САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Математико-механический факультет

Кафедра системного программирования

Дзюба Александр Александрович

Мультиагентная система поиска научных публикаций в Интернете

Выпускная квалификационная работа

Допустить к защите.
Зав. кафедрой
д. ф.-м. н., профессор …… А. Н. Терехов

Научный руководитель …… Д. Ю. Бугайченко
к. ф.-м. н.

Рецензент …… М. К. Грачев
к. ф.-м. н.

Санкт-Петербург
2010

St. Petersburg State University

Faculty of Mathematics and Mechanics

Chair of Software Engineering

Multy-agent scientific articles Internet search system

Graduate paper by

Dzyuba Aleksandr Aleksandrovich

"Admitted to proof"
(Head of the chair, Dr. of Phys. and Math. Sci.)

/Signature/_____/ Terekhov Andrey Nikolayevich

Scientific advisor
(Assistant professor, Candidate of Phys. and Math. Sci.)

/Signature/_____/ Bugaychenko Dmitry Yuryevich

Reviewer
(Candidate of Phys. and Math. Sci.)

/Signature/_____/ Grachev Mikhail Konstantinovich

St. Petersburg, 2010

Содержание

Введение

Ученые и исследователи в своей работе постоянно сталкиваются с задачей поиска научной литературы. Этот поиск ведется и на ранних этапах решения задачи, при изучении предметной области, и в ходе работы, для ознакомления с новейшими результатами исследований коллег. Причем объектом поиска может выступать не только текст публикации, но и различная дополнительная информация о ней: аннотация, год издания, место публикации.

Основные трудности такого поиска связаны с большим количеством источников данных [9] и обилием материала, имеющемся по конкретной тематике. Сейчас такие задачи наиболее эффективно решают сайты, организующие публикации в удобные для поиска структуры, специализированные поисковые системы и desktop-приложения. К первым можно отнести базы публикаций, собирающие статьи по одной или нескольким дисциплинам. Как правило, такие базы создаются в различных университетах, исследовательских институтах или библиотеках, и в них хранится от нескольких сотен тысяч до нескольких миллионов статей[1]. Поисковые системы (например Google Scholar [14]) позволяют искать публикации и информацию о них на различных Web-сайтах и в библиотеках, используя собственную поисковую машину. Desktop-приложения, помимо поиска в нескольких источниках, включающих поисковые системы и базы данных, предоставляют пользователю функции для работы с найденными статьями. К таким относится система RefNavigator[2].

В этой работе предлагается новая система для поиска публикаций в различных Интернет-источниках. Эти источники различаются по

[1] Arxiv.org (Корнелльский университет) содержит более 500 тысяч публикаций и препринтов статей по физике, математике, астрономии, информатике и биологии, а IEEE Xplore – более 2,5 миллионов статей по электротехнике, электронике, компьютерной технике и информатике.

[2] Инструмент для составления библиографии (www.refnavigator.com).

предоставляемым функциям, размерам и дисциплинам, но каждый из них обладает возможностью искать публикации. Все эти источники объединены в данной системе с мультиагентной архитектурой. Мультиагентные системы представляют собой набор взаимодействующих программных агентов [1, 5, 12]. Они успешно применяются в различных областях поиска: специального (товары, книги) и общего, интеллектуального (адаптивный поиск, социальные подходы) и сводящегося к простой агрегации результатов [3, 8]. В данной работе методики мультиагентной оптимизации поиска [4] использованы для организации поиска научных публикаций. Преимущества такой архитектуры - гибкость и масштабируемость системы, которые подразумевают, например, легкость добавления нового источника публикаций и балансирование нагрузки между различными аппаратными узлами системы.

Описываемая система использует поисковые сервисы своих источников публикаций по запросу пользователя и агрегирует полученные результаты. Порядок выдачи публикации при этом определяется как внутренним ранжированием каждого из источников, так и индивидуальными предпочтениями пользователя. Каждая поисковая система использует различные алгоритмы для определения значимости статьи в зависимости от места публикации, автора и индекса цитирования, такие как Индекс Хирша [7] и импакт-фактор [6]. Данная система, анализируя в ходе работы поведение пользователя, уточняет это ранжирование, добиваясь таким способом более релевантных результатов.

1 Постановка задачи

Задача данной работы состоит в создании прототипа системы, предназначенной для поиска научных публикаций в Интернете. Ниже перечислены этапы выполнения работы.

1. Предложить методы мультиагентного поиска информации, применимые к задаче поиска научных публикаций.
2. Разработать архитектуру мультиагентной системы для поиска научных публикаций в Интернете со следующими свойствами: использование методов мультиагентной оптимизации поиска, ориентация на гетерогенные источники данных.
3. Реализовать прототип описанной системы с двумя источниками публикаций: Arxiv.org [13] и Google Scholar [14].

2 Обзор литературы

2.1 Обзор методов поиска публикаций

Понятие *публикация*, используемое в данной работе, несколько шире, чем общеупотребительное. Обычно это понятие означает научную работу, опубликованную в книгах или периодических изданиях. В то же время Интернет-ресурсы, работающие с научными изданиями, редко ограничиваются только опубликованными работами. Некоторые из них работают с препринтами[3] (Arxiv.org, Nature Precedings[4]), некоторые – с научной литературой в целом, вне зависимости от того, была она опубликована или не предполагала публикации изначально. Поскольку в этой работе речь идет о поиске в гетерогенных источниках, в том числе в плане трактовки понятия публикация, далее оно будет означать объединение видов научных работ, предоставляемых теми Интернет-ресурсами, в которых ведется поиск.

Эти ресурсы можно разделить на два типа – библиотеки публикаций и поисковые системы. Их количество и разнообразие [9] дает большой простор для поиска, но в то же время создает проблему выбора оптимального источника, особенно если область поиска достаточно широка.

2.1.1 Сайты-библиотеки

Объединение статей в электронные библиотеки[5] является наиболее эффективным способом упорядочить большое количество публикаций в Интернете. Как правило, в одну библиотеку объединяются публикации в одной или нескольких сильно связанных областях, таких как информатика и технические науки, хотя некоторые библиотеки являются

[3] Препринт – научное издание, выпоскаемое в свет до публикации. Препринт не проходит рецензирование, а потому не имеет такой научной значимости, как публикация в научном журнале.

[4] Nature Precedings (precedings.nature.com) – архив препринтов научных работ в области химии, медицины, наук о Земле .

[5] Электронная библиотека – упорядоченная коллекция электронных документов и средства поиска в этой коллекции.

мультидисциплинарными. К примеру ScienceDirect, одна из крупнейших онлайн-коллекций публикаций, включает в себя статьи по множеству технических, естественных, общественных и гуманитарных наук.

Для всех электронных библиотек характерно большое количество собранных публикаций, однако оно не может служить показателем содержательности. Как видно на табл. 1, из 20 миллионов публикаций в хранилище PubMed, лишь около 2 миллионов имеют ссылки на полный текст статьи, в то время как все 600 тысяч препринтов Arxiv.org, помимо полного текста статей в формате PDF, имеют открытый исходный код, как правило на языке TeX, а также рисунки и таблицы отдельно от текста.

Таблица 1. Сравнение содержимого различных библиотек публикаций

Библиотека	Объем содержимого на 25.04.2010	Тип содержимого	Доступ
PubMed[6]	19,772,033	Цитаты, аннтоции и полный текст 10% статей	Бесплатный
Science Direct	10,140,303	Тексты статей	Платный доступ к текстам статей
Arxiv.org	599,923	Тексты статей, исходный код статей, рисунки и таблицы	Бесплатный

Большинство таких сайтов-хранилищ создаются при поддержке различных университетов и их библиотек. Некоторые издательства также создают электронные коллекции своих публикаций. Примером может служить ScienceDirect, созданный Elsevier, одним из крупнейших издательских домов мира.

[6] PubMed – база данных публикаций в области медицины, основанная на Национальной Библиотеке Медицины США.

2.1.1.1 Преимущества электронных библиотек

Одним из преимуществ поиска статей в электронных библиотеках является удобное деление статей по предметным областям. Это помогает отфильтровать множество нерелевантных статей. Администраторы библиотек имеют возможность разделить ту или иную область на несколько, если в ней накопится слишком много статей, таким способом поддерживая равномерное распределение публикаций по областям.

Ученые и исследователи, работающие специфических отраслях той или иной науки, имеют возможность искать лишь публикации, относящиеся к этой отрасли, в то время как при использовании поисковых систем можно указать лишь общее направление поиска.

Поисковые системы электронных хранилищ адаптированы под данные, собранные в них, и в этом заключается их главное преимущество над другими средствами поиска. Кроме возможности указать специфическую предметную область публикации, эта адаптация касается внутреннего представления статьи в библиотеке. Например, статья в библиотеке PubMed имеет 36 уникальных атрибутов, по значениям которых пользователь может произвести поиск. Другой пример такой адаптации – возможность задать при поиске ограничения, характерные медицинских статей, такие как фильтр статей по полу, возрасту предполагаемого пациента, а также является ли пациент человеком или животным. Любое другое средство поиска, агрегирующее статьи из нескольких источников, не может обеспечить столь тонкой настройки фильтра статей.

2.1.1.2 Недостатки электронных библиотек

Несмотря на удобство поисковых средств у сайтов-библиотек, их область поиска всегда меньше чем у средств, ведущих поиск по нескольким сайтам. Если пользователь заранее не знает, есть ли в библиотеке нужные ему статьи, то эффективнее воспользоваться поисковым средством, выполняющем агрегацию статей из разных электронных хранилищ.

Похожая проблема связана с узкой специализацией электронных библиотек. На большинстве сайтов собираются статьи по одной дисциплине или нескольким тесно связанным дисциплинам. Это означает, что пользователю, ведущему поиск в разнообразных предметных областях, придется использовать несколько сайтов, у каждого из которых будет своя система поиска и настройки.

2.1.1.2 Сайт Arxiv.org

В данной работе реализована система, которая использует в качестве одного из источников данных электронную библиотеку Arxiv.org [13]. Она представляет собой архив препринтов публикаций по физике, математике, статистике, информатике, и содержит более около 600 тысяч препринтов (табл. 1).

Содержимое архива представляет собой исходный код публикаций в формате TeX с рисунками и таблицами, а также тексты публикаций в форматах PDF и PostScript[7], которые генерируются автоматически.

Средство поиска по Arxiv.org состоит из двух частей: ручной поиск по архиву, доступный через браузер и интерфейс прикладного программирования (API). Обе части обладают одинаковыми возможностями, но первый ориентирован на людей и выдает ответы в виде HTML-страницы. API позволяет получить результаты в формате XML, принимая запросы через методы HTTP GET и POST[8]. Параметр этого метода, представляющий собой URL (единый указатель ресурсов), составляется из множества значений атрибутов статей, таких как:

- ключевые слова заголовка или аннотации;
- имена авторов;
- журнал публикации;

[7] PostScript – язык программирования для описания страниц.

[8] Методы HTTP GET и POST предназначены для запроса содержимого ресурса и передачи пользовательских данных ресурсу соответственно.

- предметная область;
- комментарий авторов к статье.

После обработки запроса, API возвращает результат в формате Atom 1.0. Это формат, основанный на XML и предназначенный для описания ресурсов на Web-сайтах. Он позволяет передать все атрибуты публикаций.

Ниже приведен пример описания публикации в формате atom, показывающий основные атрибуты представления статьи в библиотеке Arxiv.org:

<entry>
  <id>*Ссылка на статью*</**id**>
  <updated>*Дата обновления*</**updated**>
  <published>*Дата публикации*</**published**>
  <title>*Название публикации*</**title**>
  <summary>*Аннотация*</**summary**>
  <author>
   <name>*Имя автора*</**name**>
   <arxiv:affiliation>*Организация автора*</**arxiv:affiliation**>
  </**author**>
  <arxiv:doi> *Идентификатор цифрового объекта (DOI)*[9]</**arxiv:doi**>
  <arxiv:comment>*Комментарий автора*</**arxiv:comment**>
  <arxiv:journal_ref>*Информация о журнале*</**arxiv:journal_ref**>
  <link title="pdf" href="*Ссылка на PDF-документ с публикацией*"/>
  <arxiv:primary_category term="*Основная категория*"/>
  <category term="*Неосновная категория*"/>
</entry>

Комментарий автора к статье, как правило, содержит информацию о количестве страниц и рисунков в статье. Авторов, как и неосновных категорий может быть несколько.

Представление результатов в таком формате делает их разбор значительно проще, чем разбор HTML-страницы.

[9] DOI (идентификатор цифрового объекта) - стандарт обозначения представленной в сети информации об объекте. Состоит из идентификатора издателя и идентификатора самого объекта. Представляет собой уникальную строку из букв и цифр.

2.1.2 Поисковые системы

Обилие и разнообразие электронных библиотек [9], хранящих публикации, делает обоснованным применение поисковых систем. Для поисковых систем характерно наличие Web-интерфейса поиска и поисковой машины. Поисковая машина индексирует публикации, хранящиеся на Web-сайтах. Этот процесс состоит в извлечении из статей информации, важной для поиска. Получив запрос через Web-интерфейс, поисковая машина ищет в базе проиндексированных публикаций подходящие и ранжирует их в соответствии с их релевантностью запросу.

Индексирование большого количества публикаций позволяет оценить важность каждой из них. Для этого по отдельности оцениваются различные атрибуты статьи, такие как автор, издание, частота цитирования в уже проиндексированных публикациях.

2.1.2.1 Цитирование научных публикаций

В тех базах данных статей, которые индексируют ссылки на другие публикации, есть возможность рассчитать значимость публикаций, учитывая статьи, которые ссылаются на оцениваемую публикацию. Методы оценки уникален для каждой поисковой машины, но существуют наукометрические показатели, вычисление которых является примером использования таких методов.

Для оценки продуктивности авторов публикации может использоваться индекс Хирша [7]. Он вычисляется на основе анализа количества работ ученого и количестве цитирования его публикаций. Ученый имеет индекс h, если h из H его работ цитируются по крайней мере h раз каждая, а остальные (H - h) работ цитируются не более h раз. На рис. 1 показана кривая количества цитат для публикаций, пронумерованных согласно убыванию количества цитат.

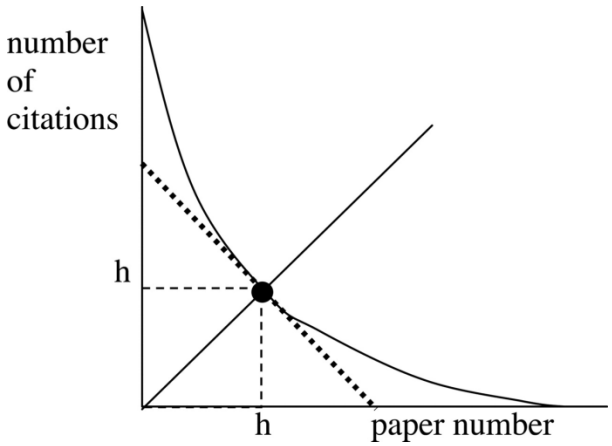

Рис. 1: Кривая количества цитат для публикаций, пронумерованных в порядке убывания количества цитат.

Как правило, такая кривая имеет форму гиперболы. На пересечении этой кривой с биссектрисой угла ∠(h,0)(0,0)(0,h) находится индекс Хирша. Этот показатель оценивает значимость ученого адекватнее, чем общее число публикаций или общее число цитирований, но существуют ситуации, для которых важность оценивается неверно.

В качестве показателя для оценки важности научного журнала можно использовать импакт-фактор [6]. Он рассчитывается на основе анализа трехлетнего периода как отношение:

$$I = A/B \qquad (1)$$

где A – число цитировании в течение последнего года в других журналах статей, опубликованных за первые два года этого периода,

а B – число статей, опубликованных за первые два года этого периода.

Импакт-фактор ежегодно рассчитывается Институтом научной информации[10]. Он рассматривает более 11 тысяч научных журналов. Несмотря на большой охват этого показателя, очевидны его недостатки. Например анализ трехлетнего промежутка не может дать полную картину цитирования, ведь фундаментальные труды цитируются на протяжении десятилетий.

[10] Институтом научной информации – организация, специализирующаяся на анализе научных публикаций и индексировании цитирования публикаций.

И индекс Хирша и импакт-фактор эффективны лишь при анализе публикаций в рамках одной предметной области. Это связано с различной интенсивностью написания статей в разных дисциплинах, например медицинские журналы имеют большие значения этих показателей, чем математические.

2.1.2.1 Преимущества поисковых систем

Главным достоинством поисковых систем является возможность искать публикации во многих источниках через один Web-интерфейс. Например, поисковая система Scirus[11] производит поиск на 380 миллионах Web-страниц и в 34 электронных библиотеках, включая поиск препринтов публикаций на Arxiv.org, поиск цитат в Pubmed и текстов публикаций в Science Direct. На примере системы Scirus видно, что разнородными могут быть не только источники, но и типы публикаций.

Кроме того, охват большой зоны поиска дает поисковым системам возможность расчета значимости публикаций, авторов, журналов на основе анализа внутренних цитирований. Эти показатели будут более достоверными, чем их аналоги для электронных библиотек, так как количество индексированных публикаций у поисковых систем на порядок выше.

2.1.2.2 Недостатки поисковых систем

Так как поиск в данном случае ведется по множеству гетерогенных источников данных, поисковый запрос невозможно составить так же точно, как это делается в поисковых средствах электронных библиотек. Если собственные поисковые сервисы каждой из электронных библиотек адаптированы под её особенности, то поисковые системы составляют запросы с помощью атрибутов публикаций, характерных для всех источников. Как правило, это ключевые слова, дата публикации, журнал и

[11] Scirus - поисковая система, принадлежащая издательскому дому Elsevier, ориентированная на научную литературу.

предметная область. Предметые области, в данном случае, не такие точные, как в сайтах-библиотеках, и они описывают лишь общую направленность работы.

Зачастую охват сайтов поисковой системы не позволяет ограничиться лишь её использованием. Некоторые издатели и электронные библиотеки не позволяют индексировать их статьи. Кроме того, не все поисковые системы публикуют список библиотек, по которым ведется поиск (например Google Scholar), что создает трудности при использовании.

2.1.2.3 Поисковая система Google Scholar

В данной работе реализована система, использующая в качестве источника данных Google Scholar [14]. Google Scholar – это поисковая система от Google, ориентированная на поиск научной литературы, опубликованной издательствами, профессиональными ассоциациями, высшими учебными заведениями и другими научными организациями. Объектом поиска являются статьи, рефераты и библиографические ссылки. Кроме информации о статье, ищется также ссылка на её полный текст.

Для определения значимости статьи, данная поисковая система оценивает авторов статьи, место публикации, частоту цитирования в научной литературе . Кроме самой статьи, также оцениваются различные её модификации, например предварительные версии. Однако в таких случаях результатом поиска будет ссылка на опубликованную официальным издателем работу (если её текст доступен для Google Scholar).

Ранжируя результаты поиска, данная система учитывает значимость статьи, а также её соответствие запросу. Однако результаты поиска, которые пользователь счет полезными, никак не учитываются в дальнейшем.

После обработки поискового запроса, Google Scholar выдает сведения о найденных статьях в виде HTML-страницы. Они включают:

- название статьи;
- ссылку на источник;

- авторов статьи;
- официального издателя;
- год публикации.

Для ознакомления с каждой статьей приведена выдержка из аннотации или текста статьи. Также пользователь может ознакомиться с статьями, цитирующими данную публикацию.

Для облегчения составления библиографии Google Scholar может генерировать цитату публикации в форматах BibTeX, EndNote, RefMan, RefWorks и WenXianWang (инструменты составления библиографии). Для каждого из указанных форматов существуют приложения, генерирующие библиографию по данным цитатам. Кроме того, данные цитаты содержат информацию о статье, не отображаемую на странице результатов, такую как номер и том журнала, в котором опубликована статья.

Формируя поисковой запрос, пользователь может задать значения таких атрибутов статьи как ключевые слова, место и дата публикации, предметная область.

Несмотря на то, что охват поисковой системы Google Scholar неизвестен, он очень большой, что, вместе с удобным интерфейсом поиска и эффективной системой ранжирования, делает Google Scholar одним из лучших средств для поиска научных публикаций.

2.1.3 Desktop-приложения для поиска

Электронные библиотеки и поисковые системы предполагают взаимодействие с пользователем через Web-интерфейс. Другой класс поисковых средств использует для этих целей desktop-приложения. Эти приложения пользуются поисковыми системами и электронными библиотеками для поиска статей, а также обладают возможностями локальной работы со статьями.

Локальная работа со статьями подразумевает создание собственной библиотеки публикаций, а также средства для работы с ней – локальный

поиск, чтение статей, составление библиографии. При этом такие приложения обладают тем же достоинством что и поисковые системы – один интерфейс для всех поддерживаемых источников данных.

Будучи размещенными на компьютере пользователя, desktop-приложения имеют широкие возможности по сбору персональной информации о пользователе. Все действия, связанные с поиском, локальной работой со статьями, позволяют оценить предпочтения пользователя и использовать их при дальнейшей работе.

Любой недостаток desktop-приложения по сравнению с Web-приложением применим к данному типу поисковых средств. В частности, это необходимость установки приложения на клиентском компьютере, а также установки обновлений.

2.1.3.1 Приложение RefNavigator

RefNavigator – это приложение для поиска научных публикаций и составления библиографии [16]. Оно ориентировано на биомедицинских исследователей, ведущих поиск публикаций в библиотеке PubMed. Поиск в этой библиотеке с помощью данного приложения также эффективен и удобен, как и поиск через Web-интерфейс.

Несмотря на медицинскую направленность, RefNavigator полезен и исследователям из других областей. В нем поддерживается поиск по крупнейшим библиотекам и поисковым системам (Google Scholar, ScienceDirect, IEEE, ACM, Scirus, Arxiv), что позволяет рассматривать данное приложение как универсальное поисковое средство.

Как и большинство desktop-приложений, RefNavigator позволяет пользователю организовать собственную библиотеку публикаций (цитат, текстов). Собранные статьи хранятся в системе с папками. Также пользователь может генерировать библиографию из найденных результатов.

К недостаткам данного приложения можно отнести отсутствие социальных функций, таких как обмен рекомендациями между

пользователями приложения. Кроме того, один сеанс поиска возможен лишь в одной системе или библиотеке, что ограничивает возможности пользователя и не позволяет сравнить между собой результаты поиска для разных источников. Личную библиотеку статей гораздо удобнее хранить в иерархической структуре по категориям, чем в папках.

2.1.4 Заключение

Существует несколько способов поиска научных публикаций. Это либо поиск непосредственно в электронных библиотеках, собирающих литературу, либо использование поисковых систем, которые ищут статьи в библиотеках и на Web-сайтах, либо работа с desktop-приложениями, использующими первые два способа. Каждый из вариантов обладает собственными достоинствами и недостатками.

Электронные библиотеки не такие масштабные, как поисковые системы, но их средства поиска оптимизированы под их содержимое. Поисковые системы не настроены на конкретные сайты-библиотеки, но они дают возможность искать статьи в разных библиотеках через один Web-интерфейс. Оба типа имеют возможность рассчитывать различные числовые показатели для своего содержимого и с их помощью оценивать качество статей. Используя поисковые системы и библиотеки, пользователь может быть уверен, что самые значимые статьи он увидит первыми.

Desktop-приложения предоставляют пользователю не только поиск, но и множество функций по работе с публикациями. Кроме того, они меньше других ограничены в функциях, которые можно добавить в приложение. Однако установка этих приложений требует некоторых усилий.

В данной работе предлагается система, которая лишена некоторых недостатков существующих систем. Благодаря мультиагентной архитектуре, описанной в 2.2, эта система рассчитана на поиск в нескольких гетерогенных источниках данных через один интерфейс, как и большинство desktop-приложений, а также удобна для добавления новых источников. А

благодаря оптимизациям поиска, характерным для мультиагентных систем (2.2.1), она учитывает предпочтения пользователя в поиске статей, делая процесс поиска более удобным и быстрым.

2.2 Обзор мультиагентных систем

В данной работе речь идет о поиске научных публикацией в Интернете. Интернет является типичным представителем *открытых систем* – системм, структура которых может динамически изменяться. Компоненты открытых систем меняются в процессе функционирования и при этом они гетерогенны (созданы для разных целей, обладают разными функциями и различны в обращении с ними). Интернет постоянно растет, делаясь все сложнее, и любая компьютерная система, работающая в нем, должна быть способна взаимодействовать с другими системами и программами без постоянного контроля пользователя. Для этого ей требуется владеть навыками кооперации и в то же время преследовать собственную выгоду. Такое взаимодействие очень похоже на взаимодействие агентов в *мультиагентных* системах [10].

В условиях, когда необходимо автоматически выполнять различные действия в постоянно меняющейся, растущей как по объему, так и по сложности системе, возрастает вероятность ошибок, не предвиденных на этапе разработки системы. Тестирование системы не может обеспечить отсутствие ошибок во всех ситуациях, если она будет работать в действительно сложной среде. Применение мультиагентных систем для таких задач является одним из подходов, направленных на решение этой проблемы [1].

Мультиагентной называется система, состоящая из набора автономных сущностей, называемых *агентами*, которые взаимодействуют между собой (обычно посредством обмена сообщениями) [1, 12]. Применимость мультиагентных систем в сложных, открытых системах объясняется свойствами, которыми обладает агент.

- Агент всегда помещен в некоторую *внешнюю среду*, в которой он может совершать автономные действия, таким образом взаимодействуя с ней. Однако агент не может контролировать эту

среду, и поэтому всегда должен быть готов к тому, что его действия не достигнут желаемого результата. В лучшем случае, агент может оказывать влияние на окружающую среду (два одинаковых действия, совершенных одна за другим могут привести к разному результату).

- *Реактивность*: агент способен воспринимать внешнюю среду и реагировать на её изменения.
- *Проактивность*: агент, взаимодействуя с внешней средой, способен выполнять автономные действия, направленные на достижения определенных целей, проявлять инициативу.
- *Социальность*: агент способен взаимодействовать с другими агентами и людьми для достижения целей, как собственных, так и общих.

Агент, проявляющий последние три свойства, называется *интеллектуальным* [11, 12].

2.2.1 Мультиагентные оптимизации поиска

Одним из примеров успешного применения мультиагентных систем в Интернете является мультиагентный поиск. Системы, ведущие поиск информации, могут быть специальными – направленными на информацию определенного рода (товары, книги), и не специальными. Также они подразделяются на не интеллектуальные и интеллектуальные, для которых характерны адаптивный[12] поиск и различные социальные подходы оптимизации поиска. Наибольший интерес представляют интеллектуальные системы поиска.

К преимуществам таких систем относятся:

[12] Адаптивный – подстраивающийся под входные данные.

- уменьшение количества действий пользователя, необходимых для осуществления поиска, за счет автоматизации какого-то количества действий;
- обучаемость, т.е запоминание предпочтений пользователя в ходе работы;
- совместное обучение пользовательских агентов, реализованное посредством анализа предпочтений пользователя и обменом этой информацией.

В то же время такие системы гораздо сложнее, чем их аналоги, за счет большого числа дополнительных данных, которые приходится собирать, хранить, анализировать и применять для оптимизации.

2.2.1.1 Учет совместного опыта поиска

Мультиагентная поисковая система Implicit, использующая совместный опыт поиска, была предложена в работе [4]. Главная идея этой системы в скрытом от пользователя сборе информации о его поведении в течении сеанса поиска информации. Причем эти знания не выражаются в явной форме, а используются для улучшения результатов поиска в дальнейшем, даже когда поиск проводит новый пользователь.

У каждого пользователя в такой системе имеется персональный агент, взаимодействующий с интерфейсом, через который пользователь вводит запросы. Для учета совместного опыта поиска в персональных агентах есть специальная компонента, состоящая из трех частей:

- *наблюдателя*, который хранит в базе данных информацию о действиях, выполненных пользователем в течении поиска;
- *классификатора*, который применяет методы интеллектуального анализа данных[13], собранных наблюдателем, для определения

[13] Интеллектуальный анализ данных (Data Mining) – это процесс обнаружения в данных ранее неизвестных, практически полезных и доступных интерпретации знаний и закономерностей.

шаблона поведения пользователя, а также других агентов с похожим поведением;

- *анализатора*, который использует информацию, собранную наблюдателем и обработанную классификатором, для анализа результатов поиска или передачи её другим агентам.

Анализатор состоит из двух частей. Задача первой – определить действия пользователя, попадающие под один шаблон поведения, найденный классификатором. Вторая часть выделяет информацию, найденную в процессе поиска, имеющую отношение к действиям, определенным первой частью.

Кроме компоненты, отвечающей за обмен совместным опытом, персональный агент пользователя содержит компоненту, взаимодействующую с Google API для поиска информации.

Таким образом, поиск в данной системе состоит из трех этапов:

- запрос необходимой информации в Google;
- анализ результатов с учетом профиля пользователя;
- связь с другими пользователями с похожим поведением и учет их опыта поиска.

В результате получаем список результатов поиска Google, ранжированный в соответствии с предпочтениями пользователя, а также в соответствии со знаниями, полученными в результате анализа поведения других пользователей системы. В такой системе даже пользователь, который не имеет истории поиска, может воспользоваться этими знаниями, что делает поиск более точным даже в начале работы с системой.

2.2.2. Инструментарий разработки мультиагентных систем

От инструмента для разработки мультиагентных систем требуется предоставлять возможность описания агентов и их взаимодействия. Помимо этого, такой инструмент должен предоставлять возможность запускать агентов на разных компьютерах в сети.

Существуют определенные стандарты, которым должны удовлетворять мультиагентные системы и технологии, связанные с ними. Это спецификации организации FIPA[14], являющаяся членом ассоциации IEEE.

2.2.2.1 Спецификации FIPA

Эти спецификации определяют правила, по которым должны существовать, действовать и управляться сообщества агентов [15]. В частности, они описывают модель агентной платформы (AP), в которой работают агенты. В этой модели выделены три ключевых роли, необходимые для управления платформой.

- Система управления агентами (Agent Management System, AMS) – агент, осуществляющий контроль над справочником уникальных идентификаторов агентов (AID), которым сопоставлены транспортные адреса агентов. Каждый агент платформы регистрируется в этом справочнике, чтобы получить AID.
- Справочник (Directory Facilitator, DF) – агент, предоставляющий функции «желтых страниц» другим агентам системы. Агенты регистрируют свои сервисы в этом справочнике. По названию сервиса DF выдает адрес агентов, зарегистрировавших его.
- Сервис передачи сообщений (Message Transport Service, MTS) - передает сообщения между агентами одной платформы, а также агентами из разных платформ (рис. 2).

[14] FIPA (Foundation for Intelligent Physical Agents) – некоммерческая организация, выпускающая стандарты для технологий, связанных с интеллектуальными агентами.

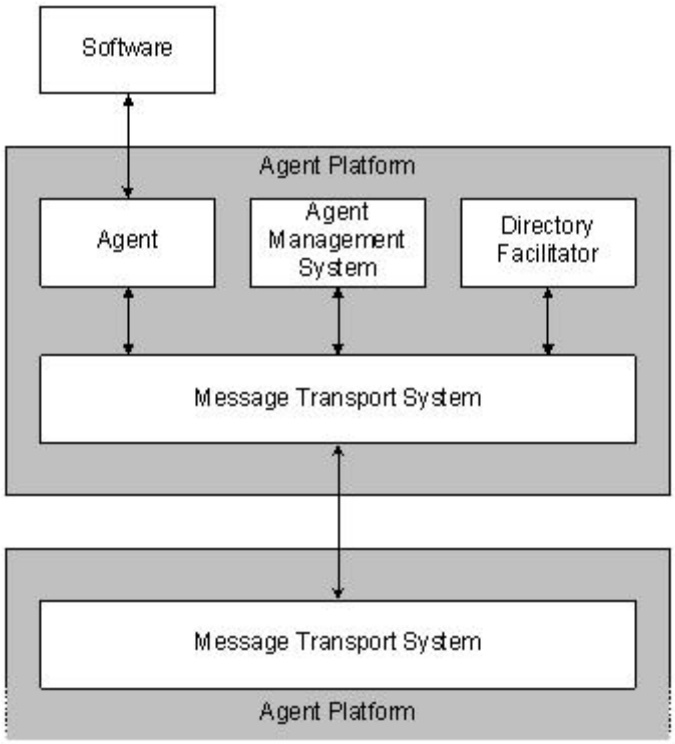

Рис. 2: Структура агентной платформы и её взаимодействие с другими платформами.

Также эти спецификации определяют язык FIPA Agent Communication Language (ACL), предназначенный для общения агентов. Они определяют лишь синтаксис и семантику языка, но не способы передачи сообщений. ACL-сообщение состоит из адресов отправителя и получателей, перформатива[15], описывающего тип акта коммуникации, содержания сообщения и информации о нем (язык, кодировка), а также нескольких полей, управляющих диалогом агентов.

Кроме того, спецификации FIPA касаются протоколов взаимодействия агентов, самым известным из которых является протокол FIPA Contract Net. Он применяется для выбора агента-поставщика необходимого сервиса среди нескольких кандидатов.

[15] Перформатив - глагол, эквивалентный выполнению действия.

2.2.2.2 Каркас JADE

JADE представляет собой программный каркас (framework), предназначенный для разработки мультиагентных систем в соответствии со спецификациями FIPA [2, 17]. JADE универсален, он позволяет описать любого агента.

Основными компонентами каркаса JADE являются:

- среда выполнения агентов;
- библиотека Java-классов для разработки мультиагентных систем;
- набор утилит для администрирования мальтиагентных систем.

Среда выполнения агентов в JADE, соответствует первым двум частям спецификаций FIPA. На каждом из компьютеров системы находится контейнер агентов, которые исполняются как Java-приложения (а агенты в них – как потоки). Вместе контейнеры составляют агентную платформу, описанную в спецификациях. В системе должен присутствовать главный контейнер, содержащий ключевых агентов платформы – DF и AMS. Сообщения между агентами передаются с помощью метода MTS, который в JADE называется Agent Communication Channel (ACC). Все взаимодействия между агентами осуществляется посредством пересылки сообщений языка FIPA ACL. Такое устройство среды выполнения агентов полностью соответствует стандартам FIPA.

Библиотека Java-классов позволяет реализовать агентов со сколь угодно сложным поведением, а также описать их взаимодействие. В основном, это классы, описывающие агентов на каждом этапе их жизненного цикла, а также их взаимодействие с графическим интерфейсом (если он есть). Действия, которые совершает агент во время выполнения, описываются с помощью *поведений*. У каждого агента есть очередь поведений, которые последовательно исполняются.

В JADE реализованы разные типы поведений:

- простые поведения:

1. OneShotBehaviour – исполняющееся ровно 1 раз;
2. CyclicBehaviour – исполняющееся постоянно;

- композитные поведения, состоящие из нескольких поведений:
  1. SequentialBehaviour – состоит из нескольких поведений, выполняющихся последовательно;
  2. FSMBehaviour – конечный автомат поведений;
  3. TickerBehaviour – цикличное поведение, исполняющееся через определенный промежуток времени;
  4. WakerBehaviour – исполняется 1 раз по истечении определенного промежутка времени.

Помимо агентов и поведений, JADE реализует FIPA-протоколы взаимодействия между агентами, такие как FIPA Contract Net Interaction Protocol и взаимодействие через топик.

Также данный каркас предоставляет набор инструментов для администрирования мультиагентных систем. В него входят такие инструменты, как Remote Management Agent (RMA), агент с графическим интерфейсом, позволяющий управлять платформой (удаление, создание агентов). Кроме него, очень полезен инструмент Sniffer, позволяющий отслеживать отправку ACL-сообщений в виде диаграммы последовательностей, а также читать отправленные сообщения.

2.2.2.3 Заключение

Подводя итог описанию каркаса JADE, можно сделать вывод, что он является средством для создания мультиагентных систем любой сложности [2]. Обширная библиотека Java-классов JADE позволяет удобно описывать различных агентов и их взаимодействие между собой. С помощью графических инструментов, предоставляемых данным каркасом, можно следить за работой созданной системы и отлаживать взаимодействие между агентами.

Помимо этого, среда выполенения агентов, удовлетворяющая спецификациям FIPA, обеспечит корректную работу полученной системы, даже если она будет запущена на нескольких компьютерах. Это особенно актуально для создания систем, работающих в Интернете, ведь для них часто необходимо взаимодействие клиентской и серверной частей.

Все эти качества делают JADE пригодным для промышленной разработки приложений, и система, разработанная в данной работе реализована с помощью этого каркаса.

3 Архитектура системы

В данной работе разработана архитектура системы, производящей поиск научных публикаций в Интернете. Система обладает мультиагентной архитектурой – она состоит из набора взаимодействующих агентов. Данная глава описывает агентов, из которых состоит система, а также их взаимодействие друг с другом.

3.1 Классы агентов

Для поиска публикаций в системе создано 4 типа агентов:

- агент пользователя;
- агент, работающий с предпочтениями пользователя;
- классификатор;
- агент-поисковик.

Кроме них в системе присутствуют агенты, соответствующие спецификациям FIPA – AMS и DF, которые необходимы для управления системой и координации агентов [15].

3.1.1 Агент пользователя

Данный агент отвечает за взаимодействие системы с пользователем при поиске. Это взаимодействие осуществляется через графический интерфейс, доступный данному агенту.

Агент пользователя имеет право отображать результаты поиска на графическом интерфейсе, а также получать из него данные, необходимые для формирования поискового запроса. Кроме того, этот агент имеет доступ к *ресурсу*, содержащему данные о предпочтениях пользователя. Однако ему не требуется работать с ними, эти данные нужно передать агенту, анализирующему предпочтения пользователя.

3.1.2 Агент, анализирующий предпочтения пользователя

Этот тип агентов отвечает за работу с *предпочтениями* пользователя. Под предпочтениями понимаются сведения, собранные данным агентом или

другими агентами этого же типа в ходе предыдущих сеансов поиска. Эти данные могут касаться сайтов, публикации которых нравятся пользователю, его любимых авторов, журналов, а также предметную область, в которой он ведет исследования. Агент пользователя передает агенту предпочтений эти данные или адрес ресурса, в котором они содержатся перед началом сеанса поиска.

У данного агента две основные задачи. Первая состоит в сборе предпочтений на основании запросов пользователя, а также тех публикаций, что были выбраны автором среди найденных. Вторая – в ранжировании результатов поиска согласно собранным предпочтениям. Общая идея состоит в следующем: если некоторые из найденных публикаций имеют автора или журнал, понравившийся пользователю ранее, то они должны стоять выше в списке результатов. При этом ранг публикации в исходном списке тоже крайне важен, ведь он составлен благодаря сложным алгоритмам ранжирования внутри источников данных (общие идеи этого описаны в 2.1.2.1). Перед определением ранга публикаций нужно выполнить агрегацию результатов из разных источников.

После завершения сеанса поиска данный агент должен сохранить собранные предпочтения (в соответствующем ресурсе, если он работает с ним, либо передать агенту пользователя).

3.1.3 Агент-классификатор

Главной задачей классификатора является определение предметной области запроса или статьи. Предметную область запроса, если она не задана пользователем, полезно определить для отсечения заведомо не интересующих пользователя публикаций. Для статьи подобная операция полезна, если из предпочтений пользователя известна предметная область, в которой он обычно работает. Тогда данные о дисциплине публикации будут полезны при ранжировании.

Классифицировать запросы и статьи можно с помощью словарей терминов, характерных той или иной дисциплине, поэтому классификатор должен иметь доступ к ресурсу с такими словарями, либо сервису, осуществляющему классификацию.

3.1.4 Агент-поисковик

Агент-поисковик взаимодействует с источником данных. В системе есть поисковики, умеющие взаимодействовать с каждым из поддерживаемых источников данных. Это взаимодействие состоит из двух этапов.

- Первый этап – перевод данных, которые задал пользователь, на язык запросов поискового средства источника данных. После чего осуществляется отправка запроса.
- Второй этап – чтение результатов поиска. Как правило, это список публикаций в формате HTML, если поисковое средство рассчитано на работу с людьми, или XML, если источник данных имеет интерфейс прикладного программирования (API).

3.2 Взаимодействие агентов и ресурсов

Основываясь на описании агентов можно выделить следующие типы ресурсов, с которым взаимодействуют агенты:

- поисковое средство источника данных;
- словари для классификации;
- ресурс предпочтений пользователя.

Схема взаимодействия агентов и ресурсов показана на рис. 3.

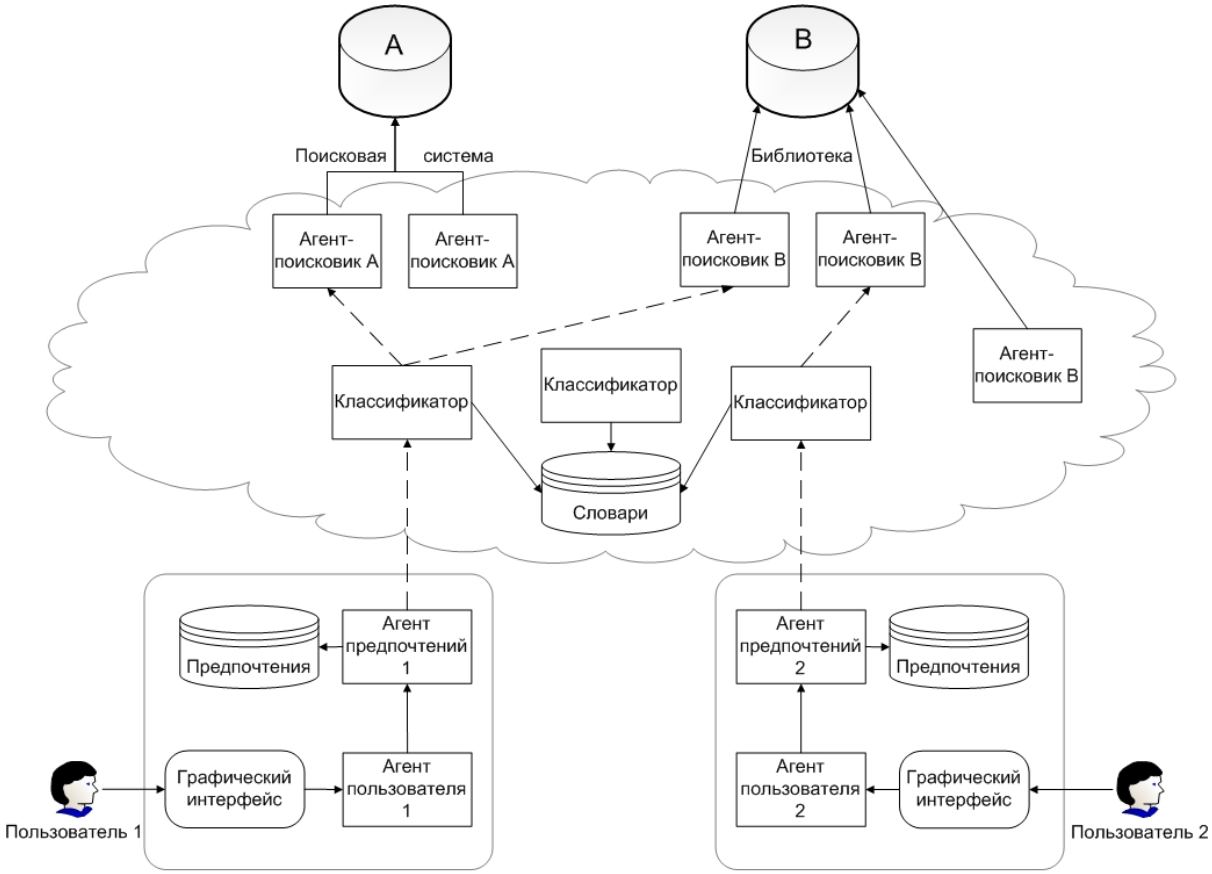

Рис. 3:Взаимодействие агентов и ресурсов в системе.

В мультиагентной системе, реализованной согласно спецификациям FIPA, физическое размещение агентов никак не влияет на работу системы – всю передачу сообщений обеспечивает система MTS. Однако в данной системе можно ввести условное деление на клиентскую и серверную часть.

На рисунке видно, что сгруппированы агент пользователя и агент его предпочтений. Оба этих агента работают с персональными данными пользователя достаточно продолжительное время, поэтому удобнее всего размещать их на клиентском компьютере. Два других агента могут обрабатывать поисковые запросы от разных пользователей по очереди, не пользуясь при этом никакими персональными данными. По этой причине их удобно запускать на сервере.

Сплошными стрелками на рисунке показаны взаимодействия, действительные между агентами в течение многих поисковых запросов. К ним относятся, например взаимодействия с ресурсами, а также

взаимодействие агента пользователя с агентом предпочтений. Эти два агента могут работать вместе долгое время, пока используется графический интерфейс поиска.

В то же время сотрудничество агента предпочтений и классификатора длится лишь в течение 1 поиска, после чего классификатор может переходить к следующему запросу от другого пользователя. Такие взаимодействия отображены на рисунке с помощью пунктирных стрелок. Однако при непостоянной связи необходимо иметь протокол поиска партнера для взаимодействия. В данной ситуации лучше всего использовать протокол FIPA Contract Net.

Оба типа отношений между агентами подразумевают передачу данных о запросе от одного агента другого к другому, а затем обратную передачу результатов поиска.

Эти данные идут по цепочке агент пользователя → агент предпочтений → классификатор → поисковик, а также обратной к ней.

Если пользователь ищет публикации в нескольких источниках, то от классификатора требуется найти поисковики, соответствующие каждому из этих источников, после чего отправить запрос каждому из них.

3.3 Обмен опытом поиска

В данной работе не предполагается оснащать систему социальными функциями, такими как возможность кооперативного поиска и обменом совместным опыта, подробно описанным в 2.2.1.1. В тоже время проектируемая система должна предусматривать дальнейшее внедрение этих практик. Задачи, которые необходимо выполнять агентам для обмена опытом поиска [4], можно разделить между агентами описываемой системы следующим образом.

Первую из задач, наблюдение за действиями пользователя, можно поручить агенту пользователя, так как он может получать данные о

состоянии графического интерфейса. Через него можно отследить такие действия, как загрузка, чтение, цитирование статьи, а также их поиск.

Функции интеллектуального анализа данных для определения шаблона поведения пользователя можно поручить любому агенту, размещенному на условной клиентской стороне, либо создать отдельного агента, потому что ни один из классов агентов данной системы не обладает функциями, от которых этот процесс зависит.

Но третья задача – использование полученных знаний для ранжирования результатов, подходит лишь агенту предпочтений, так как именно он занимается ранжированием.

Таким образом, несложно увидеть, что благодаря применению мультиагентной архитектуры описанная функциональность может быть без труда добавлена в систему позже.

4 Реализация системы

В данной работе реализован прототип системы с описанной архитектурой. Для реализации была использована библиотека Java-классов каркаса JADE [17]. Созданный прототип ведет поиск публикаций в двух источниках данных – электронной библиотеке Arxiv.org [13] и поисковой системе Google Scholar [14].

4.1 Внутреннее представление публикации

Для работы с публикациями внутри системы был разработана структура данных, с помощью которой можно хранить все важные атрибуты публикаций. Далее описаны поля этой структуры.

- Ссылка на Web-страницу, описывающую данную публикацию. Поисковые системы, ведущие поиск по сайтам, выдают эти ссылки в качестве результата. В электронных библиотеках тоже существуют собственные страницы для всех статей.
- Список авторов. В данном прототипе системы поиска, об авторе может быть известно не только имя, но и его Web-страница, адрес электронной почты и организация, в которой он работает. В перспективе все это может быть использовано для поиска публикаций на странице автора или места его работы.
- Опубликовавший статью журнал – название журнала, номер и том, а также номера страниц.
- Название, год публикации, аннотация, ссылка на полный текст, предметная область, запись BibTeX, количество страниц.

Поисковые агенты, принимая данные о публикациях от своих источников, преобразуют их в этот формат.

4.2 Формат сообщений

В ходе поиска научных публикаций агентам системы часто требуется пересылать большие количества сложной по структуре информации, такой

как публикации и данные поисковых запросов. В JADE агенты обмениваются друг с другом сообщениями на языке FIPA ACL [15], который позволяет лишь строковый текст сообщения. Для передачи сложных структур данных в строковом сообщении удобно кодировать их с помощью XML-конструкций, специфицированных XML-схемами. XML Schema – это один из языков описания структуры XML-документа. Такая схема задает правила, которым должен подчиняться документ.

Существуют технологии, позволяющие по схеме сгенерировать Java-классы, с помощью которых можно преобразовывать сложные структуры данных в вид, пригодный для передачи в сообщениях (маршалинг), а также восстанавливать их из XML-строк (демаршалинг). В данном прототипе для этого использована технология JAXB[16] [18].

XML Shema позволяет описывать типы данных, поэтому все типы, которые нужно передавать в сообщениях, были описаны с помощью схем. В частности, это типы публикации, автора и журнала.

В данной системе применяются три типа сообщений, передающих сложные структуры данных.

Первый тип – это *сообщение-запрос*, используемое при передаче данных от агента пользователя по цепочке к агенту-поисковику. Оно содержит список ключевых слов запроса, которые должны встречаться в тексте, аннотации или названии статьи, а также желаемые значения некоторых атрибутов публикации (временной диапазон, авторы, журнал, предметная область). Кроме того, в сообщении содержится список источников, в которых пользователь хочет вести поиск, а также максимальное количество результатов. Для передачи таких данных используются сообщения с перформативом CFP (Call For Proposals).

[16] JAXB (Java Architecture for XML Binding) – технология, позволяющая получать XML-представление Java-классов.

Второй тип – это *сообщения-предложения*. Данные сообщения с перформативом Propose содержат найденные поисковыми агентами публикации. Кроме атрибутов публикации, они содержат источник, в котором публикация была найдена (Google Scholar или Arxiv.org), а также позицию в их рейтинге источника.

Первые два типа сообщений используются всеми классами агентов в любом сеансе поиска. Помимо них есть сообщения, которые применяются реже, например для передачи агенту предпочтений данных о том, какие из найденных статей понравились пользователю.

Вид всех типов сообщений задается XML-схемами. С их помощью генерируются Java-классы, включающие:

- все типы данных, описанные в схемах;
- фабрику, содержащую методы для генерации экземпляров всех описанных классов.

С помощью этих классов осуществляется маршалинг и демаршалинг сложных по структуре сообщений. На рис. 4 показаны отношения между классами, предназначенными для работы с сообщением-предложением.

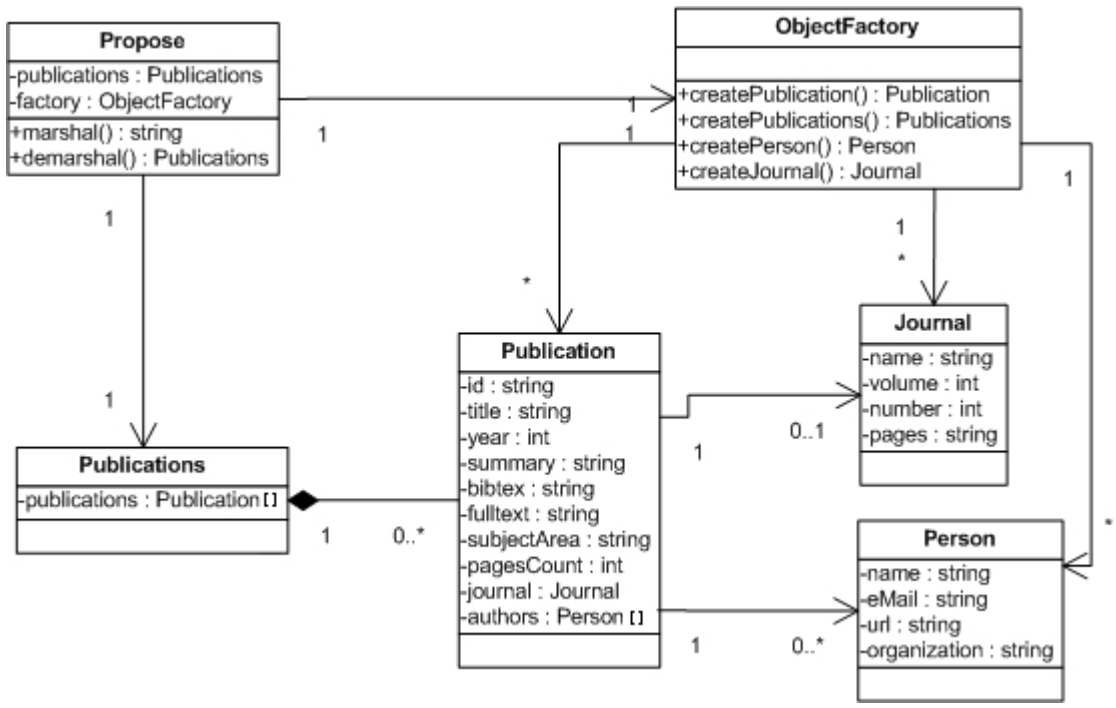

Рис. 4: Классы для работы с сообщением-предложением.

На рисунке классы Publication, Journal и Person – типы данных для публикации, журнала и автора соответственно.

Publications – класс, представляющий содержимое сообщения – список публикаций.

Propose – класс, с помощью которого происходит работа с сообщениями-предложениями. Он содержит методы преобразования списка публикаций в строку, а также преобразует XML-сообщения нужного типа в список публикаций (marshal и demarshal).

Помимо упомянутых классов, JAXB создает вспомогательный класс для работы с новыми типами данных, а также для поддержки операции маршалинга (ObjectFactory).

4.3 Реализация агентов

Все типы реализованных агентов при запуске регистрируются в реестре сервисов DF, используя в качестве названия сервиса свой тип. С помощью DF поиске они смогут находить адреса нужных классов для взаимодействия.

Агент пользователя является единственным агентом в системе, имеющим графический интерфейс. В начале работы этот агент запускает графическое приложение, через которое он будет взаимодействовать с пользователем, а также агента предпочтений пользователя, с которым он будет работать на протяжении всей поисковой сессии. Данные о предпочтениях агент пользователя получает при запуске, а затем передает их агенту предпочтений.

Предпочитаемые пользователем авторы, сайты, журналы и предметные области хранятся в XML-файле, специфицированном схемой так же, как и сообщения. Каждому из этих предпочтений соответствует числовой рейтинг, который агент предпочтения увеличивает в ходе поиска. Это происходит двумя способами – при получении поискового запроса и при сохранении

найденных публикаций для локальной работы. В этих случаях предпочтения извлекаются из запросов и публикаций.

Кроме того, получая список статей, найденных поисковыми агентами, агент предпочтений ранжирует эти статьи. К рангу статьи, присвоенному ей поисковиком добавляется рейтинг различных атрибутов найденной публикации (журнала, авторов, сайта). Затем публикации упорядочиваются в соответствии с суммарным рейтингом.

В данном прототипе системы поиска отсутствуют словари терминов различных дисциплин, поэтому классификация ключевых слов запроса и статей не производится. Агент, который должен, согласно архитектуре, выполнять эту функцию, занимается остальными предписанными ему задачами – ищет поисковых агентов и отправляет им всем запросы, а также выполняет слияние результатов.

То, как реализованы поисковые агенты, иллюстрирует следующий рисунок:

Рис. 5: Классы, описывающие поискового агента.

Для реализации поискового агента в данном прототипе необходимо создать три класса, являющиеся наследниками абстрактных классов, изображенных на рисунке, реализовав методы, также указанные на рисунке. В классе DataSourceAgent реализованы социальные функции агента-поисковика, его взаимодействие с классификатором. Поисковой агент

выполняет две основные функции – составление запроса к источнику данных и преобразование его ответов во внутренний формат публикаций.

За составление запроса отвечают наследники класса DataSourceQuery. С помощью метод calculateHref создается запрос к источнику данных, содержащий информацию, которую ввел пользователь.

Потомки класса DataSourceResponseParser должны реализовать метод request, отправляющий запрос источнику данных и получающий от него ответ. Также необходимо реализовать метод parse, разбирающий ответ источника данных и составляющий список публикаций. Методы requestDataSource и perform лишь вызывают функции классов DataSourceQuery и DataSourceResponseParser.

В данном прототипе такие классы были реализованы для взаимодействия с Google Scholar и Arxiv.org.

Сайт Arxiv.org в ответ на запрос выдает ответы в формате Atom 1.0, описанном в 2.1.1.2. Для разбора XML в этом формате использовались классы, сгенерированные с помощью JAXB по схеме, описывающей Atom.

В отличие от Arxiv.org, Google Scholar не имеет интерфейса прикладного программирования, поэтому для сбора данных о публикациях приходится анализировать HTML-страницу. Однако эта поисковая система предоставляет пользователю ссылки на BibTeX-записи для найденых публикаций, в которых информации о статьях даже больше, чем на самой странице результатов. Поэтому в HTML-странице находятся эти ссылки, после чего данные о публикациях берутся их BibTeX-записей.

При такой работе агентов, выполнение поискового запроса выглядит следующим образом:

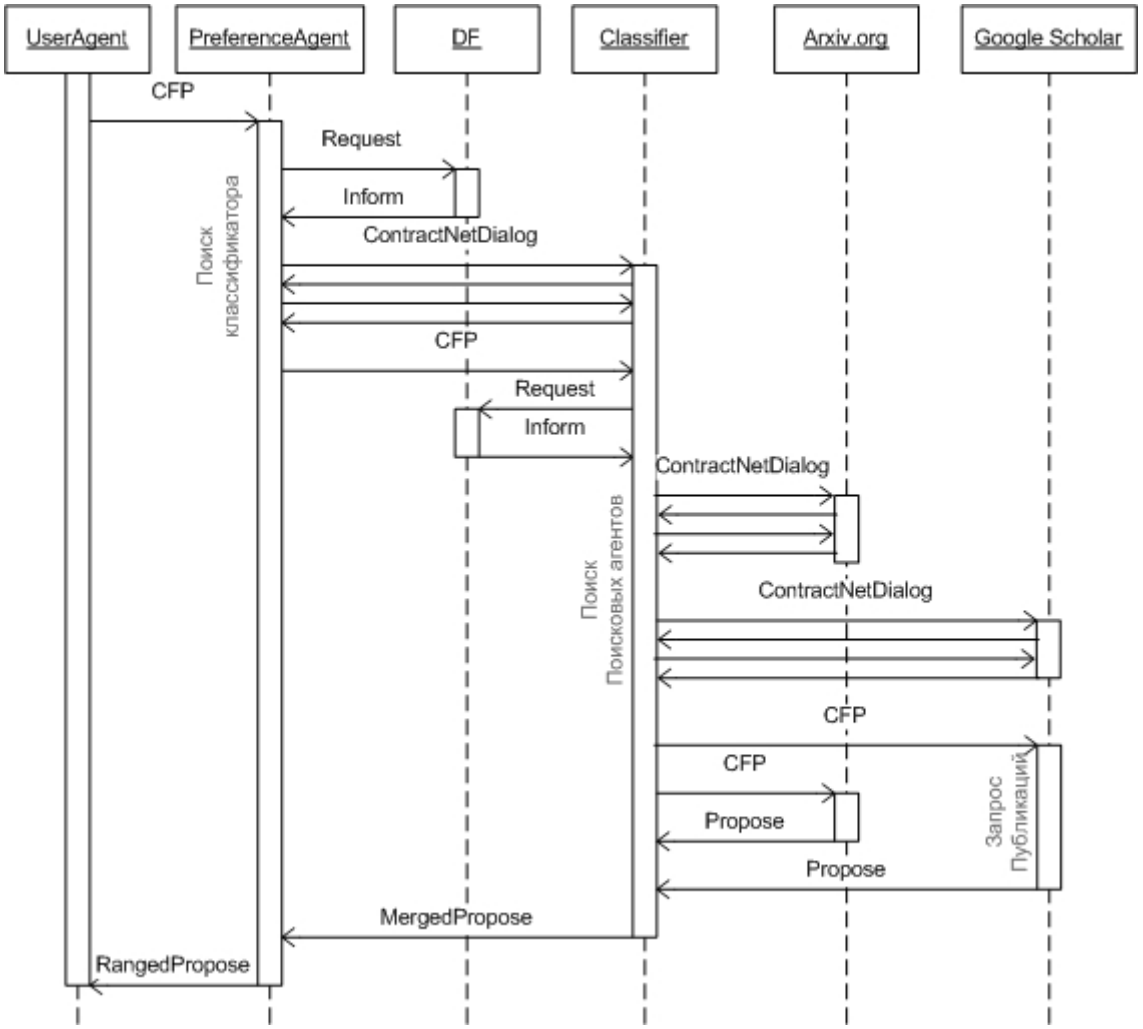

Рис. 6 : Взаимодействие агентов системы при выполнении поискового запроса.

На диаграмме изображена система с одним классификатором и одним источником данных каждого вида. В системе с большим количеством пользователей необходимо больше агентов, и тогда выбирать партнеров для взаимодействия с помощью Contract Net можно будет из многих кандидатов.

На рис. 6 сообщения, содержащие поисковой запрос помечены как CFP. Выбирая подходящего партнера для взаимодействия, агенты ведут диалог из нескольких сообщений, для краткости на рисунке он обозначен как ContractNetDialog. Помимо этого, перед применением протокола Contract Net агентам необходимо узнать список потенциальных кандидатов у DF (сообщения Request и Inform). Наконец сообщения-предложения на рисунке помещены как Propose. Они сначала сливаются вместе классификатором, а

затем скомпонованные статьи ранжируются агентом предпочтений. Ниже описан этот процесс.

Пусть Q = *(author,* $[keyword_i]_{i=1}^{m}$*, subject, jounal, ...)* – запрос пользователя к Arxiv.org и Google Scholar, n – максимальное количество результатов.

$(a_1, \ldots, a_n)$ – результат поиска в Axiv.org, $(g_1, \ldots, g_n)$ – в Google Scholar, где a_i, g_i = *(*$[author_i]_{i=1}^{m}$*,, subject, journal, site, rank,...), rank* – позиция публикации в списках результатов.

Агент предпочтений работает с данными, накопленными в ходе работы пользователя. Ему доступен числовой рейтинг сайтов, авторов, предметных областей и журналов, $rating > 0$.

1. Получая запрос Q от агента пользователя, агент предпочтений увеличивает рейтинг *author, subject* и *journal*, извлеченных из запроса. Затем Q передается классификатору.
2. Классификатор отправляет Q поисковым агентам, которые опрашивают свои источники данных и возвращают результаты $(a_1, \ldots, a_n)$ и $(g_1, \ldots, g_n)$. Получив результаты из нескольких источниов, классификатор сливает их в 1 список $(r_1, \ldots, r_{2n})$, так как в дальнейшем источник, из которого был получен результат, не будет ни на что влиять. Объединенный список публикаций отправляется агенту предпочтений.
3. Агент предпочтений вычисляет рейтинг каждой из статей как сумму её рейтинга внутри своего источника данных (он обратно пропорционален *rank*) и рейтинга её атрибутов:

$$rating(r_i) = rating(rank) + rating(site) + rating(subject) + rating(journal) + \sum_{i=1}^{m} rating(author_i),$$

после чего статьи сортируются в поряде убывания рейтинга: $(r_1`, ..., r_{2n}`)$, $rating(r_i`) \geq rating(r_{i+1}`)\ \forall i \in [1, 2n-1]$, и в таком порядке выдаются пользователю.

4. Далее пользователь выбирает среди $(r_1`, ..., r_{2n}`)$ некоторые публикации, которые нужны ему для локальной работы: $\{s_1, ..., s_k\} \subseteq \{r_1`, ..., r_{2n}`\}$. Они отправляются агенту предпочтений, который увеличивает рейтинг их атрибутов.

Благодаря шагу 4, при повторной отправке запроса *Q*, пользователь получит результаты, отсортированные в другом порядке. Позиции публикаций $\{s_1, ..., s_k\}$ будут выше, чем их позиции в результатах первого запроса.

4.4 Результаты реализации

Прототип системы поиска научных публикаций, реализованный в данной работе, ведет поиск информации на сайтах Arxiv.org и Google Scholar. В начале работы с системой поиск публикаций в каждом из источников по отдельности будет столь же эффективен, как и использование поисковых средств обоих сайтов напрямую. Однако уже в начале работы система дает возможность поиска в нескольких источниках данных через один интерфейс, а также возможность одновременного поиска с агрегацией, что делает прототип более удобным, чем те поисковые средства. которые он использует.

Накапливая знания о предпочтениях пользователя в поиске, данный прототип будет изменять порядок выдачи публикаций, если некоторые из них удовлетворяют этим предпочтениям. Благодаря такому методу, ранжирование найденных статей будет соответствовать вкусам пользователя больше, чем ранжирование, предложенное самими источниками данных, так как ни Arxiv.org, ни Google Scholar не используют персональные предпочтения. Приложение RefNavigator, описанное в 2.1.3.1, также их не учитывает.

Кроме того, архитектура системы, которую использует этот прототип, делает его удобным для оснащения в дальнейшем различными социальными методами поиска, такими как учет совместного опыта и возможность давать рекомендации, а также методами оптимизации поискового запроса, примером которых может служить классификация запроса или статьи. Для добавления нового источника данных в систему достаточно реализовать минимальный набор классов, взаимодействующих с этим источником, что делает систему удобной для расширения. Реализация упомянутых здесь методов поиска является приоритетным направлением дальнейшего развития системы. Помимо этого, важно расширить область поиска системы за счет добавления новых источников данных из числа крупных электронных библиотек и поисковых систем, а также добавить в систему новые функции для локальной работы со статьями.

5 Заключение

В ходе данной дипломной работы были достигнуты следующие результаты:

- предложены методы оптимизации Интернет-поиска, применимые к задаче поиска научных публикаций;
- разработана архитектура мультиагентной системы, применяющая эти оптимизации;
- реализован прототип такой системы в среде JADE, ведущий поиск в Google Scholar и Arxiv.org.

Список литературы

[1] *Д.Ю. Бугайченко, И. П. Соловьев*. Абстрактная архитектура интеллектуального агента и методы ее реализации / / Системное программирование, 2005б p. 36-67.

[2] *F. Bellifemine, A. Poggi, G.Rimassa*. Developing Multi-agent Systems with JADE / / Intelligent Agents VII Agent Theories Architectures and Languages, 2001, p. 42-47.

[3] *A. Birukou, E. Blanzieri, P. Giorgini*. A multi-agent system that facilitates scientific publications search / / Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, ACM, 2006, p. 272.

[4] *A. Birukou, E. Blanzieri, P. Giorgini*. Implicit: An agent-based recommendation system for web search / / Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, ACM, 2005, p. 624.

[5] J. Ferber, Multi-agent systems: an introduction to distributed artificial intelligence / / Addison-Wesley, 1999.

[6] *E. Garfield*. The impact factor and using it correctly / / Unfallchirurg, 1998, 6, Vol. 101, p. 413-414.

[7] *J.E. Hirsch*. An index to quantify an individual's scientific research output / / Proceedings of the National Academy of Sciences of the United States of America, National Acad Sciences, 102, Vol. 102, 46, p. 16569.

[8] *A.H. Keyhanipour, M. Piroozmand, B. Moshiri, Lucas*. A multi-layer/multi-agent architecture for meta-search engines / / (AIML-05), Cairo, Egypt , 2005.

[9] *M.P. Zillman*. Academic and Scholar Search Engines and Sources / / Virtual Private Library, 2006.

[10] *N. Jennings , M. Wooldridge*. Applications of Intelligent Agents / / Agent technology: Foundations, applications and markets, 1998, p. 3-28.

[11] *N. Jennings, M. Woolridge*. Intelligent agents: Theory and practice / / The knowledge engineering review, 1995, 02, Vol. 10, p. 115-152.

[12] *M. Wooldridge*. An introduction to multiagent systems / / Addison-Wesley, 2009, p. 4-23.

[13] Электронная библиотека Arxiv.org, (http://arxiv.org/).

[14] Поисковая система Google Scholar, (http://scholar.google.com/).

[15] Спецификации FIPA, (http://www.fipa.org/specs/).

[16] Приложение RefNavigator, (http://www.refnavigator.com/).

[18] Каркас JADE, (http://jade.tilab.com/).

[19] Технология JAXB, (http://java.sun.com/developer/technicalArticles/WebServices/jaxb/).