров в ходе одной итерации;

- достаточно использовать одну единственную метрику для расчёта меры близости. Главным недостатком агломеративного метода является трудность определения критерия однородности кластеров на уровне одной иерархии [3], и как следствие, необходимость пройти весь путь кластеризации от одиночных кластеров до достижения единого кластера.

П.3. Математическое описание Интернет-ресурсов и их кластеризация методом *k*-средних

В марте 2013 года проводилась экспериментальная кластеризация 98 новостных сайтов по четырем темам: «Президент», «Минобороны», «тайга» и «пробка». В указанный промежуток времени все новостные сайты на ежедневной основе затрагивали указанные выше темы, регулярно публикуя новые статьи.

Пусть $RES = \{res_1, \ldots, res_i, \ldots, res_{nof(RES)}\}$ – множество наблюдаемых ИР. Предположим, что исследуемые ИР являются статическими ресурсами, тогда произвольный $res_i \in RES$ можно представить постоянным характеристическим вектором следующего вида:

$$w_i = (w_{i,1}, \ldots, w_{i,j}, \ldots, w_{i,nof(Vw)}) \in W, \qquad (П.6)$$

где $w_{i,j}$ – вес j-ого поискового термина из глобального словаря терминов V_w, равный числу вхождений этого термина в текст i-го ресурса; $nof(V_w)$ – размер характеристического вектора ИР, равный числу слов в глобальном словаре терминов V_w. Числовые координаты $w_{i,j}$, $1 \le j \le nof(V_w)$ расположены в характеристическом векторе в том же лексикографическом порядке, что и термины в словаре V_w. Переход от вербального к числовому представлению результатов происходит за счет позиционного кодирования терминов и подсчёта числа их вхождений в ИР.

С помощью специально подготовленного тематического словаря и ассоциативного метода исследуемые Интернет-ресурсы были предварительно сгруппированы в четыре кластера и получены первые значения центров кластеров, соответствующих четырем выбранным темам. Таким образом, входным параметром метода *k*-средних явилось первоначальное число кластеров равное 4.

Полученные координаты характеристических векторов представлены в таблице П.1.

Таблица П.1 – Таблица предварительного распределения ИР по темам

Номер Интернет-ресурса *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**
1	5	0	0	0	W_1^0
2	0	0	2	4	W_4^0
4	5	30	2	2	W_2^0
5	0	0	2	2	W_4^0
6	0	0	2	2	W_3^0
7	0	0	2	2	W_4^0
8	0	0	2	2	W_3^0
9	0	18	2	0	W_2^0
11	0	0	2	0	W_3^0
12	0	0	2	0	W_3^0
13	0	0	2	0	W_3^0
14	0	0	0	0	-
16	0	0	0	4	W_4^0
17	0	0	0	0	-
19	6	0	0	0	W_1^0
20	0	0	0	4	W_4^0
21	0	0	0	4	W_4^0
22	0	0	1	0	W_3^0
25	0	0	0	0	-
26	0	9	0	0	W_2^0
31	0	14	0	0	W_2^0
38	0	13	0	0	W_2^0
41	8	0	0	0	W_1^0
48	0	24	0	0	W_2^0
52	0	16	0	0	W_2^0
54	6	11	0	0	W_2^0
62	0	0	0	0	-
64	0	0	0	0	-
68	9	0	0	0	W_1^0
70	0	0	0	4	W_4^0
77	6	0	0	0	W_1^0
83	8	0	0	0	W_1^0
86	0	12	0	0	W_2^0
87	8	0	0	0	W_1^0
88	0	0	0	0	-
89	0	0	0	0	-
93	0	13	0	0	W_2^0
98	13	0	0	0	W_1^0

Как только предварительное распределение выполнено и сформированы первоначальные кластеры (рисунок П.12), можно приступать к первой итерации.

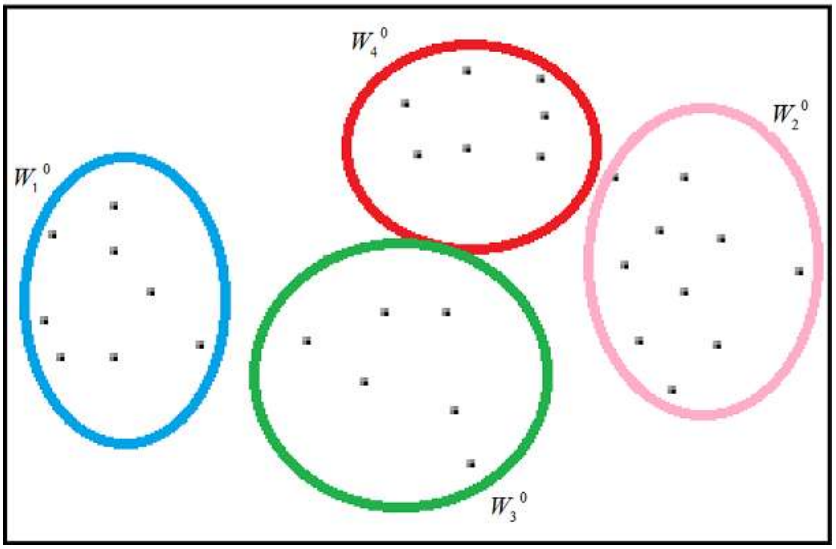

Рисунок П.12 – Иллюстрация предварительного разбиения ИР на 4 кластера

Пусть $S^0 = \{W_1^0, W_2^0, W_3^0, W_4^0\}$ – начальное разбиение, где для $1 \le l \le 4$ $W_l^0 \subset W$, $\bigcup_{l=1}^{4} W_l^0 = W$ и $W_{l'}^0 \cap W_{l''}^0 = \varnothing$. Используя данные таблицы П.1, можно рассчитать набор средних $E^0 = \{e_1^0, e_2^0, e_3^0, e_4^0\}$:

$$e_1^0 = (7.875;0;0;0);\ e_2^0 = (0.1;16;0.4;0.2);\ e_3^0 = (0;0;1.8333;0.666);\ e_4^0 = (0;0;0.857;3.429).$$

Пусть построено m-е разбиение $S^m = \{W_1^m, ..., W_{nof(S^m)}^m\}$. Вычисляем набор средних $E^m = \{e_1^m, ..., e_{nof(E^m)}^m\}$, где $nof(S^m) = nof(E^m)$ и строим минимальное дистанционное разбиение, порождаемое набором E^m, взяв его в качестве базиса для следующего шага итерации $S^{m+1} = \left\{ W_1^{m+1}, ..., W_{nof(S^{m+1})}^{m+1} \right\}$.

На первой итерации берём набор вычисленных средних E^0 и приступаем к формированию следующего по номеру, первого разбиения $S^1 = \{W_1^1, W_2^1, W_3^1, W_4^1\}$. Сформируем первое множество разбиения S^1:

$$W_1^1 = \left\{ w_i : \rho(w_i, e_1^0) = \min_{1 \le l \le 4} \rho(w_i, e_l^0) \right\}.$$

С помощью формулы (П.2) для евклидового расстояния, выявляем «самые

близкие» объекты $w_i \in W$ к центру $e_1^0 = (7.875\ ;0;0;0)$: $\rho(w_i, e_1^0) = \sqrt{\sum_{j=1}^{4} (w_{i,j} - e_{1,j}^0)^2}$ и определяем состав множества W_1^1 (таблица П.2).

Таблица П.2 – Первая таблица расстояний между всеми объектами исследования и набором средних

Номер ИР *i*	**Кластер**	$\rho(w_i, e_1^0)$	$\rho(w_i, e_2^0)$	$\rho(w_i, e_3^0)$	$\rho(w_i, e_4^0)$
41	W_1^0	0.125	17.84965	8.234285	8.745999
83	W_1^0	0.125	17.84965	8.234285	8.745999
87	W_1^0	0.125	17.84965	8.234285	8.745999
68	W_1^0	1.125	18.3142	9.208879	9.669151
19	W_1^0	1.875	17.05902	6.308997	6.963655
77	W_1^0	1.875	17.05902	6.308997	6.963655
1	W_1^0	2.875	16.73947	5.366884	6.123111
98	W_1^0	5.125	20.55748	13.14547	13.47191
22	W_3^0	7.938238	16.01281	1.066511	3.43198
11	W_3^0	8.125	16.08136	0.686619	3.614483
12	W_3^0	8.125	16.08136	0.686619	3.614483
13	W_3^0	8.125	16.08136	0.686619	3.614483
6	W_3^0	8.367534	16.18054	1.344413	1.829888
8	W_3^0	8.367534	16.18054	1.344413	1.829888
5	W_4^0	8.367534	16.18054	1.344413	1.829888
7	W_4^0	8.367534	16.18054	1.344413	1.829888
16	W_4^0	8.832645	16.45023	3.804661	1.029801
20	W_4^0	8.832645	16.45023	3.804661	1.029801
21	W_4^0	8.832645	16.45023	3.804661	1.029801
70	W_4^0	8.832645	16.45023	3.804661	1.029801
2	W_4^0	9.056248	16.52301	3.33818	1.277689
54	W_2^0	11.15866	7.746612	12.68083	13.01893
26	W_2^0	11.95891	7.014984	9.208879	9.669151
86	W_2^0	14.35324	4.026164	12.15744	12.5097
38	W_2^0	15.1992	3.034798	13.14547	13.47191
93	W_2^0	15.1992	3.034798	13.14547	13.47191
31	W_2^0	16.06286	2.051828	14.13518	14.43927
52	W_2^0	17.83299	0.458258	16.11842	16.38574
9	W_2^0	18.11077	2.570992	18.01309	18.35932
48	W_2^0	25.25897	8.013114	24.07911	24.25886
4	W_2^0	30.26988	15.02698	30.44351	30.46881

Из таблицы П.2 видно, что подмножество W_1^1 содержит все те же объекты,

что и множество W_1^0.

Сформируем второе множество:

$$W_2^1 = \left\{ w_i \in W \setminus W_1^1 : \rho\left(w_i, e_2^0\right) = \min_{2 \le l \le 4} \rho\left(w_i, e_l^0\right) \right\}.$$

На этом этапе достаточно рассматривать лишь оставшиеся объекты $w \in W \setminus W_1^1$ и набор средних $\{e_2^0, e_3^0, e_4^0\}$ (таблица П.3).

Таблица П.3 – Вторая таблица расстояний между всеми оставшимися объектами исследования и набором средних

Номер ИР *i*	**Кластер**	$\rho\left(w_i, e_2^0\right)$	$\rho\left(w_i, e_3^0\right)$	$\rho\left(w_i, e_4^0\right)$
52	W_2^0	0.458258	16.11842	16.38574
31	W_2^0	2.051828	14.13518	14.43927
9	W_2^0	2.570992	18.01309	18.35932
38	W_2^0	3.034798	13.14547	13.47191
93	W_2^0	3.034798	13.14547	13.47191
86	W_2^0	4.026164	12.15744	12.5097
26	W_2^0	7.014984	9.208879	9.669151
54	W_2^0	7.746612	12.68083	13.01893
48	W_2^0	8.013114	24.07911	24.25886
4	W_2^0	15.02698	30.44351	30.46881
22	W_3^0	16.01281	1.066511	3.43198
11	W_3^0	16.08136	0.686619	3.614483
12	W_3^0	16.08136	0.686619	3.614483
13	W_3^0	16.08136	0.686619	3.614483
6	W_3^0	16.18054	1.344413	1.829888
8	W_3^0	16.18054	1.344413	1.829888
5	W_4^0	16.18054	1.344413	1.829888
7	W_4^0	16.18054	1.344413	1.829888
16	W_4^0	16.45023	3.804661	1.029801
20	W_4^0	16.45023	3.804661	1.029801
21	W_4^0	16.45023	3.804661	1.029801
70	W_4^0	16.45023	3.804661	1.029801
2	W_4^0	16.52301	3.33818	1.277689

Из таблицы П.3 видно, что множество W_2^1 содержит все те же объекты, что и множество W_2^0.

Формируем третье множество:

$$W_3^1 = \left\{ w_i \in W \setminus \left(W_1^1 \cup W_2^1\right) : \rho(w_i, e_l^0) = \min_{3 \le l \le 4} \rho\left(w_i, e_l^0\right) \right\};$$

На этом этапе достаточно будет рассмотреть оставшиеся объекты $w_i \in W \setminus \left(W_1^1 \cup W_2^1\right)$ и набор средних $\left\{e_3^0, e_4^0\right\}$ (таблица П.4).

Таблица П.4 – Третья таблица расстояний между оставшимися объектами исследования и набором средних

Номер ИР *i*	**Кластер**	$\rho\left(w_i, e_3^0\right)$	$\rho\left(w_i, e_4^0\right)$
22	W_3^0	1.066511	3.43198
11	W_3^0	0.686619	3.614483
12	W_3^0	0.686619	3.614483
13	W_3^0	0.686619	3.614483
6	W_3^0	1.344413	1.829888
8	W_3^0	1.344413	1.829888
5	W_4^0	1.344413	1.829888
7	W_4^0	1.344413	1.829888
16	W_4^0	3.804661	1.029801
20	W_4^0	3.804661	1.029801
21	W_4^0	3.804661	1.029801
70	W_4^0	3.804661	1.029801
2	W_4^0	3.33818	1.277689

Из таблицы П.4 следует, что множество W_3^1 отличается от множества W_3^0 по содержанию объектов w_i: два объекта w_5 и w_7 переходят из множества W_4^0 в множество W_3^1 (переход иллюстрируется рисунок П.13).

Сформируем четвёртое множество разбиения S^1 из оставшихся объектов (таблица П.5):

$$W_4^1 = \left\{ w_i \in W \setminus \left(W_1^1 \cup W_2^1 \cup W_3^1\right) = W_4^0 \setminus \left\{w_5, w_7\right\} \right\}.$$

Процесс выполнения алгоритма кластеризации методом *k*-средних заканчивается в том случае, если выполняется равенство $S^1 = S^0$. Однако, после первой итерации оказалось, что $S^1 \neq S^0$, в связи с изменением составов подмножеств W_3^0 и W_3^1 с одной стороны и W_4^0 и W_4^1, с другой стороны. Поэтому продолжаем кластеризацию, формируя новое разбиение S^2 на основе расчёта множества средних E^1.

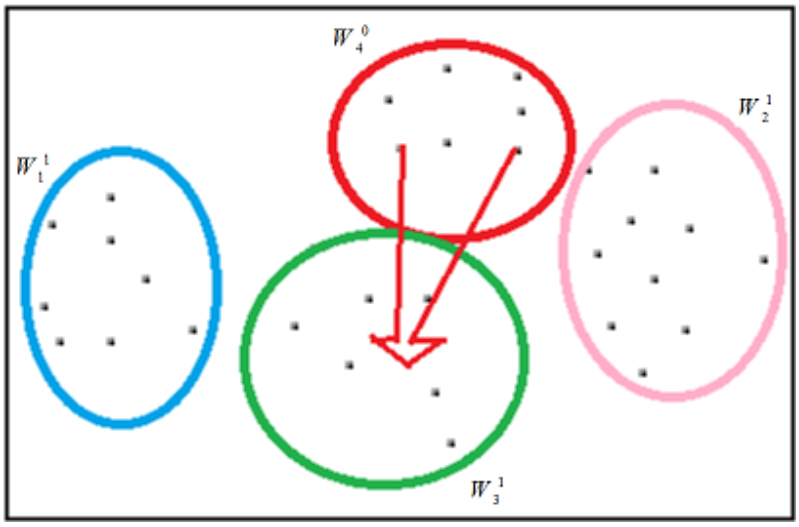

Рисунок П.13 – Иллюстрация перемещения объектов w_5 и w_7 из W_4^0 в W_3^1.

Таблица П.5 – Четвёртая таблица расстояний между всеми оставшимися объектами исследования и набором средних

Номер ИР *i*	**Кластер**	$\rho(w_i, e_4^0)$
16	W_4^0	1.029801
20	W_4^0	1.029801
21	W_4^0	1.029801
70	W_4^0	1.029801
2	W_4^0	1.277689

Продолжая работу с алгоритмом кластеризации метода *k*-средних, переходим ко второй итерации уже с разбиением $S^1 = (W_1^1, W_2^1, W_3^1, W_4^1)$ в качестве входного множества объектов. Можем рассчитать множество новых средних $E^1 = \{e_1^1, e_2^1, e_3^1, e_4^1\}$:

$$e_1^1 = e_1^0 = (7.875;0;0;0) \text{ т.к. } W_1^1 = W_1^0;$$

$$e_2^1 = e_2^0 = (0.1;16;0.4;0.2) \text{ т.к. } W_2^1 = W_2^0;$$

$$e_3^1 = (0;0;1.875;1) \text{ и } e_4^1 = (0;0;0.4;4).$$

Набор средних E^1 получен, приступаем к формированию разбиения $S^2 = (W_1^2, W_2^2, W_3^2, W_4^2)$. Имеем:

$$W_1^2 = \left\{ w_i : \rho(w_i, e_1^1) = \min_{1 \le l \le 4} \rho(w_i, e_l^1) \right\};$$

$$W_2^2 = \left\{ w_i \in W \setminus W_1^2 : \rho\left(w_i, e_l^1\right) = \min_{2 \le l \le 4} \rho\left(w_i, e_l^1\right) \right\};$$

$$W_3^2 = \left\{ w_i \in W \setminus \left(W_1^2 \cup W_2^2\right) : \rho\left(w_i, e_l^1\right) = \min_{3 \le l \le 4} \rho\left(w_i, e_l^1\right) \right\};$$

$$W_4^2 = \left\{ w_i \in W \setminus \left(W_1^2 \cup W_2^2 \cup W_3^2\right) \right\}.$$

Получаем новые результаты расчётов расстояний до средних (таблица П.6).

Таблица П.6 –Таблица расстояний между всеми объектами исследования и средними (вторая итерация)

Номер ИР	**Кластер**	$\rho\left(w_i, e_1^1\right)$	$\rho\left(w_i, e_2^1\right)$	$\rho\left(w_i, e_3^1\right)$	$\rho\left(w_i, e_4^1\right)$
41	W_1^1	0.125	17.84965	8.277417	8.953212
83	W_1^1	0.125	17.84965	8.277417	8.953212
87	W_1^1	0.125	17.84965	8.277417	8.953212
68	W_1^1	1.125	18.3142	9.247466	9.856977
19	W_1^1	1.875	17.05902	6.365189	7.222188
77	W_1^1	1.875	17.05902	6.365189	7.222188
1	W_1^1	2.875	16.73947	5.432828	6.415606
98	W_1^1	5.125	20.55748	13.17253	13.60735
52	W_2^1	17.83299	0.458258	16.1405	16.49727
31	W_2^1	16.06286	2.051828	14.16035	14.56571
9	W_2^1	18.11077	2.570992	18.02819	18.50838
38	W_2^1	15.1992	3.034798	13.17253	13.60735
93	W_2^1	15.1992	3.034798	13.17253	13.60735
86	W_2^1	14.35324	4.026164	12.1867	12.65543
26	W_2^1	11.95891	7.014984	9.247466	9.856977
54	W_2^1	11.15866	7.746612	12.70888	13.15903
48	W_2^1	25.25897	8.013114	24.09389	24.33434
4	W_2^1	30.26988	15.02698	30.4305	30.52147
11	W_3^1	8.125	16.08136	1.007782	4.308132
12	W_3^1	8.125	16.08136	1.007782	4.308132
13	W_3^1	8.125	16.08136	1.007782	4.308132
5	W_3^1	8.367534	16.18054	1.007782	2.56125
6	W_3^1	8.367534	16.18054	1.007782	2.56125
7	W_3^1	8.367534	16.18054	1.007782	2.56125
8	W_3^1	8.367534	16.18054	1.007782	2.56125
22	W_3^1	7.938238	16.01281	1.328768	4.04475
2	W_4^1	9.056248	16.52301	3.002603	1.60000
16	W_4^1	8.832645	16.45023	3.537743	0.40000
20	W_4^1	8.832645	16.45023	3.537743	0.40000
21	W_4^1	8.832645	16.45023	3.537743	0.40000

70	W_4^1	8.832645	16.45023	3.537743	0.40000

В таблице П.6 $W_1^2 = W_1^1$, $W_2^2 = W_2^1$, $W_3^2 = W_3^1$ и $W_4^2 = W_4^1$, то есть состав классов не изменился, следовательно, разбиение $S^2 = S^1$ и условие завершения алгоритма достигнуто: кластеризация выполнена (рисунок П.14)

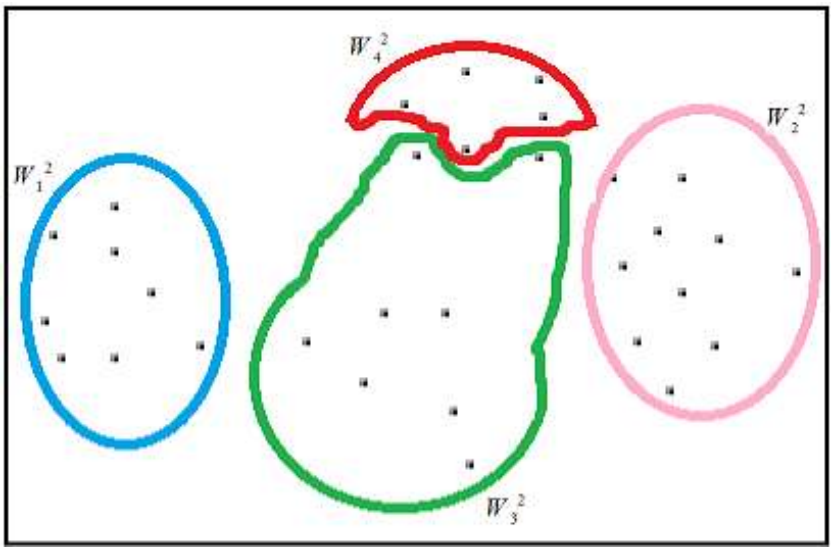

Рисунок П.14 – Иллюстрация финального состояния объектов и их распределения по кластерам

При кластеризации ИР методом *k*-средних итерации выполняются на основании центров тяжести – чем больше элементов в кластере, тем стабильнее становится сам центр тяжести и, как следствие, последующие итерации. Итеративный метод *k*-средних отличается большей степенью трудоемкости, как на вычислительном, так и на алгоритмическом уровне по сравнению с иерархическими методами.

Эффективность алгоритма на прямую зависит от первоначального числа (метод *k*-средних является точным методом) и от их содержания. В ряде случаев начальное разбиение S^0 задается, как минимальное дистанционное разбиение, порожденное некоторым набором точек $E^0 = \left(e_1^0, ..., e_k^0\right)$. Результат классификации зависит от выбора самого E^0: т.е. с разным начальным разбиением, получаем разное E^0 и конечное разбиение S^m. Предварительная классификация объектов (одним из не кластерных методов) приближает значение E^0 к эталонному, в связи с чем пропадает необходимость случайного перебора первоначального числа

кластеров.

В тех случаях, когда из априорных соображений нельзя сразу выбрать число классов k, его находят либо перебором, либо применением алгоритма Форель [3], который позволяет определить примерное число кластеров k для начального разбиения S^0.

П.4. Кластеризации Интернет-ресурсов методом Форель

Начальное множество объектов тот же, что в предыдущем параграфе П.3. Пусть $D_{r^1}(e^1) \subset R^4$ – шар радиуса r^1 с центром в точке e^1 в 4-мерном пространстве R^4. Берём точку e^1 = (5;1;1;1) в качестве центра шара с радиусом r^1 = 5 (рисунок П.15).

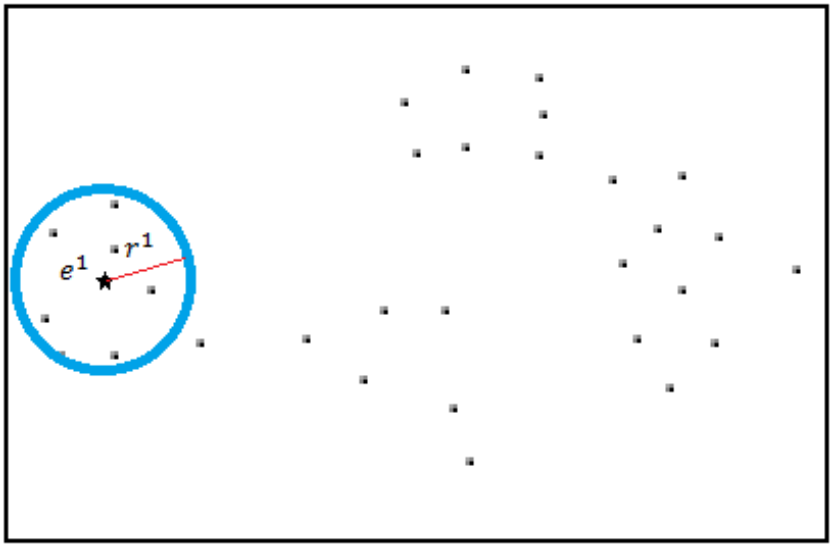

Рисунок П.15 – Иллюстрация первого шара с центром e^1 в методе Форель

В первый шар с центром в e^1 попадают семь объектов w_i, расстояние от которых до центра e^1 меньше радиуса: $\rho(w_i, e^1) \le r^1$. Если сравнивать с методом k-средних (таблица П.6), то кластер W_1 потерял объект w_{98} (таблица П.7).

Таблица П.7 – Расположение объектов в W_1 относительно центра e^1

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^1)$
1	5	0	0	0	W_1	1,732051
19	6	0	0	0	W_1	2,000000
41	8	0	0	0	W_1	3,464102
68	9	0	0	0	W_1	4,358899

77	6	0	0	0	W_1	2,000000
83	8	0	0	0	W_1	3,464102
87	8	0	0	0	W_1	3,464102
98	13	0	0	0	-	8,185353

Найдём координаты центра $e^{1(1)}$ = (7.143;0;0;0). Получаем $e^{1(1)} \neq e^1$, следовательно, выборка $W_1 = W \cap D_{r1}(e^1)$ является смещенной в $D_{r1}(e^1)$. Заменим e^1 на $e^{1(1)}$ в качестве центра первого шара и проведём расчёт расстояний относительно нового центра кластера W_1 (таблица П.8).

Таблица П.8 – Расположение объектов W_1 относительно центра $e^{1(1)}$

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^{1(1)})$
1	5	0	0	0	W_1	2,755440
19	6	0	0	0	W_1	2,075199
41	8	0	0	0	W_1	1,932472
68	9	0	0	0	W_1	2,539380
77	6	0	0	0	W_1	2,075199
83	8	0	0	0	W_1	1,932472
87	8	0	0	0	W_1	1,932472
98	13	0	0	0	-	6,107737

В таблице П.8 видно, что после замены центра шара e^1 на $e^{1(1)}$, состав кластера не изменился и получена несмещенная выборка $W_1 = W \cap D_{r1}(e^{1(1)})$. На этом первый этап алгоритма заканчивается – первый кластер определён.

На втором этапе стоит задача кластеризации объектов из множества $W \setminus W_1$.

Пусть e^2 = (0;15;0;0) – центр второго шара $D_{r2}(e^2)$ с радиусом r^2 = 5. По аналогии с таблицей П.7 выполним расчёт расстояний объектов относительно e^2 — центра кластера W_2.

Таблица П.9 – Расположение объектов в W_2 относительно центра e^2

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^2)$
4	5	30	2	2	-	16,06238
9	0	18	2	0	-	18,11077
26	0	9	0	0	-	6,00000
31	0	14	0	0	W_2	1,00000
38	0	13	0	0	W_2	2,00000
48	0	24	0	0	-	9,00000
52	0	16	0	0	W_2	1,00000
54	6	11	0	0	-	7,211103

86	0	12	0	0	W_2	3,00000
93	0	13	0	0	W_2	2,00000

Если сравнивать результаты таблицы П.9 с результатами метода *k*-средних (таблица П.6) видно, что кластер W_2 потерял 5 объектов, расстояние до центра e^2 которых превышает радиуса шара r^2.

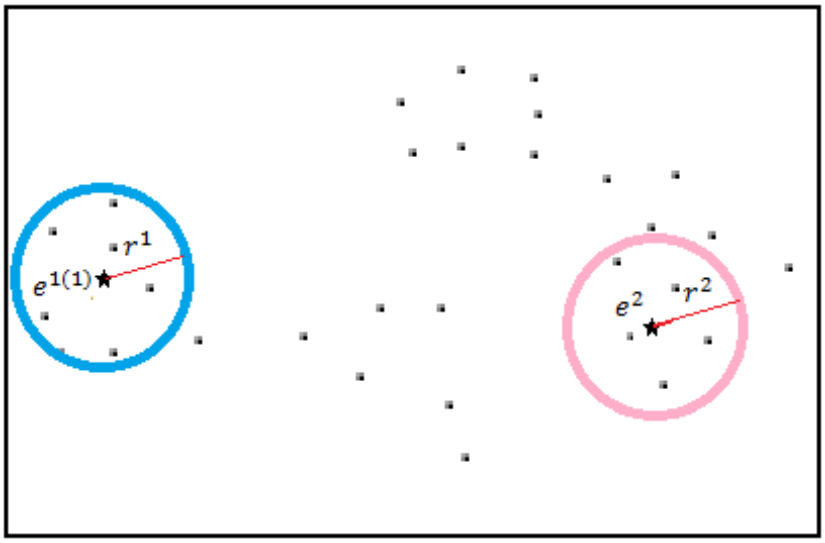

Рисунок П.16 – Иллюстрация первого кластера с центром $e^{1(1)}$ и второго шара с центром e^2 в методе Форель

Найдем координаты центра $e^{2(1)}$ = (0;13,6;0;0). Получаем, что центры не совпадают $e^2 \neq e^{2(1)}$ и, следовательно, выборка $W_2 = W \cap D_{r2}(e^2)$ является смещенной в $D_{r2}(e^2)$.

Необходимо применить рассчитанный средний вектор $e^{2(1)}$ в качестве нового центра второго шара $D_{r2}(e^{2(1)})$ и рассчитать отдалённость объектов относительно нового центра $e^{2(1)}$ кластера W_2 (таблица П.10).

Таблица П.10 – Расположение объектов в W_2 относительно центра $e^{2(1)}$.

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^{2(1)})$
4	5	30	2	2	-	17,37700
9	0	18	2	0	-	18,11077
26	0	9	0	0	W_2	4,60000
31	0	14	0	0	W_2	0,40000
38	0	13	0	0	W_2	0,60000
48	0	24	0	0	-	10,40000
52	0	16	0	0	W_2	2,40000

54	6	11	0	0	-	6,53911
86	0	12	0	0	W_2	1,60000
93	0	13	0	0	W_2	0,60000

Из таблицы П.10 видно, что после замены центра второго шара e^2 на $e^{2(1)}$ состав кластера W_2 изменился (добавился объект w_{26}), поэтому на этом этапе алгоритм не заканчивается. Предстоит новая итерация (таблица П.11) с новым средним вектором $e^{2(2)}$ = (0; 12,833; 0;0).

Таблица П.11 – Расположение объектов в W_2 относительно центра $e^{2(2)}$

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^{2(2)})$
26	0	9	0	0	W_2	3,833
31	0	14	0	0	W_2	1,167
38	0	13	0	0	W_2	0,167
52	0	16	0	0	W_2	3,167
86	0	12	0	0	W_2	0,833
93	0	13	0	0	W_2	0,167

В таблице П.11 после замены центра шара $e^{2(1)}$ на $e^{2(2)}$, состав кластера W_2 не изменился и, следовательно, получена несмещенная выборка $W_2 = W \cap D_{r2}(e^{2(2)})$. На этом второй этап алгоритма заканчивается с определением кластера W_2 (рисунок П.17).

На третьем этапе стоит задача кластеризации объектов из множества $W \setminus (W_1 \cup W_2)$. Пусть e^3 = (0;0;2;1) центр шара $D_{r3}(e^3)$ с радиусом r^3 = 5. Выполним расчёт расстояний от оставшихся объектов до e^3 – центра кластера W_3. (таблица П. 12).

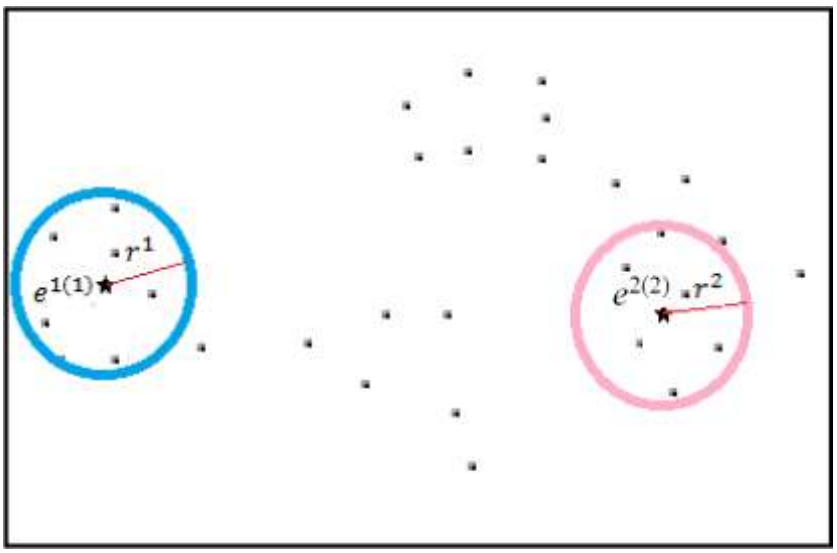

Рисунок П.17 – Иллюстрация второго кластера с центром в $e^{2(2)}$.

Таблица П.12 – Расположение объектов в W_3 относительно центра e^3

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^3)$
2	0	0	2	4	W_3	3,000000
5	0	0	2	2	W_3	1,000000
6	0	0	2	2	W_3	1,000000
7	0	0	2	2	W_3	1,000000
8	0	0	2	2	W_3	1,000000
11	0	0	2	0	W_3	1,000000
12	0	0	2	0	W_3	1,000000
13	0	0	2	0	W_3	1,000000
16	0	0	0	4	W_3	3,605551
20	0	0	0	4	W_3	3,605551
21	0	0	0	4	W_3	3,605551
22	0	0	1	0	W_3	1,414214
70	0	0	0	4	W_3	3,605551

Из таблицы П.12 видно, что кластер W_3 приобрел все 5 объектов кластера W_4, который был получен методом *k*-средних (таблица П.9), то есть кластер W_3 поглотил кластер W_4.

Найдем координаты среднего вектора $e^{3(1)}$ = (0;0;1,308;2,154). Получим $e^3 \neq e^{3(1)}$, следовательно, выборка $W_3 = W \cap D_{r3}(e^3)$ является смещенной. Продолжим расчёт уже с новым центром $e^{3(1)}$ третьего шара и проанализируем новые расстояния (таблица П.13) относительно кластера W_3.

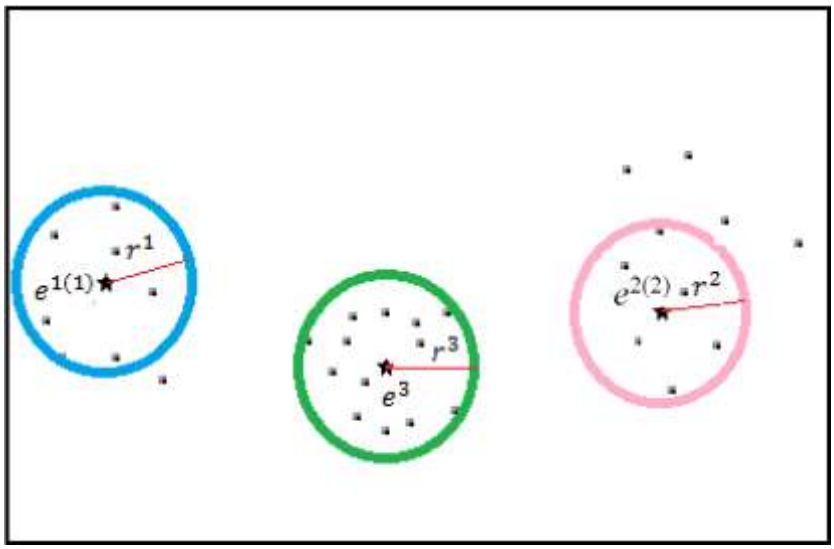

Рисунок П.18 – Иллюстрация третьего шара с центром e^3

в методе Форель

Таблица П.13 – Расположение объектов в W_3 относительно центра $e^{3(1)}$

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^{3(1)})$
2	0	0	2	4	W_4	1,971441
5	0	0	2	2	W_3	0,708929
6	0	0	2	2	W_3	0,708929
7	0	0	2	2	W_3	0,708929
8	0	0	2	2	W_3	0,708929
11	0	0	2	0	W_3	2,262428
12	0	0	2	0	W_3	2,262428
13	0	0	2	0	W_3	2,262428
16	0	0	0	4	W_4	2,262428
20	0	0	0	4	W_4	2,262428
21	0	0	0	4	W_4	2,262428
22	0	0	1	0	W_3	2,175909
70	0	0	0	4	W_4	2,262428

Из таблицы П.13 следует, что после замены центра шара e^3 на $e^{3(1)}$, состав кластера не изменился и получена несмещенная выборка $W_3 = W \cap D_r\left(e^{3(1)}\right)$. На этом третий этап алгоритма заканчивается с определением кластера W_3 (рисунок П.19).

На четвёртом этапе стоит задача кластеризации объектов $W \setminus (W_1 \cup W_2 \cup W_3)$. Пусть e^4 = (2;20;1;1) – центр шара с радиусом r^4 = 5. Кластер W_4 содержит всего два объекта – w_{48} и w_{52} (таблица П.14).

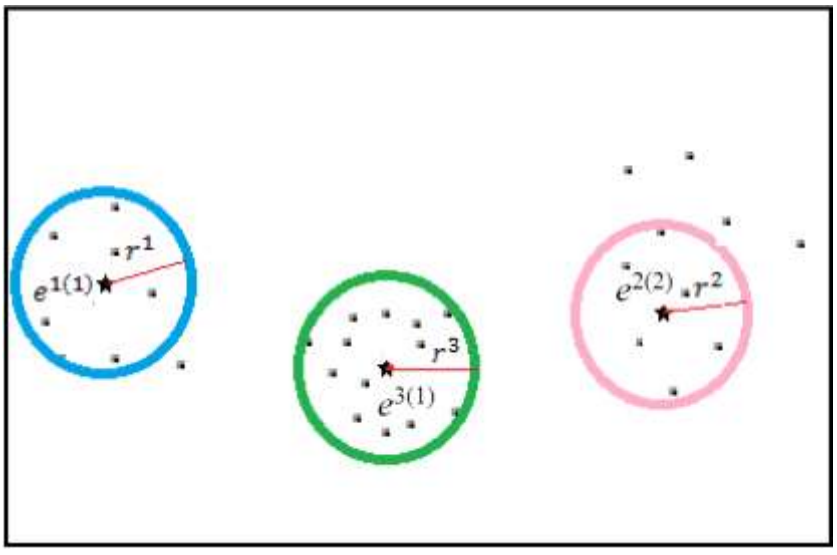

Рисунок П.19 – Формирование третьего кластера с центром $e^{3(1)}$

Таблица П.14 – Расположение объектов в W_4 относительно центра e^4

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^4)$
48	0	24	0	0	W_4	4,690416
52	0	16	0	0	W_4	4,690416

Найдя координаты среднего вектора $e^{4(1)}$ = (0;20;0;0), получим, что $e^4 \neq e^{4(1)}$ и, следовательно, выборка $W_4 = W \cap D_{r4}(e^4)$ является смещенной в $D_{r4}(e^4)$. Для получения несмещенной выборки возьмём точку $e^{4(1)}$ в качестве центра четвёртого шара и проанализируем новый состав кластера W_4 (таблица П.15).

Таблица П.15 – Расположение объектов в W_4 относительно центра $e^{4(1)}$

Номер ИР *i*	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	**Кластер**	$\rho(w_i, e^{4(1)})$
48	0	24	0	0	W_4	4,0
52	0	16	0	0	W_4	4,0

После замены центра сферы e^4 на $e^{4(1)}$ состав кластера W_4 не изменился и, следовательно, получена несмещенная подвыборка $W_4 = W \cap D_{r4}(e^{4(1)})$ в $D_{r4}(e^{4(1)})$ (рисунок П.20).

На пятом этапе стоит задача кластеризации объектов из $W_5 = W \setminus (W_1 \cup W_2 \cup W_3 \cup W_4) = \{w_4, w_9, w_{98}\}$ – множество объектов, оставшихся вне кластерной структуры.

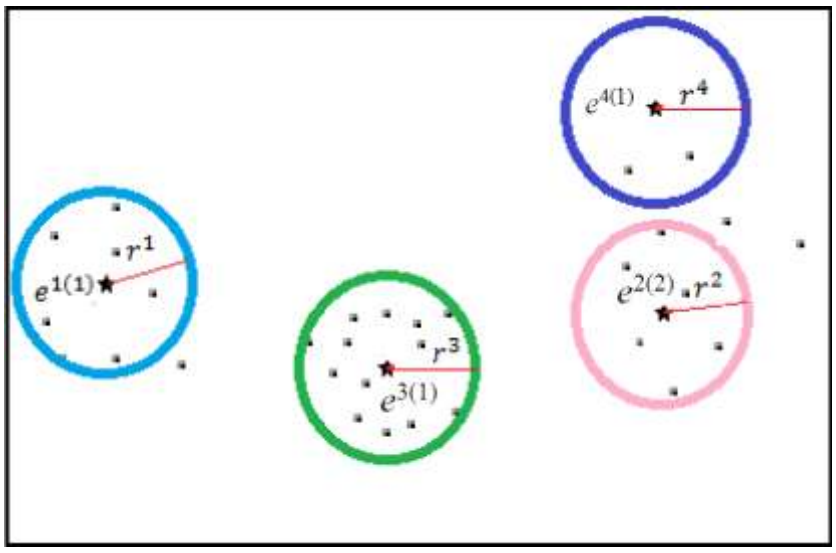

Рисунок П.20 – Формирование четвёртого кластера с центром в $e^{4(1)}$

Пусть e^5 = (5;15;2;1) центр шара с радиусом r^5 = 5. К сожалению, не удается сформировать кластер на основании 2-х и более объектов. Проверка меры

близости между оставшимися «некластеризованными» объектами показала, что минимальное расстояние от точки w_i до остальных двух точек $\rho_{min}(w_i)$ в множестве W^5 больше $2 \times r^5$: $\rho_{min}(w_4) = 13.15295 > 10$; $\rho_{min}(w_9) = 18.22087 > 10$; $\rho_{min}(w_{98}) = 22.29349 > 10$. На пятом этапе кластеризация заканчивается. В результате получаем 4 кластера, число объектов в которых составляет не менее двух, и три одиночных объекта-кластера $\{w_4\}$, $\{w_9\}$ и $\{w_{98}\}$.

Применение алгоритма Форель позволяет оценить наиболее предпочтительное число классов для заданной выборки. При этом основанием для выбора числа классов может служить многократное повторение одного и того же числа классов для нескольких последовательных значений r и его резкое возрастание для следующего шага, что и станет сигналом для фиксации оптимального числа кластеров. На основе алгоритма первого этапа метода Форель строится целое семейство алгоритмов [3], целью которых является разбиение выборки на заданное число классов, покрытие выборки W областями более сложной формы, чем шары и т. п. Подробное описание этого семейства можно найти в [22]. Имеется модификация алгоритма первого этапа метода Форель, в которой порог r является параметром, настраиваемым в ходе поиска первого кластера (например, алгоритм Пульсар § 7.3.1 в [3]).

ПРИЛОЖЕНИЕ 4. РЕАЛИЗАЦИЯ КОМБИНИРОВАННОЙ И ОБОБЩЕННОЙ КЛАСТЕРИЗАЦИИ С ПОМОЩЬЮ *SQL*-СКРИПТА

```
-- STEP2:
-- Находим ИР с упоменанием слов из поискового словаря ИП
IF (OBJECT_ID('tempdb..#pages') IS NOT NULL) DROP TABLE #pages
SELECT TOP(3) V.page_id
INTO #pages
FROM
[HTML].[dbo].[HTML_value] V
INNER JOIN
dbo.az_words W
ON V.HTML_element_value like '%'+W.word+'%'
INNER JOIN
#temp_w_dictionary D
ON D.word = W.word
GROUP BY V.page_id
ORDER BY COUNT(*) DESC

-- формирование славоря для ИР.
IF (OBJECT_ID('tempdb..#page_str') IS NOT NULL) DROP TABLE #page_str
SELECT A.[page_id], A.[HTML_element_value]
  INTO #page_str
  FROM [HTML].[dbo].[HTML_value] A
  where page_id in (SELECT page_id FROM #pages)

UPDATE #page_str
SET [HTML_element_value] = REPLACE([HTML_element_value],'.',' ')
UPDATE #page_str
SET [HTML_element_value] = REPLACE([HTML_element_value],',',' ')
UPDATE #page_str
SET [HTML_element_value] = REPLACE([HTML_element_value],'-',' ')
UPDATE #page_str
SET [HTML_element_value] = REPLACE([HTML_element_value],')',' ')
UPDATE #page_str
SET [HTML_element_value] = REPLACE([HTML_element_value],'  ',' ')

IF(OBJECT_ID('tempdb..#page_w_dictionary')    is    not    null)    DROP    TABLE
#page_w_dictionary
SELECT DISTINCT B.Item as page_w
INTO #page_w_dictionary
FROM #page_str A CROSS APPLY dbo.az_split(A.[HTML_element_value]+'t', ' ') B
WHERE B.Item not like '%/%' AND LEN(B.Item)>3 AND B.Item NOT LIKE '%&%'
AND B.Item not like '%>%' AND B.Item not like '%"%' AND B.Item not like '%|%'
AND B.Item not like '%«%'
GO
IF (OBJECT_ID('tempdb..#temp_page_w') is not null) DROP TABLE #temp_page_w
SELECT P.page_id, PD.page_w, COUNT(*) as cnt
INTO #temp_page_w
FROM #page_str P INNER JOIN #page_w_dictionary PD
ON P.HTML_element_value like '%' + PD.page_w + '%'
GROUP BY P.page_id, PD.page_w

-- Формирование таблицы комбинаций
IF    (OBJECT_ID('tempdb..#page_combination')    is    not    null)    DROP    table
#page_combination
CREATE TABLE #page_combination  (ip1 smallint not null, ip2 smallint not null, ER
FLOAT)
DECLARE @ip1 smallint, @ip2 smallint;
declare curs5 cursor local fast_forward for
select P1.page_id as ip1,  P2.page_id as ip2 FROM #pages P1
```

```
INNER JOIN #pages P2
ON P1.page_id <> P2.page_id
WHERE NOT EXISTS
(
      select ip1, ip2 FROM #page_combination PC WHERE
      (PC.ip1 = P1.page_id AND PC.ip2 = P2.page_id)
      OR
      (PC.ip2 = P1.page_id AND PC.ip1 = P2.page_id)
)
OPEN curs5
FETCH curs5 into @ip1, @ip2
WHILE @@FETCH_STATUS = 0
BEGIN
      INSERT #page_combination (ip1, ip2)
      VALUES (@ip1, @ip2)
      FETCH curs5 into @ip1, @ip2
END
CLOSE curs5
DEALLOCATE curs5
GO


-- расчёт евклидово расстояние
DECLARE @ip1 smallint, @ip2 smallint;
DECLARE curs6 cursor local FAST_FORWARD FOR
SELECT ip1, ip2 FROM #page_combination
OPEN curs6
FETCH curs6 INTO @ip1, @ip2
WHILE @@FETCH_STATUS = 0
BEGIN
      IF (OBJECT_ID('tempdb..#ER_FINAL') is not null) DROP TABLE #ER_FINAL
      SELECT page_ip1 = IP1.page_id, word_ip1 = IP1.page_w, IP1.cnt as cnt_ip1,
             resp_ip2 = IP2.page_id, word_ip2 = IP2.page_w, IP2.cnt as cnt_ip2,
             x_diffsqr = SQUARE(IP1.cnt - IP2.cnt)
      INTO #ER_FINAL
      FROM
      (
      (
      select page_id = @ip1, B.page_w, coalesce(A.cnt,0) as cnt from
            (
                  SELECT page_id, page_w, cnt FROM #temp_page_w where page_id =
@ip1
            ) A
            RIGHT JOIN
            #page_w_dictionary B
            ON A.page_w = B.page_w
      ) AS IP1
      inner join
      (
            select page_id = @ip2, B.page_w, coalesce(A.cnt,0) as cnt from
            (
                  SELECT page_id, page_w, cnt FROM #temp_page_w where page_id =
@ip2
            ) A
            RIGHT JOIN
            #page_w_dictionary B
            ON A.page_w = B.page_w
      ) AS IP2
      ON IP1.page_w = IP2.page_w
      );
      DECLARE @ER int;
      SELECT @ER = SQRT(SUM(x_diffsqr))
      FROM #ER_FINAL ERF;
```

```
      UPDATE #page_combination SET ER = @ER
      WHERE
      (ip1 = @ip1 AND ip2 = @ip2)
      OR
      (ip1 = @ip2 AND ip2 = @ip1);

      SET @ER = NULL;
      FETCH curs6 INTO @ip1, @ip2;
END
CLOSE curs6
DEALLOCATE curs6

select * from #page_combination
-- STEP3: обобщённая кластеризация
-- Формируем обобщённый словарь.
IF    (OBJECT_ID('tempdb..#total_dictionary')    IS    NOT    NULL)    DROP    TABLE
#total_dictionary
SELECT word = page_w
INTO #total_dictionary
FROM #page_w_dictionary
UNION
SELECT word FROM #temp_w_dictionary

IF (OBJECT_ID('tempdb..#temp_obj_w') is not null) DROP TABLE #temp_obj_w
SELECT obj_id = P.page_id, TD.word, COUNT(*) as cnt
INTO #temp_obj_w
FROM #page_str P INNER JOIN #total_dictionary TD
ON P.HTML_element_value like '%' + TD.word + '%'
GROUP BY P.page_id, TD.word
UNION
SELECT obj_id = R.resp_id, TD.word, COUNT(*) as cnt
FROM #temp_resp_w R INNER JOIN #total_dictionary TD
ON R.word like '%' + TD.word + '%'
GROUP BY R.resp_id, TD.word
-- Формирование таблицы комбинаций
IF (OBJECT_ID('tempdb..#obj_combination') is not null) DROP table #obj_combination
CREATE TABLE #obj_combination  (obj1 smallint not null, obj2 smallint not null, ER
FLOAT)
DECLARE @obj1 smallint, @obj2 smallint;
declare curs7 cursor local fast_forward for
select obj1.obj_id as obj1,  obj2.obj_id as obj2 FROM
(SELECT DISTINCT obj_id FROM #temp_obj_w) obj1
INNER JOIN
(SELECT DISTINCT obj_id FROM #temp_obj_w) obj2
ON obj1.obj_id <> obj2.obj_id
WHERE NOT EXISTS
(
      select obj1, obj2 FROM #obj_combination OC WHERE
      (OC.obj1 = obj1.obj_id AND OC.obj2 = obj2.obj_id)
      OR
      (OC.obj2 = obj1.obj_id AND OC.obj1 = obj2.obj_id)
)
OPEN curs7
FETCH curs7 into @obj1, @obj2
WHILE @@FETCH_STATUS = 0
BEGIN
      INSERT #obj_combination (obj1, obj2)
      VALUES (@obj1, @obj2)
      FETCH curs7 into @obj1, @obj2
END
CLOSE curs7
DEALLOCATE curs7
```

```
GO
-- расчёт евклидово расстояние
DECLARE @obj1 smallint, @obj2 smallint;
DECLARE curs8 cursor local FAST_FORWARD FOR
SELECT obj1, obj2 FROM #obj_combination
OPEN curs8
FETCH curs8 INTO @obj1, @obj2
WHILE @@FETCH_STATUS = 0
BEGIN
      IF   (OBJECT_ID('tempdb..#ER_FINAL_TOTAL')   is   not   null)   DROP   TABLE
#ER_FINAL_TOTAL
      SELECT obj_ip1 = obj1 .obj_id, word_obj1 = obj1.word, obj1.cnt as cnt_obj1,
             obj_ip2 = obj2.obj_id, word_obj2 = obj2.word, obj2.cnt as cnt_obj2,
             x_diffsqr = SQUARE(obj1.cnt - obj2.cnt)
      INTO #ER_FINAL_TOTAL
      FROM
      (
      (
      select obj_id = @obj1, B.word, coalesce(A.cnt,0) as cnt from
            (
                  SELECT obj_id, word, cnt FROM #temp_obj_w where obj_id = @obj1
            ) A
            RIGHT JOIN
            #total_dictionary B
            ON A.word = B.word
      ) AS obj1
      inner join
      (
            select obj_id = @obj2, B.word, coalesce(A.cnt,0) as cnt from
            (
                  SELECT obj_id, word, cnt FROM #temp_obj_w where obj_id = @obj2
            ) A
            RIGHT JOIN
            #total_dictionary B
            ON A.word = B.word
      ) AS obj2
      ON obj1.word = obj2.word
      );
      DECLARE @ER int;
      SELECT @ER = SQRT(SUM(x_diffsqr))
      FROM #ER_FINAL_TOTAL ERF;

      UPDATE #obj_combination SET ER = @ER
      WHERE
      (obj1 = @obj1 AND obj2 = @obj2)
      OR
      (obj1 = @obj2 AND obj2 = @obj1);

      SET @ER = NULL;
      FETCH curs8 INTO @obj1, @obj2;
END
CLOSE curs8
DEALLOCATE curs8

SELECT * FROM #obj_combination
```

ПРИЛОЖЕНИЕ 5. РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ ПРИМЕНЕНИЯ МЕТОДОВ КОМБИНИРОВАННОЙ И ОБОБЩЕННОЙ КЛАСТЕРИЗАЦИИ

	комбинированная кластеризация								*обобщённая кластеризация*							
	min_U	***max_U***	***out_U***	***min_R***	***max_R***	***out_R***	***min_UR***	***max_UR***	***min_U***	***max_U***	***out_U***	***min_R***	***max_R***	***out_R***	***min_UR***	***max_UR***
IU2IR3	28	28	0	63	121	0	91	149	28	28	0	63	121	0	93	169
IU2IR5	28	28	0	53	116	0	99	164	10	10	0	49	95	0	71	136
IU2IR10	28	28	0	29	124	0	91	165	10	10	0	24	99	0	63	137
IU2IR15	28	28	0	30	222	0	45	167	10	10	0	24	142	0	17	142
IU2IR20	28	28	0	26	222	0	45	167	10	10	0	24	142	0	17	139
IU3IR3	9	28	0	63	121	0	86	164	5	10	0	58	95	0	77	136
IU3IR5	9	28	0	53	116	0	80	164	5	10	0	49	95	0	71	136
IU3IR10	9	28	0	29	124	0	76	167	5	10	0	24	99	0	67	139
IU3IR15	9	28	0	30	222	0	25	167	5	10	0	24	142	0	16	139
IU3IR20	9	28	0	26	222	0	25	167	5	10	0	24	142	0	16	139
IU5IR3	3	28	1	63	121	0	81	164	3	10	1	58	95	0	78	136
IU5IR5	3	28	1	53	116	0	80	164	3	10	1	49	95	0	71	136
IU5IR10	3	28	1	29	124	0	66	165	3	10	1	24	99	0	63	137
IU5IR15	3	28	1	30	222	0	19	167	3	10	1	24	142	0	16	139
IU5IR20	3	28	1	26	222	0	19	167	3	10	1	24	142	0	16	139
IU10IR3	3	31	1	63	121	0	80	167	3	11	1	55	95	0	77	136
IU10IR5	3	31	1	53	116	0	71	167	3	11	1	46	95	0	68	136
IU10IR10	3	31	1	29	124	0	72	168	3	11	1	36	99	0	69	137
IU10IR15	3	31	1	30	222	0	66	191	3	11	1	28	128	0	63	139
IU10IR20	3	31	1	26	222	0	43	193	3	11	1	28	163	0	40	162
IU15IR3	2	32	1	63	121	0	80	168	3	11	1	57	96	0	78	136
IU15IR5	2	32	1	53	116	0	81	169	3	11	1	58	99	0	79	137

IU15IR10	2	32	1	29	124	0	71	192	3	11	1	36	128	0	69	160
IU15IR15	2	32	1	30	222	0	49	194	3	11	1	35	163	0	47	162
IU15IR20	2	32	1	26	222	0	49	194	3	11	1	30	179	0	47	163
IU20IR3	3	43	4	63	121	0	81	179	3	14	2	57	95	0	78	136
IU20IR5	3	43	4	53	116	0	81	180	3	14	2	58	99	0	79	137
IU20IR10	3	43	4	29	124	0	50	205	3	14	2	41	163	0	47	163
IU20IR15	3	43	4	30	222	0	51	205	3	14	2	37	163	0	48	163
IU20IR20	3	43	4	26	222	0	51	205	3	14	2	38	177	0	40	163

	min_U_comb	*min_U_un*	*max_U_comb*	*max_U_un*	*delta_U_comb*	*delta_U_un*
IU2IR3	28	28	28	28	0	0
IU2IR5	28	10	28	10	0	0
IU2IR10	28	10	28	10	0	0
IU2IR15	28	10	28	10	0	0
IU2IR20	28	10	28	10	0	0
IU3IR3	9	5	28	10	19	5
IU3IR5	9	5	28	10	19	5
IU3IR10	9	5	28	10	19	5
IU3IR15	9	5	28	10	19	5
IU3IR20	9	5	28	10	19	5
IU5IR3	3	3	28	10	25	7
IU5IR5	3	3	28	10	25	7
IU5IR10	3	3	28	10	25	7
IU5IR15	3	3	28	10	25	7
IU5IR20	3	3	28	10	25	7
IU10IR3	3	3	31	11	28	8
IU10IR5	3	3	31	11	28	8
IU10IR10	3	3	31	11	28	8
IU10IR15	3	3	31	11	28	8
IU10IR20	3	3	31	11	28	8
IU15IR3	2	3	32	11	30	8
IU15IR5	2	3	32	11	30	8
IU15IR10	2	3	32	11	30	8
IU15IR15	2	3	32	11	30	8
IU15IR20	2	3	32	11	30	8
IU20IR3	3	3	43	14	40	11
IU20IR5	3	3	43	14	40	11
IU20IR10	3	3	43	14	40	11
IU20IR15	3	3	43	14	40	11
IU20IR20	3	3	43	14	40	11

	min_R_comb	*min_R_un*	*max_R_comb*	*max_R_un*	*delta_R_comb*	*delta_R_un*
IU2IR3	63	63	121	121	58	58
IU2IR5	53	49	116	95	63	46
IU2IR10	29	24	124	99	95	75
IU2IR15	30	24	222	142	192	118
IU2IR20	26	24	222	142	196	118
IU3IR3	63	58	121	95	58	37
IU3IR5	53	49	116	95	63	46
IU3IR10	29	24	124	99	95	75
IU3IR15	30	24	222	142	192	118
IU3IR20	26	24	222	142	196	118
IU5IR3	63	58	121	95	58	37
IU5IR5	53	49	116	95	63	46
IU5IR10	29	24	124	99	95	75
IU5IR15	30	24	222	142	192	118
IU5IR20	26	24	222	142	196	118
IU10IR3	63	55	121	95	58	40
IU10IR5	53	46	116	95	63	49
IU10IR10	29	36	124	99	95	63
IU10IR15	30	28	222	128	192	100
IU10IR20	26	28	222	163	196	135
IU15IR3	63	57	121	96	58	39
IU15IR5	53	58	116	99	63	41
IU15IR10	29	36	124	128	95	92
IU15IR15	30	35	222	163	192	128
IU15IR20	26	30	222	179	196	149
IU20IR3	63	57	121	95	58	38
IU20IR5	53	58	116	99	63	41
IU20IR10	29	41	124	163	95	122
IU20IR15	30	37	222	163	192	126
IU20IR20	26	38	222	177	196	139

	min_UR_comb	*min_UR*	*max_UR_comb*	*max_UR*	*delta_UR_comb*	*delta_UR_un*
IU2IR3	91	93	149	169	58	76
IU2IR5	99	71	164	136	65	65
IU2IR10	91	63	165	137	74	74
IU2IR15	45	17	167	142	122	125
IU2IR20	45	17	167	139	122	122
IU3IR3	86	77	164	136	78	59
IU3IR5	80	71	164	136	84	65
IU3IR10	76	67	167	139	91	72
IU3IR15	25	16	167	139	142	123
IU3IR20	25	16	167	139	142	123
IU5IR3	81	78	164	136	83	58
IU5IR5	80	71	164	136	84	65
IU5IR10	66	63	165	137	99	74
IU5IR15	19	16	167	139	148	123
IU5IR20	19	16	167	139	148	123
IU10IR3	80	77	167	136	87	59
IU10IR5	71	68	167	136	96	68
IU10IR10	72	69	168	137	96	68
IU10IR15	66	63	191	139	125	76
IU10IR20	43	40	193	162	150	122
IU15IR3	80	78	168	136	88	58
IU15IR5	81	79	169	137	88	58
IU15IR10	71	69	192	160	121	91
IU15IR15	49	47	194	162	145	115
IU15IR20	49	47	194	163	145	116
IU20IR3	81	78	179	136	98	58
IU20IR5	81	79	180	137	99	58
IU20IR10	50	47	205	163	155	116
IU20IR15	51	48	205	163	154	115
IU20IR20	51	40	205	163	154	123

ПРИЛОЖЕНИЕ 6. ИСХОДНЫЙ КОД ПРОГРАММНОГО МОДУЛЯ

internet_res_search

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
//Подключаем библиотеку для работы с файлами.
using System.IO;

namespace internet_res_search
{
    public partial class Form1 : Form
    {
        //Пустая переменная названия файла.
        string Fname4Save = string.Empty;
        //Переменная о состоянии сохранения файла.
        bool NotSaved = false;

        public Form1()
        {
            InitializeComponent();
            wb.StatusTextChanged += new EventHandler(wb_StatusTextChanged);
        }

        //Функция для отражения изменений в statusbar. Подоюно Iexplorer.
        void wb_StatusTextChanged(object sender, EventArgs e)
        {
            toolStripStatusLabel1.Text = wb.StatusText;
        }

        //Функция для сброса файлов cookie.
        private void ClearCookie()
        {
            string cPath =
System.Environment.GetFolderPath(Environment.SpecialFolder.Cookies);
            System.IO.DirectoryInfo di = new
System.IO.DirectoryInfo(System.Environment.GetFolderPath(Environment.SpecialFolder.Cookies))
;
            foreach (FileInfo file in di.GetFiles("*.txt"))
            {
                file.Delete();
            }
        }

        //Обрабочик загруэки формы интерфейса программы.
        private void Form1_Load(object sender, EventArgs e)
        {
            //Вызываем функцию сброса cookie.
            ClearCookie();
        }

        //Обрабочик закрытия формы интерфейса программы.
        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
```

```
            if (NotSaved)
            {
                DialogResult dr = MessageBox.Show("Есть не сохраненные данные, Сохранить?",
                    "Внимание!", MessageBoxButtons.YesNoCancel, MessageBoxIcon.Warning);
                if (dr == DialogResult.Yes)
                {
                    //Вызываем функцию сохранения файла.
                    Save();
                }
                else if (dr == DialogResult.No)
                {
                    e.Cancel = false;
                }
                else if (dr == DialogResult.Cancel)
                {
                    e.Cancel = true;
                }
            }
        }

        //Функция для сохранения файла.
        private void Save()
        {
            saveFileDlg.FileName = Fname4Save;
            saveFileDlg.DefaultExt = "txt";
            saveFileDlg.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*";
            if (saveFileDlg.ShowDialog() == DialogResult.OK)
            {
                StreamWriter w = new StreamWriter(saveFileDlg.FileName, false,
Encoding.Default);
                foreach (string str in textBox3.Lines)
                {
                    //Записываем строку в файл.
                    w.WriteLine(str);
                }
                w.Close();
                NotSaved = false;
            }

        }

        //Функция для обработки события html-документ загружен.
        private void wb_DocumentCompleted(object sender,
WebBrowserDocumentCompletedEventArgs e)
        {
            //statusbar - индикатор "Готово"
            toolStripStatusLabel1.Text = "Готово";
            //Записываем URL-страницы на которой находится WebBrowser.
            textBox1.Text = wb.Url.ToString();
            //Получаем индекс страницы в поисковике с помощью функции html_page_index().
            int current_page_idx = Convert.ToInt16(html_page_index(wb.Url.ToString())[0]);
            //Получаем количество страниц, необходимое для сканирования.
            int user_idx = Convert.ToInt16(textBoxEnters.Text);
            //Извлекаем URL-страницы на которой находится WebBrowser.
            string wb_str = wb.Url.ToString();
            bool t = false;
            //Проверка URL и номер страницы.
            if (wb_str.StartsWith("http://yandex.ru/yandsearch?") && (current_page_idx <
user_idx))
            {
                // Небольшая пауза.
                System.Threading.Thread.Sleep(1000);
                //Объявляем переменную html-документ.
```

```
                HtmlDocument htmlDoc = wb.Document;
                //Обявляем переменную HtmlElementCollection.
                HtmlElementCollection hec = htmlDoc.GetElementsByTagName("a");
                //Проводим анализ содержания htmlElement-ов.
                foreach (HtmlElement he in hec)
                {
                    //Проверка на наличие гиперссылок в результате поиска.
                    if ((he.GetAttribute("href").ToString().Length > 0) &&
(he.OuterHtml.Contains("b-serp-item__title-link")))
                    {
                        //Переменная для устранения повторных гипперссылок.
                        //проверяем все уже имеющиеся строки в списке на предмет совпадения
с новой гиперссылкой.
                        foreach (string str in textBox3.Lines)
                            if ("-" + he.GetAttribute("href").ToString()+";" == str) t =
true;
                        //Если не было совподения добавляем новую гиперссылку в наш список.
                        if (t == false)
                        {
                            textBox3.AppendText("-" +
he.GetAttribute("href").ToString()+";");
                            //Переходим на следующую строку.
                            textBox3.Text += Environment.NewLine;
                        }
                    }
                }
                //Вызываем функцию перехода на следующую страницу.
                next_page();
            }
        }

        //Функция для определения индекса текущей страницы и URL следующей.
        private object[] html_page_index(string http_str)
        {
            //Проводим поиск номера страницы поиска.
            if (http_str.IndexOf("?p=") < 0)
            {
                http_str = "http://yandex.ru/yandsearch?p=0&text=" + textBox2.Text +
"&lr=213";
            }
            //Считывание начальной позиции номера текущей страницы.
            int pg_index = http_str.IndexOf("?p=") + 3;
            //Получаем второй кусок URL.
            string http_str_cut = http_str.Substring(pg_index, http_str.Length - pg_index);
            //Считывание конечной позиции номера текущей страницы.
            int pg_index_len = http_str_cut.IndexOf('&');
            //Получаем номер страницы поиска в виде строки.
            string html_page_index_str = http_str.Substring(pg_index, pg_index_len);
            //Преобразуем строку в число.
            int html_page_index = Convert.ToInt16(html_page_index_str);
            //Формируем URL следущей страницы поиска.
            string http_new = http_str.Substring(0, pg_index) + (html_page_index +
1).ToString() + http_str_cut.Substring(pg_index_len);
            //Записываем индекс текущей (число int) и URL следующей страницы (строка string)
в массив типа object[].
            object[] idx = {html_page_index, http_new};
            //Выводим результат функции.
            return  idx;
        }

        //Обрабочик для нажатия кнопки "Поиск"
        private void startButton_Click(object sender, EventArgs e)
        {
```

```
            NotSaved = true;
            //Выбран первый метод навигации по поиску (с имитацией).
            if (radioButtonMethod1.Checked)
            {
                HtmlElementCollection input_tag_collection =
wb.Document.GetElementsByTagName("input");
                //Находим htmlElement поисковой строки.
                foreach (HtmlElement input in input_tag_collection)
                {
                    if ((input.Name == "text"))
                    {
                        //Записываем искомое вырожение из textBox2 в htmlElement поисковой
строки.
                        input.InnerText = textBox2.Text;
                    }
                }
                //Находим htmlElement кнопки найти.
                foreach (HtmlElement input in input_tag_collection)
                {
                    if (input.OuterHtml.Contains("b-form-button__input") &&
input.GetAttribute("tabIndex") == "2")
                    /*Раньше можно было по другому определить эту кнопку*/
                    /*(input.GetAttribute("value").ToString() == "Найти"))*/
                    {
                        //Имитируем нажатия кнопки "Найти".
                        input.InvokeMember("click");
                    }

                }
            }
            //Выбран второй метод навигации с применением URL.
            else if (radioButtonMethod2.Checked)
            {
                //В URL-строки уже имеется искомое слово из textBox2.
                wb.Navigate("http://yandex.ru/yandsearch?text="+textBox2.Text+"&lr=213");
            }

        }

        //Функция для перехода на следующую страницу поиска.
        void next_page()
        {
            //Объявляем переменную html-документ.
            HtmlDocument htmlDoc = wb.Document;
            //Извлекаем URL-страницы на которой находится WebBrowser.
            string wb_str = wb.Url.ToString();
            foreach(HtmlElement he in htmlDoc.GetElementsByTagName("a"))
            //Переходим на следующую страницу результата поиска по первому методу.
            if ((he.InnerHtml == "следующая") && (radioButtonMethod1.Checked))
            {
                //Имитация нажатия гиперссылки "next page".
                he.InvokeMember("click");
            }
            //Переходим на следующую страницу результата поиска по второму методу.
            else if (radioButtonMethod2.Checked)
            {
                //Получаем индекс текущей страницы поиска с помощью функции
html_page_index().
                int p = Convert.ToInt16(html_page_index(wb_str)[0]);
                //Получаем URL следующей страницы поиска с помощью функции
html_page_index().
                string next_URL = html_page_index(wb_str)[2].ToString();
```

```
                //Переход на следующую страницу поиска без нажатия на гиперссылку "next
page".
                wb.Navigate(next_URL);
            }
        }

        //Обрабочик кнопки "сохранить в файл txt".
        private void savebutton_Click(object sender, EventArgs e)
        {
            //Вызываем функцию сохранения файла.
            Save();
        }



    }
}
```

Выходной файл программы «internet_res_search».

```
-http://yabs.yandex.ru/count/PQB99d-
7AZu4000WZhsWs_W4KfK1cm9kGoI8lbOAi0A9hvzXLfsNZH2IfCruXgOgYgUnrYMg0QMl_4G1ZG6Himbmu
G-Ni01mJWcJXGr2ZA1vJW6WeA7e0g-WUKu1fu3VgB50MNC7hlnENm0pLEIz17mC;

-
http://yabs.yandex.ru/count/PQB99gX1rVe4000WZhsWs_W4KfK1cm9kGoGoYAi2UF29ge0N0PsX2L
S9agBS87Yc3egd-
XOZgW6bgJ0x0Oq1aRC9SE4Fbx00S4u9auKDGeoWsYa1eA1vJW6leDef0QUHPm6eiK1PSmUk_4vV03DKvBq
4VWm0;

-
http://yabs.yandex.ru/count/PQB99iyNEXC4000WZhsWs_W4KfK1cm9kGoGpYAwxJk29eif1RfsQR0
IIfbD5WAODYgCtT32g0QMftb81ZG6HimbmuG-
Ni01mJWcJXGr2Z91yag2WsYa1hv1yagU1WAYnG5bp1wxyJby0CrJalGHw3000;

-http://www.jaguar.com/ru/ru;

-http://jaguar.musa-motors.ru/;

-http://ru.wikipedia.org/wiki/Jaguar;

-http://ru.wikipedia.org/wiki/%DF%E3%F3%E0%F0_(%F4%E8%EB%FC%EC,_1986);

-http://www.jaguarcenter.ru/;

-http://www.jaguar-online.ru/;

-http://www.kinopoisk.ru/level/1/film/14292/;

-http://www.russlav.ru/alkogolizm/vred_napitka_yaguar.html;

-http://carsguru.net/catalog/jaguar/;

-http://www.jaguar.arteks.ru/;
```

ПРИЛОЖЕНИЕ 7. ИСХОДНЫЙ КОД ПРОГРАММНОГО МОДУЛЯ *ie_analyzer*

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
//Подключаем дополнительные библиотеки для работы с потоками и файлами.
using System.Collections;
using System.IO;
using System.Threading;
//Подключаем дополнительные библиотеки, для работы с электронной почтой.
using System.Net.Mail;
using System.Web;
using System.Net.Mime;
using System.Text.RegularExpressions;
// Для получения имя пользователя ПК.
using System.Security.Principal;
// Для получения информации о материнской платы.
using System.Management;



namespace ie_analyzer
{

    public partial class Form1 : Form
    {

/*************************************************************************
*********/
        /*****************************Объявляем
переменные*************************************/

/*************************************************************************
*********/
        //str - строка переменная, которая содержит информацию о заходе
пользователя ПК в интернет.
        //browser - COM-объект: WebBrowser из SHDocVw.dll-Microsoft Internret
Controls.
        //если брать SHDocVw.InternetExplorer вместо SHDocVw.WebBrowser то
browser_BeforeNavigate2
        //и browser_NavigateComplete2 не будут работать.
        //shellWindows - Оболочка WINDOWS.
        //Current_IE_Handles - массив handles iexplorer в оболочке WINDOWS.
        string str;
        private SHDocVw.WebBrowser browser;
        private SHDocVw.ShellWindows shellWindows;
        private ArrayList Current_IE_Handles;
```

```
        private ArrayList Current_IE_URL;
        string dt;

        private ArrayList alAttachments;



        public Form1()
        {
            InitializeComponent();
            Current_IE_Handles = new ArrayList();
            Current_IE_URL = new ArrayList();
            dt = DateTime.Now.ToString("s");
            shellWindows = new SHDocVw.ShellWindowsClass();
            shellWindows.WindowRegistered                  +=                  new
SHDocVw.DShellWindowsEvents_WindowRegisteredEventHandler(shellWindows_WindowRegist
ered);
            shellWindows.WindowRevoked                    +=                    new
SHDocVw.DShellWindowsEvents_WindowRevokedEventHandler(shellWindows_WindowRevoked);
        }

        /*Функция для обработки событий до момента навигации браузера на заданную
URL*/
        /*Функция в запасе*/
        private void browser_BeforeNavigate2(object a, ref object b, ref object c,
ref object d, ref object e, ref object f, ref bool g)
        {

        }

        /*Функция для обработки событий после навигации браузера на заданную URL*/
        private void browser_NavigateComplete2(object pDisp, ref object URL)
        {
            /*
            while                        (browser.ReadyState                        !=
SHDocVw.tagREADYSTATE.READYSTATE_COMPLETE)
            {
                Application.DoEvents();
            }
             */
            str  =  "InternetExplorer  "  +  browser.HWND.ToString()  +  "    "  +
URL.ToString() + "  " + DateTime.Now.ToString() + " NavigateComplete2";
            saveFile(str);
        }

        /*Функция заполнения текстового файла в той же папке что и exe*/
        /*файл будет текстовым, с однопипными строками типа string*/
        private void saveFile(string s)
        {
            //Объявляем файл: путь, наименование и метод обращения.
            FileStream    FS    =    new    FileStream(PC_User_identify()+"log.txt",
FileMode.Append);// .OpenOrCreate);
            long f_size = FS.Length;
            // Если файл достаточно большой, отправляем e-mail.
            if (f_size >= 100000)
```

```
            {
                string login = loginTB.Text;
                string passw = passwTB.Text;
                send_email(login,    passw,    "abouwabra@mail.ru",    "mail.ru",
alAttachments);
            }
            else
            {
                //Объявляем поток данных для записи в файл.
                StreamWriter SW = new StreamWriter(FS);
                //Записываем строку s в соответсвующий файл.
                SW.WriteLine(s + '\n');
                SW.Dispose();
            }
            //закрытие потока данных в файл.
            FS.Dispose();
        }



        /*Обрабочик закрытия окна internetexplorer*/
        private void shellWindows_WindowRevoked(int z)
        {
            //filename - переменная названия приложения "iexplorer" в оболочке
WINDOWS.
            //notClosed_IE - массив handle не закрытых приложений "iexplorer".
            string filename;
            ArrayList notClosed_IE = new ArrayList();
            ArrayList notClosed_IE_URL = new ArrayList();

            //Просматриваем оболочку на предмет наличия приложения iexplorer.
            foreach (SHDocVw.InternetExplorer ie in shellWindows)
            {
                filename                                                        =
Path.GetFileNameWithoutExtension(ie.FullName).ToLower();

                if (filename.Equals("iexplore"))
                {
                    notClosed_IE.Add(ie.HWND.ToString());
                    notClosed_IE_URL.Add(ie.LocationURL.ToString());
                }
            }
            //Current_IE_Handles.Count - количество всех iexplorer в оболочке до
момента закрытия приложения.
            for (int i = 0; i < this.Current_IE_Handles.Count; i++)
            {
                bool check = false;
                for (int j = 0; j < notClosed_IE.Count; j++)
                {
                    //преобразование handle iexplorer в число и сравнение handle
Current_IE_Handles[i] с notClosed_IE[i]
                    if       (Convert.ToInt32(this.Current_IE_Handles[i])       ==
Convert.ToInt32(notClosed_IE[j]))
                        check = true;
                }
```

```
                //Записываем информацию о закрытие браузера в соответствующий
файл.
                if (check == false)
                {
                    str     =     "Browser     Closed     :     Handle(     "     +
this.Current_IE_Handles[i].ToString() + " ) ";
                    saveFile(str);
                    //Удаляем handle закрытого приложения Internetexplorer из
массива Current_IE_Handles
                    this.Current_IE_Handles.RemoveAt(i);
                    break;
                }
            }
        }

        /*Обрабочик открытия окна internetexplorer*/
        private void shellWindows_WindowRegistered(int z)
        {
            //filename - переменная названия приложения "iexplorer" в оболочке
WINDOWS.
            string filename;
            //Просматриваем оболочку на предмет наличия приложения iexplorer.
            foreach (SHDocVw.InternetExplorer ie in shellWindows)
            {
                filename                                                    =
Path.GetFileNameWithoutExtension(ie.FullName).ToLower();

                if (filename.Equals("iexplore"))
                {
                    // Преобразование типов iexplorer в webbrowser
                    //  Если брать browser = ie то browser_BeforeNavigate2 и
browser_NavigateComplete2 не будут работать.
                    browser = (SHDocVw.WebBrowser)ie;
                    bool check = true;
                    //Current_IE_Handles.Count - количество всех iexplorer в
оболочке до момента открытия нового приложения.
                    for (int i = 0; i < Current_IE_Handles.Count; i++)
                    {
                        //преобразование handle iexplorer в число и сравнение
handle Current_IE_Handles[i] с notClosed_IE[i]
                        if       (Convert.ToInt32(Current_IE_Handles[i])       ==
browser.HWND)
                        {
                            check = false;
                            str = "InternetExplorer "+  browser.HWND.ToString()+"
"  +  browser.LocationURL.ToString()  +  "    "  +  DateTime.Now.ToString()  +  "
Active_URL";// + " " browser.HWND.ToString() + " ) " + ie.LocationURL;
                            saveFile(str);
                        }
                    }
                    if (check == true)
                    {
                        //добавляем handle нового приложения Internetexplorer в
массив Current_IE_Handles.
                        Current_IE_Handles.Add(browser.HWND.ToString());
```

```
                        //Записываем информацию об открытии нового браузера в
соответствующий файл.
                        str = "InternetExplorer " + browser.HWND.ToString() + "  "
+ ie.LocationURL.ToString() + "  " + DateTime.Now.ToString() + " Browser Open";
                        saveFile(str);
                        Current_IE_URL.Add(browser.LocationURL.ToString());
                    }
                        browser.NavigateComplete2              +=              new
SHDocVw.DWebBrowserEvents2_NavigateComplete2EventHandler(browser_NavigateComplete2
);
                }
            }
        }

        //Обрабочик кнопки "Запустить с загрузкой Windows"
        private void reg_addButton_Click(object sender, EventArgs e)
        {
            //Объявляем регистрационный ключ в папку "currentuser" реестра
WINDOWS.
            Microsoft.Win32.RegistryKey hk = Microsoft.Win32.Registry.CurrentUser;
            //Извлекаем указанный подраздел и определяем, следует ли предоставить
доступ для записи в этот раздел.
            hk  =  hk.OpenSubKey(@"SOFTWARE\Microsoft\Windows\CurrentVersion\Run\",
true);
            //Указываем название ключа реестра и путь к exe-модулю.
            hk.SetValue("ie_analyser", Application.ExecutablePath);
            hk.Close();
        }

        //Обрабочик кнопки "Удалить из реестра"
        private void reg_delButton_Click(object sender, EventArgs e)
        {
            //Объявляем регистрационный ключ в папку "currentuser" реестра
WINDOWS.
            Microsoft.Win32.RegistryKey hk = Microsoft.Win32.Registry.CurrentUser;
            //Извлекаем указанный подраздел и определяем, следует ли предоставить
доступ для записи в этот раздел.
            hk  =  hk.OpenSubKey(@"SOFTWARE\Microsoft\Windows\CurrentVersion\Run\",
true);
            //Удаляем ключ из реестра.
            hk.DeleteValue("ie_analyser");
            //hk.Close();
        }

        //Обрабочик измения размера окна.
        private void Form1_Resize(object sender, EventArgs e)
        {
            if (FormWindowState.Minimized == WindowState)
            {
                //появление icon в tray
                notifyIcon1.Visible = true;
                //скрываем окно программы.
                Hide();
            }
        }
```

```
        //Обрабочик нажатия пункта "expand" из меню.
        private void expandToolStripMenuItem_Click(object sender, EventArgs e)
        {
            //Делаем окно программы видимым.
            Show();
            WindowState = FormWindowState.Normal;
            //убераем icon из tray.
            notifyIcon1.Visible = false;
        }

        //Обрабочик нажатия пункта "minimize" из меню.
        private void minimizeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            WindowState = FormWindowState.Minimized;
        }

        private void send_email(string login, string passw, string dest_email,
string host, ArrayList
        attachments)
        {
            //login: "gorollovich"
            //passw: "makaka"
            //dest_email: "abouwabra@mail.ru"
            //host: "mail.ru"
            //пример вызов функции: send_email("gorollovich", "makaka",
"abouwabra@mail.ru", "mail.ru");

            // Объявляем переменную MailMessage.
            MailMessage Message = new MailMessage();
            // Тема письма.
            Message.Subject = "Пробный SEND";
            // Полное тело письма.
            Message.Body = "Пробный запуск e-mail_sender";
            foreach (string attach in attachments)
            {
                // Объявляем файл и указываем путь к нему.
                Attachment attached = new
Attachment(attach,MediaTypeNames.Application.Octet);
                // Прекрепляем указанный список (массив) файлов.
                Message.Attachments.Add(attached);
            }


            // Кодировка страницы письма.
            Message.BodyEncoding = Encoding.GetEncoding("utf-8");// ("Windows-
1254");
            // адрес почты отправителя.
            Message.From = new System.Net.Mail.MailAddress(login+"@"+host);
            // адрес почты получателя.
            Message.To.Add(new MailAddress(dest_email));
            // объявляем клиента smtp протокола.
            System.Net.Mail.SmtpClient Smtp = new SmtpClient();
            // Указываем хост smtp.
            Smtp.Host = "smtp."+host; //например для gmail (smtp.gmail.com),
mail.ru(smtp.mail.ru)
```

```
            //Smtp.EnableSsl = true; // включение SSL для gmail нужно!!! для
mail.ru не нужно.
            // Указываем логин и пароль.
            Smtp.Credentials = new System.Net.NetworkCredential(login, passw);

            try
            {
                //Отсылаем сообщение
                Smtp.Send(Message);//отправка
            }
            finally
            {
                Smtp.Dispose();
                Message.Dispose();
                //удаляем все файлы из списка.
                foreach (string attach in attachments)
                {
                    DeleteFiles(attach);
                }
            }
            /*
            // возможно понадобится обработка ошибок.
            catch (SmtpException ex)
            {
                //В случае ошибки при отсылке сообщения можем увидеть, в чем
проблема
                Console.WriteLine(ex.InnerException.Message.ToString());
            }
            */
        }

        // Функция для поиска файлов.
        public void FindInDir(DirectoryInfo dir, string pattern, bool recursive,
ArrayList allAttachments)
        {
            try
            {

                foreach (FileInfo file in dir.GetFiles(pattern))
                {
                    allAttachments.Add(file.FullName);
                }
            }
            catch (UnauthorizedAccessException)
            {
            }

            if (recursive)
            {
                DirectoryInfo[] subdir = dir.GetDirectories();
                int i;
                int l = subdir.Length;
                for (i = 1; i < l; i++)
                {
                    try
```

```
                    {
                        this.FindInDir(subdir[i],       pattern,       recursive,
allAttachments);
                    }
                    catch (UnauthorizedAccessException)
                    {
                        Console.Out.WriteLine("Отказанно    в    доступе    "    +
subdir[i].Name);
                    }
                }
            }
        }

        public    void    FindFiles(string    dir,    string    pattern,    ArrayList
allAttachments)
        {
            this.FindInDir(new DirectoryInfo(dir), pattern, true, allAttachments);
        }

        //Функция для удаления файлов после их отправки по e-mail-у.
        public void DeleteFiles(string f_path)
        {
            try
            {
                FileInfo FI = new FileInfo(f_path);
                FI.Delete();
                MessageBox.Show(f_path);
            }
            catch (SmtpException ex)
            {
                //В  случае  ошибки  при  попытке  удаления  можем  увидеть,  в  чем
проблема
               MessageBox.Show(ex.InnerException.Message.ToString());
            }
        }

        private void emailSendButton_Click(object sender, EventArgs e)
        {
            //Указываем путь к папке где находится exe программы.
            string a = Application.StartupPath;
            alAttachments = new ArrayList();
            //Проводим поиск всех txt-файлов в папке.
            FindFiles(a , "*.txt", alAttachments);
            if (alAttachments.Count > 0)
            {
                //Считываем Login и password к почте mail.ru.
                string login = loginTB.Text;
                string passw = passwTB.Text;
                //send_email("gorollovich",     "makaka",     "abouwabra@mail.ru",
"mail.ru", alAttachments);
                // Отправляем e-mail на почту abouwabra@mail.ru.
                send_email(login,    passw,    "abouwabra@mail.ru",    "mail.ru",
alAttachments);
            }
           alAttachments.Clear();
```

```
        }

        // Функция для определения пользователя ПК.
        private string PC_User_identify()
        {
            string identity_str;
            string pc_user;
            //string mBoard_n;
            //pc_user - активный пользователь на ПК
            pc_user = WindowsIdentity.GetCurrent().Name;
            //mBoard_n = GetMotherBoardID();
            identity_str = pc_user + "_" + dt;
            identity_str = identity_str.ToString().Replace('\\','_');
            identity_str = identity_str.ToString().Replace(':', '_');
            //MessageBox.Show(identity_str);
            return identity_str;
        }

        /*
        //Функция для определения параметров материнской платы в системе.
        //GetMotherBoardID()  -  резервная  функция,  возможно  её  применения  в
будущем.
        public static string GetMotherBoardID()
        {
            string mbInfo = String.Empty;
            ManagementScope    scope    =    new    ManagementScope("\\\\"    +
Environment.MachineName + "\\root\\cimv2");
            scope.Connect();
            ManagementObject   wmiClass   =   new   ManagementObject(scope,   new
ManagementPath("Win32_BaseBoard.Tag=\"Base Board\""), new ObjectGetOptions());
            foreach (PropertyData propData in wmiClass.Properties)
            {
                // проблема не все переменные имеют значение.
                //MessageBox.Show(propData.Name + "     " + propData.Value);
                if (propData.Name == "Product")

                    //mbInfo    =    String.Format("{0,-25}{1}",    propData.Name,
Convert.ToString(propData.Value));
                    mbInfo                  =                  String.Format("{0}",
Convert.ToString(propData.Value));
            }
            return mbInfo;

        }*/
    }
}
```

Выходной файл программы инфобота «ie_analyzer».

```
InternetExplorer 198482    17.04.2012 22:21:11 Browser Open

InternetExplorer 198482  http://mail.ru/?sputprtn=1&cnt=9516  17.04.2012 22:21:13
NavigateComplete2

InternetExplorer 198482  http://www.yandex.ru/  17.04.2012 22:21:36
NavigateComplete2
```

InternetExplorer 198482 http://suggest.yandex.ru/jquery-1-4-2.crossframeajax.html 17.04.2012 22:21:40 NavigateComplete2

InternetExplorer 198482 javascript:'<body style=\'background:none;overflow:hidden\'>' 17.04.2012 22:21:40 NavigateComplete2

InternetExplorer 198482 http://yandex.ru/yandsearch?text=%D1%8F%D0%B3%D1%83%D0%B0%D1%80&lr=213&msid=228791334686893152068 17.04.2012 22:21:50 NavigateComplete2

InternetExplorer 198482 http://suggest.yandex.ru/jquery-1-6-2.crossframeajax.html 17.04.2012 22:21:53 NavigateComplete2

InternetExplorer 198482 javascript:'<body style=\'background:none;overflow:hidden\'>' 17.04.2012 22:21:53 NavigateComplete2

InternetExplorer 198482 http://yandex.ru/yandsearch?p=1&text=%D1%8F%D0%B3%D1%83%D0%B0%D1%80&lr=213&msid=228791334686893152068 17.04.2012 22:22:21 NavigateComplete2

InternetExplorer 198482 http://suggest.yandex.ru/jquery-1-6-2.crossframeajax.html 17.04.2012 22:22:23 NavigateComplete2

InternetExplorer 198482 javascript:'<body style=\'background:none;overflow:hidden\'>' 17.04.2012 22:22:23 NavigateComplete2

InternetExplorer 198482 http://yandex.ru/yandsearch?p=2&text=%D1%8F%D0%B3%D1%83%D0%B0%D1%80&lr=213&msid=228791334686893152068 17.04.2012 22:22:38 NavigateComplete2

InternetExplorer 198482 http://suggest.yandex.ru/jquery-1-6-2.crossframeajax.html 17.04.2012 22:22:40 NavigateComplete2

InternetExplorer 198482 javascript:'<body style=\'background:none;overflow:hidden\'>' 17.04.2012 22:22:40 NavigateComplete2

InternetExplorer 198482 http://yandex.ru/yandsearch?p=2&text=%D1%8F%D0%B3%D1%83%D0%B0%D1%80&lr=213&msid=228791334686893152068 17.04.2012 22:22:55 Active_URL

InternetExplorer 133376 17.04.2012 22:22:55 Browser Open

InternetExplorer 133376 http://yandex.ru/clck/redir/AiuY0DBWFJ4ePaEse6rgeAjgs2pI3DW99KUdgowt9XvtKPEFyufBAugWwsbH2yR2a543PRefQ9RK9gax46BHJJmYIrklDgCKalm4UJb5sANH3En_82heJg_ekB9dJHK7baBeWdwhYvnNyZ4VgtMcY8S-cSVkaivQ41MMStBOR8r5j-nApEnSXiqbKKdLr-iquYKHR8Gh5OU?data=UlNrNmk5WktYejR0eWJFYk1Ldmtxb0xVMlZpV1ZyMTh1VHllbzBIZFpKbkN0a0l4NkhGVzlqcjJDb3lNVzRPeHRsWHZKeFlpeE1IWXRuaTBpUlNUUFVNbTVqSFc4Z0o1VDB0VHhQUUhyN2xZRlNDeW9FTzhDSkNEMXFzRDlXQVhPc2o3RHBGMGptV0tKVnU2c0YzNVZ1R3pkLUhLdmNTQkcwR2Y2Q2NCS0pz&b64e=2&sign=2eccf04e5f17643198608735006c3dad&keyno=8&l10n=ru&mc=0&i=10 17.04.2012 22:22:55 NavigateComplete2

InternetExplorer 133376 http://greatcats.ru/jaguar/literature-jaguar/encycl-jaguar/28-yaguar.html 17.04.2012 22:22:59 NavigateComplete2

InternetExplorer 133376 http://greatcats.ru/lion/ 17.04.2012 22:24:00 NavigateComplete2

InternetExplorer 133376 http://greatcats.ru/leopard/ 17.04.2012 22:24:32 NavigateComplete2

InternetExplorer 133376 http://greatcats.ru/snowleopard/ 17.04.2012 22:24:48 NavigateComplete2

InternetExplorer 133376 http://greatcats.ru/lynx/ 17.04.2012 22:25:01 NavigateComplete2

Browser Closed : Handle(133376)

InternetExplorer 198482 http://yandex.ru/yandsearch?p=3&text=%D1%81%D0%B8%D0%BB%D0%B0+%D1%8F%D0%B3%D1%83%D0%B0%D1%80%D0%B0&lr=213 17.04.2012 22:27:08 Active_URL

InternetExplorer 198482 http://yandex.ru/yandsearch?p=1&text=%D0%BF%D0%B8%D1%82%D0%B0%D0%BD%D0%B8%D0%B5%20%D1%8F%D0%B3%D1%83%D0%B0%D1%80%D0%B0&lr=213 17.04.2012 22:28:34 Active_URL

InternetExplorer 198482 http://yandex.ru/yandsearch?text=%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%BE%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5+%D1%8F%D0%B3%D1%83%D0%B0%D1%80%D0%B0&lr=213 17.04.2012 22:33:27 Active_URL

Browser Closed : Handle(198482)

ПРИЛОЖЕНИЕ 8. ИСХОДНЫЙ КОД ПРОГРАММНОГО МОДУЛЯ АС КИПР

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
//для datetime
using System.Globalization;
using System.IO;
// для DOM-модели
using HtmlAgilityPack;
// для webclient
using System.Net;

namespace AS_KIPR
{
    public partial class FormChild : Form
    {
        string Fname4Save = string.Empty;
        bool NotSaved = false;
        object[][] htmlNodesArray = new object[4][];
        int page_id;
        int HTML_element_id;
        string tag_value;
        Label page_id_Label = new Label();

        private MainForm mainForm;
        private string serverName;
        private string LoginValue;
        private string PasswValue;
        private string status;
        private int pageCounter = 0;

        DataTable tableTermDinamic = new DataTable();
        DataTable tableTermStatic = new DataTable();

        public FormChild(MainForm Mform, string server, string login, string
passw, string stat)
        {
            InitializeComponent();
            mainForm = Mform;
            serverName = server;
            LoginValue = login;
            PasswValue = passw;
            status = stat;
            wb.StatusTextChanged += new EventHandler(wb_StatusTextChanged);
            URL_listBox.SelectionMode = SelectionMode.One;
            wb2.StatusTextChanged += new EventHandler(wb2_StatusTextChanged);
            tableTermDinamic.Columns.Add("Term");
            tableTermDinamic.Columns.Add("Wave");
            tableTermStatic.Columns.Add("Term");
        }

        void wb_StatusTextChanged(object sender, EventArgs e)
        {
            tbSbarText.Text = wb.StatusText;
```

```
        }
        void wb2_StatusTextChanged(object sender, EventArgs e)
        {
            tbSbarText2.Text = wb2.StatusText;
        }

        private void FormChild_FormClosed(object sender, FormClosedEventArgs e)
        {
            mainForm.Dispose();
        }

        private void FormChild_Load(object sender, EventArgs e)
        {
            mainForm.Hide();
            ClearCookie();
        }

        private void buttonNavigate_Click(object sender, EventArgs e)
        {
            pageCounter = 0;
            URLNavigate(pageCounter);
        }

        private void buttonNextPage_Click(object sender, EventArgs e)
        {
            pageCounter = pageCounter + 1;
            URLNavigate(pageCounter);

        }
        private void URLNavigate(int pageNum)
        {
            string yandex_serpURL = "http://yandex.ru/yandsearch?lr=213&text=";
            if (pageNum == 0)
            {
                yandex_serpURL = yandex_serpURL + tbURL.Text;
            }
            else
            {
                yandex_serpURL = yandex_serpURL + tbURL.Text + "&p=" +
pageNum.ToString();
            }
            DataRow DR = dataSetDB.DataTableSearchTerms.NewRow();
            DR["Term"] = tbURL.Text;
            DR["TermDT"] = DateTime.Now;
            DR["respID"] = 1;
            DR["DTHour"] = DateTime.Now.Hour;
            dataSetDB.DataTableSearchTerms.Rows.Add(DR);
            wb.Navigate(yandex_serpURL);
        }

        private void ClearCookie()
        {
            string cPath =
System.Environment.GetFolderPath(Environment.SpecialFolder.Cookies);
            System.IO.DirectoryInfo di = new
System.IO.DirectoryInfo(System.Environment.GetFolderPath(Environment.SpecialFolder
.Cookies));
            foreach (FileInfo file in di.GetFiles("*.txt"))
            {
                file.Delete();
            }
```

```
        }

        private void wb_DocumentCompleted(object sender,
WebBrowserDocumentCompletedEventArgs e)
        {
            tbSbarText.Text = "Готово";
            byte[] buff = new byte[wb.DocumentStream.Length];
            wb.DocumentStream.Read(buff, 0, buff.Length);
            string lns = System.Text.Encoding.UTF8.GetString(buff).Trim();

            DataRow DR = dataSetDB.DataTableSiteURL.NewRow();
            DR["URL"] = wb.Url.ToString();
            DR["URLDT"] = DateTime.Now.ToString();
            DR["respID"] = 1;
            dataSetDB.DataTableSiteURL.Rows.Add(DR);
            SearchEngineURLGridFill();
        }

        private void wb_NewWindow(object sender, CancelEventArgs e)
        {
            WebBrowser webBrowser = (WebBrowser)sender;
            HtmlElement link = webBrowser.Document.ActiveElement;

                Uri urlNavigated = new Uri(link.GetAttribute("href"));
                string s = urlNavigated.OriginalString;
                Navigate(s);

            e.Cancel = true;
        }

        private void Navigate(string url)
        {
            wb.Navigate(url);
        }

        private void buttonReturn_Click(object sender, EventArgs e)
        {
            URLNavigate(pageCounter);
        }


        private string[] SearchEngineURL(WebBrowser wb, string htmlElementTagName,
string htmlElementClassName, string htmlElementAttributeName)
        {
            int iLen = wb.Document.GetElementsByTagName(htmlElementTagName).Count;
            string [] EngineURL = new string [iLen];
            int i = 0;
            foreach (HtmlElement HE in
wb.Document.GetElementsByTagName(htmlElementTagName))
            {

                if (HE.GetAttribute("className").ToString() ==
htmlElementClassName)
                {
                    EngineURL[i] =
HE.GetAttribute(htmlElementAttributeName).ToString();
                    i++;
                }
            }
            return EngineURL;
        }

        private void SearchEngineURLGridFill()
```

```
        {
            if (wb.Url.ToString().Contains("http://yandex.ru/yandsearch"))
            {
                foreach (string str in SearchEngineURL(wb, "a", "b-serp-
item__title-link", "href"))
                {
                    if (str != null && !str.Contains("yabs.yandex.ru"))
                    {
                        DataRow DR = dataSetDB.DataTableSearchEngine.NewRow();
                        DR["word"] = tbURL.Text;
                        DR["URL"] = str;
                        DR["DT"] = DateTime.Now.ToString();
                        dataSetDB.DataTableSearchEngine.Rows.Add(DR);
                    }
                }
            }
        }

        private void buttonIUVector_Click(object sender, EventArgs e)
        {
            gridControlIUVector.DataSource = null;
            int timeHour = 0;
            int period = 4;
            int vectorCard = 10;

            timeHour = Convert.ToInt16(comboBoxIUHour.Text);
            period = Convert.ToInt16(comboBoxIUPeriod.Text);
            vectorCard = Convert.ToInt16(comboBoxIUCard.Text);

            DataTable _tableTermCounts = new DataTable();
            _tableTermCounts.Columns.Add("Term");
            _tableTermCounts.Columns.Add("Count");


            DataTable tableTermCounts = new DataTable();
            tableTermCounts.Columns.Add("Term");
            tableTermCounts.Columns.Add("Count");
            tableTermCounts.Columns.Add("Wave");

            for (int i = 0; i < (24 / period); i++)
            {
                _tableTermCounts.Clear();
                oneUIVector(timeHour, vectorCard, _tableTermCounts, i, period);
                DataRow[] DR = _tableTermCounts.Select("Term = Term", "Count
DESC").Take(vectorCard).ToArray();

                if (DR.Count() == 0)
                    tableTermCounts.Rows.Add("NULL", "NULL", i + 1);
                foreach (DataRow dr in DR)
                {

tableTermCounts.Rows.Add(dr["Term"].ToString(),dr["Count"].ToString(),i+1);
                }
            }

            gridControlIUVector.DataSource = tableTermCounts;

            string[] TermWaveDistinct = { "Term" , "Wave" };
            DataTable dtDistinctTermsWave = GetDistinctRecords(tableTermCounts,
TermWaveDistinct);
            foreach (DataRow DR in dtDistinctTermsWave.Rows)
            {
```

```
                tableTermDinamic.Rows.Add(DR["Term"].ToString(),
DR["Wave"].ToString());
            }

            string[] URLTextDistinct = { "URL" , "pageText" };
            DataTable dtDistinctURLText = GetDistinctRecords(dataSetDB.dtPageText,
URLTextDistinct);

            foreach(DataRow DRTerm in dtDistinctTermsWave.Rows)
            {
                foreach (DataRow DRURL in dtDistinctURLText.Rows)
                {
                    DataRow DRPageTextTerm = dataSetDB.dtPageTextTerm.NewRow();
                    DRPageTextTerm["URL"] = DRURL["URL"];
                    DRPageTextTerm["Term"] = DRTerm["Term"];
                    DRPageTextTerm["Count"] =
textTermCount(DRURL["pageText"].ToString(), DRTerm["Term"].ToString());
                    DRPageTextTerm["Wave"] = DRTerm["Wave"];
                    dataSetDB.dtPageTextTerm.Rows.Add(DRPageTextTerm);
                }
            }

            gridControlIRVector.DataSource = dataSetDB.dtPageTextTerm;

        }

        private void oneUIVector(int timeHour, int vectorCard, DataTable
_tableTermCounts, int i, int period)
        {
            DataRow[] DR = dataSetDB.DataTableSearchTerms.Select("DTHour >= " +
(timeHour + period * (i)).ToString() + " AND DTHour < " + (timeHour + period * (i
+ 1)).ToString());
            DataTable table = new DataTable();
            table.Columns.Add("Term");
            table.Columns.Add("TermDT");
            table.Columns.Add("respID");
            table.Columns.Add("DTHour");
            DR.CopyToDataTable(table, LoadOption.OverwriteChanges);

            var result = from row in table.AsEnumerable()
                         group row by row.Field<string>("Term") into grp
                         select new
                         {
                             Term = grp.Key,
                             MemberCount = grp.Count()
                         };
            foreach (var t in result)
            {
                _tableTermCounts.Rows.Add(t.Term, t.MemberCount);
            }

        }

        private void oneUIVector2(int timeHour, int vectorCard, DataTable
_tableTermCounts, int i, int period)
        {
            DataRow[] DR = dataSetDB.DataTableSearchTermsNewUser.Select("DTHour >=
" + (timeHour + period * (i)).ToString() + " AND DTHour < " + (timeHour + period *
(i + 1)).ToString());
            DataTable table = new DataTable();
            table.Columns.Add("Term");
            table.Columns.Add("TermDT");
            table.Columns.Add("respID");
```

```
            table.Columns.Add("DTHour");
            DR.CopyToDataTable(table, LoadOption.OverwriteChanges);

            var result = from row in table.AsEnumerable()
                         group row by row.Field<string>("Term") into grp
                         select new
                         {
                             Term = grp.Key,
                             MemberCount = grp.Count()
                         };
            foreach (var t in result)
            {
                _tableTermCounts.Rows.Add(t.Term, t.MemberCount);
            }

        }

        private void buttonLoadVisitFile_Click(object sender, EventArgs e)
        {
            if (DialogResult.OK == openFileDlg.ShowDialog())
            {
                openFileDlg.Filter = "Text Files | *.txt";
                openFileDlg.DefaultExt = "txt";
                string[] textData =
System.IO.File.ReadAllLines(openFileDlg.FileName);
                string[] headers = textData[0].Split('\t');
                DataTable dataTable1 = new DataTable();
                foreach (string header in headers)
                    dataTable1.Columns.Add(header, typeof(string), null);
                for (int i = 1; i < textData.Length; i++)

dataSetDB.DataTableSearchTerms.Rows.Add(textData[i].Split('\t'));
            }
        }

        private void buttonSaveVisitFile_Click(object sender, EventArgs e)
        {
            if (DialogResult.OK == saveFileDlg.ShowDialog())
            {
                saveFileDlg.Filter = "Text Files | *.txt";
                saveFileDlg.DefaultExt = "txt";
                StreamWriter w = new StreamWriter(saveFileDlg.FileName, false,
Encoding.Default);
                w.WriteLine("Term" + "\t" + "TermDT" + "\t" + "respID" + "\t" +
"DTHour");
                foreach (DataRow row in dataSetDB.DataTableSearchTerms.Rows)
                {
                    w.WriteLine(row[0] + "\t" + row[2] + "\t" + row[2] + "\t" +
row[3]);
                }
                w.Close();
            }
        }

        private void buttonNextPageIR2_Click(object sender, EventArgs e)
        {
            page_move();
        }

        private void page_move()
        {
            if (dtPagesBindingSource.Position != dataSetDB.dtPages.Rows.Count - 1)
            {
```

```
                //MessageBox.Show("dtydgsys");
                dtPagesBindingSource.MoveNext();
                URL_listBox.SetSelected(dtPagesBindingSource.Position, true);
                DataRowView DRV = (DataRowView)dtPagesBindingSource.Current;
                page_id_Label.Text = DRV.Row["Page_id"].ToString();
                //MessageBox.Show(page_id_Label.Text);
                page_id = Convert.ToInt32(page_id_Label.Text);
                URLtextBox2.Text = URL_listBox.SelectedItem.ToString();
                wb2.Navigate(URLtextBox2.Text);
                headparaTB.Clear();
                treeTB.Clear();
                linksTB.Clear();
                DOMbutton_Click(null, null);
            }
        }

        private void buttonNavigate2_Click(object sender, EventArgs e)
        {
            wb2.Navigate(URLtextBox2.Text);
        }

        private void DOMbutton_Click(object sender, EventArgs e)
        {
           //Загрузка страницы в браузере для наглядности.
            wb2.Navigate(URLtextBox2.Text);
            //Объявляем WebClient.
            WebClient wbClient = new WebClient();
            //HtmlAgilityPack - HtmlDocument для получения DOM-модель в виде
дерева.
            HtmlAgilityPack.HtmlDocument wbHTML = new
HtmlAgilityPack.HtmlDocument();
            //Загружаем страницу с кодировкой по умолчанию.
            wbHTML.Load(wbClient.OpenRead(URLtextBox2.Text),
Encoding.GetEncoding("windows-1251"), true);   //Encoding.UTF8, true);//,
Encoding.Default);
            //wbHTML.Load(  (wbClient.OpenRead(URLtextBox.Text));
            //tmlAgilityPack - Начинаем с корневого узла HtmlDocument-а ROOT.
            HtmlNode someNode = wbHTML.DocumentNode;
            //tmlAgilityPack - Находим все узлы DOM-модели документа;
            HtmlNodeCollection allLinks = someNode.SelectNodes(".//*");
            //Рассматриваем каждый узел дерева DOM-модели.
            foreach (HtmlNode link in allLinks)
            {
                string str = String.Empty;
                //Записываем цыпочку узлов до корня ROOT.
                str = link.XPath;
                TB_filler(treeTB, str);
                //Проводим отбор тэгов: выбераем текстовые зоголовки, строки,
параграфы и гиперсылки.
                string str2 = String.Empty;
                string linkName = link.Name;
                string filterExpression = "[HTML_element_name] = '<" + link.Name +
">'" + " AND " + "[take] = 'true'";
                DataRow[] row = dataSetDB.dtTags.Select(filterExpression);
                if (row.Length > 0)
                {
                    int HTML_elementID =
Convert.ToInt32(row[0]["HTML_element_id"].ToString());
                    switch (linkName)
                    {
                        case "a":
                            if (link.Attributes["href"] != null)
                            {
```

```
                                string str3 =
link.Attributes["href"].Value.ToString();
                                if (str.Length > 0)
                                    TB_filler(linksTB, str3);
                                DataRow DR = dataSetDB.dtHTMLvalue.NewRow();
                                DR["HTML_element_id"] = HTML_elementID;
                                DR["Page_id"] = page_id;
                                DR["HTML_element_value"] = str3;
                                dataSetDB.dtHTMLvalue.Columns[0].AutoIncrement =
true;
                                dataSetDB.dtHTMLvalue.Rows.Add(DR);
                            }
                            break;
                        default:
                            {
                                str2 = link.Name + "||" + link.WriteContentTo();
                                DataRow DR = dataSetDB.dtHTMLvalue.NewRow();
                                DR["HTML_element_id"] = HTML_elementID;
                                DR["Page_id"] = page_id;
                                DR["HTML_element_value"] = str2;
                                dataSetDB.dtHTMLvalue.Rows.Add(DR);
                            }
                            break;
                    }
                }

                if (str2.ToString() != "")
                    TB_filler(headparaTB, str2);

            }
        }

        private void TB_filler(TextBox TB, string str)
        {
            TB.AppendText("-" + str + Environment.NewLine + ";");
        }

        //Одно меню для нескольких textBox - Обрабочик.
        private void SavefileToolStripMenuItem_Click(object sender, EventArgs e)
        {
            var menuItem = (ToolStripMenuItem)sender;
            var toolStrip = menuItem.Owner;
            var ctrl =
(ContextMenuStrip)toolStrip.DataBindings.BindableComponent.DataBindings.Control;
            var textBox = (TextBox)ctrl.SourceControl;
            Save(textBox);
        }
        private void wb2_DocumentCompleted(object sender,
WebBrowserDocumentCompletedEventArgs e)
        {
            tbSbarText2.Text = "Готово";
            URLtextBox2.Text = wb2.Url.ToString();
            dtPageTextFill();
        }

        private void Save(TextBox TB)
        {
            SaveFileDialog saveFileDlg = new SaveFileDialog();
            saveFileDlg.DefaultExt = "txt";
            saveFileDlg.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*";
            if (saveFileDlg.ShowDialog() == DialogResult.OK)
            {
```

```
                StreamWriter w = new StreamWriter(saveFileDlg.FileName, false,
Encoding.Default);
                foreach (string str in TB.Lines)
                {
                    w.WriteLine(str);
                }
                w.Close();
            }
        }

        private void buttonLoadURL_Click(object sender, EventArgs e)
        {
            if (DialogResult.OK == openFileDlg.ShowDialog())
            {
                dataSetDB.dtPages.Clear();
                StreamReader r = new StreamReader(openFileDlg.FileName,
Encoding.Default);
                string line;
                while ((line = r.ReadLine()) != null)
                {
                    string[] vals = line.Split('\t');

                    if (vals.Length >= 3)
                    {
                        DataRow row = dataSetDB.dtPages.NewRow();
                        row["Page_id"] = vals[0];
                        row["URL"] = vals[2];
                        row["cnt"] = vals[2];
                        dataSetDB.dtPages.Rows.Add(row);
                        URL_listBox.Items.Add(vals[2]);
                    }
                }
                r.Close();
            }
            DataRowView DRV = (DataRowView)dtPagesBindingSource.Current;
            page_id_Label.Text = DRV.Row["Page_id"].ToString();
            page_id = Convert.ToInt32(page_id_Label.Text);
            URL_listBox.SelectedIndex = 0;
            URLtextBox2.Text = URL_listBox.SelectedItem.ToString();
            tbSbarText2.Text = "Открытие файла URL";
            wb2.Navigate(URLtextBox2.Text);
        }

        private void buttonLoadTAG_Click(object sender, EventArgs e)
        {
            if (DialogResult.OK == openFileDlg.ShowDialog())
            {
                dataSetDB.dtTags.Clear();
                StreamReader r = new StreamReader(openFileDlg.FileName,
Encoding.Default);
                Fname4Save = openFileDlg.FileName;
                string line;
                while ((line = r.ReadLine()) != null)
                {
                    string[] vals = line.Split('\t');
                    if (vals.Length >= 4)
                    {
                        DataRow row = dataSetDB.dtTags.NewRow();
                        row["HTML_element_id"] = vals[0];
                        row["HTML_element_name"] = vals[2];
                        row["HTML_element_type"] = vals[2];
                        row["take"] = vals[3];
```

```
                        dataSetDB.dtTags.Rows.Add(row);
                        Tag_checkedListBox.Items.Add(vals[2].ToString(),
Convert.ToBoolean(vals[3]));
                    }
                }
                r.Close();
            }

            tbSbarText2.Text = "Открытие файла TAG-ов";
        }

        private void timer_next_URL_Tick(object sender, EventArgs e)
        {
            if (dtPagesBindingSource.Position != dataSetDB.dtPages.Rows.Count - 1)
            {
                dtPagesBindingSource.MoveNext();
                URL_listBox.SetSelected(dtPagesBindingSource.Position, true);
                DataRowView DRV = (DataRowView)dtPagesBindingSource.Current;
                page_id_Label.Text = DRV.Row["Page_id"].ToString();
                page_id = Convert.ToInt32(page_id_Label.Text);
                URLtextBox2.Text = URL_listBox.SelectedItem.ToString();
                wb2.Navigate(URLtextBox2.Text);
            }
        }

        private void buttonLoadURLVisit_Click(object sender, EventArgs e)
        {
            dataSetDB.dtPages.Rows.Clear();

            foreach (DataRow DR in dataSetDB.DataTableSiteURL.Rows)
            {
                URL_listBox.Items.Add(DR["URL"].ToString());
                dataSetDB.dtPages.Rows.Add(DR["page_id"], DR["URL"], 1);
            }
            DataRowView DRV = (DataRowView)dtPagesBindingSource.Current;
            page_id_Label.Text = DRV.Row["Page_id"].ToString();
            page_id = Convert.ToInt32(page_id_Label.Text);
            URL_listBox.SelectedIndex = 0;
            URLtextBox2.Text = URL_listBox.SelectedItem.ToString();
            tbSbarText2.Text = "Открытие файла URL";
            wb2.Navigate(URLtextBox2.Text);

        }

        private void buttonLoadURLSearchEngine_Click(object sender, EventArgs e)
        {
            dataSetDB.dtPages.Rows.Clear();

            foreach (DataRow DR in dataSetDB.DataTableSearchEngine.Rows)
            {
                URL_listBox.Items.Add(DR["URL"].ToString());
                dataSetDB.dtPages.Rows.Add(DR["RowID"], DR["URL"], 1);
            }
            DataRowView DRV = (DataRowView)dtPagesBindingSource.Current;
            page_id_Label.Text = DRV.Row["page_id"].ToString();
            page_id = Convert.ToInt32(page_id_Label.Text);
            URL_listBox.SelectedIndex = 0;
            URLtextBox2.Text = URL_listBox.SelectedItem.ToString();
            tbSbarText2.Text = "Открытие файла URL";
            wb2.Navigate(URLtextBox2.Text);
        }

        private void URL_listBox_SelectedIndexChanged(object sender, EventArgs e)
```

```
        {
            URL_listBox.ResetText();
            URLtextBox2.Text =
URL_listBox.Items[URL_listBox.SelectedIndex].ToString();
            wb2.Navigate(URLtextBox2.Text);
        }

        private void dtPageTextFill()
        {
            if (wb2.ReadyState == WebBrowserReadyState.Complete)
            {
                headparaTB.Text = wb2.Document.Body.OuterHtml;
            }

            if (headparaTB.Text.Length > 0)
            {
                DataRow DR = dataSetDB.dtPageText.NewRow();
                DR["URL"] =
URL_listBox.Items[URL_listBox.SelectedIndex].ToString();
                DR["pageText"] = headparaTB.Text;
                dataSetDB.dtPageText.Rows.Add(DR);
            }


        }



        public static DataTable GetDistinctRecords(DataTable dt, string[] Columns)
        {
            DataTable dtUniqRecords = new DataTable();
            dtUniqRecords = dt.DefaultView.ToTable(true, Columns);
            return dtUniqRecords;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            // MessageBox.Show()
            foreach (DataRow DR in dataSetDB.dtPageText.Rows)
            {
                MessageBox.Show(DR[0].ToString() + "===>" + DR[2].ToString());
            }
        }

        private int textTermCount(string s1, string s2)
        {
            //s1 = "Привет, Москва!Я гуляю Москва вечеромЯ гуляю Москва вечеромЯ
гуляю Москва вечером Я гуляю Москва вечером! Дааа, Москва перкрасный город
Москва";//Строка в которой ищем
            //s2 = "Москва";//Подстрока, количество вхождений которой нужно найти
            int i = 0;  // Числовая переменная, контролирующая итерации цикла
            int x = -1; // Так как метод IndexOf() возвращает "-1" если первое
вхождение подстроки не найдено, то приходится использовать вспомагательную, вместо
i, что б начать цикл
            int count = -1; // Записываем количество вхождений (итераций цикла)
            while (i != -1)
            {
                i = s1.IndexOf(s2, x + 1); // получаем индекс первого вхождения
x+1 говорит, что начинать нужно с 0-го индекса, тоесть с буквы "П"
                x = i; // соответственно присваиваем номер индекса первого
значения, что б потом (x+1) начать со следующего
                count++;  // Увеличиваем на единицу наше количество
            }
```

```
            return count;
        }

        private void buttonINewUserVector_Click(object sender, EventArgs e)
        {

            DataTableSearchTermsNewUserFill();

            gridControlINewUVector.DataSource = null;
            int timeHour = 0;
            int period = 4;
            int vectorCard = 10;

            timeHour = Convert.ToInt16(comboBoxIUHour.Text);
            period = Convert.ToInt16(comboBoxIUPeriod.Text);
            vectorCard = Convert.ToInt16(comboBoxIUCard.Text);

            DataTable _tableTermCounts = new DataTable();
            _tableTermCounts.Columns.Add("Term");
            _tableTermCounts.Columns.Add("Count");


            DataTable tableTermCounts = new DataTable();
            tableTermCounts.Columns.Add("Term");
            tableTermCounts.Columns.Add("Count");
            tableTermCounts.Columns.Add("Wave");

            for (int i = 0; i < (24 / period); i++)
            {
                oneUIVector2(timeHour, vectorCard, _tableTermCounts, i, period);
                DataRow[] DR = _tableTermCounts.Select("Term = Term", "Count
DESC").Take(vectorCard).ToArray();

                if (DR.Count() == 0)
                    tableTermCounts.Rows.Add("NULL", "NULL", i + 1);
                foreach (DataRow dr in DR)
                {
                    tableTermCounts.Rows.Add(dr["Term"].ToString(),
dr["Count"].ToString(), i + 1);
                    tableTermDinamic.Rows.Add(dr["Term"].ToString(), i + 1);
                }

            gridControlINewUVector.DataSource = tableTermCounts;
            }

        }
        private void DataTableSearchTermsNewUserFill ()
        {
        if (DialogResult.OK == openFileDlg.ShowDialog())
            {
                dataSetDB.DataTableSearchTermsNewUser.Clear();
                StreamReader r = new StreamReader(openFileDlg.FileName,
Encoding.Default);
                string line;
                while ((line = r.ReadLine()) != null)
                {
                    string[] vals = line.Split('\t');

                    if (vals.Length >= 4)
                    {
                        DataRow row =
dataSetDB.DataTableSearchTermsNewUser.NewRow();
                        row["Term"] = vals[0];
```

```
                    row["TermDT"] = vals[2];
                    row["respID"] = vals[2];
                    row["DTHour"] = vals[3];

                    dataSetDB.DataTableSearchTermsNewUser.Rows.Add(row);
                }
            }
            r.Close();
        }
    }

    private string [] splitTexttoWords(string text)
    {
        string[] split = text.Split(new Char[] { ' ', ',', '.', ':', '\t' });

        string[] w = new string[split.Length] ;
        int i = 0;
        foreach (string s in split)
        {

            if (s.Trim().Length >= 5)
            {
                w[i] = s.Trim();
                i++;
            }
        }
        return w;
    }

    private void buttonINewResourceVector_Click(object sender, EventArgs e)
    {
        DataTable DTURLWords = new DataTable();
        DTURLWords.Columns.Add("URL");
        DTURLWords.Columns.Add("Term");

        DataTable DTURLWordsCount = new DataTable();
        DTURLWordsCount.Columns.Add("URL");
        DTURLWordsCount.Columns.Add("Term");
        DTURLWordsCount.Columns.Add("Count");

        string[] URLTextDistinct = { "URL" , "pageText" };
        DataTable dtDistinctURLText = GetDistinctRecords(dataSetDB.dtPageText,
URLTextDistinct);


        foreach (DataRow DRURL in dtDistinctURLText.Rows)
        {
            string pageText = DRURL["pageText"].ToString();
            foreach(string word in splitTexttoWords(pageText).Distinct())
            {
                DTURLWords.Rows.Add(DRURL["URL"].ToString(),word);
            }

            var result = from row in DTURLWords.AsEnumerable()
                         group row by row.Field<string>("Term") into grp
                         select new
                         {
                             Term = grp.Key,
                             MemberCount = grp.Count()
                         };
            foreach (var t in result)
            {
                DTURLWordsCount.Rows.Add(DRURL["URL"], t.Term, t.MemberCount);
```

```
                }

            }


            gridControlIRNewVector.DataSource = DTURLWordsCount;
            string[] TermText = { "Term" };
            tableTermStatic = GetDistinctRecords(DTURLWordsCount, TermText);




        }

        private void tabControl1_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (tabControl1.SelectedIndex == 3)
            {
                gridControlDinamic.DataSource = tableTermDinamic;
                gridControlStatic.DataSource = tableTermStatic;

            }
        }

    }
}
```

ПРИЛОЖЕНИЕ 9. *SQL*-СКРИПТЫ СОЗДАНИЯ КОМПОНЕНТОВ БД *InternetDB*

Необходимые *SQL*-скрипты для создания таблиц *az_cities*, *az_resps* и *az_visits* в БД *InternetDB*:

```
CREATE TABLE [dbo].[az_cities]
(
      [city_id] [int] NOT NULL,
      [city] [varchar](255) NOT NULL
      PRIMARY KEY CLUSTERED ([city_id] ASC)
)
```

Описание столбцов таблицы *[dbo].[az_cities]*:

city_id - идентификатор города, для Москвы = 1, Санкт-Петербурга = 2, Другие города России >= 3; Первычный ключ;

city - Наименование города;

```
CREATE TABLE [dbo].[az_resps]
(
      [resp_id] [int] NOT NULL,
      [city_id] [int] NOT NULL,
      [sex] [tinyint] NOT NULL,
      [bday] [datetime] NOT NULL,
      [firstname] [varchar](255) NULL,
      [middlename] [varchar](255) NULL,
      [lastname] [varchar](255) NULL,
      PRIMARY KEY CLUSTERED ([resp_id] ASC)
)

ALTER TABLE [dbo].[az_resps]  WITH CHECK ADD FOREIGN KEY([city_id])
REFERENCES [dbo].[az_cities] ([city_id])
```

Описание столбцов таблицы *[dbo].[az_resps]*:

resp_id - Идентификатор интернет-пользователя; Первычный ключ таблицы;

city_id - Идентификатор города в котором проживает интернет-пользователь; Внешний ключ, привязан к таблице *[dbo].[az_cities]*

sex - Для мужского пола будет = 1; женского пола = 2;

bday - дата рождения интернет пользователя;

firstname, middlenmae и lastname - формируют полное имя ИП;

```
CREATE TABLE [dbo].[az_visits]
(
```

```
      [visit_id] [bigint] NOT NULL,
      [resp_id] [int] NOT NULL,
      [day_id] [int] NOT NULL,
      [dt] [datetime] NOT NULL,
      [url] [varchar](max) NULL,
PRIMARY KEY CLUSTERED ([visit_id] ASC)
)

ALTER TABLE [dbo].[az_visits]  WITH CHECK ADD FOREIGN KEY([resp_id])
REFERENCES [dbo].[az_resps] ([resp_id])
```

Описание столбцов таблицы заходов *[DBO].[az_visits]*:

visit_id - Идентификатор захода на *URL*; Первичный ключ таблицы *[dbo].[az_visits]*;

resp_id - Идентификатор интернет-пользователя; Внешний ключ, привязан к таблице *[dbo].[az_resps]*;

day_id - Идентификатор дня; получен вычислительным путём с помощью функции *DATEDIFF()*;

dt - Момент захода интернет-пользователя *YYYY-MM-DD HH:MM:SS.MSC*;

URL - *url* страницы на которой выполняет заход ИП.

Дополнительные сущности в БД заходов для работы с соц-дем группами:

```
CREATE TABLE [dbo].[az_sd]

(
      [sd_id] [int] NOT NULL,
      [Sex] [tinyint] NOT NULL,
      [so_dem] [varchar](255) NOT NULL,
      [Min_Age] [tinyint] NOT NULL,
      [Max_Age] [tinyint] NOT NULL,
PRIMARY KEY CLUSTERED ([sd_id] ASC)
)
```

Описание столбцов таблицы *[dbo].[az_sd]*:

sd_id - Идентификатор социально-демографической группы; Первичный ключ таблицы *[dbo].[az_sd]*;

Sex - Если мужской пол то = 1, женский пол будет = 2;

sodem - Наименование социально-демографической группы; Всего 10 групп.

Min_Age - Минимальный возраст (включительно) соответствующей *sodem* группы;

Max_Age - Максимальный возраст (включительно) соответствующей *sodem* группы;

Функция *fnGetDomainFromUrl* получения доменного имени из *URL*:

```
CREATE FUNCTION [dbo].[fnGetDomainFromUrl](@URL varchar(2200))
RETURNS varchar(512)
AS
begin
  DECLARE
    @S varchar(512),
    @beg int,
    @end int
    SET @URL = LTRIM(RTRIM(ISNULL(@URL, '')))

    IF(@URL = '#AuFURL#')
     begin
     SET @S = @URL
     RETURN @S
     end

    SET @beg = CHARINDEX('//', @URL) + 2
    SET @end = CHARINDEX('/', @URL, @beg)
    IF(@beg > @end)
      SET @S = ''
    ELSE
      SET @S = SUBSTRING(@URL, @beg, @end-@beg)
    RETURN @S
end
```

Функция *[dbo].[fnGetDomainFromUrl] (<URL>)* извлекает доменное имя второго или третьего уровня из *URL*.

Например:

```
Declare @URL varchar(2200)
SET @URL = 'http://support.microsoft.com/kb/811031'
select [fil].[fnGetDomainFromUrl](@URL)
```

возращает : *support.microsoft.com*

Таблица доменов:

```
CREATE TABLE [dbo].[az_domain]
(
      domain_id - int not null identity(1,1) Primary Key,
      domain - varchar(512),
      is_search - tinyint default 0
)
```

Описание столбцов таблицы *[dbo].[az_domain]*:
domain_id – Идентификатор домена; Первычный ключ таблицы *[dbo].[az_domain]*;

domain – Домен сайта на который интернет пользователь выполняет заход;

is_search – столбец признак поисковой системы (0 – ложь; 1 – истина);

Уникальные *URL* записываем в новую таблицу *[DBO].[az_pages]*:

```
CREATE TABLE [dbo].[az_pages]
(
      page_id – bigint not null identity(1,1) Primary Key,
      url – varchar(MAX) – unique index,
      domain_id – int foreign key references dbo.az_domain(domain_id)
)
```

Описание столбцов таблицы *[dbo].[az_pages]*:

page_id – Идентификатор страницы; Первычный ключ таблицы *[dbo].[az_domain]*;

url – уникальная *URL*-страница сайта на который ИП выполняет заход;

domain_id – Идентификатор домена; Внешний ключ, привязан к таблице *[dbo].[az_domain]*;

Создание таблицы масок *[dbo].[az_mask]*:

```
CREATE TABLE [dbo].[az_mask]
(
      mask_id  int not null identity(1,1) Primary Key,
      mask  varchar(512)
)

alter table [InternetDB].[dbo].[az_mask]
  add constraint unique_mask unique (mask)
```

Описание столбцов таблицы *[dbo].[az_mask]*:

mask_id – Идентификатор маски; Первычный ключ таблицы *[dbo].[az_mask]*.

mask – Маска для *URL*;

Создание таблицы масок *[dbo].[az_domain_mask]*:

```
CREATE TABLE [dbo].[az_domain_mask](
      [mask_id] [int] NOT NULL foreign key references [dbo].[az_mask](mask_id),
      [domain_id] [int] NOT NULL foreign key references
[dbo].[az_domain](domain_id),
      PRIMARY KEY([mask_id], domain_id)
      )
```

Описание столбцов таблицы *[dbo].[az_domain_mask]*:

mask_id – Идентификатор маски; Внешний ключ, привязан к таблице *[dbo].[az_mask]*.

domain_id – Идентификатор домена; Внешний ключ, привязан к таблице *[dbo].[az_domain]*.

Обобщённый код получения отфильтрованной *URL* строки:

```
declare @mask varchar(500)
declare @domain_id bigint
```

```
declare curs cursor local fast_forward for
select A.mask, B.domain_id from dbo.az_mask A
inner join
dbo.az_domain_mask B
ON A.mask_id = B.mask_id

open curs
fetch curs into @mask, @domain_id
while @@FETCH_STATUS = 0
begin
      update dbo.az_pages
      set decoded_url = REPLACE(decoded_url,@mask,'#')
      where domain_id = @domain_id

      fetch curs into @mask, @domain_id
end
close curs
deallocate curs
```

Добавление таблиц *az_key_word* в БД *InternetDB*:

```
create table dbo.az_key_word
(
      key_word varchar(30) not null,
      domain_id int not null
      foreign key references dbo.az_domain(domain_id),
      primary key (key_word, domain_id)
)
```

Описание столбцов таблицы dbo.az_key_word:
key_word - Ключевое слово, после которого идёт поисковое выражение.
domain_id - Идентификатор домена сайта; Внешний ключ, привязан к таблице *az_domain*.
key_word и *domain_id* - Первычний ключ таблицы;

Заполняем таблицу *dbo.az_key_word*:

```
update p
set p.decoded_url = null from
dbo.az_pages p inner join dbo.az_key_word kw
ON p.domain_id = kw.domain_id
and p.decoded_url like '%'+kw.key_word+'%'


update p
set p.decoded_url = REPLACE(p.decoded_url,'&#','#')
FROM dbo.az_pages p

update dbo.az_pages set decoded_url = decoded_url+'#'

-- Получаем поисковые слова.
update p
set p.decoded_url =
SUBSTRING(p.decoded_url,CHARINDEX(kw.key_word,p.decoded_url)+len(kw.key_word),CHAR
INDEX('#',p.decoded_url,CHARINDEX(kw.key_word,p.decoded_url)+len(kw.key_word))-
(CHARINDEX(' kw.key_word ',p.decoded_url)+ len(kw.key_word)))
from
dbo.az_pages p inner join dbo.az_key_word kw
ON p.domain_id = kw.domain_id
```

```
update dbo.az_pages set decoded_url = REPLACE(decoded_url,'&',' ')
update dbo.az_pages set decoded_url = rtrim(decoded_url)
```

Создаём таблицу слов *[dbo].[az_words]*:

```
Create table [dbo].[az_words]
(
      word_id bigint not null identity(1,1) Primary key,
      word varchar(40) not null,
      power bigint not null default 0
)

alter table [az_words]
add constraint word_unq unique (word)
```

Описание столбцов таблицы *[dbo].[az_words]*:

word_id – Идентификатор слова; Первичный ключ таблицы *[dbo].[az_words]*.

word – Поисковое слово.

power – Число появления слова в уникальных поисковых строках таблицы *[dbo].[az_pages]*

Заполняем таблицу слов *[dbo].[az_words]*:

```
insert [dbo].[az_words]
SELECT  rs.Item as word, COUNT(distinct p.page_id) cnt_pages from dbo.az_pages p
cross apply [dbo].[ribr_split](rtrim(decoded_url),' ') rs
where p.decoded_url is not null
and len(rs.Item) > 1 and LEN(rs.Item) <= 40
GROUP BY rs.Item
HAVING COUNT(distinct p.page_id)>1
order by cnt_pages desc

DELETE from dbo.az_words where word like '%??%'
```

Создание таблицы *[dbo].[az_pages_words]*:

```
Create table [dbo].[az_pages_words]
(
      page_id bigint not null
            Foreign Key references [dbo].[az_pages](page_id),
      word_id bigint not null
            Foreign key references [dbo].[az_words](word_id)
-- нет UNIQUE т.к. возможно повторных слов в поисковой строке.
)
```

Описание столбцов таблицы *[dbo].[az_pages_words]*:

page_id – Идентификатор *URL*, на которой выполнен заход; Внешний ключ таблицы *[dbo].[az_pages]*.

word_id – Идентификатор слова; Внешний ключ таблицы *[dbo].[az_words]*.

Заполняем таблицу *[dbo].[az_pages_words]*:

```
insert az_pages_words (page_id, word_id)
select A.page_id, B.word_id, from
(select * from [dbo].[az_pages] where decoded_url is not null) as A
inner join
(select * from [dbo].[az_words] where power>1) as B
ON ' '+A.decoded_url+' ' like '% '+replace(B.word,'%','')+' %'
```

Функция преобразования даты рождения в социальную группу:

```
CREATE FUNCTION [dbo].[az_sd_detect]
(
      -- Add the parameters for the function here
      @bd datetime, @sex tinyint
)
RETURNS int
AS
BEGIN
      -- Declare the return variable here
      DECLARE @age int;
      declare @SD_ID int;

      -- Add the T-SQL statements to compute the return value here
      SELECT @age = CASE WHEN month(SYSDATETIME()) < month(@bd) OR
                                                (month(SYSDATETIME()) = month(@bd)
AND day(SYSDATETIME()) < day(@bd)) THEN datediff(year, @bd, SYSDATETIME()) - 1
                                               ELSE datediff(year,
@bd,SYSDATETIME()) END
    SELECT @SD_ID = SD_ID FROM dbo.az_sd WITH (nolock)
    where  @sex = Sex AND @age BETWEEN Min_Age AND Max_Age

      -- Return the result of the function
      RETURN @SD_ID

END
```

Функция получения слов из строки:

```
CREATE FUNCTION [dbo].[az_split]
/* This function is used to split up multi-value parameters */
(
@ItemList NVARCHAR(4000),
@delimiter CHAR(1)
)
RETURNS @IDTable TABLE (Item VARCHAR(500))
AS
BEGIN
DECLARE @tempItemList NVARCHAR(4000)
SET @tempItemList = @ItemList

DECLARE @i INT
DECLARE @Item NVARCHAR(4000)

SET @tempItemList = REPLACE (@tempItemList, @delimiter + ' ',
@delimiter)
SET @i = CHARINDEX(@delimiter, @tempItemList)
```

```
WHILE (LEN(@tempItemList) > 0)
BEGIN
IF @i = 0
SET @Item = @tempItemList
ELSE
SET @Item = LEFT(@tempItemList, @i - 1)

INSERT INTO @IDTable(Item) VALUES(substring(@Item,1,500))

IF @i = 0
SET @tempItemList = ''
ELSE
SET @tempItemList = RIGHT(@tempItemList, LEN(@tempItemList) - @i)

SET @i = CHARINDEX(@delimiter, @tempItemList)
END
RETURN
END
```

Функция кодирования и декодирования с применением библиотеки *HttpUtility.dll*:

Для автоматизации процесса декодирования поисковых строк, необходимо написать специальную библиотеку динамической компоновки (*DLL*) на *C#* с подключением *HttpUtility.dll*. Новая библиотека динамической компоновки будет интегрирована в ядро сервера БД для дальнейшего применения доступных функций кодирования и декодирования.

Когда основные таблицы созданы, приступим к расшифрованию *ULR* поисковых страниц. Для этого неоходимо сначала написапь динамическую библиотеку компоновки, зарегистрировать её на *SQL* сервере, затем объявить нужные нам функции.

С помощью среды программирования *Microsoft Visual Studio 2010*, приступаем к созданию динамической библиотеки компоновки:

```
--  Подключаем библиотеки windows, .NET и Microsoft SQL Server;
using System;
using System.Text;
using HttpUtilityTest;
using Microsoft.SqlServer.Server;

-- Объявляем пространсиво имён SQLHTTPUtility;
namespace SQLHTTPUtility
{
    -- Создаём класс функций UrlDecodeFunctions
    public partial class UrlDecodeFunctions
    {

        [SqlFunction(IsDeterministic = true, IsPrecise = true)]
        -- Первая функция декодирования;
        public static string UrlDecode(string encodeString)
```

```
        {
            return HttpUtility.UrlDecode(encodeString);
        }


        [SqlFunction(IsDeterministic = true, IsPrecise = true)]
        -- Вторая функция декодирования;
        public static string UrlDecode2(string encodeString, string codepage = ""
)
        {
            if (codepage == "")
                return HttpUtility.UrlDecode(encodeString);
            else
            {
                Encoding encode;
                try
                {
                    encode = System.Text.Encoding.GetEncoding(codepage);
                }
                catch (Exception)
                {
                    encode = Encoding.Default;
                }

                return HttpUtility.UrlDecode(encodeString, encode);
            }
        }
    }
}
```

После компиляции выше представленнового кода и получении скомпилированного файла "*SQLHTTPUtility .dll*", приступаем к регистрации этой библотеки в ядро *MS SQL Server 2012*, для этого нужны права администратора с целю подключения интеграции со средой *CLR* (*sp_configure 'clr enabled', 1*). Сейчас всё готово для интеграции динамической библиотеки "*SQLHTTPUtility.dll*" в ядро сервера БД. Приступаем к созданию управляемый модуль приложений (*ASSEMBLY*):

```
CREATE ASSEMBLY [SQLHTTPUtility]
AUTHORIZATION [dbo]
FROM 'C:\Windows\SQLHTMLUTILITY_DLL\SQLHTTPUtility.dll'
WITH PERMISSION_SET = SAFE
```

Управляемый модуль приложений "*SQLHTTPUtility*" создан, приступаем к объявлению нужных нам функций декодирования содержания *URL*-страниц:

```
USE [InternetDB]
GO
-- Первая функция декодирования;
CREATE FUNCTION [dbo].[UrlDecode](@encodeString [nvarchar](max))
RETURNS [nvarchar](max) WITH EXECUTE AS CALLER
AS
EXTERNAL NAME [SQLHTTPUtility].[SQLHTTPUtility.UrlDecodeFunctions].[UrlDecode]
GO
```

```
-- Вторая функция декодирования;
CREATE FUNCTION [dbo].[UrlDecode2](@encodeString [nvarchar](max), @codepage
[nvarchar](30))
RETURNS [nvarchar](max) WITH EXECUTE AS CALLER
AS
EXTERNAL NAME [SQLHTTPUtility].[SQLHTTPUtility.UrlDecodeFunctions].[UrlDecode2]
GO
```

Исходный *SQL*-код очистки содержания интернет-страниц:

```
USE [HTML]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER function [dbo].[az_HTML_value_parser](@tag_value varchar(5000))
returns varchar(5000)
as
BEGIN
declare @t Table (mask varchar(50))
insert @t(mask) values
('h3||'),
('h1||'),
('h2||'),
('h4||'),
('h5||'),
('h6||'),
('p||'),
('title||'),
(' '),
('&raquo;'),
('&ndash;'),
('&laquo;'),
('&mdash;');

while PATINDEX('%<%>%', @tag_value) > 0
begin
      declare @mask varchar(1000)
      select @mask = substring(@tag_value, charindex('<',@tag_value),
charindex('>',@tag_value)- charindex('<',@tag_value)+1)
      set @tag_value = REPLACE(@tag_value,@mask, ' ')
end

declare curs cursor local fast_forward for
select mask from @t
open curs
fetch curs into @mask
while @@FETCH_STATUS = 0
BEGIN
      SET @tag_value = REPLACE(@tag_value,@mask, ' ')
      fetch curs into @mask
END
close curs
deallocate curs
SET @tag_value = REPLACE(@tag_value,'  ', ' ')
return @tag_value
END


SELECT [page_id], count([Item]) as cnt_words
```

```
into #t1
FROM [HTML].[dbo].[az_words_count_in_pages]
group by [page_id]
order by cnt_words desc


SELECT A.tema, B.page_id
into #t2
FROM
(SELECT [tema], [item]
  FROM [HTML].[dbo].[new_tema]) AS A
  INNER JOIN
 (SELECT [page_id]
      ,[Item]
  FROM [HTML].[dbo].[az_words_count_in_pages]) AS B
  ON A.item = B.item
  GROUP BY  A.tema, B.page_id

  select B.tema, SUM(A.cnt_words) as cluster_size
  from #t1 A inner join #t2 B
  ON A.page_id = B.page_id
  GROUP BY B.tema
  ORDER BY cluster_size desc

  DROP TABLE #t1
  DROP TABLE #t2
```

ПРИЛОЖЕНИЕ 10. ТАБЛИЦА СООТВЕТСТВИЯ КОДИРОВАННЫХ СИМВОЛОВ В ПОИСКОВЫХ СИСТЕМАХ *RU*-НЕТА

Код в системе HEX	Символ	кодировка	Код в системе HEX	символ	кодировка	Код в системе HEX	символ	кодировка	Код в системе HEX	символ	Кодировка
410	А	90	420	Р	A0	430	А	B0	440	р	80
411	Б	91	421	С	A1	431	Б	B1	441	с	81
412	В	92	422	Т	A2	432	В	B2	442	т	82
413	Г	93	423	У	A3	433	Г	B3	443	у	83
414	Д	94	424	Ф	A4	434	Д	B4	444	ф	84
415	Е	95	425	Х	A5	435	Е	B5	445	х	85
416	Ж	96	426	Ц	A6	436	Ж	B6	446	ц	86
417	З	97	427	Ч	A7	437	З	B7	447	ч	87
418	И	98	428	Ш	A8	438	И	B8	448	ш	88
419	Й	99	429	Щ	A9	439	Й	B9	449	щ	89
41A	К	9A	42A	Ъ	AA	43A	К	BA	44A	ъ	8A
41B	Л	9B	42B	Ы	AB	43B	Л	BB	44B	ы	8B
41C	М	9C	42C	Ь	AC	43C	М	BC	44C	ь	8C
41D	Н	9D	42D	Э	AD	43D	Н	BD	44D	э	8D
41E	О	9E	42E	Ю	AE	43E	О	BE	44E	ю	8E
41F	П	9F	42F	Я	AF	43F	П	BF	44F	я	8F

ПРИЛОЖЕНИЕ 11. ИСХОДНЫЙ КОД ПРОГРАММОГО МОДУЛЯ *HTMLDocDom*

```
// необходимо скачать HtmlAgilityPack.dll для работы с DOM-деревом
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Web;
using HtmlAgilityPack;
using System.Net;
using System.IO;

//http://olussier.net/2010/03/30/easily-parse-html-documents-in-csharp/
namespace HTMLDocDom
{
    public partial class Form1 : Form
    {
        string Fname4Save = string.Empty;
        bool NotSaved = false;
        object[][] htmlNodesArray = new object[4][];
        int page_id;
        int HTML_element_id;
        string tag_value;
        Label page_id_Label = new Label();


        public Form1()
        {
            InitializeComponent();
            URL_listBox.SelectionMode = SelectionMode.One;
            wb.StatusTextChanged += new EventHandler(wb_StatusTextChanged);
        }

        void wb_StatusTextChanged(object sender, EventArgs e)
        {
            toolStripLabel1.Text = wb.StatusText;
        }

        private void DOMbutton_Click(object sender, EventArgs e)
        {
            wb.Navigate(URLtextBox.Text);



            //Загрузка страницы в браузере для наглядности.
            wb.Navigate(URLtextBox.Text);
            //Объявляем WebClient.
            WebClient wbClient = new WebClient();
            //HtmlAgilityPack - HtmlDocument для получения DOM-модель в
виде дерева.
```

```
            HtmlAgilityPack.HtmlDocument wbHTML = new
HtmlAgilityPack.HtmlDocument();
            //Загружаем страницу с кодировкой по умолчанию.
            wbHTML.Load(wbClient.OpenRead(URLtextBox.Text), Encoding.UTF8,
true);//, Encoding.Default);
            //wbHTML.Load(  (wbClient.OpenRead(URLtextBox.Text));
            //tmlAgilityPack - Начинаем с корневого узла HtmlDocument-a
ROOT.
            HtmlNode someNode = wbHTML.DocumentNode;
            //tmlAgilityPack - Находим все узлы DOM-модели документа;
            HtmlNodeCollection allLinks = someNode.SelectNodes(".//*");
            //Рассматриваем каждый узел дерева DOM-модели.
            foreach (HtmlNode link in allLinks)
            {
                string str = String.Empty;
                //Записываем цыпочку узлов до корня ROOT.
                str = link.XPath;
                TB_filler(treeTB, str);
                //Проводим отбор тэгов: выбераем текстовые зоголовки,
строки, параграфы и гиперсылки.
                string str2 = String.Empty;
                string linkName = link.Name;
                string filterExpression = "[HTML_element_name] =
'<"+link.Name+">'"+" AND "+"[take] = 'true'";
                DataRow[] row = dSURL.dtTags.Select(filterExpression);
                if (row.Length > 0)
                {
                    int HTML_elementID =
Convert.ToInt32(row[0]["HTML_element_id"].ToString());
                    switch (linkName)
                    {
                        case "a":
                            if (link.Attributes["href"] != null)
                            {
                                string str3 =
link.Attributes["href"].Value.ToString();
                                if (str.Length > 0)
                                    TB_filler(linksTB, str3);
                                DataRow DR = dSURL.dtHTMLvalue.NewRow();
                                DR["HTML_element_id"] = HTML_elementID;
                                DR["Page_id"] = page_id;
                                DR["HTML_element_value"] = str3;
                                dSURL.dtHTMLvalue.Columns[0].AutoIncrement
= true;
                                dSURL.dtHTMLvalue.Rows.Add(DR);
                            }
                            break;
                        default:
                            {
                                str2 = link.Name + "||" +
link.WriteContentTo();
                                DataRow DR = dSURL.dtHTMLvalue.NewRow();
                                //DR["id"] = dSURL.dtHTMLvalue.Count + 1;
                                DR["HTML_element_id"] = HTML_elementID;
                                DR["Page_id"] = page_id;
                                DR["HTML_element_value"] = str2;
                                dSURL.dtHTMLvalue.Rows.Add(DR);
```

```
                            }
                            break;
                        }
                    }

                    //if (row.Length <= 0)
                       // MessageBox.Show("NO");
                    /*
                    switch (linkName)
                    {
                        case "title":
                            str2 = link.Name +"||"+ link.WriteContentTo();
                            break;
                        case "h1":
                            str2 = link.Name + "||" + link.WriteContentTo();
                            break;
                        case "h2":
                            str2 = link.Name + "||" + link.WriteContentTo();
                            break;
                        case "h3":
                            str2 = link.Name + "||" + link.WriteContentTo();
                            break;
                        case "h4":
                            str2 = link.Name + "||" + link.WriteContentTo();
                            break;
                        case "h5":
                            str2 = link.Name + "||" + link.WriteContentTo();
                            break;
                        case "h6":
                            str2 = link.Name + "||" + link.WriteContentTo();
                            break;
                        case "p":
                            str2 = link.Name + "||" + link.WriteContentTo();
                            break;


                        case "a":
                            if (link.Attributes["href"] != null)
                            {
                                string str3 =
link.Attributes["href"].Value.ToString();
                                if (str.Length > 0)
                                    TB_filler(linksTB, str3);
                            }
                            break;
                        default:
                            break;

                    }
                    */
                    if (str2.ToString() != "")
                        TB_filler(headparaTB, str2);

                }
            }

            private void TB_filler(TextBox TB, string str)
```

```
        {
            TB.AppendText("-" + str + Environment.NewLine + ";");
        }

        private void Save(TextBox TB)
        {
            SaveFileDialog saveFileDlg = new SaveFileDialog();
            saveFileDlg.DefaultExt = "txt";
            saveFileDlg.Filter = "Text files (*.txt)|*.txt|All files
(*.*)|*.*";
            if (saveFileDlg.ShowDialog() == DialogResult.OK)
            {
                StreamWriter w = new StreamWriter(saveFileDlg.FileName,
false, Encoding.Default);
                foreach (string str in TB.Lines)
                {
                    w.WriteLine(str);
                }
                w.Close();
            }

        }

        //Одно меню для нескольких textBox - Обрабочик.
        private void SavefileToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            var menuItem = (ToolStripMenuItem)sender;
            var toolStrip = menuItem.Owner;
            var ctrl =
(ContextMenuStrip)toolStrip.DataBindings.BindableComponent.DataBindings.Con
trol;
            var textBox = (TextBox)ctrl.SourceControl;
            Save(textBox);
        }

        private void wb_DocumentCompleted(object sender,
WebBrowserDocumentCompletedEventArgs e)
        {
            toolStripLabel1.Text = "Готово";
            URLtextBox.Text = wb.Url.ToString();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            wb.Navigate(URLtextBox.Text);
        }

        private void Load_URL_Click(object sender, EventArgs e)
        {
            if (DialogResult.OK == openFileDlg.ShowDialog())
            {
                dSURL.dtPages.Clear();
                //SocDemList.Clear();
                StreamReader r = new StreamReader(openFileDlg.FileName,
Encoding.Default);
                string line;
```

```
                while ((line = r.ReadLine()) != null)
                {
                    string[] vals = line.Split('\t');
                    //w.WriteLine(lname + "\t" + fname + "\t" + email +
"\t" + pass + "\t" + age.ToString() + "\t" + sd.Sex);
                    if (vals.Length >= 3)
                    {
                        DataRow row = dSURL.dtPages.NewRow();
                        row["Page_id"] = vals[0];
                        row["URL"] = vals[2];
                        row["cnt"] = vals[2];
                       /* if (vals.Length > 6)
                        {
                            row["rez"] = vals[6];
                        }
                        */
                        dSURL.dtPages.Rows.Add(row);
                        URL_listBox.Items.Add(vals[2]);
                        //SocDemList.Add(new SocDem(vals[0], vals[2],
vals[2], vals[3], Convert.ToInt32(vals[4]), vals[5][0]));
                    }
                }
                r.Close();
            }
            //MessageBox.Show(page_id_Label.Text);
            DataRowView DRV = (DataRowView)dtPagesBindingSource.Current;
            page_id_Label.Text = DRV.Row["Page_id"].ToString();
            //MessageBox.Show(page_id_Label.Text);
            page_id = Convert.ToInt32(page_id_Label.Text);
            URL_listBox.SelectedIndex = 0;
            URLtextBox.Text = URL_listBox.SelectedItem.ToString();
            toolStrip1.Text = "Открытие файла URL";
            wb.Navigate(URLtextBox.Text);

        }

        private void button3_Click(object sender, EventArgs e)
        {
            DataRow[] foundTags;
            foundTags = dSURL.dtTags.Select("[take] = true");
            MessageBox.Show(foundTags.Count().ToString());
        }

        private void timer_next_URL_Tick(object sender, EventArgs e)
        {
            if (dtPagesBindingSource.Position != dSURL.dtPages.Rows.Count -
1)
                {
                    MessageBox.Show("dtydgsys");
                    dtPagesBindingSource.MoveNext();
                    URL_listBox.SetSelected(dtPagesBindingSource.Position,
true);
                    DataRowView DRV =
(DataRowView)dtPagesBindingSource.Current;
                    page_id_Label.Text = DRV.Row["Page_id"].ToString();
                    //MessageBox.Show(page_id_Label.Text);
                    page_id = Convert.ToInt32(page_id_Label.Text);
```

```
                    URLtextBox.Text = URL_listBox.SelectedItem.ToString();
                    wb.Navigate(URLtextBox.Text);
                    //DOMbutton_Click(null,null);
                }
        }

        private void Load_TAGs_Click(object sender, EventArgs e)
        {
            if (DialogResult.OK == openFileDlg.ShowDialog())
            {
                dSURL.dtTags.Clear();
                //SocDemList.Clear();
                StreamReader r = new StreamReader(openFileDlg.FileName,
Encoding.Default);
                Fname4Save = openFileDlg.FileName;
                string line;
                while ((line = r.ReadLine()) != null)
                {
                    string[] vals = line.Split('\t');
                    //w.WriteLine(lname + "\t" + fname + "\t" + email +
"\t" + pass + "\t" + age.ToString() + "\t" + sd.Sex);
                    if (vals.Length >= 4)
                    {
                        DataRow row = dSURL.dtTags.NewRow();
                        row["HTML_element_id"] = vals[0];
                        row["HTML_element_name"] = vals[2];
                        row["HTML_element_type"] = vals[2];
                        row["take"] = vals[3];

                        /* if (vals.Length > 6)
                         {
                             row["rez"] = vals[6];
                         }
                         */
                        dSURL.dtTags.Rows.Add(row);
                        Tag_checkedListBox.Items.Add(vals[2].ToString(),
Convert.ToBoolean(vals[3]));
                        //SocDemList.Add(new SocDem(vals[0], vals[2],
vals[2], vals[3], Convert.ToInt32(vals[4]), vals[5][0]));
                    }
                }
                r.Close();
            }

            toolStrip1.Text = "Открытие файла TAG-ов";
            //wb.Navigate(tbUrl.Text);

        }

        private void Tag_checkedListBox_ItemCheck(object sender,
ItemCheckEventArgs e)
        {
            if (Tag_checkedListBox.SelectedIndex >=0)
                if
(Tag_checkedListBox.GetItemChecked(Tag_checkedListBox.SelectedIndex) ==
false)
                {
```

```
                    //is checked
                    string filterExpression = "[HTML_element_id] = " +
HTML_element_id.ToString();
                    foreach (DataRow row in
dSURL.dtTags.Select(filterExpression))
                    {
                        row["take"] = "true";
                        NotSaved = true;
                    }

                }
                else
                if
(Tag_checkedListBox.GetItemChecked(Tag_checkedListBox.SelectedIndex) ==
true)
                {
                    //is unchecked
                    string filterExpression = "[HTML_element_id] = '" +
HTML_element_id.ToString()+"'";
                    foreach (DataRow row in
dSURL.dtTags.Select(filterExpression))
                    {
                        row["take"] = "false";
                        NotSaved = true;
                    }
                }
        }

        private void Tag_checkedListBox_SelectedValueChanged(object sender,
EventArgs e)
        {
            string tag_name = Tag_checkedListBox.SelectedItem.ToString();
            string filterExpression = "[HTML_element_name] = '" + tag_name
+ "'";
            DataRow [] DR = dSURL.dtTags.Select(filterExpression);
            //MessageBox.Show(DR[0]["HTML_element_id"].ToString());
            HTML_element_id =
Convert.ToInt32(DR[0]["HTML_element_id"].ToString());
        }

        private void Save_tags()
        {
            dSURL.AcceptChanges();
            //saveFileDlg.FileName = Fname4Save;
            saveFileDlg.Filter = "txt files (*.txt)|*.txt|dat files
(*.dat)|*.dat";
            if (saveFileDlg.ShowDialog() == DialogResult.OK)
            {
                StreamWriter w = new StreamWriter(saveFileDlg.FileName,
false, Encoding.Default);
                foreach (dSURL.dtTagsRow row in dSURL.dtTags.Rows)
                {
                    w.WriteLine(row.HTML_element_id + "\t" +
row.HTML_element_name + "\t" + row.HTML_element_type + "\t" + row.take);
                }
                w.Close();
```

```
            }
            NotSaved = false;
            //tsChanged.Text = "";
        }

        private void Save_result()
        {
            dSURL.AcceptChanges();
            //saveFileDlg.FileName = Fname4Save;
            saveFileDlg.Filter = "txt files (*.txt)|*.txt|dat files
(*.dat)|*.dat";
            if (saveFileDlg.ShowDialog() == DialogResult.OK)
            {
                StreamWriter w = new StreamWriter(saveFileDlg.FileName,
false, Encoding.Default);
                foreach (dSURL.dtHTMLvalueRow row in
dSURL.dtHTMLvalue.Rows)
                {
                    w.WriteLine(row.id + "\t" + row.HTML_element_id + "\t"
+ row.page_id + "\t" + row.HTML_element_value);
                }
                w.Close();

            }
            //NotSaved = false;
            //tsChanged.Text = "";
        }


        private void page_move()
        {
            if (dtPagesBindingSource.Position != dSURL.dtPages.Rows.Count -
1)
            {
                dtPagesBindingSource.MoveNext();
                URL_listBox.SetSelected(dtPagesBindingSource.Position,
true);
                DataRowView DRV =
(DataRowView)dtPagesBindingSource.Current;
                page_id_Label.Text = DRV.Row["Page_id"].ToString();
                //MessageBox.Show(page_id_Label.Text);
                page_id = Convert.ToInt32(page_id_Label.Text);
                URLtextBox.Text = URL_listBox.SelectedItem.ToString();
                wb.Navigate(URLtextBox.Text);
                headparaTB.Clear();
                treeTB.Clear();
                linksTB.Clear();
                DOMbutton_Click(null,null);
            }
        }
        private void Form1_FormClosing(object sender, FormClosingEventArgs
e)
        {
            if (NotSaved == true)
            {
                Save_tags();
            }
```

```
            Save_result();

        }

        private void button3_Click_1(object sender, EventArgs e)
        {
            page_move();
        }


    }
}
```

ПРИЛОЖЕНИЕ 12. ПРОЦЕДУРЫ КЛАСТЕРНОГО АНАЛИЗА, РЕАЛИЗОВАННЫЕ В СРЕДЕ *MS SQL Server* 2012

Процедуры эксперимента по оценке попадания в целевую группу.

```
USE [InternetDB]
GO
/****** Object:  StoredProcedure [dbo].[az_clustering_step1]    Script Date: 05/31/2014 23:08:38
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[az_clustering_step1] (@day_id_begin int, @node1_coef int, @node2_coef int,
@nodesearch_coef int, @hour_step smallint, @online_hours_day smallint, @online_days_week SMALLINT,
@word_len smallint, @recount bit)
AS
-- EXEC az_clustering_step1 @day_id_begin = 4494, @node1_coef = 10, @node2_coef = 5, @nodesearch_coef =
50, @hour_step = 4, @online_hours_day = 8, @online_days_week = 7, @word_len = 5, @recount = 1;
/*
ALTER DATABASE InternetDB SET RECOVERY SIMPLE;
USE InternetDB;
GO
CHECKPOINT;
GO
CHECKPOINT;
GO
-- Уменьшаем лог-файл БД.
DBCC SHRINKFILE(InternetDB_log, 200);
GO
*/
SET NOCOUNT ON
SET XACT_ABORT ON
-- переменные курсора
declare @curs_cnt_all_day_users int;
declare @curs_day_dt date;

declare @curs3_day_dt date;
declare @curs3_hour_start smallint;
declare @curs3_cnt_resp_id int;

declare @curs4_day_dt date;
declare @curs4_hour_start smallint;
declare @curs4_cnt_page_id int;

declare @curs5_day_dt date
declare @curs5_hour_start smallint
declare @curs5_cnt_obj int;

declare @curs6_day_dt date;
declare @curs6_hour_start smallint;
declare @curs6_cnt_dict_word int;
declare @curs7_page_id int;
declare @curs7_day_id int;
declare @curs7_hour_start smallint;

declare @curs8_day_id int;
declare @curs8_hour_start smallint;

declare @author varchar(100) = 'Author: Zein A.N. - PhD Student, MPEI, VMSS (ZeynAN@mpei.ru)';
PRINT '**********Prepare
Procedure*******************************************************************************';
PRINT '**************************' + @author + '******************';
PRINT
'*************************************************************************************************
***';
-- формируем список ИП у которых не менее 7 часов активности в Интернете в течении недели.
IF OBJECT_ID('tempdb..#resp_day') is not null DROP TABLE #resp_day
CREATE TABLE #resp_day ([resp_id] int not null, [day_id] int not null, [cnt_hours_day] int, PRIMARY KEY
([resp_id],[day_id]))
INSERT #resp_day ([resp_id], [day_id], [cnt_hours_day])
```

```
SELECT [resp_id], [day_id], cnt_hours = COUNT(DISTINCT DATEPART(hour,[dt]))
FROM [dbo].[az_visits] with (nolock)
-- проставляем интервал в неделю
WHERE [day_id] between @day_id_begin AND (@day_id_begin+6)
GROUP BY [resp_id], [day_id]
-- отбор активных ИП с 8 часовым посещением в интернете.
HAVING count(distinct DATEPART(hour,[dt])) >= @online_hours_day
ORDER BY [resp_id];

-- отбор ИП с ежедневным заходом в интернет.
DELETE FROM #resp_day where [resp_id] in
(
	SELECT [resp_id] FROM #resp_day
	GROUP BY [resp_id]
	HAVING COUNT(DISTINCT [day_id]) < @online_days_week
)
OR [resp_id] = -1 -- устраняем не определившихся ИП.

PRINT '---------------------------------Отчёт по количеству ИП по дням---------------------------------
--------'
DECLARE curs CURSOR LOCAL FAST_FORWARD FOR
	SELECT		DATEADD(day,[day_id],'19900101'),  COUNT(DISTINCT [resp_id]) FROM #resp_day
	GROUP BY	DATEADD(day,[day_id],'19900101')
	ORDER BY	DATEADD(day,[day_id],'19900101')
OPEN curs
FETCH curs INTO @curs_day_dt, @curs_cnt_all_day_users
WHILE @@FETCH_STATUS = 0
BEGIN
	PRINT 'Количество ИП в ' + CONVERT(varchar(10), @curs_day_dt, 120) + ': ' +
CONVERT(varchar(10), @curs_cnt_all_day_users) +';'
	FETCH curs INTO @curs_day_dt, @curs_cnt_all_day_users
END
CLOSE curs;
DEALLOCATE curs;
PRINT'';

-- Для каждого часа весь список ИП. этот список нужен для формирования векторов характеристик.
IF OBJECT_ID('tempdb..#resp_current') is not null DROP TABLE #resp_current
CREATE TABLE #resp_current ([resp_id] int not null, [day_id] int not null, [hour] int not null, PRIMARY
KEY ([resp_id], [day_id], [hour]))
INSERT #resp_current ([resp_id], [day_id], [hour])
SELECT V.[resp_id], V.[day_id], hour = DATEPART(hour,V.[dt])
FROM [dbo].[az_visits] V with (nolock)
INNER JOIN
#resp_day R
ON V.[resp_id] = R.[resp_id]
AND V.[day_id] = R.[day_id]
GROUP BY V.[resp_id], V.[day_id], DATEPART(hour,V.[dt])

-- отбираем список ИП в каждом периоде @hour_step
IF OBJECT_ID('tempdb..#resp_currentPeriod') IS NOT NULL DROP TABLE #resp_currentPeriod
CREATE TABLE #resp_currentPeriod ([resp_id] INT not null , [day_id] INT, [hour_start] smallint, PRIMARY
KEY([resp_id], [day_id], [hour_start]))
INSERT #resp_currentPeriod ([resp_id], [day_id], [hour_start])
SELECT fR.[resp_id], fR.[day_id], hour_start = (fR.[hour]-fR.[hour]%@hour_step)
FROM #resp_current fR WHERE fR.hour = (fR.[hour]-fR.[hour]%@hour_step)


PRINT '---------------------------------------Отчёт по количеству ИП по периодам----------------------
------------'
DECLARE curs3 CURSOR LOCAL FAST_FORWARD FOR
	SELECT		DATEADD(day,[day_id],'19900101'), [hour_start], COUNT(distinct [resp_id])  FROM
#resp_currentPeriod
	GROUP BY	DATEADD(day,[day_id],'19900101'), [hour_start]
	ORDER BY	DATEADD(day,[day_id],'19900101'), [hour_start]
OPEN curs3
FETCH curs3 INTO @curs3_day_dt, @curs3_hour_start, @curs3_cnt_resp_id
WHILE @@FETCH_STATUS = 0
BEGIN
	PRINT 'В ' + CONVERT(varchar(10), @curs3_day_dt, 120) + ' в периоде с     ' +
CONVERT(varchar(2), @curs3_hour_start) + '  по      ' + CONVERT(varchar(2), @curs3_hour_start
+(@hour_step-1)) + '  количество ИП оказалось:' + CONVERT(varchar(10), @curs3_cnt_resp_id) +';'
	FETCH curs3 INTO @curs3_day_dt, @curs3_hour_start, @curs3_cnt_resp_id
END
CLOSE curs3;
DEALLOCATE curs3;
PRINT'';

-- Фоормируем список URL-ов для каждого ИП.
```

```
IF OBJECT_ID('tempdb..#resp_hour_page') IS NOT NULL DROP TABLE #resp_hour_page
CREATE TABLE #resp_hour_page([resp_id] int not null, [day_id] int not null, [hour] smallint NOT NULL,
[page_id] int not null, [cnt] int)
INSERT #resp_hour_page([resp_id], [day_id], [hour], [page_id], [cnt])
SELECT V.[resp_id], V.[day_id], hour = DATEPART(hour,V.[dt]), P.[page_id], COUNT(*) as cnt
FROM [dbo].[az_visits] V with (nolock)
INNER JOIN
#resp_day R
ON V.[resp_id] = R.[resp_id]
AND V.[day_id] = R.[day_id]
INNER JOIN
dbo.az_pages P
ON V.[url] = P.[url]
WHERE P.[page_id] >= 1439963 -- берём нужную неделю
GROUP BY V.[resp_id], V.[day_id], DATEPART(hour,V.[dt]), P.[page_id]
ORDER BY [day_id], [hour], [resp_id]

-- получаем таблицу связки ИП, времени(день+период) и ИР
IF @recount = 1 AND OBJECT_ID('resp_page_hour_current_period') IS NOT NULL DROP TABLE
resp_page_hour_current_period
IF OBJECT_ID('resp_page_hour_current_period') IS NULL
BEGIN
       CREATE TABLE resp_page_hour_current_period([resp_id] int NOT NULL, [day_id] int NOT NULL,
[hour_start] smallint NOT NULL, [page_id] int NOT NULL, [cnt] int NOT NULL, PRIMARY KEY([resp_id],
[day_id], [hour_start], [page_id], [cnt]))
       INSERT INTO resp_page_hour_current_period([resp_id], [day_id], [hour_start], [page_id], [cnt])
       SELECT [resp_id], [day_id], hour_start = ([hour] - [hour]%@hour_step), [page_id], cnt =
SUM([cnt])
       FROM #resp_hour_page
       GROUP BY [resp_id], [day_id], ([hour] - [hour]%@hour_step), [page_id]
       ORDER BY [day_id], hour_start, [resp_id], [page_id]
END

-- отбираем список ИР в каждом периоде @hour_step
IF OBJECT_ID('tempdb..#page_currentPeriod') IS NOT NULL DROP TABLE #page_currentPeriod
CREATE TABLE #page_currentPeriod ([page_id] INT not null , [day_id] INT, [hour_start] smallint, PRIMARY
KEY([page_id], [day_id], [hour_start]))
INSERT #page_currentPeriod ([page_id], [day_id], [hour_start])
SELECT rphcp.[page_id], rphcp.[day_id], rphcp.[hour_start]
FROM resp_page_hour_current_period rphcp
GROUP BY rphcp.[page_id], rphcp.[day_id], rphcp.[hour_start]

PRINT '--------------------------------------Отчёт по количеству ИР по периодам----------------------
------------'
DECLARE curs4 CURSOR LOCAL FAST_FORWARD FOR
       SELECT         DATEADD(day,[day_id],'19900101'), [hour_start], COUNT(distinct [page_id])  FROM
#page_currentPeriod
       GROUP BY       DATEADD(day,[day_id],'19900101'), [hour_start]
       ORDER BY       DATEADD(day,[day_id],'19900101'), [hour_start]
OPEN curs4
FETCH curs4 INTO @curs4_day_dt, @curs4_hour_start, @curs4_cnt_page_id
WHILE @@FETCH_STATUS = 0
BEGIN
       PRINT 'В ' + CONVERT(varchar(10), @curs4_day_dt, 120) + ' в периоде с     ' +
CONVERT(varchar(2), @curs4_hour_start) + '  по      ' + CONVERT(varchar(2), @curs4_hour_start
+(@hour_step-1)) + '  количество ИР оказалось:' + CONVERT(varchar(10), @curs4_cnt_page_id) +';'
       FETCH curs4 INTO @curs4_day_dt, @curs4_hour_start, @curs4_cnt_page_id
END
CLOSE curs4;
DEALLOCATE curs4;
PRINT'';

PRINT '-------------------------------------Отчёт по количеству ИП и ИР по периодам--------------------
------------'
IF OBJECT_ID('tempdb..#obj_total_cnt') IS NOT NULL DROP TABLE #obj_total_cnt;
CREATE TABLE #obj_total_cnt ([day_dt] date not null, [hour_start] smallint not null, [cnt] int not
null, PRIMARY KEY ([day_dt], [hour_start], [cnt]));
WITH total_obj as
(
SELECT         day_dt = DATEADD(day,[day_id],'19900101'), [hour_start], cnt = COUNT(distinct [resp_id])
FROM #resp_currentPeriod
       GROUP BY       DATEADD(day,[day_id],'19900101'), [hour_start]
UNION ALL
SELECT         day_dt = DATEADD(day,[day_id],'19900101'), [hour_start], cnt = COUNT(distinct [page_id])
FROM #page_currentPeriod
       GROUP BY       DATEADD(day,[day_id],'19900101'), [hour_start]
)
INSERT #obj_total_cnt ([day_dt], [hour_start], [cnt])
SELECT [day_dt], [hour_start], SUM([cnt]) FROM total_obj
```

```
GROUP BY [day_dt], [hour_start]
ORDER BY [day_dt], [hour_start]

DECLARE curs5 CURSOR LOCAL FAST_FORWARD FOR
       SELECT [day_dt], [hour_start], [cnt] FROM #obj_total_cnt ORDER BY [day_dt], [hour_start]
OPEN curs5
FETCH curs5 INTO @curs5_day_dt, @curs5_hour_start, @curs5_cnt_obj
WHILE @@FETCH_STATUS = 0
BEGIN
       PRINT 'В ' + CONVERT(varchar(10), @curs5_day_dt, 120) + ' в периоде с     ' +
CONVERT(varchar(2), @curs5_hour_start) + '  по      ' + CONVERT(varchar(2), @curs5_hour_start
+(@hour_step-1)) + '  количество объектов (ИП + ИР) оказалось:' + CONVERT(varchar(10), @curs5_cnt_obj)
+';'
       FETCH curs5 INTO @curs5_day_dt, @curs5_hour_start, @curs5_cnt_obj
END
CLOSE curs5;
DEALLOCATE curs5;
PRINT'';

-- Формируем обобщённую таблицу объектов (ИП + ИР)
IF @recount = 1 AND OBJECT_ID('obj_total') IS NOT NULL DROP TABLE obj_total
IF OBJECT_ID('obj_total') IS NULL
BEGIN
       CREATE TABLE obj_total ([obj_id] int NOT NULL IDENTITY(1,1), [resp_id] int NULL, [page_id] int
NULL, [day_id] int NOT NULL, [hour_start] smallint NOT NULL, PRIMARY KEY([obj_id], [day_id],
[hour_start]))
       INSERT obj_total ([resp_id], [page_id], [day_id], [hour_start])
       SELECT resp_id = NULL, [page_id], [day_id], [hour_start] FROM #page_currentPeriod
       UNION ALL
       SELECT [resp_id], [page_id] = NULL, [day_id], [hour_start] FROM #resp_currentPeriod
       ORDER BY [day_id], [hour_start], [resp_id], [page_id]
END
-- Расчёт коэффициентов усиления
IF OBJECT_ID('tempdb..#header_coef') IS NOT NULL DROP TABLE #header_coef
  CREATE TABLE #header_coef ([page_id] int NOT NULL PRIMARY KEY, [K1] float NOT NULL, [K2] float NOT
NULL, [word_type] smallint)
  INSERT #header_coef ([page_id], [K1], [K2], [word_type])
  SELECT HEADER.[page_id], [K1] = 10,
  [K2] = case
                WHEN

       ((670*60)*CONVERT(float,ISNULL(BODY.[cnt_word_body],HEADER.[cnt_header])))/((670*30)*CONVERT(fl
oat,HEADER.[cnt_header])) < 1 THEN 1
                WHEN

       ((670*60)*CONVERT(float,ISNULL(BODY.[cnt_word_body],HEADER.[cnt_header])))/((670*30)*CONVERT(fl
oat,HEADER.cnt_header)) > 10 THEN 10
                ELSE
((670*60)*CONVERT(float,ISNULL(BODY.[cnt_word_body],HEADER.[cnt_header])))/((670*30)*CONVERT(float,HEAD
ER.[cnt_header]))
              END,
  1
  FROM
  (
         SELECT P.[page_id], cnt_header = count(n1.[word_node_1])
         FROM dbo.az_pages P inner join
         dbo.decode_node1 n1
         ON P.[decoded_url] = n1.[decoded_url]
         GROUP BY P.[page_id]
  ) as HEADER
  LEFT JOIN
  (
         SELECT BODY.[page_id], cnt_word_body = SUM(BODY.[cnt_word_body]) FROM
         (
                select P.[page_id], cnt_word_body = count(n2.[word_node_2])
                from dbo.az_pages P inner join
                dbo.decode_node2 n2
                ON P.[decoded_url] = n2.[decoded_url]
                GROUP BY P.[page_id]
                UNION ALL
                select P.[page_id], cnt_word_body = count(n3.[word_node_3])
                from dbo.az_pages P inner join
                dbo.[decode_node3] n3
                ON P.[decoded_url] = n3.[decoded_url]
                GROUP BY P.[page_id]
         ) as BODY
         GROUP BY BODY.[page_id]
  ) AS BODY
  ON HEADER.[page_id] = BODY.[page_id]
```

```
  order by HEADER.[page_id]

-- Формирование обобщённого словаря по интервалам.
PRINT '-----------------------------------Отчёт по размеру глобального словаря терминов по периодам--
------------------------------'
--declare @word_len smallint
--SET @word_len = 4
IF OBJECT_ID('tempdb..#dictionaryPeriod') IS NOT NULL DROP TABLE #dictionaryPeriod
CREATE TABLE #dictionaryPeriod ([word_id] int NOT NULL, [day_id] int not null, [hour_start] smallint,
PRIMARY KEY([word_id], [day_id], [hour_start]))
INSERT #dictionaryPeriod
SELECT apwp.[word_id], ot.[day_id], ot.[hour_start] FROM
obj_total ot
inner join
dbo.az_pages_words_power apwp
ON ot.[page_id] = apwp.[page_id]
INNER JOIN
dbo.az_pages ap
ON apwp.[page_id] = ap.[page_id]
GROUP BY apwp.[word_id], ot.[day_id], ot.[hour_start]
ORDER BY  ot.[day_id], ot.[hour_start], apwp.[word_id]

DELETE dP FROM
#dictionaryPeriod dP
inner join
dbo.az_words W
ON dP.[word_id] = W.[word_id]
WHERE LEN(W.[word]) < @word_len;

-- создаем словаря корней
IF OBJECT_ID('tempdb..#smalldictionaryPeriod') IS NOT NULL DROP TABLE #smalldictionaryPeriod
CREATE TABLE #smalldictionaryPeriod ([smallword_id] int NOT NULL, [day_id] int not null, [hour_start]
smallint NOT NULL, PRIMARY KEY ([smallword_id], [day_id], [hour_start]))
INSERT #smalldictionaryPeriod ([smallword_id], [day_id], [hour_start])
SELECT asn.[smallword_id], dP.[day_id], dP.[hour_start]
FROM dbo.az_smallwords_new asn
INNER JOIN
dbo.link_word_small lws
ON lws.[smallword_id] = asn.[smallword_id]
INNER JOIN
#dictionaryPeriod dP
ON dP.[word_id] = lws.[word_id]
GROUP BY asn.[smallword_id], dP.[day_id], dP.[hour_start]

DECLARE curs6 CURSOR LOCAL FAST_FORWARD FOR
       SELECT day_dt = DATEADD(day,[day_id],'19900101'), [hour_start], cnt = COUNT([smallword_id])
       FROM #smalldictionaryPeriod
       GROUP BY DATEADD(day,[day_id],'19900101'), [hour_start]
       ORDER BY day_dt, [hour_start]
OPEN curs6
FETCH curs6 INTO @curs6_day_dt, @curs6_hour_start, @curs6_cnt_dict_word
WHILE @@FETCH_STATUS = 0
BEGIN
       PRINT 'В ' + CONVERT(varchar(10), @curs6_day_dt, 120) + ' в периоде с     ' +
CONVERT(varchar(2), @curs6_hour_start) + '  по      ' + CONVERT(varchar(2), @curs6_hour_start
+(@hour_step-1)) + '  размер глобального словаря терминов:' + CONVERT(varchar(10),
@curs6_cnt_dict_word) +';'
       FETCH curs6 INTO @curs6_day_dt, @curs6_hour_start, @curs6_cnt_dict_word
END
CLOSE curs6;
DEALLOCATE curs6;
PRINT'';

CREATE NONCLUSTERED INDEX idx_temp_#dictionaryPeriod_day_id_hour_start
ON #dictionaryPeriod ([day_id],[hour_start])
INCLUDE ([word_id])

CREATE NONCLUSTERED INDEX idx_temp_#smalldictionaryPeriod_day_id_hour_start
ON #smalldictionaryPeriod ([day_id],[hour_start])
INCLUDE ([smallword_id])


-- Формируем статистику ИР (13 мин).
IF OBJECT_ID('tempdb..#page_word_power_period') IS NOT NULL DROP TABLE #page_word_power_period
CREATE TABLE #page_word_power_period ([id] int NOT NULL IDENTITY(1,1) PRIMARY KEY, [page_id] int NOT
NULL, [word_id] int NOT NULL, [power] float NOT NULL, [word_type] smallint NULL, [K1] float NOT NULL,
[K2] float NOT NULL, [day_id] int not NULL, [hour_start] smallint NOT NULL)
declare curs7 cursor LOCAL FAST_FORWARD FOR
SELECT DISTINCT [page_id], [day_id], [hour_start] FROM obj_total
```

```
WHERE [page_id] is NOT NULL
ORDER BY [day_id], [hour_start]
OPEN curs7
FETCH curs7 INTO @curs7_page_id, @curs7_day_id, @curs7_hour_start
WHILE @@FETCH_STATUS = 0
BEGIN
	INSERT #page_word_power_period ([page_id], [word_id], [power], [word_type], [K1], [K2],
[day_id], [hour_start])
	SELECT page_id = @curs7_page_id, dP.word_id,
	[power] = CASE WHEN apwp.[power] IS NULL THEN 0 ELSE apwp.[power] END,
	apwp.[word_type],
	K1 = CASE WHEN apwp.[word_type] IS NULL THEN 0 WHEN apwp.[word_type] = 0 THEN @nodesearch_coef
WHEN apwp.[word_type] = 1 THEN hc.[K1] WHEN apwp.[word_type] = 2 THEN @node2_coef WHEN apwp.[word_type]
= 3 THEN 1 END,
	K2 = CASE WHEN apwp.[word_type] IS NULL THEN 0 WHEN apwp.[word_type] = 0 THEN 1 WHEN
apwp.[word_type] = 1 THEN hc.[K2] WHEN apwp.[word_type] = 2 THEN 1 WHEN apwp.[word_type] = 3 THEN 1 END
	, dp.[day_id], dP.[hour_start]
	  FROM
(SELECT * FROM dbo.az_pages_words_power WHERE [page_id] = @curs7_page_id) apwp
LEFT JOIN
#header_coef hc
ON apwp.[page_id] = hc.[page_id] AND apwp.[word_type] = hc.[word_type]
Right join
(SELECT * FROM #dictionaryPeriod WHERE [hour_start] = @curs7_hour_start AND [day_id] = @curs7_day_id)
dP
ON apwp.[word_id] = dP.[word_id]
OPTION (RECOMPILE)
	FETCH curs7 INTO @curs7_page_id, @curs7_day_id, @curs7_hour_start
END
CLOSE curs7
DEALLOCATE curs7

-- Индекс для ускорения (9 мин)
CREATE NONCLUSTERED INDEX idx_temp_#page_word_power_period
ON [dbo].[#page_word_power_period] ([day_id],[hour_start])
INCLUDE ([page_id],[word_id],[power],[K1],[K2])

-- Непосредственно вектор ИР (14 мин).
IF @recount = 1 AND OBJECT_ID('vector_page_period') IS NOT NULL DROP TABLE vector_page_period
IF OBJECT_ID('vector_page_period') IS NULL
BEGIN
	CREATE TABLE vector_page_period([page_id] int NOT NULL, [word_id] int NOT NULL, [day_id] int
NOT NULL, [hour_start] smallint NOT NULL, [power] float NOT NULL, PRIMARY KEY([page_id], [word_id],
[day_id], [hour_start]))
	DECLARE curs8 CURSOR LOCAL FAST_FORWARD FOR
	SELECT [day_id], [hour_start] FROM #page_word_power_period
	GROUP BY [day_id], [hour_start]
	ORDER BY [day_id], [hour_start]
	OPEN curs8
	FETCH curs8 INTO @curs8_day_id, @curs8_hour_start
	WHILE @@FETCH_STATUS = 0
	BEGIN
		INSERT vector_page_period([page_id], [word_id], [day_id], [hour_start], [power])
		select pwpp.[page_id], lws.[smallword_id], pwpp.[day_id], pwpp.[hour_start],
SUM(pwpp.[power]*pwpp.[K1]*pwpp.[K2])
		FROM #page_word_power_period pwpp
		inner join
		dbo.link_word_small lws
		ON pwpp.[word_id] = lws.[word_id]
		WHERE [day_id] = @curs8_day_id AND [hour_start] = @curs8_hour_start
		GROUP BY pwpp.[page_id], lws.[smallword_id], pwpp.[day_id], pwpp.[hour_start]
		FETCH curs8 INTO @curs8_day_id, @curs8_hour_start
	END
	CLOSE curs8
	DEALLOCATE curs8

	-- Индекс для ускорения выполнения запросов
	CREATE NONCLUSTERED INDEX idx_vector_page_period_1
	ON [dbo].[vector_page_period] ([day_id],[hour_start])
	INCLUDE ([page_id],[word_id],[power])
END

IF OBJECT_ID('tempdb..#page_word_power_period') IS NOT NULL DROP TABLE #page_word_power_period

-- Непосредственно вектор ИП (10 мин).
IF @recount = 1 AND OBJECT_ID('vector_user_period') IS NOT NULL DROP TABLE vector_user_period
IF OBJECT_ID('vector_user_period') IS NULL
BEGIN
```

```
	CREATE TABLE vector_user_period([resp_id] int NOT NULL, [word_id] int NOT NULL, [day_id] int
NOT NULL, [hour_start] smallint NOT NULL, [power] float NOT NULL, PRIMARY KEY([resp_id], [word_id],
[day_id], [hour_start]))
	INSERT vector_user_period([resp_id], [word_id], power, [day_id], [hour_start])
	SELECT ot.[resp_id], vpp.[word_id], MAX(vpp.[power]), vpp.[day_id], vpp.[hour_start]
	FROM resp_page_hour_current_period ot inner join
	vector_page_period vpp
	on ot.[page_id] = vpp.[page_id]
	AND ot.[day_id] = vpp.[day_id]
	AND ot.[hour_start] = vpp.[hour_start]
	GROUP BY [resp_id], [word_id], vpp.[day_id], vpp.[hour_start]

	-- Индекс для ускорения выполнения запросов
	CREATE NONCLUSTERED INDEX idx_vector_user_period_1
	ON [dbo].[vector_user_period] ([day_id],[hour_start])
	INCLUDE ([resp_id],[word_id],[power])
END

IF OBJECT_ID('tempdb..#resp_day') is not null DROP TABLE #resp_day
IF OBJECT_ID('tempdb..#firstResps') IS NOT NULL DROP TABLE #firstResps
IF OBJECT_ID('tempdb..#resp_current') is not null DROP TABLE #resp_current
IF OBJECT_ID('tempdb..#resp_hour_page') IS NOT NULL DROP TABLE #resp_hour_page
IF OBJECT_ID('tempdb..#page_currentPeriod') IS NOT NULL DROP TABLE #page_currentPeriod
IF OBJECT_ID('tempdb..#obj_total_cnt') IS NOT NULL DROP TABLE #obj_total_cnt;
IF OBJECT_ID('tempdb..#dictionaryPeriod') IS NOT NULL DROP TABLE #dictionaryPeriod
IF OBJECT_ID('tempdb..#smalldictionaryPeriod') IS NOT NULL DROP TABLE #smalldictionaryPeriod
IF OBJECT_ID('tempdb..#page_word_power_period') IS NOT NULL DROP TABLE #page_word_power_period
IF OBJECT_ID('tempdb..#header_coef') IS NOT NULL DROP TABLE #header_coef
IF OBJECT_ID('tempdb..#resp_currentPeriod') IS NOT NULL DROP TABLE #resp_currentPeriod



USE [InternetDB]
GO
/****** Object:  StoredProcedure [dbo].[az_clustering_step2]    Script Date: 10/06/2014 23:27:08
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[az_clustering_step2] (@day_id int, @hour_start smallint, @clust_num smallint,
@dist_type tinyint)
AS
-- EXEC az_clustering_step2 @day_id = 4494, @hour_start = 0, @clust_num =3
SET NOCOUNT ON
SET XACT_ABORT ON
-- Список переменных
--@dist_type -выбор меры близости:1 - евклидово расстояние; 2 - квадрат евклидового расстояния; 3 -
менхэтенское расстояния.
declare @user_id int;
declare @min_euklide float;
declare @max_euklide float;
declare @cluster_num int;
declare @geometry varchar(max);
declare @author varchar(100) = 'Author: Zein A.N. - PhD Student, MPEI, VMSS (ZeynAN@mpei.ru)';
PRINT '*******Clustering
Procedure********************************************************************************';
PRINT '**************************' + @author + '******************';
PRINT
'**************************************************************************************************
***';

IF @clust_num <=1
BEGIN
	RAISERROR('Значение clust_num должно быть больше 1!!!',11,1);
	RETURN -1;
END

Set @clust_num = @clust_num - 1;

-- переменные курсора
declare @curs3_object_id1 int;
declare @curs3_object_id2 int;
declare @curs3_euklideDist int;

declare @curs2_cluster int;
declare @curs2_cntOBJ int;
declare @curs2_OBJS varchar(max);
```

```
-- Указываем первый кластер.
--set @cluster_num = 1;
IF OBJECT_ID('tempdb..#GlobalObjectClusterTable') IS NOT NULL DROP TABLE #GlobalObjectClusterTable
Create table #GlobalObjectClusterTable ([object_id] int not null, [word] varchar(50), [power] float,
[cluster] int)

-- Заполняем обобщённую таблицу
INSERT #GlobalObjectClusterTable ([object_id] , [word], [power], [cluster])
select [page_id] , CAST([word_id] as varchar(20)), [power], [cluster] = 0
from vector_page_period vpp WITH (NOLOCK)
WHERE day_id = @day_id AND hour_start = @hour_start

INSERT #GlobalObjectClusterTable ([object_id] , [word], [power], [cluster])
select -[resp_id] , CAST([word_id] as varchar(20)), [power], [cluster] = 0
from vector_user_period vpp WITH (NOLOCK)
WHERE day_id = @day_id AND hour_start = @hour_start

PRINT '====================================Начало
отчёта=========================================================';
PRINT '';
-- расчёт меры близости и группировка сталбцов во избежании повторений
PRINT
'/*************************************************************************************************
**\'
PRINT '|************************************Отчёт по Евклидово
расстояние*************************************|'
PRINT
'\*************************************************************************************************
**/'
IF OBJECT_ID('tempdb..#GlobalObjectEuclide') IS NOT NULL DROP TABLE #GlobalObjectEuclide
CREATE TABLE #GlobalObjectEuclide ([object_id1] int NOT NULL, [object_id2] int NOT NULL, [euklideDist]
float, PRIMARY KEY([object_id1],[object_id2]) )
INSERT #GlobalObjectEuclide ([object_id1], [object_id2], [euklideDist])
SELECT object_id1 = CASE WHEN GOCT.[object_id] < GOCT2.[object_id] THEN GOCT.[object_id] ELSE
GOCT2.[object_id] END,
       object_id2 = CASE WHEN GOCT2.[object_id] > GOCT.[object_id] THEN GOCT2.[object_id] ELSE
GOCT.[object_id] END,
EuklidDist = CASE
                              WHEN @dist_type = 1 then POWER(SUM(POWER((GOCT.[power] - GOCT2.[power]),
2)),0.5)
                              WHEN @dist_type = 2 then POWER(SUM(POWER((GOCT.[power] - GOCT2.[power]),
2)),1)
                              WHEN @dist_type = 3 then SUM(ABS(GOCT.[power] - GOCT2.[power]))
                       END
FROM
#GlobalObjectClusterTable GOCT
INNER JOIN
#GlobalObjectClusterTable GOCT2
ON GOCT.[word] = GOCT2.[word]
AND GOCT.[object_id] <> GOCT2.[object_id]
GROUP BY
       CASE WHEN GOCT.[object_id] < GOCT2.[object_id] THEN GOCT.[object_id] ELSE GOCT2.[object_id]
END,
       CASE WHEN GOCT2.[object_id] > GOCT.[object_id] THEN GOCT2.[object_id] ELSE GOCT.[object_id] END
ORDER BY
       CASE WHEN GOCT.[object_id] < GOCT2.[object_id] THEN GOCT.[object_id] ELSE GOCT2.[object_id]
END,
       CASE WHEN GOCT2.[object_id] > GOCT.[object_id] THEN GOCT2.[object_id] ELSE GOCT.[object_id] END

-- курсор для вывода евклидово расстояния
DECLARE curs3 cursor LOCAL FAST_FORWARD for
              SELECT [object_id1], [object_id2], [euklideDist] FROM #GlobalObjectEuclide
       OPEN curs3
       FETCH curs3 INTO @curs3_object_id1, @curs3_object_id2, @curs3_euklideDist
       WHILE @@FETCH_STATUS = 0
       BEGIN
              PRINT 'Мера близости между объектами с номерами: ' +
CONVERT(varchar(100),@curs3_object_id1) + ' и ' + CONVERT(varchar(100), @curs3_object_id2) + ' равно: '
+ CONVERT(varchar(100),@curs3_euklideDist) + '.';
              FETCH curs3 INTO @curs3_object_id1, @curs3_object_id2, @curs3_euklideDist
       END
       CLOSE curs3;
       DEALLOCATE curs3;

-- находим максимальное евклидово расстояние между ИП (object_id < 0)
PRINT 'находим максимальное значение меры близости между ИП (object_id < 0)'
IF OBJECT_ID('tempdb..#max_euc') IS NOT NULL DROP TABLE #max_euc
CREATE TABLE #max_euc (euklide_distance float not null)
INSERT #max_euc (euklide_distance)
```

```
SELECT DISTINCT TOP(@clust_num) GOE.[euklideDist] FROM #GlobalObjectEuclide GOE
INNER JOIN
(SELECT [object_id] FROM #GlobalObjectClusterTable WHERE [cluster] = 0) as A1
ON GOE.object_id1 = A1.[object_id] AND A1.[object_id] < 0
INNER JOIN
(SELECT [object_id] FROM #GlobalObjectClusterTable WHERE [cluster] = 0) as A2
ON GOE.object_id2 = A2.[object_id] AND A2.[object_id] < 0
ORDER BY GOE.[euklideDist] DESC
PRINT 'Максимальное значение меры близости между ИП (object_id < 0) найдено'

-- находим ИП на самом отдалённом растоянии между собой.
PRINT 'Находим ИП на самом отдалённом растоянии между собой'
IF OBJECT_ID('tempdb..#firstClasters') IS NOT NULL DROP TABLE #firstClasters
CREATE TABLE #firstClasters ([object_id] int NOT NULL, [clust_id] smallint identity(1,1), PRIMARY
KEY([object_id]))
INSERT #firstClasters ([object_id])
select [object_id1] from #GlobalObjectEuclide GOE where GOE.[euklideDist] in
(select [euklide_distance] FROM #max_euc) AND ([object_id1] < 0 OR [object_id2] < 0)
UNION
select [object_id2] from #GlobalObjectEuclide GOE where GOE.[euklideDist] in
(select [euklide_distance] FROM #max_euc) AND ([object_id1] < 0 OR [object_id2] < 0)
PRINT 'ИП на самом отдалённом растоянии между собой найдены'

-- подготовка для расчёта среднего вектора.
IF OBJECT_ID('tempdb..#clust_page') IS NOT NULL DROP TABLE #clust_page
CREATE TABLE #clust_page ([clust_id] smallint NOT NULL, [page_id] int NOT NULL)
INSERT #clust_page ([clust_id], [page_id])
SELECT fC.[clust_id], us.[page_id]
FROM #firstClasters fC
inner join
[dbo].[resp_page_hour_current_period] us
ON fC.[object_id] = -us.[resp_id] AND us.[day_id] = @day_id AND us.[hour_start] = @hour_start
UNION
select
fC.[clust_id], fC.[object_id] FROM #firstClasters fC

-- средние вектора
IF OBJECT_ID('tempdb..#avgVectorTable') IS NOT NULL DROP TABLE #avgVectorTable
CREATE TABLE #avgVectorTable([clust_id] int NOT NULL, [word_id] int NOT NULL, [power] float NOT NULL,
PRIMARY KEY([clust_id], [word_id], [power]))
INSERT INTO #avgVectorTable([clust_id], [word_id], [power])
SELECT cp.[clust_id], vpp.[word_id], power = AVG([power])
FROM #clust_page cp inner join vector_page_period vpp
ON cp.[page_id] = vpp.[page_id]
WHERE vpp.[day_id] = @day_id AND vpp.[hour_start] = @hour_start
GROUP BY cp.[clust_id], vpp.[word_id]

IF OBJECT_ID('tempdb..#euklidDistTable') IS NOT NULL DROP TABLE #euklidDistTable
CREATE TABLE #euklidDistTable ([object_id] int, [avgCluster] int,  [EuklidDist] float, PRIMARY KEY
([object_id], [avgCluster]))

PRINT 'Расчёт меры близости относительно средних векторов';
INSERT #euklidDistTable ([object_id], [avgCluster],  [EuklidDist])
SELECT GOCT.[object_id], avgCluster = aVT.[clust_id],
       EuklidDist = CASE
                                          WHEN @dist_type = 1 then POWER(SUM(POWER((GOCT.[power] -
aVT.[power]), 2)),0.5)
                                          WHEN @dist_type = 2 then POWER(SUM(POWER((GOCT.[power] -
aVT.[power]), 2)),1)
                                          WHEN @dist_type = 3 then SUM(ABS(GOCT.[power] -
aVT.[power]))
                                    END
FROM #GlobalObjectClusterTable GOCT
INNER JOIN
(
       select [clust_id], [word_id], [power] from #avgVectorTable
) AS aVT
ON cast(aVT.[word_id] as varchar(50)) = GOCT.[word]
GROUP BY GOCT.[object_id], aVT.[clust_id]
OPTION (RECOMPILE);
PRINT 'Расчёт меры близости относительно средних векторов выполнено';

-- для каждого объекта находим кластер с минимальным растоянием к центру.
IF OBJECT_ID('tempdb..#euklidDistTableMin') IS NOT NULL DROP TABLE #euklidDistTableMin
CREATE TABLE #euklidDistTableMin ([object_id] int not null, [avgCluster] int, [EuklidDistMin] float)
INSERT #euklidDistTableMin ([object_id], [avgCluster], [EuklidDistMin])
select top(1) with ties [object_id], [avgCluster], [EuklidDist]
FROM #euklidDistTable eDT
order by row_number() over (partition by [object_id] order by EuklidDist ASC)
```

```
UPDATE GOCT SET GOCT.[cluster] = C.[avgCluster]
FROM
#GlobalObjectClusterTable GOCT
INNER JOIN
#euklidDistTableMin C
ON GOCT.[object_id] = C.[object_id]

IF EXISTS (SELECT NULL FROM #GlobalObjectClusterTable WHERE [cluster] = 0)
BEGIN
	RAISERROR('Имеются не кластеризованные объекты',11,1);
END

PRINT
'/*************************************************************************************************
**\'
PRINT '|*********************Результат предварительной
кластеризации*****************************************|'
PRINT
'\*************************************************************************************************
**/'
IF OBJECT_ID('tempdb..#ClusterContent') IS NOT NULL DROP TABLE #ClusterContent
	CREATE TABLE #ClusterContent ([cluster] int NOT NULL PRIMARY KEY, [cluster_cntOBJ] int, [OBJS]
varchar(max))
	INSERT #ClusterContent ([cluster], [cluster_cntOBJ])
	select [cluster], COUNT(DISTINCT [object_id])
		FROM #GlobalObjectClusterTable
		GROUP BY [cluster]
		ORDER BY [cluster]

	UPDATE CC SET
    CC.OBJS =  REPLACE(
				REPLACE(
					CAST(
						(
							select distinct convert(varchar(1000),
[object_id]) AS 'Itm'
							FROM #GlobalObjectClusterTable GOCT
							WHERE GOCT.cluster = CC.cluster
							FOR XML PATH('')
						)AS varchar(MAX)
					), --//получаем строку вида
<Itm>ids1</Itm><Itm>Ids2</Itm><Itm>Ids3</Itm>...
			'<Itm>',  ''),
		  '</Itm>',  ';')
	from
		#ClusterContent CC

--// удаляем последнюю точку-запятую
	UPDATE CC SET CC.OBJS = CASE
							WHEN SUBSTRING(CC.OBJS, LEN(CC.OBJS), 1) =
';' THEN LEFT(CC.OBJS, LEN(CC.OBJS) -1)
							ELSE CC.OBJS
						END
	FROM #ClusterContent CC

	DECLARE curs2 cursor LOCAL FAST_FORWARD for
		SELECT [cluster], [cluster_cntOBJ], [OBJS] FROM #ClusterContent
	OPEN curs2
	FETCH curs2 INTO @curs2_cluster, @curs2_cntOBJ, @curs2_OBJS
	WHILE @@FETCH_STATUS = 0
	BEGIN
		PRINT 'В кластере с номером: ' + CONVERT(varchar(100),@curs2_cluster) + ' содержится: '
+ CONVERT(varchar(100), @curs2_cntOBJ) + ' объектов:';
		PRINT '{';
		PRINT ' ' + @curs2_OBJS;
		PRINT '}';
		FETCH curs2 INTO @curs2_cluster, @curs2_cntOBJ, @curs2_OBJS;
	END
	CLOSE curs2;
	DEALLOCATE curs2;

PRINT '=============================================Конец
отчёта================================================';
SELECT [object_id], [word], [power], [cluster] FROM #GlobalObjectClusterTable
```

```
USE [InternetDB]
GO
/****** Object:  StoredProcedure [dbo].[az_clustering_step3]    Script Date: 10/06/2014 23:42:57
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[az_clustering_step3] (@dist_type tinyint)
-- время выполнения 1 мин 30 сек.
--EXEC az_clustering_step3
AS
--@dist_type -выбор меры близости:1 - евклидово расстояние; 2 - квадрат евклидового расстояния; 3 -
менхэтенское расстояния.

SET NOCOUNT ON
SET XACT_ABORT ON
-- переменные
declare @iteration int;
declare @cluster_cnt int;
declare @obj_cnt int;
declare @author varchar(100) = 'Author: Zein A.N. - PhD Student, MPEI, VMSS (ZeynAN@mpei.ru)';
PRINT '*******Clustering
Procedure********************************************************************************';
PRINT '**************************' + @author + '******************';
PRINT
'*********************************************************************************************************
***';

set @iteration = 0;

-- переменные курсора: @curs_object_id, @curs_clusterOld, @curs_clusterNew, @curs_iteration
-- @curs1_cluster, @curs1_word, @curs1_power
declare @curs1_cluster int;
declare @curs1_word varchar(100);
declare @curs1_power float;

declare @curs2_object_id int;
declare @curs2_clusterOld int;
declare @curs2_clusterNew int;
declare @curs2_iteration int;

declare @curs3_cluster int;
declare @curs3_cntOBJ int;
declare @curs3_OBJS varchar(max);

-- создаём обобщённую таблицу для объектов ИП и ИР
IF OBJECT_ID('tempdb..#GlobalObjectClusterTable') IS NOT NULL DROP TABLE #GlobalObjectClusterTable
Create table #GlobalObjectClusterTable ([object_id] int not null, [word] varchar(100), [power] float,
[cluster] int, [iteration] int)

-- Заполняем обобщённую таблицу
INSERT #GlobalObjectClusterTable ([object_id] , [word], [power], [cluster], [iteration])
select [object_id] , [word], [power], [cluster], [iteration] = 0 from #GlobalObjectClusterTableMAIN

-- Метка начала цикла
begining:
-- Находим номер последней итерации
SELECT @iteration = MAX(GOCT.[iteration]) FROM #GlobalObjectClusterTable GOCT
PRINT '====================================Начало отчёта - итерация: '+CONVERT(varchar(10),
@iteration)+'========================================='
IF OBJECT_ID('tempdb..#ClusterContent') IS NOT NULL DROP TABLE #ClusterContent
CREATE TABLE #ClusterContent ([cluster] int NOT NULL PRIMARY KEY, [cluster_cntOBJ] int, [OBJS]
varchar(max))
INSERT #ClusterContent ([cluster], [cluster_cntOBJ])
select [cluster], COUNT(DISTINCT [object_id])
       FROM #GlobalObjectClusterTable
       WHERE [iteration] = @iteration
       GROUP BY [cluster]
       ORDER BY [cluster]

UPDATE CC SET
CC.OBJS =  REPLACE(
                     REPLACE(
                            CAST(
                                    (
```

```
                                                    select distinct convert(varchar(1000), [object_id])
AS 'Itm'
                                                    FROM #GlobalObjectClusterTable GOCT
                                                    WHERE GOCT.[cluster] = CC.[cluster]
                                                    AND GOCT.[iteration] = @iteration
                                                    FOR XML PATH('')
                                          )AS varchar(MAX)
                                        ), --//получаем строку вида
<Itm>ids1</Itm><Itm>Ids2</Itm><Itm>Ids3</Itm>...
       '<Itm>',  ''),
     '</Itm>',  ';')
from
       #ClusterContent CC

--// удаляем последнюю точку-запятую
UPDATE CC SET CC.[OBJS] = CASE
                                                    WHEN SUBSTRING(CC.[OBJS], LEN(CC.[OBJS]), 1) = ';'
THEN LEFT(CC.OBJS, LEN(CC.OBJS) -1)
                                                    ELSE CC.[OBJS]
                                             END
FROM #ClusterContent CC

DECLARE curs3 cursor LOCAL FAST_FORWARD for
       SELECT [cluster], [cluster_cntOBJ], [OBJS] FROM #ClusterContent
OPEN curs3
FETCH curs3 INTO @curs3_cluster, @curs3_cntOBJ, @curs3_OBJS
WHILE @@FETCH_STATUS = 0
BEGIN
       PRINT 'В кластере с номером: ' + CONVERT(varchar(100),@curs3_cluster) + ' содержится: ' +
CONVERT(varchar(100), @curs3_cntOBJ) + ' объектов:';
       PRINT '{';
       PRINT ' ' + @curs3_OBJS;
       PRINT '}';
       FETCH curs3 INTO @curs3_cluster, @curs3_cntOBJ, @curs3_OBJS;
END
CLOSE curs3;
DEALLOCATE curs3;

-- расчёт среднего вектора
PRINT
'/**********************************************************************************************
**\'
PRINT '|********************Отчёт по средним векторам в кластерной структуре - итерация:
'+CONVERT(varchar(10), @iteration)+' ********************|'
PRINT
'\**********************************************************************************************
**/'
IF OBJECT_ID('tempdb..#avgVectorTable') IS NOT NULL DROP TABLE #avgVectorTable
select [cluster], [word], [power] = avg([power])
INTO #avgVectorTable
FROM #GlobalObjectClusterTable
WHERE [iteration] = @iteration
GROUP BY [cluster], [word]
ORDER BY [cluster], [word]

       -- выводим результат средних векторов
       DECLARE curs1 CURSOR LOCAL FAST_FORWARD FOR
              select [cluster], [word], [power]
              from #avgVectorTable
              GROUP BY [cluster], [word], [power]
              ORDER BY [cluster], [word]
       OPEN curs1
       FETCH curs1 INTO @curs1_cluster, @curs1_word, @curs1_power
       WHILE @@FETCH_STATUS = 0
       BEGIN
              PRINT 'Итерация: ' + CONVERT(varchar(10), @iteration) +'   Кластер с номером: ' +
CONVERT(varchar(10), @curs1_cluster) + '    для координаты: ' + @curs1_word + '  имеет среднее
значение: ' + CONVERT(varchar(10), @curs1_power) + ';'
              FETCH curs1 INTO @curs1_cluster, @curs1_word, @curs1_power
       END
       CLOSE curs1
       DEALLOCATE curs1

PRINT '';

-- Мера близости
IF OBJECT_ID('tempdb..#euklidDistTable') IS NOT NULL DROP TABLE #euklidDistTable
CREATE TABLE #euklidDistTable ([object_id] int, [avgCluster] int,  [EuklidDist] float)
```

```
INSERT #euklidDistTable ([object_id], [avgCluster],  [EuklidDist])
SELECT GOCT.[object_id], avgCluster = aVT.[cluster],
EuklidDist = CASE
                              WHEN @dist_type = 1 then POWER(SUM(POWER((GOCT.[power] - aVT.[power]),
2)),0.5)
                              WHEN @dist_type = 2 then POWER(SUM(POWER((GOCT.[power] - aVT.[power]),
2)),1)
                              WHEN @dist_type = 3 then SUM(ABS(GOCT.[power] - aVT.[power]))
                        END
FROM #GlobalObjectClusterTable GOCT
OUTER APPLY
(
       select [cluster], [word], [power] from #avgVectorTable
) AS aVT
WHERE aVT.word = GOCT.word
AND GOCT.[iteration] = @iteration
GROUP BY GOCT.[object_id], aVT.[cluster]
OPTION (RECOMPILE);

-- для каждого объекта находим кластер с минимальным растоянием к центру.
IF OBJECT_ID('tempdb..#euklidDistTableMin') IS NOT NULL DROP TABLE #euklidDistTableMin
CREATE TABLE #euklidDistTableMin ([object_id] int not null, [avgCluster] int, [EuklidDistMin] float)
INSERT #euklidDistTableMin ([object_id], [avgCluster], [EuklidDistMin])
select top(1) with ties [object_id], [avgCluster], [EuklidDist]
FROM #euklidDistTable eDT
order by row_number() over (partition by [object_id] order by EuklidDist ASC)

-- Проверка на перемещение объектов из одного кластера в другой (более близкий)
IF EXISTS
(
       SELECT top(1) NULL FROM
       #euklidDistTableMin eDTM
       inner join
       #GlobalObjectClusterTable GOCT
       ON eDTM.[object_id] = GOCT.[object_id]
       WHERE eDTM.[avgCluster] <> GOCT.[cluster]
       AND GOCT.[iteration] = @iteration
)
BEGIN
       -- Создаём временную таблицу для хранения новой кластерной структуры.
       IF OBJECT_ID('tempdb..#NewGOCT') IS NOT NULL DROP TABLE #NewGOCT
       CREATE TABLE #NewGOCT ([object_id] int not null, [word] varchar(100), [power] float,
[clusterOld] int, [clusterNew] int, [iteration] int)
       INSERT #NewGOCT ([object_id], [word], [power], [clusterOld], [clusterNew], [iteration])
       SELECT GOCT.[object_id], GOCT.[word], GOCT.[power], clusterOld = GOCT.[cluster], clusterNew =
eDTM.[avgCluster], iterationInc = GOCT.[iteration] + 1 FROM
       #euklidDistTableMin eDTM
       inner join
       #GlobalObjectClusterTable GOCT
       ON eDTM.[object_id] = GOCT.[object_id]
       WHERE GOCT.iteration = @iteration

       PRINT
'/*****************************************************************************************************
**\'
       PRINT '|************************Отчёт по изменениям в кластерной структуре - итерация:
'+CONVERT(varchar(10), @iteration)+' **********************|'
       PRINT
'\*****************************************************************************************************
**/'
       DECLARE curs2 CURSOR LOCAL FAST_FORWARD FOR
              select [object_id], [clusterOld], [clusterNew], [iteration]
              from #NewGOCT where clusterOld <> clusterNew
              GROUP BY [object_id], [clusterOld], [clusterNew], [iteration]
              ORDER BY [object_id]
       OPEN curs2
       FETCH curs2 INTO @curs2_object_id, @curs2_clusterOld, @curs2_clusterNew, @curs2_iteration
       WHILE @@FETCH_STATUS = 0
       BEGIN
              PRINT 'Итерация: ' + CONVERT(varchar(10), @iteration) +'   Объект с номером: ' +
CONVERT(varchar(10), @curs2_object_id) + '  был перенесён из кластера: ' + CONVERT(varchar(10),
@curs2_clusterOld) + 'в другой кластер под номером: ' + CONVERT(varchar(10), @curs2_clusterNew) + ';'
              FETCH curs2 INTO @curs2_object_id, @curs2_clusterOld, @curs2_clusterNew,
@curs2_iteration
       END
       CLOSE curs2
       DEALLOCATE curs2

       -- Заполняем глобальную таблицу кластерного анализа #GlobalObjectClusterTable
```

```
	INSERT #GlobalObjectClusterTable([object_id] , [word], [power], [cluster], [iteration])
	select GOCT.[object_id], GOCT.[word], GOCT.[power], NGOCT.[clusterNew], NGOCT.[iteration]
	from
	(SELECT [object_id], [clusterNew], [iteration] FROM #NewGOCT GROUP BY [object_id],
[clusterNew], [iteration]) AS NGOCT
	inner join
	#GlobalObjectClusterTable GOCT
	ON NGOCT.[object_id] = GOCT.[object_id]
	WHERE GOCT.iteration = 0

	PRINT '=====================================Конец отчёта- итерация: '+CONVERT(varchar(10),
@iteration)+'=========================================';
	-- Увеличиваем счётчик.
	SET @iteration = @iteration + 1;
	PRINT '-------------------------------------------------------------------------------------------------
-----------------';
	PRINT '';
	-- Переходим к следующей итерации.
	goto begining
END
ELSE
BEGIN
	DECLARE @max_euklide_distance FLOAT;
	SET @max_euklide_distance = 0;
	PRINT '====================Нет Изменений в кластерной структуре - Кластеризация
завершена======================';
	select @cluster_cnt = COUNT(DISTINCT [cluster]) from #GlobalObjectClusterTable WHERE
[iteration] = @iteration
	PRINT 'Конечное число кластеров: '    + CONVERT(varchar(10), @cluster_cnt)  +';';
	select @obj_cnt = COUNT(DISTINCT [object_id]) from #GlobalObjectClusterTable WHERE [iteration]
= @iteration
	PRINT 'Число задействованных объектов: '     + CONVERT(varchar(10), @obj_cnt)  +';';
	SELECT [object_id] , [word], [power], [cluster] FROM #GlobalObjectClusterTable
	WHERE iteration = @iteration;


END

SET NOCOUNT OFF
SET XACT_ABORT OFF




USE [InternetDB]
GO
/****** Object:  StoredProcedure [dbo].[az_clustering_step4]    Script Date: 10/06/2014 23:49:04
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[az_clustering_step4] (@day_id int, @hour_start smallint, @dist_type tinyint/*,
@max_euklide_distance float*/)
-- время выполнения 1 мин 30 сек.
--EXEC az_clustering_step4
--@dist_type -выбор меры близости:1 - евклидово расстояние; 2 - квадрат евклидового расстояния; 3 -
менхэтенское расстояния.

AS
SET NOCOUNT ON
SET XACT_ABORT ON
-- переменные
declare @iteration int;
declare @cluster_cnt int;
declare @obj_cnt int;
declare @cnt_resp_new_OK int;
declare @cnt_resp_new_ALL int;
declare @min_cluster_euklide float;
declare @author varchar(100) = 'Author: Zein A.N. - PhD Student, MPEI, VMSS (ZeynAN@mpei.ru)';
PRINT '*******Clustering
Procedure*********************************************************************************';
PRINT '**************************' + @author + '******************';
PRINT
'*************************************************************************************************
***';

-- переменные курсора: @curs_object_id, @curs_clusterOld, @curs_clusterNew, @curs_iteration
```

```
-- @curs1_cluster, @curs1_word, @curs1_power
declare @curs1_cluster int;
declare @curs1_word varchar(100);
declare @curs1_power float;

declare @curs2_object_id int;
declare @curs2_avgCluster int;
declare @curs2_EuklidDistMin float;

declare @curs3_object_id int;
declare @curs3_avgCluster int;
declare @curs3_EuklidDistMin float;

declare @curs3_cluster int;
declare @curs3_cntOBJ int;
declare @curs3_OBJS varchar(max);

-- Заполняем обобщённую таблицу
IF OBJECT_ID('tempdb..#GlobalObjectClusterTable') IS NOT NULL DROP TABLE #GlobalObjectClusterTable
Create table #GlobalObjectClusterTable ([object_id] int not null, [word] varchar(50), [power] float,
[cluster] int)

-- Заполняем обобщённую таблицу
INSERT #GlobalObjectClusterTable ([object_id] , [word], [power], [cluster])
select vpp.[page_id] , CAST(vpp.[word_id] as varchar(20)), vpp.[power], [cluster] = 0
from vector_page_period vpp WITH (NOLOCK)
WHERE [day_id] = case when @hour_start = 20 THEN @day_id + 1 ELSE @day_id end
AND [hour_start] = case when @hour_start = 20 THEN 0 else @hour_start + 4 end

PRINT '==================================================Начало
отчёта=================================================='
-- расчёт среднего вектора
PRINT
'/*************************************************************************************************
**\'
PRINT '|********************Отчёт по средним векторам в кластерной структуре - итерация:
'+CONVERT(varchar(10), @iteration)+' ********************|'
PRINT
'\*************************************************************************************************
**/'
IF OBJECT_ID('tempdb..#avgVectorTable') IS NOT NULL DROP TABLE #avgVectorTable
select [cluster], [word], [power] = avg([power])
INTO #avgVectorTable
FROM #GlobalObjectClusterTableMAINFINAL
GROUP BY [cluster], [word]
ORDER BY [cluster], [word]

-- выводим результат средних векторов
DECLARE curs1 CURSOR LOCAL FAST_FORWARD FOR
       select [cluster], [word], [power]
       from #avgVectorTable
       GROUP BY [cluster], [word], [power]
       ORDER BY [cluster], [word]
OPEN curs1
FETCH curs1 INTO @curs1_cluster, @curs1_word, @curs1_power
WHILE @@FETCH_STATUS = 0
BEGIN
       PRINT 'Итерация: ' + CONVERT(varchar(10), @iteration) +'   Кластер с номером: ' +
CONVERT(varchar(10), @curs1_cluster) + '    для координаты: ' + @curs1_word + '  имеет среднее
значение: ' + CONVERT(varchar(10), @curs1_power) + ';'
       FETCH curs1 INTO @curs1_cluster, @curs1_word, @curs1_power
END
CLOSE curs1
DEALLOCATE curs1

PRINT '';

IF OBJECT_ID('tempdb..#euklidDistTableAVG') IS NOT NULL DROP TABLE #euklidDistTableAVG
CREATE TABLE #euklidDistTableAVG ([avgCluster] int,  [EuklidDist] float)

INSERT #euklidDistTableAVG ([avgCluster],  [EuklidDist])
select aVT1.[cluster],
EuklidDist = CASE
                            WHEN @dist_type = 1 then POWER(SUM(POWER((aVT1.[power] - aVT2.[power]),
2)),0.5)
                            WHEN @dist_type = 2 then POWER(SUM(POWER((aVT1.[power] - aVT2.[power]),
2)),1)
                            WHEN @dist_type = 3 then SUM(ABS(aVT1.[power] - aVT2.[power]))
                     END
```

```
from #avgVectorTable aVT1
OUTER APPLY
(
	select [cluster], [word], [power] from #avgVectorTable
) aVT2
WHERE aVT1.word = aVT2.word
AND aVT1.cluster <> aVT2.cluster
GROUP BY aVT1.[cluster]
OPTION (RECOMPILE);

SELECT @min_cluster_euklide = AVG([EuklidDist]) FROM #euklidDistTableAVG
--SELECT @min_cluster_euklide = @max_euklide_distance
-- Расчёт меры близости между объектами и центроидами
IF OBJECT_ID('tempdb..#euklidDistTable') IS NOT NULL DROP TABLE #euklidDistTable
CREATE TABLE #euklidDistTable ([object_id] int, [avgCluster] int,  [EuklidDist] float)

INSERT #euklidDistTable ([object_id], [avgCluster],  [EuklidDist])
SELECT GOCT.[object_id], avgCluster = aVT.[cluster],
EuklidDist = CASE
                                WHEN @dist_type = 1 then POWER(SUM(POWER((GOCT.[power] - aVT.[power]),
2)),0.5)
                                WHEN @dist_type = 2 then POWER(SUM(POWER((GOCT.[power] - aVT.[power]),
2)),1)
                                WHEN @dist_type = 3 then SUM(ABS(GOCT.[power] - aVT.[power]))
                        END
FROM #GlobalObjectClusterTable GOCT
OUTER APPLY
(
	select [cluster], [word], [power] from #avgVectorTable
) aVT
WHERE aVT.word = GOCT.word
GROUP BY GOCT.[object_id], aVT.[cluster]
OPTION (RECOMPILE);

-- для каждого объекта находим кластер с минимальным растоянием к центру.
IF OBJECT_ID('tempdb..#euklidDistTableMinIN') IS NOT NULL DROP TABLE #euklidDistTableMinIN
CREATE TABLE #euklidDistTableMinIN ([object_id] int not null, [avgCluster] int, [EuklidDistMin] float)
INSERT #euklidDistTableMinIN ([object_id], [avgCluster], [EuklidDistMin])
select top(1) with ties [object_id], [avgCluster], [EuklidDist]
FROM #euklidDistTable eDT WHERE EuklidDist < @min_cluster_euklide
order by row_number() over (partition by [object_id] order by EuklidDist ASC)

-- для каждого объекта находим кластер с минимальным растоянием к центру.
IF OBJECT_ID('tempdb..#euklidDistTableMinOUT') IS NOT NULL DROP TABLE #euklidDistTableMinOUT
CREATE TABLE #euklidDistTableMinOUT ([object_id] int not null, [avgCluster] int, [EuklidDistMin] float)
INSERT #euklidDistTableMinOUT ([object_id], [avgCluster], [EuklidDistMin])
select top(1) with ties [object_id], [avgCluster], [EuklidDist]
FROM #euklidDistTable eDT WHERE EuklidDist >= @min_cluster_euklide
-- Условие для оставщихся не кластеризованных объектов
AND object_id NOT IN (SELECT object_id FROM #euklidDistTableMinIN)
order by row_number() over (partition by [object_id] order by EuklidDist ASC)

PRINT
'/*************************************************************************************************
**\'
PRINT '|*****************************Отчёт по
распределению*************************************************|'
PRINT
'\*************************************************************************************************
**/'
DECLARE curs2 CURSOR LOCAL FAST_FORWARD FOR
	select [object_id], [avgCluster], [EuklidDistMin]
	from #euklidDistTableMinIN
	ORDER BY [object_id]
OPEN curs2
FETCH curs2 INTO @curs2_object_id, @curs2_avgCluster, @curs2_EuklidDistMin
WHILE @@FETCH_STATUS = 0
BEGIN
	PRINT 'Объект с номером: ' + CONVERT(varchar(10), @curs2_object_id) + '   был распределён в
кластер: ' + CONVERT(varchar(10), @curs2_avgCluster) + ';'
	FETCH curs2 INTO @curs2_object_id, @curs2_avgCluster, @curs2_EuklidDistMin
END
CLOSE curs2
DEALLOCATE curs2

DECLARE curs3 CURSOR LOCAL FAST_FORWARD FOR
	select [object_id], [avgCluster], [EuklidDistMin]
	from #euklidDistTableMinOUT
```

```
	ORDER BY [object_id]
OPEN curs3
FETCH curs3 INTO @curs3_object_id, @curs3_avgCluster, @curs3_EuklidDistMin
WHILE @@FETCH_STATUS = 0
BEGIN
	PRINT 'Объект с номером: ' + CONVERT(varchar(10), @curs3_object_id) + '   не был распределён по
кластерам, ближайший кластер: ' + CONVERT(varchar(10), @curs3_avgCluster) + ';'
	FETCH curs3 INTO @curs3_object_id, @curs3_avgCluster, @curs3_EuklidDistMin
END
CLOSE curs3
DEALLOCATE curs3

--
declare @new_page_cnt float, @new_page_cnt_macro float
SELECT @new_page_cnt = COUNT(DISTINCT eDTMIN.[object_id])  FROM #euklidDistTableMinIN eDTMIN
SELECT @new_page_cnt = @new_page_cnt + COUNT(DISTINCT eDTMOUT.[object_id]) FROM #euklidDistTableMinOUT
eDTMOUT
IF OBJECT_ID('tempdb..#new_pages_clustering_Macro') IS NOT NULL DROP TABLE #new_pages_clustering_Macro
CREATE TABLE #new_pages_clustering_Macro (page_id int NOT NULL, resp_id int NOT NULL, PRIMARY
KEY(page_id, resp_id))
INSERT #new_pages_clustering_Macro (page_id, resp_id)
select page_id = eDTM.[object_id], resp_id = -GOCTMF.[object_id]
from #euklidDistTableMinIN  eDTM
INNER JOIN
#GlobalObjectClusterTableMAINFINAL GOCTMF
ON eDTM.avgCluster = GOCTMF.cluster
WHERE GOCTMF.object_id < 0 --(<0 признак ИП)
GROUP BY  eDTM.[object_id], -GOCTMF.[object_id]

SELECT @new_page_cnt_macro = COUNT(DISTINCT [page_id]) FROM #new_pages_clustering_Macro

--SELECT case WHEN rphcp.resp_id IS NULL THEN 0 ELSE 1 END, COUNT(DISTINCT npcM.page_id)
--FROM #new_pages_clustering_Macro npcM
--LEFT JOIN
--dbo.resp_page_hour_current_period rphcp
--ON npcM.resp_id = rphcp.resp_id
--AND npcM.page_id = rphcp.page_id
--AND rphcp.day_id = @day_id AND rphcp.hour_start = @hour_start
--GROUP BY case WHEN rphcp.resp_id IS NULL THEN 0 ELSE 1 END

--количество ИП для которых подошёл прогноз после кластеризации
SELECT @cnt_resp_new_OK = COUNT(DISTINCT rphcp.resp_id)
FROM #new_pages_clustering_Macro npcM
inner JOIN
dbo.resp_page_hour_current_period rphcp
ON npcM.resp_id = rphcp.resp_id
AND npcM.page_id = rphcp.page_id
AND rphcp.day_id = @day_id AND rphcp.hour_start = @hour_start

-- количество всех ИП в новом интервале.
SELECT @cnt_resp_new_ALL = COUNT(DISTINCT rphcp.resp_id)
FROM
dbo.resp_page_hour_current_period rphcp
inner JOIN
#new_pages_clustering_Macro npcM
ON rphcp.resp_id = npcM.resp_id
WHERE rphcp.day_id = @day_id AND rphcp.hour_start = @hour_start

select cnt_resp_new_OK = @cnt_resp_new_OK, cnt_resp_new_ALL = @cnt_resp_new_ALL, prognoz =
cast(@cnt_resp_new_OK as float)*100/cast(@cnt_resp_new_ALL as FLOAT)
-- процент кластеризации
SELECT new_page_cnt_macro = @new_page_cnt_macro, new_page_cnt = @new_page_cnt, clustering_percentage =
@new_page_cnt_macro*100/@new_page_cnt


SET NOCOUNT OFF
SET XACT_ABORT OFF
```

Процедуры по оценке качества персонализации поиска

```
-- az_pages -- интернет-ресурсы
-- az_word -- словарь терминов (длина терминов >=4)
-- az_words_small -- упрощённый словарь терминов
USE InternetDB2
SET NOCOUNT ON
SET XACT_ABORT ON

DECLARE @cnt_dic int, @cnt_dic_small int
```

```
SELECT @cnt_dic = COUNT(*) FROM [InternetDB2].[dbo].[az_words]
PRINT 'Размер обобщённого словаря терминов: ' + CAST(@cnt_dic as varchar(10)) + ';'

SELECT @cnt_dic_small = COUNT(*) FROM [InternetDB2].[dbo].[az_words_small]
PRINT 'Размер упращенноо обобщённого словаря терминов:     ' + CAST(@cnt_dic_small as varchar(10)) +
';'

-- Формируем таблицу не нулевых координат
IF OBJECT_ID('tempdb..#page_vector') IS NOT NULL DROP TABLE #page_vector
CREATE TABLE #page_vector (page_id int not null, word_id int not null, power float not null)
INSERT #page_vector(page_id,word_id, power)
SELECT P.page_id, word_small_id = WS.word_id, power = SUM(PW.power) FROM dbo.az_pages P
inner join dbo.az_page_word PW
ON P.page_id = PW.page_id
inner join dbo.az_words W
ON PW.word_id = W.word_id
inner join dbo.link_word_small_w lws
ON W.word_id = lws.word_id
inner join dbo.az_words_small WS
ON WS.word_id = lws.word_small_id
GROUP BY P.page_id, WS.word_id
ORDER BY P.page_id, WS.word_id


-- Формируем характеристические вектора
IF OBJECT_ID('tempdb..#vector_page') IS NOT NULL DROP TABLE #vector_page
CREATE TABLE #vector_page (page_id int not null, word_id int not null, power float not null)
declare @page_id int
DECLARE curs CURSOR LOCAl FAST_FORWARD FOR
select page_id FROM az_pages
OPEN curs
FETCH curs INTO @page_id
WHILE @@FETCH_STATUS = 0
BEGIN
       INSERT #vector_page (page_id, word_id, power)
       SELECT page_id = @page_id, WS.word_id, power = CASE when A.power IS null THEN 100 ELSE A.power
END FROM
       (SELECT * FROM #page_vector pv where page_id = @page_id) A
       RIGHT join
       dbo.az_words_small WS
       ON A.word_id = WS.word_id
       FETCH curs INTO @page_id
END
CLOSE curs
DEALLOCATE curs

-- Пусть имеем 3 ИП выполняющие заходы на ИР в зависимости от поисковых интересов
-- Имитируем действия ИП.
-- ИП будут использованы в качестве центроидов при первой инициализации объектов
IF OBJECT_ID('tempdb..#vector_user') IS NOT NULL DROP TABLE #vector_user
CREATE TABLE #vector_user (user_id smallint not null, word_id int not null, power float not null)
INSERT #vector_user (user_id, word_id, power)
SELECT 1, word_id, MAX(power)
FROM #vector_page
WHERE page_id IN (1, 48, 5/* ,13*/) and page_id <> 504
GROUP BY word_id
UNION
SELECT 1, word_id, power = 50
FROM #vector_page WHERE page_id = 504
GROUP BY word_id



INSERT #vector_user (user_id, word_id, power)
SELECT 2, word_id, MAX(power)
FROM #vector_page
WHERE page_id IN (15, 25/*, 33*/) and page_id <> 504
GROUP BY word_id
UNION
SELECT 2, word_id, power = 50
FROM #vector_page WHERE page_id = 504
GROUP BY word_id

INSERT #vector_user (user_id, word_id, power)
SELECT 3, word_id, MAX(power)
FROM #vector_page
WHERE page_id IN (34, 42/*, 45*/)and page_id <> 504
GROUP BY word_id
UNION
```

```
SELECT 1, word_id, power = 50
FROM #vector_page WHERE page_id = 504
GROUP BY word_id


INSERT #vector_user (user_id, word_id, power)
SELECT 4, word_id, MAX(power)
FROM #vector_page
WHERE page_id IN (8, 20, 22, 44, 41, 43, 46)and page_id <> 504
GROUP BY word_id
UNION
SELECT 1, word_id, power = 50
FROM #vector_page WHERE page_id = 504
GROUP BY word_id


IF OBJECT_ID('tempdb..#euklidDistTable') IS NOT NULL DROP TABLE #euklidDistTable
CREATE TABLE #euklidDistTable ([object_id] int, [avgCluster] int,  [EuklidDist] float, PRIMARY KEY
([object_id], [avgCluster]))

PRINT 'Расчёт евклидово расстояние относительно центроидов'
INSERT #euklidDistTable ([object_id], [avgCluster], [EuklidDist])
SELECT  GOCT.[page_id],  avgCluster  =  aVT.[user_id],  EuklidDist  =  POWER(SUM(POWER((GOCT.[power]  -
aVT.[power]), 2)),0.5)
FROM #vector_page GOCT
INNER JOIN
(
       select [user_id], [word_id], [power] from #vector_user
) AS aVT
ON aVT.[word_id] = GOCT.[word_id]
GROUP BY GOCT.[page_id], aVT.[user_id]
ORDER BY GOCT.[page_id], aVT.[user_id]
OPTION (RECOMPILE);
PRINT 'Расчёт евклидово расстояние относительно центроидов выполнено'

-- для каждого объекта находим кластер с минимальным растоянием к центру.
IF OBJECT_ID('tempdb..#euklidDistTableMin') IS NOT NULL DROP TABLE #euklidDistTableMin
CREATE TABLE #euklidDistTableMin ([object_id] int not null, [avgCluster] int, [EuklidDistMin] float)
INSERT #euklidDistTableMin ([object_id], [avgCluster], [EuklidDistMin])
select top(1) with ties [object_id], [avgCluster], [EuklidDist]
FROM #euklidDistTable eDT
order by row_number() over (partition by [object_id] order by EuklidDist ASC)

IF OBJECT_ID('tempdb..#GlobalObjectClusterTable1') IS NOT NULL DROP TABLE #GlobalObjectClusterTable1
Create table #GlobalObjectClusterTable1([object_id] int not null, [word] varchar(50), [power] float,
[cluster] int)
-- Загружаем ИР
INSERT INTO #GlobalObjectClusterTable1 ([object_id], [word], [power], [cluster])
SELECT page_id, CAST(word_id as varchar(10)), power, 0 FROM #vector_page
-- Загружаем ИП
--INSERT INTO #GlobalObjectClusterTable1 ([object_id], [word], [power], [cluster])
--SELECT -[user_id], CAST(word_id as varchar(10)), power, [user_id] FROM #vector_user


UPDATE GOCT SET GOCT.[cluster] = C.[avgCluster]
FROM
#GlobalObjectClusterTable1 GOCT
INNER JOIN
#euklidDistTableMin C
ON GOCT.[object_id] = C.[object_id]

IF EXISTS (SELECT NULL FROM #GlobalObjectClusterTable1 WHERE [cluster] = 0)
BEGIN
       RAISERROR('Имеются не кластеризованные объекты',11,1);
END

PRINT
'/*************************************************************************************************
**\'
PRINT                          '|********************Результат                              предварительной
кластеризации*****************************************|'
PRINT
'\*************************************************************************************************
**/'
IF OBJECT_ID('tempdb..#ClusterContent1') IS NOT NULL DROP TABLE #ClusterContent1
       CREATE TABLE #ClusterContent1 ([cluster] int NOT NULL PRIMARY KEY, [cluster_cntOBJ] int, [OBJS]
varchar(max))
       INSERT #ClusterContent1 ([cluster], [cluster_cntOBJ])
       select [cluster], COUNT(DISTINCT [object_id])
```

```
              FROM #GlobalObjectClusterTable1
              GROUP BY [cluster]
              ORDER BY [cluster]

       UPDATE CC SET
    CC.OBJS =  REPLACE(
                           REPLACE(
                                  CAST(
                                         (
                                                select    distinct    convert(varchar(1000),
[object_id]) AS 'Itm'
                                                FROM #GlobalObjectClusterTable1 GOCT
                                                WHERE GOCT.cluster = CC.cluster
                                                FOR XML PATH('')
                                         )AS varchar(MAX)
                                  ),            --//получаем            строку            вида
<Itm>ids1</Itm><Itm>Ids2</Itm><Itm>Ids3</Itm>...
           '<Itm>',  ''),
         '</Itm>',  ';')
       from
              #ClusterContent1 CC

--// удаляем последнюю точку-запятую
       UPDATE CC SET CC.OBJS = CASE
                                                WHEN  SUBSTRING(CC.OBJS,  LEN(CC.OBJS),  1)  =
';' THEN LEFT(CC.OBJS, LEN(CC.OBJS) -1)
                                                ELSE CC.OBJS
                                         END
       FROM #ClusterContent1 CC

       declare @curs2_cluster int;
       declare @curs2_cntOBJ int;
       declare @curs2_OBJS varchar(max);
       DECLARE curs2 cursor LOCAL FAST_FORWARD for
              SELECT [cluster], [cluster_cntOBJ], [OBJS] FROM #ClusterContent1
       OPEN curs2
       FETCH curs2 INTO @curs2_cluster, @curs2_cntOBJ, @curs2_OBJS
       WHILE @@FETCH_STATUS = 0
       BEGIN
              PRINT 'В кластере с номером: ' + CONVERT(varchar(100),@curs2_cluster) + ' содержится:   '
+ CONVERT(varchar(100), @curs2_cntOBJ) + ' объектов:';
              PRINT '{';
              PRINT ' ' + @curs2_OBJS;
              PRINT '}';
              FETCH curs2 INTO @curs2_cluster, @curs2_cntOBJ, @curs2_OBJS;
       END
       CLOSE curs2;
       DEALLOCATE curs2;

PRINT                                                 '=============================================Конец
отчёта================================================';


IF     OBJECT_ID('tempdb..#GlobalObjectClusterTableMAIN')     IS     NOT     NULL     DROP     TABLE
#GlobalObjectClusterTableMAIN
CREATE TABLE #GlobalObjectClusterTableMAIN
([object_id] int NOT NULL, [word] varchar(50) NOT NULL, [power] float NOT NULL, [cluster] int NOT NULL,
PRIMARY KEY ([object_id], [word]))
INSERT INTO #GlobalObjectClusterTableMAIN ([object_id], [word], [power], [cluster])
SELECT [object_id], [word], [power], [cluster] FROM #GlobalObjectClusterTable1


IF    OBJECT_ID('tempdb..#GlobalObjectClusterTableMAINFINAL')    IS    NOT    NULL    DROP    TABLE
#GlobalObjectClusterTableMAINFINAL
CREATE TABLE #GlobalObjectClusterTableMAINFINAL
([object_id] int NOT NULL, [word] varchar(50) NOT NULL, [power] float NOT NULL, [cluster] int NOT NULL,
PRIMARY KEY ([object_id], [word]))
INSERT INTO #GlobalObjectClusterTableMAINFINAL ([object_id], [word], [power], [cluster])
exec InternetDB2.dbo.az_clustering_step3


PRINT
'/*************************************************************************************************
**\'
PRINT                        '|********************Результат                          *****************
кластеризации*****************************************|'
PRINT
'\*************************************************************************************************
**/'
```

```
IF OBJECT_ID('tempdb..#ClusterContent') IS NOT NULL DROP TABLE #ClusterContent
       CREATE TABLE #ClusterContent ([cluster] int NOT NULL PRIMARY KEY, [cluster_cntOBJ] int, [OBJS]
varchar(max))
       INSERT #ClusterContent ([cluster], [cluster_cntOBJ])
       select [cluster], COUNT(DISTINCT [object_id])
              FROM #GlobalObjectClusterTableMAINFINAL
              GROUP BY [cluster]
              ORDER BY [cluster]

       UPDATE CC SET
    CC.OBJS =  REPLACE(
                           REPLACE(
                                  CAST(
                                         (
                                                       select    distinct    convert(varchar(1000),
[object_id]) AS 'Itm'
                                                       FROM #GlobalObjectClusterTableMAINFINAL GOCT
                                                       WHERE GOCT.cluster = CC.cluster
                                                       FOR XML PATH('')
                                          )AS varchar(MAX)
                                   ),            --//получаем            строку            вида
<Itm>ids1</Itm><Itm>Ids2</Itm><Itm>Ids3</Itm>...
            '<Itm>',  ''),
          '</Itm>',  ';')
       from
              #ClusterContent CC

--// удаляем последнюю точку-запятую
       UPDATE CC SET CC.OBJS = CASE
                                                       WHEN  SUBSTRING(CC.OBJS,  LEN(CC.OBJS),  1)  =
';' THEN LEFT(CC.OBJS, LEN(CC.OBJS) -1)
                                                       ELSE CC.OBJS
                                                END
       FROM #ClusterContent CC

       declare @curs3_cluster int;
       declare @curs3_cntOBJ int;
       declare @curs3_OBJS varchar(max);
       DECLARE curs3 cursor LOCAL FAST_FORWARD for
              SELECT [cluster], [cluster_cntOBJ], [OBJS] FROM #ClusterContent
       OPEN curs3
       FETCH curs3 INTO @curs3_cluster, @curs3_cntOBJ, @curs3_OBJS
       WHILE @@FETCH_STATUS = 0
       BEGIN
              PRINT 'В кластере с номером: ' + CONVERT(varchar(100),@curs3_cluster) + ' содержится:   '
+ CONVERT(varchar(100), @curs3_cntOBJ) + ' объектов:';
              PRINT '{';
              PRINT ' ' + @curs3_OBJS;
              PRINT '}';
              FETCH curs3 INTO @curs3_cluster, @curs3_cntOBJ, @curs3_OBJS;
       END
       CLOSE curs3;
       DEALLOCATE curs3;
```

Шаблон выходного отчёта

```
**********Prepare
Procedure**************************************************************************
*************************Author: Zein A.N. - PhD Student, MPEI, VMSS
(ZeynAN@mpei.ru)*****************
*****************************************************************************************************
**
---------------------------------Отчёт по количеству ИП по дням----------------------------------------
-
Количество ИП в 2014-04-21: 352;
Количество ИП в 2014-04-22: 352;
Количество ИП в 2014-04-23: 352;
Количество ИП в 2014-04-24: 352;
Количество ИП в 2014-04-25: 352;
Количество ИП в 2014-04-26: 352;
Количество ИП в 2014-04-27: 352;

-----------------------------------Отчёт по номеру первого ИП по периодам------------------------------
В 2014-04-21 в периоде с 0    по     3      первым ИП оказался:282618;
В 2014-04-21 в периоде с 4    по     7      первым ИП оказался:287234;
В 2014-04-21 в периоде с 8    по     11     первым ИП оказался:206266;
В 2014-04-21 в периоде с 12   по     15     первым ИП оказался:243214;
В 2014-04-21 в периоде с 16   по     19     первым ИП оказался:378532;
В 2014-04-21 в периоде с 20   по     23     первым ИП оказался:211797;
```

В 2014-04-22 в периоде с 0 по 3 первым ИП оказался:176763;
В 2014-04-22 в периоде с 4 по 7 первым ИП оказался:261224;
В 2014-04-22 в периоде с 8 по 11 первым ИП оказался:176763;
В 2014-04-22 в периоде с 12 по 15 первым ИП оказался:271211;
В 2014-04-22 в периоде с 16 по 19 первым ИП оказался:205939;
В 2014-04-22 в периоде с 20 по 23 первым ИП оказался:361413;
В 2014-04-23 в периоде с 0 по 3 первым ИП оказался:174361;
В 2014-04-23 в периоде с 4 по 7 первым ИП оказался:275099;
В 2014-04-23 в периоде с 8 по 11 первым ИП оказался:277577;
В 2014-04-23 в периоде с 12 по 15 первым ИП оказался:215822;
В 2014-04-23 в периоде с 16 по 19 первым ИП оказался:354571;
В 2014-04-23 в периоде с 20 по 23 первым ИП оказался:358268;
В 2014-04-24 в периоде с 0 по 3 первым ИП оказался:217178;
В 2014-04-24 в периоде с 4 по 7 первым ИП оказался:197123;
В 2014-04-24 в периоде с 8 по 11 первым ИП оказался:306283;
В 2014-04-24 в периоде с 12 по 15 первым ИП оказался:275885;
В 2014-04-24 в периоде с 16 по 19 первым ИП оказался:377511;
В 2014-04-24 в периоде с 20 по 23 первым ИП оказался:358268;
В 2014-04-25 в периоде с 0 по 3 первым ИП оказался:379940;
В 2014-04-25 в периоде с 4 по 7 первым ИП оказался:200738;
В 2014-04-25 в периоде с 8 по 11 первым ИП оказался:379940;
В 2014-04-25 в периоде с 12 по 15 первым ИП оказался:378532;
В 2014-04-25 в периоде с 16 по 19 первым ИП оказался:379940;
В 2014-04-25 в периоде с 20 по 23 первым ИП оказался:250399;
В 2014-04-26 в периоде с 0 по 3 первым ИП оказался:225776;
В 2014-04-26 в периоде с 4 по 7 первым ИП оказался:367010;
В 2014-04-26 в периоде с 8 по 11 первым ИП оказался:200738;
В 2014-04-26 в периоде с 12 по 15 первым ИП оказался:226439;
В 2014-04-26 в периоде с 16 по 19 первым ИП оказался:334614;
В 2014-04-26 в периоде с 20 по 23 первым ИП оказался:372363;
В 2014-04-27 в периоде с 0 по 3 первым ИП оказался:414487;
В 2014-04-27 в периоде с 4 по 7 первым ИП оказался:361413;
В 2014-04-27 в периоде с 8 по 11 первым ИП оказался:174392;
В 2014-04-27 в периоде с 12 по 15 первым ИП оказался:340443;
В 2014-04-27 в периоде с 16 по 19 первым ИП оказался:362508;
В 2014-04-27 в периоде с 20 по 23 первым ИП оказался:351347;

----------Отчёт по количеству ИП по периодам----------

В 2014-04-21 в периоде с 0 по 3 количество ИП оказалось:113;
В 2014-04-21 в периоде с 4 по 7 количество ИП оказалось:54;
В 2014-04-21 в периоде с 8 по 11 количество ИП оказалось:183;
В 2014-04-21 в периоде с 12 по 15 количество ИП оказалось:249;
В 2014-04-21 в периоде с 16 по 19 количество ИП оказалось:267;
В 2014-04-21 в периоде с 20 по 23 количество ИП оказалось:248;
В 2014-04-22 в периоде с 0 по 3 количество ИП оказалось:124;
В 2014-04-22 в периоде с 4 по 7 количество ИП оказалось:52;
В 2014-04-22 в периоде с 8 по 11 количество ИП оказалось:185;
В 2014-04-22 в периоде с 12 по 15 количество ИП оказалось:229;
В 2014-04-22 в периоде с 16 по 19 количество ИП оказалось:257;
В 2014-04-22 в периоде с 20 по 23 количество ИП оказалось:253;
В 2014-04-23 в периоде с 0 по 3 количество ИП оказалось:114;
В 2014-04-23 в периоде с 4 по 7 количество ИП оказалось:58;
В 2014-04-23 в периоде с 8 по 11 количество ИП оказалось:185;
В 2014-04-23 в периоде с 12 по 15 количество ИП оказалось:242;
В 2014-04-23 в периоде с 16 по 19 количество ИП оказалось:264;
В 2014-04-23 в периоде с 20 по 23 количество ИП оказалось:254;
В 2014-04-24 в периоде с 0 по 3 количество ИП оказалось:120;
В 2014-04-24 в периоде с 4 по 7 количество ИП оказалось:61;
В 2014-04-24 в периоде с 8 по 11 количество ИП оказалось:183;
В 2014-04-24 в периоде с 12 по 15 количество ИП оказалось:241;
В 2014-04-24 в периоде с 16 по 19 количество ИП оказалось:241;
В 2014-04-24 в периоде с 20 по 23 количество ИП оказалось:261;
В 2014-04-25 в периоде с 0 по 3 количество ИП оказалось:134;
В 2014-04-25 в периоде с 4 по 7 количество ИП оказалось:61;
В 2014-04-25 в периоде с 8 по 11 количество ИП оказалось:187;
В 2014-04-25 в периоде с 12 по 15 количество ИП оказалось:239;
В 2014-04-25 в периоде с 16 по 19 количество ИП оказалось:253;
В 2014-04-25 в периоде с 20 по 23 количество ИП оказалось:251;
В 2014-04-26 в периоде с 0 по 3 количество ИП оказалось:143;
В 2014-04-26 в периоде с 4 по 7 количество ИП оказалось:57;
В 2014-04-26 в периоде с 8 по 11 количество ИП оказалось:158;
В 2014-04-26 в периоде с 12 по 15 количество ИП оказалось:232;
В 2014-04-26 в периоде с 16 по 19 количество ИП оказалось:231;
В 2014-04-26 в периоде с 20 по 23 количество ИП оказалось:249;
В 2014-04-27 в периоде с 0 по 3 количество ИП оказалось:145;
В 2014-04-27 в периоде с 4 по 7 количество ИП оказалось:42;
В 2014-04-27 в периоде с 8 по 11 количество ИП оказалось:163;
В 2014-04-27 в периоде с 12 по 15 количество ИП оказалось:243;

```
В 2014-04-27 в периоде с      16      по      19      количество ИП оказалось:239;
В 2014-04-27 в периоде с      20      по      23      количество ИП оказалось:259;

---------------------------------------Отчёт по количеству ИР по периодам------------------------------
-----
В 2014-04-21 в периоде с      0       по      3       количество ИР оказалось:96;
В 2014-04-21 в периоде с      4       по      7       количество ИР оказалось:97;
В 2014-04-21 в периоде с      8       по      11      количество ИР оказалось:249;
В 2014-04-21 в периоде с      12      по      15      количество ИР оказалось:349;
В 2014-04-21 в периоде с      16      по      19      количество ИР оказалось:294;
В 2014-04-21 в периоде с      20      по      23      количество ИР оказалось:185;
В 2014-04-22 в периоде с      0       по      3       количество ИР оказалось:79;
В 2014-04-22 в периоде с      4       по      7       количество ИР оказалось:104;
В 2014-04-22 в периоде с      8       по      11      количество ИР оказалось:234;
В 2014-04-22 в периоде с      12      по      15      количество ИР оказалось:332;
В 2014-04-22 в периоде с      16      по      19      количество ИР оказалось:306;
В 2014-04-22 в периоде с      20      по      23      количество ИР оказалось:300;
В 2014-04-23 в периоде с      0       по      3       количество ИР оказалось:109;
В 2014-04-23 в периоде с      4       по      7       количество ИР оказалось:94;
В 2014-04-23 в периоде с      8       по      11      количество ИР оказалось:219;
В 2014-04-23 в периоде с      12      по      15      количество ИР оказалось:290;
В 2014-04-23 в периоде с      16      по      19      количество ИР оказалось:273;
В 2014-04-23 в периоде с      20      по      23      количество ИР оказалось:344;
В 2014-04-24 в периоде с      0       по      3       количество ИР оказалось:106;
В 2014-04-24 в периоде с      4       по      7       количество ИР оказалось:145;
В 2014-04-24 в периоде с      8       по      11      количество ИР оказалось:282;
В 2014-04-24 в периоде с      12      по      15      количество ИР оказалось:336;
В 2014-04-24 в периоде с      16      по      19      количество ИР оказалось:374;
В 2014-04-24 в периоде с      20      по      23      количество ИР оказалось:374;
В 2014-04-25 в периоде с      0       по      3       количество ИР оказалось:92;
В 2014-04-25 в периоде с      4       по      7       количество ИР оказалось:137;
В 2014-04-25 в периоде с      8       по      11      количество ИР оказалось:298;
В 2014-04-25 в периоде с      12      по      15      количество ИР оказалось:319;
В 2014-04-25 в периоде с      16      по      19      количество ИР оказалось:293;
В 2014-04-25 в периоде с      20      по      23      количество ИР оказалось:267;
В 2014-04-26 в периоде с      0       по      3       количество ИР оказалось:126;
В 2014-04-26 в периоде с      4       по      7       количество ИР оказалось:111;
В 2014-04-26 в периоде с      8       по      11      количество ИР оказалось:202;
В 2014-04-26 в периоде с      12      по      15      количество ИР оказалось:289;
В 2014-04-26 в периоде с      16      по      19      количество ИР оказалось:224;
В 2014-04-26 в периоде с      20      по      23      количество ИР оказалось:217;
В 2014-04-27 в периоде с      0       по      3       количество ИР оказалось:157;
В 2014-04-27 в периоде с      4       по      7       количество ИР оказалось:104;
В 2014-04-27 в периоде с      8       по      11      количество ИР оказалось:278;
В 2014-04-27 в периоде с      12      по      15      количество ИР оказалось:261;
В 2014-04-27 в периоде с      16      по      19      количество ИР оказалось:224;
В 2014-04-27 в периоде с      20      по      23      количество ИР оказалось:249;

-----------------------------------Отчёт по количеству ИП и ИР по периодам---------------------------
-----
В 2014-04-21 в периоде с      0       по      3       количество объектов (ИП + ИР) оказалось:209;
В 2014-04-21 в периоде с      4       по      7       количество объектов (ИП + ИР) оказалось:151;
В 2014-04-21 в периоде с      8       по      11      количество объектов (ИП + ИР) оказалось:432;
В 2014-04-21 в периоде с      12      по      15      количество объектов (ИП + ИР) оказалось:598;
В 2014-04-21 в периоде с      16      по      19      количество объектов (ИП + ИР) оказалось:561;
В 2014-04-21 в периоде с      20      по      23      количество объектов (ИП + ИР) оказалось:433;
В 2014-04-22 в периоде с      0       по      3       количество объектов (ИП + ИР) оказалось:203;
В 2014-04-22 в периоде с      4       по      7       количество объектов (ИП + ИР) оказалось:156;
В 2014-04-22 в периоде с      8       по      11      количество объектов (ИП + ИР) оказалось:419;
В 2014-04-22 в периоде с      12      по      15      количество объектов (ИП + ИР) оказалось:561;
В 2014-04-22 в периоде с      16      по      19      количество объектов (ИП + ИР) оказалось:563;
В 2014-04-22 в периоде с      20      по      23      количество объектов (ИП + ИР) оказалось:553;
В 2014-04-23 в периоде с      0       по      3       количество объектов (ИП + ИР) оказалось:223;
В 2014-04-23 в периоде с      4       по      7       количество объектов (ИП + ИР) оказалось:152;
В 2014-04-23 в периоде с      8       по      11      количество объектов (ИП + ИР) оказалось:404;
В 2014-04-23 в периоде с      12      по      15      количество объектов (ИП + ИР) оказалось:532;
В 2014-04-23 в периоде с      16      по      19      количество объектов (ИП + ИР) оказалось:537;
В 2014-04-23 в периоде с      20      по      23      количество объектов (ИП + ИР) оказалось:598;
В 2014-04-24 в периоде с      0       по      3       количество объектов (ИП + ИР) оказалось:226;
В 2014-04-24 в периоде с      4       по      7       количество объектов (ИП + ИР) оказалось:206;
В 2014-04-24 в периоде с      8       по      11      количество объектов (ИП + ИР) оказалось:465;
В 2014-04-24 в периоде с      12      по      15      количество объектов (ИП + ИР) оказалось:577;
В 2014-04-24 в периоде с      16      по      19      количество объектов (ИП + ИР) оказалось:615;
В 2014-04-24 в периоде с      20      по      23      количество объектов (ИП + ИР) оказалось:635;
В 2014-04-25 в периоде с      0       по      3       количество объектов (ИП + ИР) оказалось:226;
В 2014-04-25 в периоде с      4       по      7       количество объектов (ИП + ИР) оказалось:198;
В 2014-04-25 в периоде с      8       по      11      количество объектов (ИП + ИР) оказалось:485;
В 2014-04-25 в периоде с      12      по      15      количество объектов (ИП + ИР) оказалось:558;
В 2014-04-25 в периоде с      16      по      19      количество объектов (ИП + ИР) оказалось:546;
```

```
В 2014-04-25 в периоде с      20      по      23      количество объектов (ИП + ИР) оказалось:518;
В 2014-04-26 в периоде с      0       по      3       количество объектов (ИП + ИР) оказалось:269;
В 2014-04-26 в периоде с      4       по      7       количество объектов (ИП + ИР) оказалось:168;
В 2014-04-26 в периоде с      8       по      11      количество объектов (ИП + ИР) оказалось:360;
В 2014-04-26 в периоде с      12      по      15      количество объектов (ИП + ИР) оказалось:521;
В 2014-04-26 в периоде с      16      по      19      количество объектов (ИП + ИР) оказалось:455;
В 2014-04-26 в периоде с      20      по      23      количество объектов (ИП + ИР) оказалось:466;
В 2014-04-27 в периоде с      0       по      3       количество объектов (ИП + ИР) оказалось:302;
В 2014-04-27 в периоде с      4       по      7       количество объектов (ИП + ИР) оказалось:146;
В 2014-04-27 в периоде с      8       по      11      количество объектов (ИП + ИР) оказалось:441;
В 2014-04-27 в периоде с      12      по      15      количество объектов (ИП + ИР) оказалось:504;
В 2014-04-27 в периоде с      16      по      19      количество объектов (ИП + ИР) оказалось:463;
В 2014-04-27 в периоде с      20      по      23      количество объектов (ИП + ИР) оказалось:508;

Warning: Null value is eliminated by an aggregate or other SET operation.
-----------------------------------Отчёт по размеру глобального словаря терминов по периодам---------
----------------------
В 2014-04-21 в периоде с      0       по      3       размер глобального словаря терминов:1989;
В 2014-04-21 в периоде с      4       по      7       размер глобального словаря терминов:2944;
В 2014-04-21 в периоде с      8       по      11      размер глобального словаря терминов:4374;
В 2014-04-21 в периоде с      12      по      15      размер глобального словаря терминов:4306;
В 2014-04-21 в периоде с      16      по      19      размер глобального словаря терминов:4155;
В 2014-04-21 в периоде с      20      по      23      размер глобального словаря терминов:3093;
В 2014-04-22 в периоде с      0       по      3       размер глобального словаря терминов:1875;
В 2014-04-22 в периоде с      4       по      7       размер глобального словаря терминов:3105;
В 2014-04-22 в периоде с      8       по      11      размер глобального словаря терминов:3665;
В 2014-04-22 в периоде с      12      по      15      размер глобального словаря терминов:4592;
В 2014-04-22 в периоде с      16      по      19      размер глобального словаря терминов:4300;
В 2014-04-22 в периоде с      20      по      23      размер глобального словаря терминов:4019;
В 2014-04-23 в периоде с      0       по      3       размер глобального словаря терминов:2935;
В 2014-04-23 в периоде с      4       по      7       размер глобального словаря терминов:2211;
В 2014-04-23 в периоде с      8       по      11      размер глобального словаря терминов:4652;
В 2014-04-23 в периоде с      12      по      15      размер глобального словаря терминов:4365;
В 2014-04-23 в периоде с      16      по      19      размер глобального словаря терминов:4518;
В 2014-04-23 в периоде с      20      по      23      размер глобального словаря терминов:4489;
В 2014-04-24 в периоде с      0       по      3       размер глобального словаря терминов:2991;
В 2014-04-24 в периоде с      4       по      7       размер глобального словаря терминов:3813;
В 2014-04-24 в периоде с      8       по      11      размер глобального словаря терминов:4438;
В 2014-04-24 в периоде с      12      по      15      размер глобального словаря терминов:4609;
В 2014-04-24 в периоде с      16      по      19      размер глобального словаря терминов:4140;
В 2014-04-24 в периоде с      20      по      23      размер глобального словаря терминов:3613;
В 2014-04-25 в периоде с      0       по      3       размер глобального словаря терминов:2553;
В 2014-04-25 в периоде с      4       по      7       размер глобального словаря терминов:3192;
В 2014-04-25 в периоде с      8       по      11      размер глобального словаря терминов:4367;
В 2014-04-25 в периоде с      12      по      15      размер глобального словаря терминов:5005;
В 2014-04-25 в периоде с      16      по      19      размер глобального словаря терминов:4180;
В 2014-04-25 в периоде с      20      по      23      размер глобального словаря терминов:3814;
В 2014-04-26 в периоде с      0       по      3       размер глобального словаря терминов:2356;
В 2014-04-26 в периоде с      4       по      7       размер глобального словаря терминов:2784;
В 2014-04-26 в периоде с      8       по      11      размер глобального словаря терминов:3251;
В 2014-04-26 в периоде с      12      по      15      размер глобального словаря терминов:3909;
В 2014-04-26 в периоде с      16      по      19      размер глобального словаря терминов:3122;
В 2014-04-26 в периоде с      20      по      23      размер глобального словаря терминов:2873;
В 2014-04-27 в периоде с      0       по      3       размер глобального словаря терминов:1597;
В 2014-04-27 в периоде с      4       по      7       размер глобального словаря терминов:1799;
В 2014-04-27 в периоде с      8       по      11      размер глобального словаря терминов:3452;
В 2014-04-27 в периоде с      12      по      15      размер глобального словаря терминов:3180;
В 2014-04-27 в периоде с      16      по      19      размер глобального словаря терминов:2761;
В 2014-04-27 в периоде с      20      по      23      размер глобального словаря терминов:3114;
```

```
*******Clustering
Procedure*********************************************************************************

**************************Author:      Zein      A.N.      -      PhD      Student,      MPEI,      VMSS
(ZeynAN@mpei.ru)******************

**********************************************************************************************
**

====================================Начало
отчёта=============================================================
```

```
/*****************************************************************************************************
*\

|**********************************Отчёт                          по                          Евклидово
расстояние************************************|

\*****************************************************************************************************
*/

Евклидово расстояние между объектами с номерами: -416122   и -408665 равно: 122.

Евклидово расстояние между объектами с номерами: -416122   и -395772 равно: 28.

Евклидово расстояние между объектами с номерами: -416122   и -382151 равно: 567.

Евклидово расстояние между объектами с номерами: -416122   и -380227 равно: 70.

Евклидово расстояние между объектами с номерами: -416122   и -375607 равно: 361.

Евклидово расстояние между объектами с номерами: -416122   и -353269 равно: 434.

Евклидово расстояние между объектами с номерами: -416122   и -352925 равно: 0.

Евклидово расстояние между объектами с номерами: -416122   и -352122 равно: 360.

Евклидово расстояние между объектами с номерами: -416122   и -339606 равно: 467.

...

...

...

Евклидово расстояние между объектами с номерами: 1456864   и 1457043 равно: 297.

Евклидово расстояние между объектами с номерами: 1456865   и 1456876 равно: 573.

Евклидово расстояние между объектами с номерами: 1456865   и 1457042 равно: 300.

Евклидово расстояние между объектами с номерами: 1456865   и 1457043 равно: 300.

Евклидово расстояние между объектами с номерами: 1456876   и 1457042 равно: 575.

Евклидово расстояние между объектами с номерами: 1456876   и 1457043 равно: 575.

Евклидово расстояние между объектами с номерами: 1457042   и 1457043 равно: 0.

находим максимальное евклидово расстояние между ИП (object_id < 0)

Максимальное евклидово расстояние между ИП (object_id < 0) найдено

Находим ИП на самом отдалённом растоянии между собой

ИП на самом отдалённом растоянии между собой найдены

Расчёт евклидово расстояние относительно средних векторов

Расчёт евклидово расстояние относительно средних векторов выполнено

/*****************************************************************************************************
*\

|********************Результат                                                          предварительной
кластеризации*************************************|

\*****************************************************************************************************
*/

В кластере с номером: 1       содержится: 12 объектов:

{

        1442130;1442045;-181454;1441297;1442175;-232947;-333009;-382151;1442129;1442165;1442178;-208031
```

```
}

В кластере с номером: 2       содержится: 56 объектов:

{

       1444024;1451881;-398;1456324;1444097;1452257;1444377;1443277;1452597;-304572;1456688;1456876;-
270523;1452258;-211797;1452284;-352122;-21232;1456090;-375607;1451973;-
172457;1456128;1451657;1444376;1442936;1442968;1452283;1443966;1444025;1456145;1444023;1451879;1442863;
-48674;-193297;-258015;-
353269;1452596;1444374;1442874;1443318;1444375;1443317;1451901;1443282;1443461;1443965;-
339606;1456728;-35185;1444393;1456865;-164981;1456864;1440301

}

В кластере с номером: 3       содержится: 73 объектов:

{

       1441295;-416122;1440302;1442693;1455484;-352925;1444718;1457042;1440306;-189175;-
226326;1441300;-40838;-328658;-181006;1451174;1451471;-12059;1455482;-180163;1440307;1442138;-
227683;1449449;1450941;-200738;1442670;-17510;-181687;1439964;1452256;-
266044;1450875;1450880;1455647;1441303;-251667;1441305;1440308;1442709;1441304;1444719;-408665;-
180266;1442657;1442705;1457043;1440303;1442655;-395772;1440304;-217178;1440305;1455789;1442697;-
380227;1440309;1454181;-250399;-277577;-262211;1441301;1442081;1442131;-
221165;1444720;1452765;1442025;1442622;1442135;1442180;-277900;1443964

}

==========Конец
отчёта==========
```

```
*******Clustering
Procedure*********************************************************************************

**************************Author:      Zein      A.N.      -      PhD      Student,      MPEI,      VMSS
(ZeynAN@mpei.ru)******************

**********************************************************************************************************
**

====================================Начало               отчёта               -               итерация:
0=========================================

В кластере с номером: 1       содержится: 12 объектов:

{

       1442130;1442045;-181454;1441297;1442175;-232947;-333009;-382151;1442129;1442165;1442178;-208031

}

В кластере с номером: 2       содержится: 56 объектов:

{

       1444024;1451881;-398;1456324;1444097;1452257;1444377;1443277;-304572;1452597;1456688;1456876;-
270523;1452258;-211797;1452284;-352122;-21232;1456090;-375607;1451973;-
172457;1456128;1451657;1444376;1442936;1442968;1452283;1443966;1444025;1456145;1444023;1451879;1442863;
-48674;-193297;-258015;-
353269;1452596;1444374;1442874;1443318;1444375;1443317;1451901;1443282;1443461;1443965;-
339606;1456728;-35185;1444393;1456865;-164981;1456864;1440301

}

В кластере с номером: 3       содержится: 73 объектов:

{

       1441295;-416122;1440302;1442693;-352925;1455484;1444718;1457042;1440306;-189175;-
226326;1441300;-40838;-328658;-181006;1451174;1451471;-12059;1455482;-180163;1440307;1442138;-
227683;1449449;1450941;-200738;1442670;-17510;-181687;1439964;-266044;1452256;1450875;1450880;1455647;-
```

```
251667;1441303;1441305;1440308;1442709;1441304;1444719;-408665;-
180266;1442657;1442705;1457043;1440303;1442655;-395772;-217178;1440304;1440305;1455789;1442697;-
380227;1440309;1454181;-250399;-277577;-262211;1441301;1442081;1442131;-
221165;1444720;1452765;1442025;1442622;1442135;1442180;-277900;1443964

}

/*************************************************************************************************
*\

|********************Отчёт по средним векторам в кластерной структуре - итерация: 0
********************|

\*************************************************************************************************
*/

Итерация: 0    Кластер с номером: 1  для координаты: 1     имеет среднее значение: 0;

Итерация: 0    Кластер с номером: 1  для координаты: 10008 имеет среднее значение: 0;

Итерация: 0    Кластер с номером: 1  для координаты: 10050 имеет среднее значение: 0.0833333;

Итерация: 0    Кластер с номером: 1  для координаты: 10066 имеет среднее значение: 0.0833333;

Итерация: 0    Кластер с номером: 1  для координаты: 10087 имеет среднее значение: 0;

Итерация: 0    Кластер с номером: 1  для координаты: 10131 имеет среднее значение: 0.25;

Итерация: 0    Кластер с номером: 1  для координаты: 10138 имеет среднее значение: 0;

...

...

...

Итерация: 0    Кластер с номером: 3  для координаты: 9900  имеет среднее значение: 0;

Итерация: 0    Кластер с номером: 3  для координаты: 9942  имеет среднее значение: 0.0136986;

Итерация: 0    Кластер с номером: 3  для координаты: 9956  имеет среднее значение: 0.0273973;

Итерация: 0    Кластер с номером: 3  для координаты: 9966  имеет среднее значение: 0;

Итерация: 0    Кластер с номером: 3  для координаты: 9967  имеет среднее значение: 0;


/*************************************************************************************************
*\

|************************Отчёт по изменениям в кластерной структуре - итерация: 0
**********************|

\*************************************************************************************************
*/

Итерация: 0    Объект с номером: -375607    был перенесён из кластера: 2  в другой кластер под номером:
3;

Итерация: 0    Объект с номером: -352122    был перенесён из кластера: 2  в другой кластер под номером:
3;

Итерация: 0    Объект с номером: -328658    был перенесён из кластера: 3  в другой кластер под номером:
2;

Итерация: 0    Объект с номером: -172457    был перенесён из кластера: 2  в другой кластер под номером:
3;

Итерация: 0    Объект с номером: 1440301    был перенесён из кластера: 2  в другой кластер под номером:
3;

Итерация: 0    Объект с номером: 1442045    был перенесён из кластера: 1  в другой кластер под номером:
3;
```

Итерация: 0 Объект с номером: 1444097 был перенесён из кластера: 2 в другой кластер под номером: 3;

Итерация: 0 Объект с номером: 1456728 был перенесён из кластера: 2 в другой кластер под номером: 3;

==========Конец отчёта- итерация: 0==========

==========Начало отчёта - итерация: 1==========

В кластере с номером: 1 содержится: 11 объектов:

{

1442130;-181454;1441297;1442175;-232947;-333009;-382151;1442129;1442165;1442178;-208031

}

В кластере с номером: 2 содержится: 51 объектов:

{

1444024;1451881;-398;1456324;1452257;1444377;1443277;-304572;1452597;1456688;1456876;-270523;1452258;-211797;1452284;-21232;-328658;1456090;1451973;1456128;1451657;1444376;1442936;1442968;1452283;1443966;1444025;1456145;1444023;1451879;1442863;-48674;-193297;-258015;-353269;1452596;1444374;1442874;1443318;1444375;1443317;1451901;1443282;1443461;1443965;-339606;-35185;1444393;1456865;-164981;1456864

}

В кластере с номером: 3 содержится: 79 объектов:

{

1441295;-416122;1442045;1440302;1444097;1442693;-352925;1455484;1444718;1457042;1440306;-189175;-226326;1441300;-40838;-352122;-375607;-181006;-172457;1451174;1451471;-12059;1455482;-180163;1440307;1442138;-227683;1449449;1450941;-200738;1442670;-17510;-181687;1439964;-266044;1452256;1450875;1450880;1455647;-251667;1441303;1441305;1440308;1442709;1441304;1444719;-408665;-180266;1442657;1442705;1457043;1440303;1442655;-395772;-217178;1440304;1440305;1455789;1442697;-380227;1440309;1454181;-250399;-277577;-262211;1441301;1442081;1442131;-221165;1444720;1452765;1456728;1442025;1442622;1442135;1442180;-277900;1440301;1443964

}

/***\

|********************Отчёт по средним векторам в кластерной структуре - итерация: 1********************|

***/

Итерация: 1 Кластер с номером: 1 для координаты: 1 имеет среднее значение: 0;

Итерация: 1 Кластер с номером: 1 для координаты: 10008 имеет среднее значение: 0;

Итерация: 1 Кластер с номером: 1 для координаты: 10050 имеет среднее значение: 0.0909091;

Итерация: 1 Кластер с номером: 1 для координаты: 10066 имеет среднее значение: 0.0909091;

Итерация: 1 Кластер с номером: 1 для координаты: 10087 имеет среднее значение: 0;

Итерация: 1 Кластер с номером: 1 для координаты: 10131 имеет среднее значение: 0.272727;

...

```
...

...

Итерация: 1    Кластер с номером: 3  для координаты: 9900  имеет среднее значение: 0;

Итерация: 1    Кластер с номером: 3  для координаты: 9942  имеет среднее значение: 0.0126582;

Итерация: 1    Кластер с номером: 3  для координаты: 9956  имеет среднее значение: 0.0253165;

Итерация: 1    Кластер с номером: 3  для координаты: 9966  имеет среднее значение: 0;

Итерация: 1    Кластер с номером: 3  для координаты: 9967  имеет среднее значение: 0;


====================Нет      Изменений      в      кластерной      структуре      -      Кластеризация
завершена======================

Конечное число кластеров: 3;

Число задействованных объектов: 141;
```

ПРИЛОЖЕНИЕ 13. АКТ О ВНЕДРЕНИИ РЕЗУЛЬТАТОВ РАБОТЫ

TNS Россия
127018, Россия, Москва
Ул. Двинцев, д. 12, корпус 1
Бизнес-центр «Двинцев»
т.+7 (495) 935-87-18 ф. +7 (495) 626-52-28
www.tns-global.ru

«Утверждаю»
Заместитель Генерального директора
ЗАО «ТНС ГЭЛЛАП ЭДФАКТ»
Калинин А.Г. _____

"ТНС ГЭЛЛАП ЭДФАКТ" "TNS Gallup Adfact" МОСКВА

АКТ

о внедрении результатов кандидатской диссертационной работы Зейна А.Н.

Комиссия в составе:

- председателя: руководителя отдела автоматизации проектов *Media Intelligence*, Грибова В.Н.

- членов комиссии: заместителя начальника отдела разработки высоконагруженных систем, к.т.н., Онеса И.В.

аналитика, к.т.н., Савватеева С.С.

составила настоящий акт в том, что результаты диссертационной работы Зейна А.Н. «Исследование и разработка методов автоматической кластеризации Интернет-пользователей и Интернет-ресурсов для персонализации поиска» внедрены в ЗАО «ТНС ГЭЛЛАП ЭДФАКТ» а именно:

- Метод применения числовых характеристик с учётом особенностей *DOM*-модели интернет-страниц на начальной стадии группировки нестандартных рекламных баннеров «баннеры-витрина», состоящих из множество мелких элементов.
- Алгоритм фильтрации узлов *DOM*-модели с помощью специальных процедур формирования масок и словарных таблиц, что позволило сократить количество объектов для первоначальной выборки под кластерный анализ.
- Метод обобщенной кластеризации и принцип применения обобщённых характеристических векторов позволили кластеризовать динамические элементы, принадлежащие одному баннеру, но полученные в разные моменты времени.

Предложенные методы вошли в систему проверки описания рекламных кампаний. А кластеризация элементов в *DOM*-модели позволила собрать сложные баннеры для дальнейшей обработки.

ПРЕДСЕДАТЕЛЬ КОМИССИИ

Руководитель отдела автоматизации проектов *Media Intelligence*. В.Н. Грибов

ЧЛЕНЫ КОМИССИИ

Заместитель начальника отдела разработки высоконагруженных систем, к.т.н. И. В. Онес

Аналитик, к.т.н. С.С. Савватеев