1.1. Семантический веб. Онтологии в интернете

Всемирная Паутина первого поколения строилась с ориентацией на обработку содержащейся в ней информации человеком. Поэтому язык HTML направлен, скорее, на разметку визуализации, чем на семантическую разметку, доступную для восприятия не только пользователями-людьми, но и компьютерными программами.

Технологии Всемирной Паутины нового поколения должны обеспечивать возможности автоматизированной интерпретации и обработки информации с целью представления в интернете знаний, а не просто данных. По мнению экспертов, такая Семантическая Всемирная Паутина (Semantic Web) будет проявлять признаки искусственного интеллекта.

Архитектура Семантической Всемирной Паутины предполагает наличие у любой информации, находящейся в Сети, связанного с этой информацией точного смысла, который нельзя было бы перепутать даже в случае совпадения фраз или слов, встреченных в разных контекстах. Фактически это означает, что любая информация связана с некоторым неотделимым от нее контекстом.

Сейчас информация для людей и компьютеров готовится отдельно. Для людей — в виде текстов, образов и звуков, для машин — в виде специальных кодов. Семантическая Всемирная Паутина предусматривает объединение разных видов информации в единую структуру, где каждому элементу «человеческой» информации будет соответствовать машинный код – специальный смысловой тэг. Все тэги будут составлять единую иерархическую структуру, на основе которой и будет работать Семантическая Паутина. При этом метаданные будут в обязательном порядке включать сведения о том, как, где и кем была собрана данная информация и как она структурирована.

Обратимся к традиционным конструкциям языка HTML, которые могут быть использованы для семантической разметки интернет-документов. [Гаврилова, Хорошевский, 2001] Это, в первую очередь, тэги <*TITLE*>, <*META*>, <*HR*>, <*A...*>. Текст в заголовке документа – тэг <*TITLE*> – как правило, отражает его назначение или содержание. Тэги <*META*> вводят имена атрибутов и их значения, в частности, могут содержать информацию о языке документа, о ключевых словах. Тэги <*HR*> выделяют заголовки и разделы в тексте. Гиперссылки и якоря – тэги <*A…*> – фиксируют отношение между отдельными документами и частями одного документа, соответственно. В некоторых случаях это отношение можно определить как *SeeAlso*, в других – как *IsA*, *PartOf* и т.д. Однако выделение значимых семантических конструкций остается экспертной задачей, и каждый автор ее решает по-своему.

Термин «онтология» позаимствован из философии, где определяет учение о бытии, о сущем. Предмет онтологии составляет изучение таких философских категорий как бытие, субстанция, причина, действия, явление. В свое время онтология как наука и часть метафизики претендовала на полное объяснение причин всех явлений.

Интерпретацию онтологии для использования в рамках ИИ-сообщества дал Т.Грубер.

Онтологией называется эксплицитная (то есть выраженная явно) спецификация определенной темы. Онтологию также можно определить как базу знанию специального типа, которая может «читаться» и пониматься, отчуждаться от разработчика и/или физически разделяться ее пользователями.

С позиции теории формальных систем, онтология состоит из словаря терминов, образующих таксономию, их определений и атрибутов, а также связанных с ними аксиом и правил вывода.

Под формальной моделью онтологии *O* будем понимать упорядоченную тройку вида:

$$O = \langle X, R, \Phi \rangle,$$

где X – конечное множество концептов (понятий, терминов) предметной области, которую представляет онтология *O*;

R – конечное множество отношений между концептами заданной предметной области;

Φ – конечное множество функций интерпретации (аксиоматизация), заданных на концептах и/или отношениях онтологии *O*.

Ограничением для множества X, является его конечность и непустота. Значение R и Φ могут принимать пустые значения и должны быть конечными множествами в определении онтологии *O*.

Пусть $R = \varnothing$ и $\Phi = \varnothing$. Тогда онтология O трансформируется в простой словарь:

$$O = V = \langle X, \{\}, \{\} \rangle.$$

Такая вырожденная онтология может быть полезна для спецификации, пополнения и поддержки словарей предметной области. Онтологии-словари имеют довольно ограниченное использование, хотя в некоторых случаях, когда используемые термины принадлежат очень узкому (например, техническому) словарю и их смыслы уже заранее хорошо согласованы в пределах определенного сообщества, такие онтологии применяются на практике. Известными примерами онтологии данного типа являются индексы машин поиска информации в сети Интернет.

Однако в тех случаях, когда используются термины естественного языка (ЕЯ) или общаются программные агенты, совершенно необходимо характеризовать предполагаемый смысл элементов словаря с помощью подходящей аксиоматизации, цель использования которой – в исключении нежелательных моделей и в том, чтобы интерпретация была единой для всех участников общения.

Рассмотрим вариант, в котором $R = \varnothing$, $\Phi \neq \varnothing$. Тогда каждому элементу множества терминов из X может быть поставлена в

соответствие функция интерпретации f из Φ. Формально это утверждение может быть записано следующим образом.

Пусть $X = X_1 \cup X_2$, причем $X_1 \cap X_2 = \varnothing$, где X_1 – множество интерпретируемых терминов, X_2 – множеств о интерпретирующих терминов. Тогда $\exists \left(x \in X_1, y^1, y^2, ..., y^k \in X_2\right)$, такие что $x = f\left(y^1, y^2, ..., y^k\right)$, где $f \in \Phi$.

Вид отображения f из Φ определяет выразительную мощность и практическую полезность этого вида онтологии. Так, если предположить, что функция интерпретации задается оператором присваивания значений $(X_1 := X_2,$ где X_1 – имя интерпретации $X_2)$, то онтология трансформируется в пассивный словарь V^p:

$$O = V^p = \langle\ X_1 \cup X_2, \varnothing, \{:=\}\ \rangle.$$

Такой словарь пассивен, так как все определения терминов из X_1 берутся из уже существующего фиксированного множества X_2. Практическая ценность его выше, чем простого словаря, но явно недостаточна, например, для представления знаний в задачах обработки информации в интернете в силу динамического характера этой среды.

Для того чтобы учесть последнее обстоятельство, используется активный словарь. Предположим, что часть интерпретирующих терминов из множества X_2 задается процедурно, а не декларативно. Смысл таких терминов “вычисляется” каждый раз при их интерпретации:

$$O = V^A = \langle\ X_1 \langle X_1 \cup (X_2 \cup p_2), \varnothing, \Phi\ \rangle.$$

Для представления модели онтологии, которая может быть использована для решения задач обработи информации в интернете, очевидно, потребуется отказаться от предположения $R = \varnothing$.

Введем в рассмотрение специальный подкласс онтологий – простую таксономию вида:

$$O = T = \langle\ X, \{is\_a\}, \varnothing\ \rangle.$$

Под таксономической структурой будем понимать иерархическую систему понятий, связанных между собой отношением «быть элементом класса». Отношение *is_a* имеет фиксированную заранее семантику и позволяет организовать структуру понятий онтологии в виде дерева. Такой подход в общем случае является адекватным и удобным для представления иерархии понятий.

Под обобщенный формальной моделью онтологии [Тарасов, 2002], расширяющей рассмотренную модель, будем понимать тройку вида:

$$O=(U, Im(R), Ф),$$

где U – множество понятий предметной области, $|U| \neq \varnothing$, $\mathrm{Im}(R) = \{w \mid w : U^n \to [0,1]\}$ – множество нечетких отношений между понятиями предметной области; $\Phi = \{f\}$ – конечное множество функций интерпретации (аксиоматизация), заданных на понятиях и/или отношениях онтологии O, $f : D^n \to [0,1]$, D – область интерпретации.

Запишем в общем виде алгоритм онтологического инжиниринга [Гаврилова, Хорошевский, 2001]:

1. выделение концептов – базовых понятий предметной области;
2. определение «высоты дерева онтологий» – числа уровней абстракции;
3. распределение концептов по уровням;
4. построение связей между концептами – определение отношений и взаимодействий базовых понятий;
5. консультации с экспертами для исключения противоречий и неточностей.

Комплекс рекомендаций W3C, связанных с Семантической Всемирной Паутиной, имеет следующую структуру:

– XML обеспечивает синтаксис для структурированных документов, но не налагает никаких семантических ограничений на значение этих документов;

– XML Schema определяет структуру документов XML, а также дополняет XML конкретными типами данных;
– RDF позволяет описать модель данных для объектов-ресурсов и отношения между ними, обеспечивает простую семантику для этой моделей данных, представляя их в XML синтаксисе;
– RDF Schema предоставляет средства для описания свойств и классов RDF-ресурсов, а также семантику для иерархий-обобщений таких свойств и классов;
– OWL добавляет еще больше возможностей для того, чтобы описать свойства и классы: в частности, отношения между классами (например, непересекаемость), кардинальность (например, «точно один»), равенство, больше типов свойств, характеристик свойств (например, симметрия), и перечисляемые классы.

Особенность XML-языков является наличие стандартов на определение синтаксиса и средств введения новых тэгов, что позволяет создавать новые языки разметки и обеспечивать возможность различным приложениям «понимать» и обрабатывать XML-документы.

Каждый XML-документ обладает определенной логической и физической структурой. Физически это композиция элементов, называемых единицами, которые могут быть связаны взаимными ссылками. Логически документ состоит из деклараций, единиц, комментариев, собственно текстов и инструкций обработки, причем каждая конструкция XML маркируется специальными тэгами. Все тэги – парные, а конструкции могут быть вложены друг в друга. Например, конструкция <Item Attribute1=”Value1”></Item> определяет единицу с именем Item и списком пар “атрибут-значение”, который в нашем случае представлен единственным атрибутом с именем Atribute1, имеющим значение “Value 1”.

Рассмотрим содержательный пример XML-документа, описывающего домашнюю страницу ученого Иванова:

```
<?xml version="1.0"?>
<Homepage>
        <Name>Домашняя страница Иванова</Name>
  <Person>
        <firstName>Иван</firstName>
        <lastName>Иванов</lastName>
        <married ToHomepage="http://...">Мария Иванова</marriedTo>
        <employee Homepage="http://www.tsure.ru">ТРТУ</employee>
    <publications>
        <book title="Учебное пособие" />
        <book title="Методические указания" />
    </publications>
  </Person>
</Homepage>
```

XML-документ структурирован существенно лучше, чем HTML-текст, но пока не имеет «смысла», так как из него не следует, как интерпретируются единицы типа Person, publications, book и т.д. С этой целью используется специальная спецификация определения типа документа Document Type Definition (DTD). По сути, это грамматика языка разметки, в рамках которой определяется, какие элементы могут присутствовать в документе, какие атрибуты они имеют и как элементы соотносятся друг с другом. Каждая спецификации DTD определяет новый язык разметки.

Рассмотрим пример DTD-спецификации:

```
<!ELEMENT Homepage (Name, Person)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Person (firstName, lastName, marriedTo?, employee?,
                  publications?, Homepage?)>
```

<!ATTLIST Person Homepage xml:link CDATA>
<!ELEMENT firstName (#PCDATA)>
<!ELEMENT lastName (#PCDATA)>
<!ELEMENT marriedTo (Person)>
<!ELEMENT employee <organization)>
<!ATTLIST organization Homepagexml:link CDATA>
<!ELEMENT publications (book, paper*, report*)>*
<!ATTLIST book title CDATA #REQUIRED, coauthor Person,
publisher CDATA, year CDATA)>
<!ELEMENT paper (title, coauthor, journal, year, vol?, number?)>*
<!ELEMENT report (title, coauthor, organization, year?)>*

Для спецификации общих ресурсов широко используется Resource Definition Framework (RDF). RDF определяет простую модель данных, состоящую из троек (субъект, предикат, объект), где значениями являются Uniform Resource Identifiers (URI) – универсальные идентификаторы ресурсов. Таким образом, с помощью конструкций RDF можно описывать отношения между различными объектами. Например, "(Париж) (является столицей) (Франция)". Все элементы тройки RDF – это гиперссылки, ведущие туда, где определяются термины "Париж", "является столицей" и "Франция".

Основным современным направлением работ по созданию Семантической Всемирной Паутины является разработка развитого языка описаний онтологий, на роль которого претендует Ontology Web Language (OWL). Язык OWL строится на основе стандартов RDF и обогащает предоставляемые ими возможности для описания свойств и классов. Например, для классов можно указывать, что они не пересекаются, указывать их кардинальность, определять эквивалентность. Поддерживаются перечисляемые классы. OWL располагает более богатой системой типов и позволяет указывать характеристики свойств, например, симметричность.

Поскольку Семантическая Сеть, по определению, распределена, OWL должен позволять собирать информацию из распределенных источников. Это частично обеспечивается возможностью онтологий быть связанными, включая прямой импорт информации из других онтологий.

Кроме того, OWL предполагает открытость. То есть, описания ресурсов не ограничены единственным файлом или темой. В то время как класс C1 первоначально может быть определен в онтологии O1, он может быть расширен в других онтологиях. Следствия из этих дополнительных суждений о C1 являются монотонными. Новая информация не может опровергать предыдущую информацию. Новая информация может быть противоречащей, но факты и следствия могут только добавляться, и не могут удаляться.

Объединение OWL-онтологий осуществляется простым объединением списков утверждений, поэтому для интеграции веб-ресурсов необходимо выполнение лишь следующих требований: 1) публикация онтологий в интернете для свободного доступа, 2) следование практике стандартизации и повторного использования существующих онтологий, 3) использование средств OWL для соотнесения понятий разных онтологий.

OWL имеет три различных по выразительности диалекта, спроектированных для использования отдельными сообществами разработчиков и пользователей. Разрабатывая онтологию, нужно решить, какой из диалектов лучше подходит к конкретной задаче:

- OWL Lite предназначен для тех пользователей, которые нуждаются, прежде всего, в классификационной иерархии и простых ограничениях. Например, при том, что он поддерживает ограничения кардинальности (количества элементов), допускаются значения кардинальности только 0 или 1.
- OWL DL ориентирован на тех пользователей, которые хотят максимальной выразительности при сохранении полноты

вычислений (все заключения гарантировано будут вычисляемыми), и разрешаемости (все вычисления завершатся в определенное время). OWL DL включает все языковые конструкции OWL, но они могут использоваться только согласно определенным ограничениям (например, в то время как класс может быть подклассом многих классов, класс не может быть представителем другого класса). OWL DL так назван из-за его соответствия дескриптивной логике – дисциплине, в которой разработаны логики, формирующие формальную основу OWL.

– OWL Full предоставляет максимальную выразительность и синтаксическую свободу RDF без гарантий вычисления. Например, в OWL Full класс может рассматриваться одновременно как собрание индивидов и как один индивид в своем собственном значении. OWL Full допускает такие онтологии, которые расширяют состав предопределенного словаря.

Разработке онтологий посвящено отдельное направление инженерии знаний – онтологический инжиниринг.

Инженерия онтологий – сложный и занимающий много времени процесс. Чтобы облегчить его, в середине 90-х годов начали создаваться первые среды для разработки онтологий. Они обеспечили интерфейсы, которые позволили выполнять концептуализацию, реализацию, проверку непротиворечивости и документирование онтологий. В настоящее время существует целый ряд инструментов для создания и поддержки онтологий, которые, помимо общих функций редактирования и просмотра, обеспечивают импорт и экспорт онтологий разных форматов и языков, поддержку графического редактирования, управление библиотеками онтологий и т.д. [Овдей, Проскудина, 2004]

Одним из наиболее известных инструментов инженерии онтологий является Protégé, свободно распространяемая локальная Java-программа, разработанная группой медицинской информатики Стэнфордского

университета (http://protege.stanford.edu/). Это интегрированное инструментальное программное средство используется разработчиками систем и экспертами по предметным областям для разработки систем, основанных на знаниях. Приложения, разработанные при помощи Protégé, используются при решении задач и принятии решений в конкретной предметной области.

Protégé представляет собой редактор онтологий, позволяющий проектировать онтологии, разворачивая иерархическую структуру абстрактных или конкретных классов и слотов. На основе сформированной онтологии Protégé может генерировать формы получения знаний для введения экземпляров классов и подклассов.

В то время как в классической системе баз данных отдельно определялись классы и хранились экземпляры этих классов, в Protégé отдельный экземпляр может быть использован на уровне описания класса, а класс можно хранить как экземпляр. Protégé основан на фреймовой модели представления знания OKBC (Open Knowledge Base Connectivity) и снабжен рядом плагинов, что позволяет адаптировать его для редактирования моделей, хранимых в разных форматах (стандартный текстовый, в базе данных JDBC, UML, XML, XOL, SHOE, RDF и RDFS, DAML+OIL, а также OWL).

Protégé имеет унифицированный графический интерфейс пользователя, верхний уровень которого представляет собой перекрывающиеся вкладки для компактного представления частей и удобного их совместного редактирования. Такой дизайн делает возможным интеграцию: 1) моделирования онтологии классов, описывающей определенную дисциплину; 2) создания средства приобретения знаний для сбора информации; 3) ввода отдельных экземпляров данных и создания базы знаний; 4) выполнения приложений внутри интегрированной среды. При этом онтология определяет набор понятий и их отношения. Средство приобретения знаний разрабатывается специально для предметной области, позволяя экспертам легко и

свободно вводить свои знания в предметной области. Конечная база знаний затем может быть использована вместе с методом поиска решения задач для ответа на вопросы и решения задач в предметной области. Таким образом, приложение является конечным продуктом, в котором база знаний применяется для решения задачи конечного пользователя с использованием подходящих методов поиска решения задач, методов «эксперт-система» и методов поддержки принятия решений.

В традиционном представлении разработка баз знаний – сложный и дорогостоящий процесс. Разработкой системы, основанной на знаниях, обычно занимается команда как из разработчиков, так и из экспертов по предметной области, которые могут быть в меньшей степени знакомы с компьютерным программным обеспечением. Protégé предполагает описание предметной области уже в процессе разработки системы. Инструментальные средства среды позволяют разработчикам повторно использовать онтологии предметных областей и методы поиска решения задач, таким образом уменьшая время, необходимое для разработки и поддержки программы. Несколько приложений могут использовать одну и ту же онтологию предметной области для решения различных задач; один и тот же метод поиска решения задач может быть использован с различными онтологиями.

1.2. Интеллектуальные агенты и многоагентные системы в интернете

Современный словарь дает следующее определение понятия «агент» – лицо или организация, наделенное полномочиями представлять другое лицо или организацию и вести их дела. Это определение весьма удачно передает сущность искусственных агентов.

Как таковое научное направление, изучающее интеллектуальных агентов и мультиагентные системы, сформировалось в конце 70-х годов

на основе результатов, полученных в рамках работ по распределенному решению задач, распределенному и параллельному искусственному интеллекту. Уже первые исследования «умных» агентов в 1977-м году изначально были посвящены анализу принципов взаимодействия между агентами, декомпозиции решаемых задач на подзадачи и распределении полученных задач между отдельными агентами, координации и кооперации агентов, разрешении конфликтов путем переговоров и т.д. В 90-х годах внимание научного сообщества переключилось на программные агенты

К настоящему времени сформировалось несколько различных трактовок понятия «искусственный агент».

Еще в 1975 году ученый Холланд исследовал искусственные организмы, развивающиеся в популяции себе подобных, стремящиеся обучаться и адаптироваться к внешней среде, чтобы выжить в ней и победить конкурентов. Эта интерпретация опирается на теоретические подходы и модели искусственной эволюции и принципы искусственной жизни. Она тесно связана с робототехникой, вопросами информационной безопасности и компьютерной вирусологии, а также эволюционирующим программным обеспечением.

Вторая трактовка «искусственного агента» – это персональный помощник пользователя или интеллектуальный посредник между пользователем и компьютерной средой, в которой он работает. Истоки данной концепции относятся к теории диалога «человек-компьютер» и средствам развития интеллектуальных интерфейсов. С развитием Интернета начался настоящий бум в области программных агентов. В качестве примеров можно привести информационных агентов PointCast, которые доставляют новости и сообщают об изменениях на избранных сайтах. Агенты покупок BargainFinder сравнивают цены в электронных магазинах. Роботы-пауки бродят по ссылкам и индексируют информацию для поисковых машин.

Третья интерпретация «искусственного агента» – это новая парадигма программирования как развитие идей объектно-ориентированного программирования, основанная на социальном взгляде на компьютерные вычисления.

Для определенности далее будем считать, что агент – аппаратная или программная сущность, способная действовать автономно в интересах достижения целей, поставленных перед ним владельцем или пользователем. [Гаврилова, Хорошевский, 2001]

Существуют два подхода к построению агентно-ориентированных систем: реализация единственного автономного агента и разработка мультиагентной системы. Автономный агент взаимодействует с пользователем и реализует весь спектр функциональных возможностей. В свою очередь, мультиагентная система – это своего рода агентство, программно-вычислительный комплекс, где взаимодействуют различные агенты для решения поставленной перед ними задачи.

На практике, используя понятие «агент», каждый автор обычно определяет своего собственного агента с конкретным набором свойств, в зависимости от целей, решаемых задач, техники реализации и прочих критериев. Как следствие, появилось множество типов агентов, например: автономные агенты, мобильные агенты, персональные ассистенты, интеллектуальные агенты, социальные агенты и т.д.

Приведем одну из классификаций агентов (таблица 2.1):

Таблица 2.1. Классификация агентов

Типы агентов/ Характеристики	**Простые**	**Смышленые**	**Интеллектуальные**
Автономное выполнение	+	+	+
Взаимодействие с другими агентами и/или пользователями	+	+	+
Слежение за окружением	+	+	+

Продолжение таблицы 2.1.

Способность использования абстракций		+	+
Способность использования предметных знаний		+	+
Возможность адаптивного поведения для достижения целей			+
Обучение из окружения			+
Толерантность к ошибкам			+
Исполнение в режиме реального времени			+
Взаимодействие на естественном языке			+

Как следует из таблицы, целесообразное поведение проявляется только на уровне интеллектуальных агентов, для которых характерно не только наличие целей функционирования, но и возможность использования достаточно сложных знаний о среде, партнерах и самих себе. Понятно, что характеристики более простых агентов при этом наследуются.

Один из распространенных подходов к определению агента – составление списка свойств, которыми он обладает.

Приведем типовой список свойств интеллектуальных агентов:

– автономность – способность функционировать без вмешательства со стороны своего влаждельца и осщуествлять контроль внутреннего состояния и своих действий;

– социальное поведение – возможность взаимодействия и коммуникации с другими агентами;

– реактивность – адекватное восприятие среды и соответствующие реакции на ее изменения;

– активность – способность генерировать цели и действовать рациональным образом для их достижения;

– базовые знания – знания агента о себе, окружающей среде, включая других агентов, которые не меняются в рамках жихненного цикла агента;

– убеждения – переменная часть базовых знаний, которые могут меняться во времени, хотя агент может об этом не знать и продолжать из использовать для своих целей;

– цели – совокупность состояний, на достижение которых направлено текущее поведение агента;

– желания – состояния и/или ситуации, достиэение которых для агента важно;

– обязательства – задачи, которые берет на себя агент по просьбе и/или поручению других агентов;

– намерения – то, что агент должен сделать в силу своих обязательств и/или желаний.

Иногда в этот перечень добавляются такие свойства, как рациональность, правдивость, благожелательность, а также мобильность агента.

В общем случае, агент *a* обладает *n* различными функциональными свойствами в разной степени. Его можно охарактеризовать вектором значений принадлежности $\mu = (\mu_1(x),...,\mu_n(x))$, выражающим степень удовлетворения этим свойствам. $\mu : X \to [0,1]^n$. Соответственно, идеальный агент характеризуется вектором $\mu^0 = (1,...,1)$.

Взаимодействие между агентами в МАС удобно описать при помощи математического аппарата теории отношений (нечетких отношений). Запись *aRb* означает, что агент *a* находится в отношении *R* с

агентом b, а $\mu_R(a,b)$ понимается как степень интенсивности (сила) проявления этого отношения.

Для отношений могут выполняться свойства рефлексивности, антирефлексивности, симметричности, асимметричности, антисимметричности, полноты, транзитивности и т.д.

Рассмотрим, например, условие транзитивности, которую можно интерпретировать следующим образом: если агенты a и c связаны опосредованно через агента b, то они могут быть связаны и непосредственно друг с другом, причем эта связь такая же или сильнее, чем опосредованная.

Математическое выражение запишется в следующем виде:

$$R \circ R \subseteq R,\ \mu_R(a,c) \geq \mu_R(a,b) * \mu_R(b,c),\ \forall a,b,c \in A\ .$$

В зависимости от концепции организации многоагентных систем, выделяют три класса архитектур МАС:

– архитектуры, которые базируются на принципах и методах работы со знаниями (делиберативные);

– архитектуры, основанные на поведенческих моделях типа «стимул-реакция»;

– гибридные архитектуры.

Архитектуру МАС или агентов, которые используют точное представление картины мира в символьной форме, а решения при этом (например, о действиях) принимают на основе формальных рассуждений и использования методов сравнения по образцу, принято определять как делиберативные.

Иными словами, речь идет о разумных агентах и архитектурах, имеющих в качестве основы проектирования и реализации модели, методы и средства искусственного интеллекта. Иначе их называют когнитивными агентами, агентами, базирующимися на знаниях, или интеллектуальными агентами.

Принципы реактивной архитектуры возникли как альтернативный подход к архитектуре интеллектуальных агентов. По мере развития исследований в этой области стало ясно, что некоторые ментальные свойства агентов, как, например, убеждения, желания, намерения, обязательства по отношению к другим агентам, невыразимы в терминах исчисления предикатов первого порядка классической логической парадигмы ИИ. Поэтому были использованные специальные расширения логических исчислений, а также разработаны новые архитектуры, в частности, архитектуры типа «Убеждение-Желание-Намерение» (Belief-Desire-Intention).

Первым тезис о том, что интеллектуальное поведение может быть реализовано без символьного представления знаний, принятого в классическом ИИ, выдвинул Брукс.

Реактивными называются агенты и архитектуры, где нет эксплицитно представленной модели мира, а функционирование отдельных агентов и всей системы в целом осуществляется по правилам типа «ситуация-действие». Под ситуацией понимается потенциально сложная комбинация внутренних и внешних состояний.

Примером реализации таких реактивных архитектур являются системы, где реакции агентов на внешние события генерируются соответствующими конечными автоматами. В качестве еще одного примера можно привести систему STRIPS, где используется логический подход, расширенный за счет предусловий и постусловий. Считается, что реактивные архитектуры пока не могут справиться с задачами реального уровня сложности.

Поскольку ни первый, ни второй подходы не дают оптимального результата, предпринимаются попытки комбинирования этих подходов, которые привели к появлению гибридных структур. Именно гибридные архитектуры используются в настоящее время практически во всех современных проектах и системах. К их недостаткам можно отнести специфичность для приложений, под которые они разрабатываются.

Подытожим список существующих архитектур МАС и их характеристики (таблица 2.2.):

Таблица 2.2. Архитектуры МАС и их характеристики

Архитектура	**Представление знаний**	**Модель мира**	**Решатель**
Интеллектуальная	Символьное	Исчисление	Логический
Реактивная	Автоматное	Граф	Автомат
Гибридная	Смешанное	Гибридная	Машина вывода

При рассмотрении МАС говорят о взаимодействии агентов, его характеристиках и видах. Выделяют кооперацию и конкуренцию агентов, а также конфликты.

К моделям кооперации агентов относятся:

– модель контрактный сетей Смита;

– модель аукциона;

– модель монотонных минимальных уступок (по Розенштейну и Злоткину);

– модель социальных зависимостей Кастельфранши и Контэ.

Рассмотрим пример описания процессов взаимодействия между агентами на основе аппарата теории нечетких отношений.

Будем обозначать положительное взаимодействие между агентами (содействие или кооперацию) через R+, а отрицательное взаимодействие – через R–. Соответственно, невзаимодействие (или уклонение от взаимодействия) обозначим как R_0.

Тогда весь диапазон взаимодействия R можно представить в виде:

$$R = R^+ \cup R^- \cup R^0$$

Если при этом полагать, что взаимодействие между агентами характеризуется не только знаком, но и различной силой (интенсивностью) взаимодействия, то различные виды взаимодействия агентов в МАС можно представить нечеткими отношениями вида:

$$g_R : A \times A \to [-1;1] \text{ } \textit{(по Коско)}$$

Отношение, обладающее свойством слабой рефлексивности

$$g_{R+}(a_i, a_i) > g_{R+}(a_i, a_j),\ \forall a_i, a_j \in A$$

будет означать, что агент в большей мере содействует себе, чем любому другому агенту.

Отношение антирефлексивности

$$g_R(a_i, a_i) = 0, \forall a_i \in A$$

будет характеризовать абсолютное безразличие агента по отношению к самому себе.

Связывая $g_R(a_i, a_i)$ с отношением агента к своим индивидуальным целям (интересам), а $g_R(a_i, a_j)$, наоборот, с его отношением к целям других агентов, можно также выделить благонамеренных, эгоистичных, злонамеренных и альтруистичных агентов.

Например, злонамеренный агент

$$g_{R+}(a_i, a_i) = +1,\ g_{R-}(a_i, a_j) = -1,\ \forall a_i, a_j \in A$$

соблюдает свои интересы, но в максимальной степени противодействует достижению целей всех других агентов.

Далее рассмотрим вопросы проектирования и реализации агентов и многоагентных систем.

Как мы уже говорили выше, если процедуры, функции, методы и классы – это давно известные абстракции, которые разработчики программного обеспечения используют ежедневно, то программные агенты – это принципиально новая парадигма.

Средства разработки и исполнения распределенных приложений МАС основываются на следующих двух подходах:

– статический подход, когда передаются только данные приложения;

– динамический подход, при котором передается исполняемый код.

В случае динамического подхода речь идет о так называемых мобильных агентах.

Мобильные агенты – это программы, которые могут перемещаться по сети, они покидают клиентский компьютер и перемещаются на удаленный сервер для выполнения своих действий, после чего возвращаются обратно.

Сформулируем преимущества мобильных агентов**:**

– уменьшают время и стоимость передачи данных (например, при больших объемах данных вместо передачи всей необработанной информации по сети на сервер посылается агент, который выбирает только необходимую информацию и передает ее пользователю);

– позволяют преодолеть ограничение локальных ресурсов (например, если возможности процессора и объем памяти клиентского компьютера малы, целесообразнее использовать мобильных агентов, выполняющих вычисления на сервере);

– облегчают координацию (например, запросы к удаленным серверам выполняются мобильными агентами как отдельные задачи, поэтому не нуждаются в координации);

– позволяют выполнять асинхронные вычисления (например, запустив агента, можно переключиться на другое приложение и даже отсоединиться от сети, а результат будет доставлен агентом адресату после выполнения задания).

Единых стандартов разработки мобильных агентов пока не существует. Остаются нерешенными вопросы легального перемещения агентов по сети, верификации агентов (чтобы их можно было отличить от вирусов), соблюдения агентами прав частной собственности и конфиденциальности информации, перенаселения сети агентами, а также совместимости кода агента и программно-аппаратных средств машины, на которой он исполняется.

В качестве программных средств реализации мобильных агентов используется программирование сокетов и вызов удаленных процедур, в частности технологии:

- Remote Procedure Control (RPC);
- Microsoft Distributed Component Object (DCOM);
- Java Remote Method Invocation (Java RMI);
- Common Object Request Broker Architecture (CORBA).

На сегодня известно около 30 коммерческих инструментальных средств разработки.

Примером инструментария для построения МАС является пакет AgentBuilder компании Reticular Systems.

Данный инструментарий имеет все средства для организации предметной области создаваемой МАС, средства спецификации архитектуры агентства и поведения агентов, а также средства отладки агентных приложений и наблюдения за поведением созданных агентов.

Ключевым понятием жизненного цикла агента в системе AgentBuilder является ментальная модель, которая включает описание исходных (или текущих) намерений, полаганий, обязательств и возможностей, а также спецификации правил поведения.

Рассмотрим жизненный цикл агента в системе AgentBuilder:

1. Обработка новых сообщений.
2. Определение правил поведения, применимых в текущей ситуации.
3. Выполнение действий, определенных правилами поведения.
4. Обновление ментальной модели в соответствие с заданными правилами.
5. Планирование

Областью практического использования агентных технологий являются в первую очередь управление информационными потоками и сетями, управление воздушным движением, информационный поиск, электронная коммерция, обучение и целый ряд других областей. Как

правило, во всех случаях речь идет о сборе информации, ее фильтрации и использовании для принятия решений.

Рассмотрим систему MARRI, использующую онтологии для поиска веб-страниц, релевантных запросам в определенной предметной области.

Разработчики системы предположили, что релевантные тексты состоят из значимых для предметной области предложений, содержащих фрагменты онтологии предметной области. Одни агенты – агенты сети – для предварительного отбора используют стандартные машины поиска. После чего специализированные агенты осуществляют поверхностный анализ результатов, проверяя веб-страницы на соответствие онтологическому тесту, и возвращают пользователю лишь те страницы, которые прошли данный тест.

В ходе онтологического теста осуществляется морфологический и синтаксический анализ предложений полученного от агентов сети текста, строится синтаксическое дерево, определяется тип предложений и для дальнейшего анализа выбираются только простые утвердительные предложения с четко заданной структурой. Если подтверждается, что анализируемое предложение описывает некоторый концепт, это означает, что значимые для предметной области слова присутствуют в онтологии.

С точки зрения архитектуры, MARRI является сетью специализированных агентов четырех типов: агент пользователя, агент-брокер, агент сети и агент обработки текста.

Каждый агент – это автономная Java-программа, взаимодействующая с другими агентами с помощью языка ACL (Agent Communication Language).

Агент пользователя обеспечивает интеллектуальное взаимодействие с пользователем. Он помогает при формулировании запросов и представляет результаты поиска в виде списка релевантных веб-страниц. Он же на основе выбора пользователя запрашивает соответствующую онтологию из базы данных и информирует агентов сети, какая онтология будет использоваться.

Агент сети осуществляет подключение к заданной веб-странице, ее считывание и анализ. Ко всему прочему, он умеет обрабатывать внештатные ситуации, например, когда страница недоступна или неинтересна по содержанию.

Агенты-брокеры контролируют список адресов и распределяют тексты веб-страниц между агентами обработки текста.

Наконец, агенты обработки текста осуществляют семантический анализ веб-страниц для проверки их релевантности на основе соответствующей онтологии.

Нужно сказать, что такая архитектура является традиционной для современных многоагентных систем.

1.3. Персонализация. Адаптивные веб-ресурсы

Открывая разные издания классического учебника по маркетингу Ф.Котлера «Основы маркетинга», можно заметить, что большая часть информации остается практически без изменений, и лишь глава «Интерактивный маркетинг» уверенно растет в объеме.

За последние десять лет маркетинг прошел четыре основные стадии [Хэнсон, 2001].

Первая стадия характеризовалась ориентацией на продукт. Это было время, когда производство не достигло еще того массового уровня, который мы имеем сегодня, и количество товаров было очень ограничено. Товары сами по себе пользовались спросом, поэтому усилия по маркетингу сводились к минимуму.

Вторая стадия – так называемая, ориентация на продажи, – наступила, когда производство достигло определенного уровня развития, рынок наполнился товарами, а компании начали вести активную торговлю.

Третья стадия, известная как стадия сегментной ориентации, ознаменовала переход к продажам, направленным не на весь рынок, как это было ранее, а на определенные сегменты с совершенно конкретными задачами. Компании адаптировали товары, услуги и методы взаимодействия с потребителями к конкретным целевым сегментам рынка.

Наконец, четвертая стадия – стадия ориентации на потребителя – наступила тогда, когда компании решили собирать информацию об отдельных участниках целевого сегмента рынка.

Другими словами, можно проследить эволюцию маркетинга от массового, предполагающего совершенно одинаковый подход ко всем потребителям, к целевому маркетингу сегментов, который допускает выделение в пределах рынка групп потребителей, к микромаркетингу,

при котором компания ориентирует свои маркетинговые программы на нужды локальных потребительских групп и отдельных потребителей. В своей крайней форме микромаркетинг проявляется как индивидуальный маркетинг, маркетинг рынков одного потребителя, маркетинг на заказ, маркетинг один-на-один.

Индивидуальный маркетинг – это приспособление товарного ассортимента и маркетинговых программ к нуждам и предпочтениям отдельных потребителей в соответствие с их личными потребностями.

Поскольку нужды и потребности каждого покупателя уникальны, в теории каждого покупателя, действительно, можно рассматривать в качестве отдельного рынка. Следовательно, в идеале продавец может для каждого из них разработать отдельную маркетинговую программу.

До недавнего времени такое полномасштабное сегментирование многочисленных мелких покупателей вызывало определенные сложности. Мощные компьютеры, базы данных, быстродействующие и работающие в интерактивном режиме средства коммуникаций, интернет, сделали возможным в массовых масштабах создавать товары и услуги, разработанные на индивидуальной основе таким образом, чтобы удовлетворить требования каждого конкретного потребителя.

Концепция индивидуального маркетинга получила свое развитие в работе Д.Пайна, который в начале 90-х годов ввел термин “массовая индивидуализация” для описания явления, предсказанного в романе “Идеальное будущее” С.Дэвисом, а позже в книге “Виртуальная корпорация” В.Дэвидоу и М.Малоуном: «индивидуализированная продукция XIX века и массовая продукция XX века могут быть совмещены в продукции XXI века, где массовое производство будет сочетаться с индивидуализацией». Авторы предсказывали, что компания будет устанавливать базовые модули, которые могут быть индивидуально скомпонованы каждым покупателем. Такая технология позволяет клиенту принимать непосредственное участие в разработке продукции и услуг, и

фактически “арендовать” производство, логистику и другие ресурсы предприятия.

Говорят, что индивидуальный маркетинг направлен на «виртуального клиента» новой формации, который ведет себя как инопланетянин, попавший в условия земных рынков. Благодаря новым каналам получения информации, открытым с помощью современных технологий, он более информирован и требователен, чем раньше. У него совершенно другие ожидания, он по-другому строит отношения с компаниями, в которых приобретают продукцию и услуги. «Виртуальный клиент» изначально ожидает персональных предложений, – от продукции и услуг до информации и даже цены, которую он готов заплатить.

Инструментами индивидуального маркетинга в интернете являются механизмы персонализации, которые позволяют настроить веб-ресурс на нужды конкретного пользователя и в результате каждому посетителю предложить информацию и товары, соответствующие его индивидуальным вкусам и предпочтениям.

Персонализация – это специфическая форма дифференциации товаров, позволяющая переходить от стандартного продукта или услуги к решению, учитывающему вкусы и особенности каждого отдельного потребителя.

Можно выделить следующие основные функции персонализации:

1. *Демократизация товаров.* Персонифицированные услуги всегда были символом богатства и статуса, прерогативой избранных. Однако благодаря развитию технологий, уникальные услуги все шире становятся достоянием рядового потребителя. Интернет делает возможным открытый и дешевый доступ к персонифицированным услугам и советам в отношении очень широкого ассортимента продуктов и услуг с учетом разных уровней доходов потребителей. В этом ключе можно говорить о демократизации товаров. Важной стороной персонализации является и более эффективное потребление. Правильный совет помогает каждому

найти наилучший способ, как истратить свои деньги и в максимальной степени получить от этого удовлетворение.

2. *Превращение необычных товаров в привычные.* Не всегда потребитель может заранее указать важнейшие характеристики товаров и оценить все выгоды, которые он получит при их потреблении. Прежде всего, потому, что такие продукты и услуги могут обладать многими различными свойствами, быть слишком сложными для всесторонней оценки, требовать от потребителя определенных навыков и умений. Кроме того, товары могут восприниматься очень субъективно, так как личные пристрастия часто являются важным фактором в определении их полезности. Персонализация является эффективным инструментом превращения необычных товаров в привычные, давая, какую продукцию следует выбирать, а какую – избегать. Потребитель получает выгоду за счет снижения неопределенности в отношении необычных для себя товаров.

3. *Помощь потребителю при его выборе.* Огромное разнообразие потребительских товаров и услуг сильно затрудняет выбор при их приобретении. Ведь вместо прежнего относительно узкого диапазона коммерческих предложений потребитель сталкивается с сотнями и тысячами вариантов. Системы персонализации в интернете комбинируют проверенные приемы маркетинга с возможностями новой информационной среды. Они охватывают большой диапазон продукции, стараются определить индивидуальные вкусы и потребности и на основании этого выдать персональные рекомендации, подсказывающие, какой вариант для потребителя наиболее предпочтителен. Если такие рекомендации оказываются востребованными, они способствуют созданию дополнительной ценности и лояльности. Более того, оказание помощи потребителю при его выборе способствует тому, что потребитель точнее и самостоятельнее определяет свои собственные вкусы.

4. *Подгонка продукции на заказ.* Упрощенный выбор из обширного ассортимента является мощным маркетинговым инструментом, однако

полностью индивидуализированной продукции он не обеспечивает, так как число потребителей с разными вкусами очень велико. Хранение на складе большого числа вариантов для удовлетворения изначально неопределенно широкого спроса для производителей, дистрибьюторов и розничных торговцев слишком накладно. Расчетная и производственная гибкость современного бизнеса позволяет предложить другой подход. На первый план выходит подгонка продукции на заказ – комбинирование информации индивидуального уровня и гибкого дизайна продукта. Подгонка на заказ в интернете особенно наглядна. Используя специализированное программное обеспечение и учитывая полученный опыт, онлайновые службы предлагают своим потребителям уникальные и динамично персонализированные ресурсы, функционирующие в режиме реального времени.

Выделяют следующие виды подгонки продукции на заказ:

1. *Адаптивная подгонка на заказ* – это стандартное предложение со многими разновидностями, когда пользователь сам совершает выбор при помощи системы фильтров.

2. *Косметическая подгонка на* заказ – это изменение презентации продукта. Страница с разными заголовками выглядит уникально, при этом суть предложений не меняется.

3. *Прозрачная подгонка на заказ* производится “за кулисами”. Уникальные продукты и услуги доставляются без информирования потребителей об изменениях. Важной составляющей прозрачной подгонки является выявление индивидуальных потребностей и предпочтений, которое происходит без опрашивания потенциального потребителя. При этом, чтобы оперативно учесть динамику вкусов пользователя, продукция меняется автоматически. Таким образом работают, например, “умные” рекламные объявления.

4. *Совместная подгонка на заказ* – желанная цель многих маркетологов – это диалог с потребителями, помогающий им точнее

выразить собственные предпочтения и найти наиболее приемлемые коммерческие предложения.

Можно сделать вывод, что движение в сторону индивидуального маркетинга отражает тенденции к увеличению роли двустороннего диалога и развитию маркетингового самообслуживания. Выступая в роли индивидуальных заказчиков, потребители несут большую ответственность за то, какие товары и каких марок им покупать. Вместе с тем, они получают большие возможности для удовлетворения своих запросов.

Индивидуальный маркетинг является эффективным способом привлечь и удержать потребителей, преуспеть в рыночной борьбе в условиях быстрого затоваривания и острой ценовой конкуренции новой «виртуальной» экономики.

Обратимся к методами и приемам персонализации, применяемым в интернете.

Полуавтоматическая персонализация на основе настроек пользователя предполагает конфигурирование содержимого страницы пользователем вручную – обычно при помощи списков множественного выбора. Пользователь непосредственно указывает, какая информация ему интересна. Он может задать последовательность вывода фрагментов информации на экран или выбрать комфортную цветовую гамму. При повторных заходах на страницу система обращается к пользователю по имени и организует содержимое страницы в соответствие с его персональными настройками. Эта простейшая форма персонализации получила широкое распространение на интернет- порталах.

Полуавтоматическая персонализация на базе правил также включает в себя этап регистрации и авторизации пользователей, сбора сведений демографического характера и суждений пользователей о собственных интересах и предпочтениях с последующим применением к ним условных правил бизнес-логики, устанавливаемых продавцом в соответствие с текущей маркетинговой программой. Это могут быть

скидки и специальные коммерческие предложения, ориентированные на определенный сегмент покупателей, стимулирование продаж дополнительных продуктов из другой товарной категории (cross-selling) и более дорогостоящих продуктов и услуг (up-selling). Коммерческие системы персонализации – Broadvision, Vignette и др. – предлагают гибкие механизмы формулирования таких правил, например, эффективные инструменты визуализации и оперативного анализа данных.

Основным недостатком полуавтоматической персонализации является трудоемкая процедура настройки. По статистике, только каждый десятый пользователь знает о возможностях такой персонализации. Кроме того, суждения пользователя о собственных интересах зачастую субъективны и пристрастны. Статические профили быстро теряют свою актуальность, в то время как пользователи склонны менять свое представление о необходимой информации. Сложной задачей представляется разработка гибкого инструмента формирования профиля, способный собрать исчерпывающие сведения об особенностях каждого пользователя.

Развиваются методы автоматической персонализации, включающие в себя подходы на основе интеллектуального анализа данных, информационного поиска и совместной фильтрации. Большая работа в этом русле ведется в рамках современного направления исследований в области интеллектуальных пользовательских интерфейсов – адаптивные гипермедиа-системы.

Системы адаптивной гипермедиа формируют модель целей, предпочтений и знаний конкретного пользователя и используют ее в процессе взаимодействия с пользователем для адаптации к его предпочтениям и потребностям. [Брусиловский, 1996]

Пользователи с различными совокупностями целей и уровнями знаний могут быть заинтересованы в получении различной информации, представленной на сайте, и могут использовать различные ссылки для навигации. Системы адаптивной гипермедиа делает попытку преодолеть

эту проблему, используя данные, хранящиеся в модели пользователя, для адаптации информации и ссылок, предоставляемых пользователю. При этом система может выглядеть совершенно по-разному для пользователей с различными моделями.

Выделяют следующие основные цели адаптации:

1. Предоставление пользователю оптимального маршрута навигации (наиболее посещаемых пользователем или подходящих для него разделов и ресурсов, предметных областей и т.д.).
2. Обеспечение комфортного режима работы с системой.
3. Подстройка системы под уровень пользователя (чаще всего применяется для образовательных систем).
4. Реализация советующих функций и контекстной помощи.
5. Сохранение и применение по мере необходимости наборов действий пользователя – транзакций (например, в электронных магазинах).
6. Обратная связь (рассылки и оповещения, настроенные под конкретного пользователя).
7. Выявление общих трендов и моделей в поведении групп пользователей для настройки и оптимизации системы.
8. Пополнение наборов правил адаптации на основе анализа действий пользователя.

Архитектуру адаптивной информационной системы можно представить в виде трех составных частей:

- Модель системы, или модель предметной области, которая описывает, каким образом структурировано содержание приложения.
- Модель пользователя, которая представляет предпочтения, знания, цели, историю навигации и другие характеризующие аспекты пользователя.

– Механизмы адаптации, лежащие в основе процесса генерации адаптивного представления и обновления модели пользователя. [Дикарев, Целых, 2005]

К основным приемам адаптации визуального ряда и содержания информационных ресурсов относятся:

1. Изменение порядка следования информационных ресурсов в текущей предметной области.

2. Изменение навигационной схемы на основе предпочтений пользователя и его модели.

3. Формирование релевантных, то есть «интересных» пользователю, наборов ссылок на другие информационные ресурсы на основе различных схем адаптации: анализ множества ключевых слов в модели пользователя, истории его посещений, семантического пространства текущей темы.

4. Подсветка ссылок текущей предметной области. Подсветка другим цветом ссылок более высокого уровня и более низкого уровня.

5. Сокрытие ссылок, выпадающих за или выше текущего уровня пользователя или сложности ресурса.

6. Сокрытие информационных фрагментов и подсветка их по аналогичным критериям

7. Адаптация интерфейса. Выбор пользователем стиля, интересующих модулей и интерфейсных решений.

8. Всплывающие подсказки на ключевых словах – основных понятиях (концептах) ресурса.

Дадим формальную постановку задачи персонализации. [Целых, 2005]

Пусть $U = \{url_1, url_2, ..., url_{N_u}\}$ множество из N_U страниц веб-ресурса, посещенных в рамках пользовательских сессий s_j, $j = 1, 2, ..., N_s$, где N_S – общее число сессий для всех пользователей. $P = \{p_1, p_2, ..., p_{N_p}\}$ – множество из N_P профилей, каждый из которых представлен в виде вектора длины

N_U, состоящего из p_{ik} – веса страницы url_k в i-м профиле. Если страница url_k отсутствует в i-м профиле, p_{ik} принимает нулевое значение.

На основе вектора текущей пользовательской сессии $s_j = [s_{j1}, s_{j2}, ..., s_{jN_u}]$ требуется предсказать множество страниц, отвечающих потребностям пользователя. При этом рекомендации для s_j также удобно представить в виде вектора $r_j = [r_{j1}, r_{j2}, ..., r_{jN_u}]$, где r_{jk} – оценка релевантности для каждой страницы url_k.

Тривиальный подход к выдаче рекомендаций в системах персонализации заключается в нахождении профиля, содержащего страницы из текущей пользовательской сессии. При этом пользователю можно рекомендовать посетить остальные страницы, входящие в данный профиль. Сами рекомендации ранжируются по весу каждой из страниц в профиле.

Однако, такой подход имеет ряд недостатков. В частности, при ранжировании рекомендаций не учитывается степень близости между текущей сессией и ближайшим профилем. Кроме того, затруднена выдача рекомендаций для пользовательских сессий, релевантных двум и более профилям, либо отличных от всех установленных профилей.

Рассмотрим альтернативный подход на основе нечеткой логики и применения правила нечеткого вывода.

В работе [Заде, 1976] предложен подход, позволяющий, если известны некоторые нечеткие понятия $\widetilde{A}, \widetilde{B}, \widetilde{A}'$, сделать вывод о значении следствия $\widetilde{B}' = \widetilde{A}' \circ (\widetilde{A} \to \widetilde{B})$.

Определим нечеткое отношение R на множестве $X \times Y$, где X – множество профилей, Y – множество страниц, а R устанавливает соответствие между профилями в X и страницами в Y с различными степенями релевантности. Другими словами, строки матрицы R задаются векторами профилей с релевантными весами каждой страницы в профиле.

На входе процедуры вывода имеем нечеткое подмножество $A' \subseteq X$, которое задает степени принадлежности текущей сессии к каждому из

профилей X. На выходе – нечеткое подмножество $B' \subseteq Y$, представляющее вероятность того, что страница веб-ресурса релевантна текущему профилю, а значит, в определенной степени, отвечает интересам и потребностям пользователя.

Для вычисления степени принадлежности текущей сессии s_j к $i-$му профилю p_i будем использовать оценку близости:

$$\mu_S(i) = sim(s_j, p_i).$$

В качестве такой оценки может применяться косинусовая мера близости:

$$\mu_{si}^{\cos} = \frac{\sum_{k=1}^{N_U} p_{ik} s_k}{\sqrt{\sum_{k=1}^{N_U} p_{ik} \sum_{k=1}^{N_U} s_k}}.$$

Если нужно учесть особенности иерархической структуры веб-ресурса, можно произвести следующую модификацию косинусовой меры близости:

$$\mu_{si}^{web} = \max\left\{ \frac{\sum_{i=1}^{N_U} \sum_{k=1}^{N_U} p_{il} S_u(l,k) s_k}{\sum_{k=1}^{N_U} p_{ik} \sum_{k=1}^{N_U} s_k}, \mu_{si}^{\cos} \right\},$$

где S_U – матрица близости страниц, вычисляемая на основе величины пересечения путей, ведущих от корня веб-ресурса к каждой из страниц рассматриваемой пары

$$S_u(i,j) = \min\left(1, \frac{\left|p_i \cap p_j\right|}{\max\left(1, \max\left(\left|p_i\right|, \left|p_j\right|\right) - 1\right)}\right).$$

Тогда вычисление рекомендаций будет осуществляться по формуле:

$$\mu_{rk} = \mu_r(URL_k) = \mu_S(i) \circ R = \max_i[\min(\mu_S(i), R_{ik})].$$

С целью оценки эффективности выдаваемых рекомендаций может привлекаться мнение эксперта, субъективно трактующего их полезность.

Можно также ввести количественную оценку эффективности рекомендаций на основе методов информационного поиска.

Будем рассматривать завершенную сессию (транзакцию) s_T в качестве целевой, а подмножество этой сессии s_j в качестве неполной сессии. Другими словами, рекомендации предсказывают транзакцию.

Пусть $r_j^* = r_j - s_j$ – набор страниц веб-ресурса в рекомендации, за исключением тех, что уже входят в текущую подсессию, а $s_j^* = s_T - s_j$ – набор страниц в транзакции, кроме тех, что уже входят в текущую подсессию.

Тогда оценка точности рекомендаций определяется выражением:

$$prec_j = \frac{\sum_k^{N_U} r_{jk}^* s_{jk}^*}{\sum_k^{N_U} \left(r_{jk}^*\right)^2}.$$

Оценка полноты определяется выражением:

$$\text{cov}_j = \frac{\sum_k^{N_U} r_{jk}^* s_{jk}^*}{\sum_k^{N_U} \left(s_{jk}^*\right)^2}.$$

Выбирая различные меры близости при вычислении функций принадлежности $\mu_S(i)$ и различные *t*-нормы и *t*-конормы при определении операции композиции, используя пороговые величины (α – уровневые множества) при построении матрицы отношений *R* и итогового множества рекомендаций, мы будем получать различные значения оценок точности и полноты.

1.4. Информационный поиск в интернете

Поисковые системы предоставили человеку неограниченный и практически мгновенный доступ к гигантским массивам информации. На сегодня в мире известны сотни поисковых систем и тысячи разновидностей поисковых функций, реализованных в самых разных программах. Однако как бы ни был реализован процесс поиска, на

какую бы математическую модель он не опирался, основные идеи и программы, реализующие поиск, достаточно просты.

Известны четыре класса поисковых алгоритмов. [Сегалович, 2002] Три из них требуют «индексирования», то есть предварительной обработки документов, при которой создается вспомогательный файл, призванный упростить и ускорить поиск. Это алгоритмы инвертированных файлов, суффиксных деревьев, сигнатур. В четвертом случае предварительный этап индексирования отсутствует, а поиск происходит при помощи последовательного просмотра документов. Такой поиск иначе называется прямым.

Инвертированные файлы (конкордансы) – это упорядоченные по алфавиту исчерпывающие списки слов из одного текста (расширенный глоссарий), в которых для каждого слова перечислены все позиции (главы и страницы), на которых встречается слово. Поисковый алгоритм, по сути, состоит в отыскании нужного слова и загрузке в память развернутого списка позиций. Чтобы сэкономить на дисковом пространстве и ускорить поиск, прибегают к приемам уменьшения размера инвертированного файла. В самом подробном варианте в инвертированном файле может храниться и номер слова, и смещение в байтах от начала текста, и цвет, и размер шрифта. Но чаще просто указывают только номер документа, скажем, книгу Библии, и число употреблений этого слова в нем. Именно такая упрощенная структура считается основной в классической теории информационного поиска. Размер инвертированного файла, как правило, составляет от 7 до 30 процентов от размера исходного текста. У прямых алгоритмов, работающих непосредственно по оригинальным документам безо всяких искажений, есть принципиально беспроигрышные отличительные черты. Например, неограниченные возможности по организации приближенного и нечеткого поиска. Ведь любое индексирование всегда сопряжено с упрощением и нормализацией терминов, а, следовательно, с потерей информации.

В современных подходах каждый документ обычно характеризуется множеством ключевых слов – термов, семантика которых позволяет описать его основное содержание. Аналогичным образом представляются поисковые запросы или профили пользователей.

Когда речь заходит о повышении качества поиска, о большом объеме информации, о потоке пользовательских запросов, оказывается полезным оперировать каким-нибудь, пусть и несложным теоретическим аппаратом, ввести математическую модель.

Выделяют три класса моделей информационного поиска:

– теоретико-множественные (булевская, расширенная булевская, нечетких множеств);
– алгебраические (векторная, обобщенная векторная, латентно-семантическая, нейросетевая)
– вероятностные.

В простейших булевских моделях документ считается найденным, если он содержит терм из профиля пользователя. Есть слово – документ найден, нет слова – документ не найден.

Жесткость таких моделей и непригодность для ранжирования обусловила появление векторной модели, которая позволяет ранжировать результаты поиска на основе статистического наблюдения, что, чем выше локальная частота термина в документе (TF) и чем выше «редкость» (т.е. обратная встречаемость в документах) термина в коллекции (IDF), тем больше вес данного документа по отношению к термину. Обозначение TF*IDF широко используется как синоним векторной модели. В векторной модели близость документа к запросу оценивается как корреляция между векторами описаний, например, их скалярное произведение. Вес термов вычисляется на основе нормализованной частоты использования, дискриминационной силы терма и других подходов.

Релевантность в вероятностной модели информационного поиска рассматривается как вероятность того, что данный документ может оказаться интересным пользователю. При этом подразумевается наличие уже существующего первоначального набора релевантных документов, непосредственно выбранных пользователем либо полученных автоматически при упрощенном предположении. Вероятность оказаться релевантным для каждого следующего документа рассчитывается на основании соотношения встречаемости терминов в релевантном (уже существующем) наборе и в остальной, “нерелевантной” части коллекции. Вероятностные модели обладают некоторым теоретическим преимуществом, располагая документы в порядке убывания вероятности оказаться релевантным.

Рассмотренные нами простейшие модели информационного поиска каждого из семейств исходят из предположения о взаимонезависимости слов и обладают условием фильтрации – документы, не содержащие слов запроса, никогда не бывают найденными.

Альтернативные модели позволяют находить и ранжировать документы, близкие по смыслу, при этом они могут и не содержать слов из запроса. Именно эту особенность моделей зачастую считают признаком искусственного интеллекта.

Пожалуй, самой распространенной моделью этого типа является латентно-семантический анализ. Эта алгебраическая модель основана на сингулярном разложении прямоугольной матрицы, ассоциирующей слова с документами. Элементами матрицы выступают частотные характеристики, отражающие степень связи слова и документа, например, *TF*IDF*. Причем, вместо исходной большой матрицы рассматриваются обычно от 50 до 300 “скрытых смыслов”, соответствующих первым главным компонентам сингулярного разложения матрицы.

Операции поиска или нахождения похожих документов резко упрощаются, так как каждому слову и каждому документу сопоставляется относительно короткий вектор из k смыслов, представленный строками и столбцами соответствующих матриц. Однако, если во вспомогательных целях, включающих автоматическую фильтрацию, классификацию, разделение коллекций и предварительное снижение размерности для использования других моделей этот метод находит широкое применение, по причине малой осмысленности “скрытых смыслов” использование лантентно-семантического анализа не получило самого широкого распространения.

В последнее время получили развитие теоретико-графовые подходы к семантическому поиску.

В семантической сети знания представляются в виде ориентированного графа, узлы которого соответствуют фактам или понятиям. В приложении к задаче информационного поиска – это слова и словосочетания. Дуги соответствуют отношениям или ассоциациям между ними. При построении семантической сети текста в нее включаются наиболее часто встречающиеся слова, которые несут основную смысловую нагрузку. Для каждого понятия формируется набор ассоциативных (смысловых) связей, то есть список других понятий, в сочетании с которыми оно встречалось в тексте.

Когда информация о документе сохранена в виде такой семантичексой сети, запрос на поиск информации может быть выражен в терминах нечеткой логики. При этом задача поиска заключается в нахождении нечеткого вхождения семантической сети знаний пользователя в семантическую сеть документа. Другими словами, запрос на поиск тоже формулируется в виде графа. Степень релевантности может определяться степенью изоморфизма графов, то есть степенью нечеткого вхождения одной графовой структуры в другую.

Среди методов оценки изоморфизма и изоморфного вложения нечетких графов выделяют подходы на основе формул сходства нечетких множеств и на основе нечетких инвариантов нечетких графов.

В основе первого подхода лежит использование формул сходства нечетких множеств, наиболее распространенными из которых являются сходство по Заде, сходство по Лукасевичу-Хэммингу, сходство по Дейку, сходство по Танимото. Исследуя нечеткое отображение (соответствие) между множествами вершин сравниваемых графов, можно вычислить степень включения (сходства) для соответствующих вершин и ребер. При этом степень включения (сходства) для всего графа определяется как среднее арифметическое либо как минимальное значение из найденных. Если отображение заранее не задано, осуществляется перебор вариантов установления соответствий и проверка для каждого случая степени *t*-вложения. Для сокращения перебора используются оптимизации и отсечения.

Другой подход заключается в исследовании нечетких инвариантов нечетких графов: нечетких множеств внешней и внутренней устойчивости, нечетких ядер, нечетких клик, нечетких баз и антибаз, нечетких сильных компонент, нечетких хроматических множеств. [Берштейн, Боженюк, 2005]

Со временем стало очевидно, что поиск в интернете не может быть корректно выполнен, будучи основанным на анализе одного лишь текста документа. Большую роль играют внетекстовые (off-page) факторы: расположение информации на сайте, посещаемость и авторитетность источника, частота обновления, цитируемость страницы и ее авторов – все это нельзя сбрасывать со счета.

Многие из проблем, с которыми столкнулись разработчики поисковых систем в интернете, никогда прежде не рассматривались в традиционной науке информационного поиска:

1. Искусственная генерация входных страниц, насыщенных популярными словами, техника клоакинга (поисковый спам) и многие другие приемы, предназначенные для обмана поисковых систем.

2. Проблема корректного ранжирования (качество ранжирования).

Не все внетекстовые критерии полезны в равной мере. Ключевым фактором с 1999 года является ссылочная популярность и производные от нее. С их помощью поисковые системы научились самостоятельно ранжировать ответы на короткие частотные запросы, составляющие значительную часть поискового потока.

Простейшая идея статического учета ссылочной популярности состоит в подсчете числа ссылок, указывающих на страницы. Примерно то, что в традиционном библиотековедении называют индексом цитирования. Этот критерий использовался в поисковых системах еще до 1998 года. Однако он легко подвергается накрутке, кроме того, он не учитывает вес источников. Естественным развитием этой идеи можно считать предложенный Брином и Пейджем (создателями Гугла) в 1998 году алгоритм PageRank – итеративный алгоритм, подобный тому, что используется в задаче определения победителя в шахматной турнире по швейцарской системе. В сочетании с поиском по лексике ссылок, указывающих на страницу, эта мера позволила резко повысить качество поиска.

Кроме решения проблемы качества ранжирования, создателям поисковых систем в интернете пришлось решать задачу обновления и синхронизации колоссальной по размеру коллекции с гетерогенными форматами, способами доставки, языками, кодировками, массой бессодержательных и дублирующихся текстов (качество индекса).

По статистике, «редкие» запросы, то есть те, по которым находится менее 100 документов, составляют в сумме около 30% от всей массы поисков – весьма значительную часть. Это делает размер базы одним из самых критичных параметров системы.

Происхождение копий документов в Интернете может быть различным. Один и тот же документ на одном и том же сервере может отличаться по техническим причинам: быть представлен в разных кодировках и форматах; может содержать динамические элементы – рекламу или дату. Широкий класс документов в вебе активно копируется и редактируется – ленты новостных агентств, документация и юридические документы, прейскуранты магазинов, ответы на часто задаваемые вопросы и т.д. Необходимо адекватно реагировать на мусор, повторы и т.п.

С этой целью были разработаны методы, позволяющие исключать как полные повторы, так и незначительно измененные документы, «почти-дубликаты». Например, алгоритм шинглов (чешуек). Для каждого десятисловия текста рассчитывается контрольная сумма (шингл). Причем, десятисловия идут внахлест, с перекрытием. Затем из всего множества контрольных сумм отбираются только те, которые, скажем, делятся на 25. Один совпавший шингл в выборке соответствует примерно 25 совпавшим десятисловиям в полном тексте. Очевидно, что так можно определять процент перекрытия текстов, выявлять все его источники и т.д.

Этот изящный алгоритм воплотил давнюю мечту преподавателей: с его помощью можно легко вычислить, «у кого студент списывал», и даже оценить степень плагиата.

1.5. Грид-технологии

По словам ученого Макса Гоффа, «системы сетевых распределенных вычислений обуславливают непрекращающуюся техническую революцию, которая уже продила интернет и вскоре превратит мир в одно вездесущее, всепроникающее информационное поле».

И действительно, поскольку типичный пользователь расходует лишь малую часть вычислительных ресурсов настольного компьютера, часть

тактов центрального процессора можно заимствовать для выполнения программы без ущерба для повседневной работы.

Можно проследить сходство вычислительных массивов с электросетями энергетических компаний (power grid). Включая свет, мы получаем энергию не обязательно от ближайшей электростанции. Мы подключаемся к региональной сети станций. Если станция, от которой обычно питаются наши электроприборы, не работает, то электричество подается от другой станции. Подобным образом вычислительный массив обеспечивает доступ к сетевым вычислительным ресурсам.

Еще в 1965 году ученые университета MIT, разрабатывая операционную систему Multics, предвидели некую компьютерную организацию, функционирующую как предприятие по электро- и водоснабжению.

В 1998 году американский ученый Иен Фостер предложил термин «грид-вычисления» для описания множества компьютеров, соединенных в сеть с целью проведения вычислительных операций, требующих колоссальных мощностей. Совокупная мощность такой развитой сети вычислений может значительно превышать производительность самых мощных суперкомпьютеров в мире. [Foster, 2002]

Грид-вычисления – это инфраструктура «железа» и программного обеспечения для кластеризации и интеграции высокопроизводительных компьютеров, сетей, баз данных и научных инструментов из разных источников с целью организации виртуального суперкомпьютера, с которым могут совместно работать пользователи.

Объединив в массив группу машин, можно заимствовать их свободные или мало используемые ресурсы, независимо от изготовителя, модели, аппаратной конфигурации, а иногда – операционной системы каждого из компьютеров. В состав массива могут входить специализированне клиентские ПК, рабочие станции сотрудников, серверы и большие ЭВМ. Таким образом, идет речь о распределении гетерогенных ресурсов и виртуализации компьютерных ресурсов.

Можно выполнять задачи только на одной машине или начать работу на одном, а затем переместиться на другой компьютер, на котором высвободились системные ресурсы. Задачи, которые удается разделить на отдельные фрагменты, можно запускать на нескольких машинах.

Технические подробности скрыты от конечных пользователей, им известно лишь то, что необходимые вычислительные ресурсы выделены. Административный компонент более сложен. Выбрав ЛВС или территориально-распределенную сеть для вычислительного массива, необходимо организовать на серверах центральную консоль управления. Затем следует выбрать машины-доноры ресурсов и загрузить на них небольшие программы-агенты для связи с центральной консолью. Центральная консоль обеспечивает общее управление вычислительным массивом, давая поручения агентам, размещенным в сети. С консоли можно передать исполняемый модуль приложения для запуска на удаленной машине, запустить уже установленную на ПК программу или извлечь нужный фрагмент данных.

Грид-система характеризуется тремя следующими характеристиками:

– координирует ресурсы, которые не администрируются централизованно;
– использует открытые стандарты, протоколы и интерфейсы;
– оперирует нетривиальными качествами услуг, такими как: время отклика, пропускная способность, доступность и безопасность.

Вычисления в грид-массивах не следует путать с одноранговой (peer-to-peer) технологией, которая позволяет использовать множество машин, объединенных в сеть. Важное отличие – программы P2P часто одновременно выполняют мелкие, не связанные между собой задачи, и клиенты в разных услах сети работают независимо друг от друга. Вычислительный массив обычно предназначен для выполнения единой всеобъемлющей задачи, и одна центральная машина управляет всеми клиентами, работающими над этим общим заданием.

В отличие от кластерных вычислений, когда вычислительный массив обычно территориально находится в одном месте, грид состоит из множества удаленных кластеров и других видов ресурсов (сетей, сервисов хранения данных и т.д.).

Вычислительные грид-массивы не подходят для решения простых, малых задач. Для организации массива и связи с отдаленными машинами требуется потратить некоторое время. Вычислительные массивы наиболее эффективны для программ, которые на одной машине не работают или выполняются слишком медленно. Вместе с тем, не всякую крупную программу можно ускорить с помощью вычислительных массивов.

Можно выделить две общие черты, характерные для всех грид-приложений. Во-первых, в каждом из них производится много последовательных вычислений над огромными наборами данных. Для такого гигантского числа операций требуется больше вычислительных ресурсов, чем может предоставить ода рабочая станция за разумный период времени. Во-вторых, каждая вычислительная операция выполняется не над всем набором, а лишь над небольшой его частью. При этом можно легко разослать данные по компьютерам массива и параллельно обрабатывать их на нескольких машинах.

Массивы используются не только для наращивания вычислительной мощности. Через массив можно заимствовать данные, ресурсы памяти и пропускную способность сетевых каналов связи.

Такие инструменты, как Entropia DCGrid и United Devices MetaProcessor, позволяют выполнять сложные манипуляции с вычислительными массивами. После того, как поставлена задача, они отыскивают доступные машины, а затем координируют обмен данными между центральной консолью и клиентскими агентами. Для защиты и обеспечения целостности данных используются шифрование и алгоритмы коррекции данных. Функционирование пакетов не замедляет нормальной работы клиентской машины, поскольку каждое задание помещается в

виртуальную «песочницу», изолированную от других программ. Если машина отключается от сети или выходит из строя, задание перенаправляется на другой компьютер.

Наиболее известные вычислительные проекты используют простаивающие мощности обычных компьютеров. Как правило, это программа-скринсейвер, которая включается, когда компьютер долго стоит без дела. Программа обрабатывает фрагменты данных, поставляемых с центрального сервера проекта, и отсылает результаты обратно. Обычно такие проекты направлены на решение одной конкретно взятой задачи. Принятие участие в проекте может любой пользователь, подключенный к Интернету, установив на своем компьютере клиентскую программу. Стимулом участия в проекте может быть денежный приз или соавторство при написании научных работ, в том случае, если машина пользователя вычислит новый или интересный, с точки зрения проекта, результат. Многие проекты позволяют участникам объединяться в команды и участвовать в командных зачетах. Общие результаты проекта зачастую публикуются в открытом доступе.

Примеры вычислительных проектов:

– SETI@home (SETI = the Search for ExtraTerrestrial Intelligence) – пожалуй, один из самых известных проектов распределенных вычислений в Интернете. Клиентская программа получает пакет данных, полученных с радиотелескопа, и анализирует их на предмет наличия «разумных сигналов», т.е. сигналов, имеющих не хаотическую структуру.

– Find-a-Drug – проект, направленный на поиск лекарств от рака, СПИДа, атипичной пневмонии и других заболеваний, а также на создание безопасных гербицидов и пестицидов. Клиентская часть забирает с сервера список молекул и анализирует их влияние на белковые соединения.

– Distributed.net RC5 – нацелен на доказательство ненадежности алгоритма шифрования RC5 путем "взлома" зашифрованного с помощью него сообщения.

– Climateprediction.net – имеет целью выяснить точность современных климатических моделей и установить поправки, которые должны быть в них внесены. Между собой сравниваются результаты запущенных климатических моделей со слегка различающимися начальными данными.

– Fermat Search – проект российских ученых по поиску делителей для чисел Ферма - чисел вида $F_m = 2^{2m} + 1$.

– Lifemapper – ставит цель составить электронный атлас живой природы Земли. Определяются текущие места обитания животных и растений, территории, на которых они могли бы обитать, пути распространения биологических видов.

На принципе использования простаивающих мощностей компьютеров основана программа IBM World Community Grid. Корпорация Oracle присвоила звание Grid своей СУБД Oracle 10g, хотя речь идет всего лишь о распределении единой базы данных. Sun Microsystems продвигает идею глобальной вычислительной сети, за использование ресурсов которой люди будут платить так же, как за пользование электрическими сетями. Hewlett-Packard планирует использовать принципы распределенных вычислений в рамках своей программы Adaptive Effort, которая должна повысить гибкость корпоративных вычислительных сетей.

«Отец» грида Иен Фостер стоял у истоков первой коммерческой компании Univa, предоставляющей услуги и техническую поддержку проектов, связанных с распределенными вычислениями. В качестве технологической платформы используется Glovus Toolkit, программа с открытым исходным кодом, которая используется в большом числе исследовательских проектов, а также крупными коммерческими компаниями, такими как IBM и SAP.

Рынку коммерческих распределенных вычислений прочат дальнейший рост. Компьютерный анализ наиболее востребован в таких сложных областях, как телекоммуникации, финансовые услуги и

транспорт, где объем вычислений может быть очень велик, а результаты требуются быстро.

1.6. Социальные сети и социальный интеллект

Очень часто распределенные системы, в том числе сети сотовой связи, компьютерные сети и Мировая Паутина обладают сложной топологией и имеют в своей основе социальные процессы.

Последние годы растет интерес к теории социальных сетей. Ряд исследований показал, что такие системы представляют собой результат действия самоорганизующихся процессов, управляемых общими, довольно простыми законами. Причем, такие системы, состоящие из большого числа взаимодействующих элементов, могут демонстрировать интересные и весьма неожиданные временные и пространственные эффекты. Так, многие из них довольно устойчивы к ошибкам и внешним воздействиям (например, хакерским атакам). Несмотря на то, что ключевые элементы время от времени срабатывают неправильно, локальная поломка на деле редко ведет к потере глобальной способности сети эффективно работать с информацией. [Дмитриев, 2001]

Для изучения подобных явлений в сложных сетях необходим адекватный теоретический аппарат.

Одним из интересных подходов к анализу таких объектов является теория малого мира (small world). Говоря метафорически, обитателей такого "мира" отделяют друг от друга несколько рукопожатий.

Население земного шара превысило 6 миллиардов человек. Тем не менее, несмотря на огромное число людей на планете, структура социальных сетей такова, что все мы очень тесно связаны друг с другом. Оказывается, два случайно выбранных человека, как правило, связаны весьма короткой цепочкой промежуточных знакомств.

Экспериментально вычисленная длина такой цепочки неизменна с 1967 года (по результатам почтового исследования Стэнли Милграма) и составляет 6 звеньев.

Позже эффект малого мира изучался в разных контекстах - в частности, при прогнозировании возможных эпидемий. Если инфекция передается по воздуху, то все 150 пассажиров самолета, летящего из Сингапура в Париж, связаны в малый мир. Если у одного из них грипп, заражены будут многие. А вот дальнейшее распространение болезни зависит от того, как устроено общество. Если в нем типичны кластеры из 100-200 тесно общающихся между собой людей и эти кластеры пересекаются друг с другом - получится стремительный волнообразный процесс массового заражения. Если же кластеры очень маленькие - скажем, семья из нескольких человек, - массовой эпидемии не будет.

Сетям социальных связей, имеющим структуру малого мира, характерны технологии "массовой мобилизации". Если в такие сети "вбросить" яркие, мобилизующие образы, они будут распространяться там, как эпидемия. Примеры таких образов и сетей: события 9 сентября в Нью-Йорке и волнения во Франции. Первое имело мобилизующее влияние в ряде стран мусульманского мира, второе – в локальных сообществах (neighborhood), потому что сети для распространения подобных образов были уже готовы: малые миры прихожан мечетей; обширные на Ближнем Востоке сообщества людей, связанных отношениями в торговле.

Для Европы в целом нехарактерно существование малых миров большого размера (порядка ста человек). В Швеции, например, вообще не образуются локальные коммьюнити такой численности. А вот иммигранты образуют малые миры из нескольких сот человек, постоянно общающихся друг с другом. Нечто вроде колонии эмигрантов из СССР на Брайтон-Бич в Нью-Йорке, - почти замкнутая среда, где многие по-прежнему говорят только по-русски.

Можно ли создать такие мобилизующие сети искусственно? Теоретически - да, но нужно не просто перезнакомить людей друг с другом - нужно их чем-то связать. Связи в малых мирах фундаментально неидеологические. Это миры совместного проживания, миры повседневности. Группа мигрантов во враждебном окружении. Люди, работающие на одном предприятии и каждый вечер пьющие вместе пиво. Таким миром может быть даже двор в городском доме (но сейчас дворов почти не осталось). Поэтому сильными являются сообщества, связанные с религиозными организациями и с университетской средой.

Например, в интернете получило развите такое явление как флешмоб. Флешмоб – это акция, организованная через интернет или иные современные средства коммуникации, в которой большая группа людей внезапно собирается в общественном месте, в течение нескольких минут выполняет заранее оговоренные действия, и затем быстро расходится.

Особенностью флешмоба являются: одновременное начало и окончание акции ее участниками, точное следование сценарию; нескопление на мобплейсе до и после акции, кажущаяся спонтанность действия.

Не существует единого мнения по поводу того, каковы цели проведения флешмобов. Участники одного и того же мероприятия могут преследовать различные цели. Среди возможных вариантов – развлечение, нарушение обыденного хода жизни, произведение впечатления на окружающих, ощущение причастности к общему делу, самоутверждение.

Экспериментально изучались и свойства малого мира в интернете на примере гиперссылок между документами во Всемирной Паутине. В 2000 году в момент проведения исследований Всемирная Паутина насчитывала ~ 8•108 документов. Средняя длина цепочки («расстояние» между документами) составляла около 19 переходов.

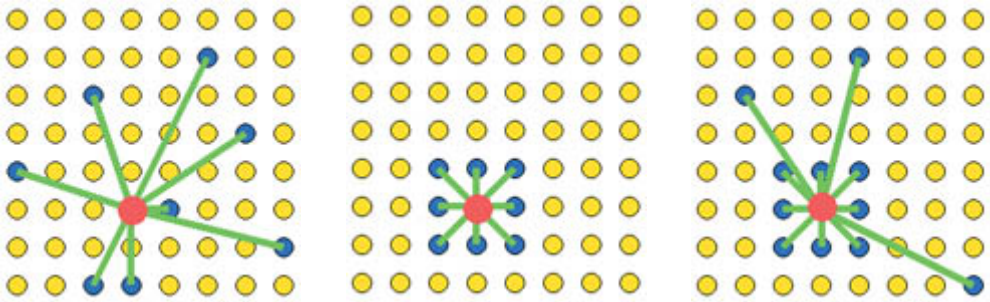

Рис.3.3. Модели локальных и «дальних» связей

Первой попыткой математического объяснения эффекта малого мира было использование модели случайного графа (модель Эрдоса-Реньи).

Предположим, что имеется сообщество, состоящее из N членов. В среднем каждый из них имеет z знакомств (рис. 3.3а). Это означает, что во всем сообществе имеется Nz/2 контактов между людьми. Модель предполагает, что эти связи устанавливаются между случайно выбранными парами. (Имея N вершин, мы соединяем каждые 2 из них с вероятностью p.) Оказалось, что такая модель действительно демонстрирует эффект малого мира. Средняя длина D цепочки, соединяющей двух случайных людей, согласно этой модели равна D = lg N/lg z. Поскольку lg N медленно увеличивается с ростом N, то D также мало даже для очень больших систем.

Однако модель случайных графов не определяет всех свойств реальных сетей. В частности, она не предсказывает кластеризацию в сети. В случайном графе вероятность того, что две персоны будут знакомы между собой, не зависит от того, какие персоны выбраны. С другой стороны, во многих сетях существуют кластеры пользователей (группы связанных между собой элементов).

В модель можно ввести коэффициент кластеризации C, представляющий собой среднюю долю таких пар в общем числе пар соседей узла, которые также являются соседями друг друга. В полностью связанной сети, в которой каждый знает каждого, C = 1. В случайном

графе C ~ 1/N, что является очень малой величиной при очень большом размере сети. В реальных сетях значение C хотя и много меньше единицы, однако существенно превосходит 1/N.

Противоположностью случайным графам в некотором смысле являются решетки с полностью упорядоченными связями каждого элемента с некоторым числом соседей (рис. 3.3б). Легко видеть, что часть непосредственных соседей каждого элемента всегда связана друг с другом, благодаря чему обеспечиваются кластерные свойства сети. Однако такая сеть не обладает эффектом малого мира - из-за отсутствия «дальних» связей.

Модель, узлы которой имеют одновременно некоторое количество локальных и случайных «дальних» связей (рис. 3.3в), демонстрирует и эффект малого мира, и кластеризацию.

Совместная фильтрация. Широкое распространение при решении задач персонализации получили методы совместной (общей, социальной) фильтрации. В их основе лежит идея автоматизации процесса обмена устными рекомендациями, – одного из ключевых факторов, влияющих на принятие решений о выборе и покупках.

Классические методы совместной фильтрации предлагают соотносить каждого пользователя с пользовательской группой с аналогичными предпочтениями с целью выдачи рекомендаций по объектам, на данный момент не имеющим оценки этого пользователя.

В общем случае эта задача представляет собой частный случай задачи добычи данных. Спецификой подхода к ее решению является предварительное построение модели пользовательских оценок, для чего, как правило, используя алгоритмы машинного обучения на основе байесовских сетей, кластеризации, индукции ассоциативных правил и др.

Рассмотрим метод хортинга, оригинального теоретико-графового подхода к совместной фильтрации. [Aggarwal и др., 1999]

Пусть пользователь i оценил некоторое подмножество объектов R_i, причем каждому объекту j из R_i дал оценку $r_{i,j}$. Для двух пользователей i_1 и i_2 подмножество объектов, получивших оценку обоих пользователей, определяется выражением $C(i_1,i_2) = R_{i_1} \cap R_{i_2}$. Предстоит выяснить, имеется ли достаточное число объектов с оценкой обоих пользователей, чтобы можно было предсказать оценку одного пользователя по оценке другого пользователя. Если это справедливо, говорят, что пользователь i_1 *хортингует* пользователя i_2.

По оценке пользователя i_2 можно предсказать оценку пользователя i_1 для данного объекта, если

$$\frac{|C(i_1,i_2|}{|R_{i_1}|} \geq F \text{, либо } C(i_1,i_2) \geq G,$$

где $F \leq 1$ и G – установленные пороги.

Заметим, что отношение хортинга обладает свойствами рефлексивности, несимметричныести и нетранзитивности.

Предсказание оценки подразумевает поиск соответствия между оценками пользователей i_2 и i_1.

Пусть существует линейное преобразование $T_{s,t}(r) = sr + t$, где $s \in \{-1;+1\}$, $t \in \{t_{s,0} \ldots t_{s,1}\}$.

Определим расстояние “по Манхэттэну” между оценками пользователя i_1 и i_2 как

$$D_{i_1,i_2}(s,t) = \sum_{j \in C(i_1,i_2)} | r_{i_1,j} - T_{s,t}(r_{i_2,j}) |.$$

Говорят, что пользователь i_2 предсказывает оценку пользователя i_1, если i_1 хортингует i_2 и выполняется следующее условие

$$s \in \{-2v+1,\ldots,2v-1\},\ \frac{D_{i_1,i_2}(s,t)}{|C(i_1,i_2)|} < U,$$

где U - заранее установленный порог, показывающий, какая разница в оценках допустима.

Выбор s и t, удовлетворяющих условию, дает требуемое нам соответствие.

В случае, когда $s=1$, $t=0$, поведение пользователя i_2 близко к поведению пользователя i_1. В ситуации, когда $D_{s,t}=0$ для $s=1$, $t=0$, пользователь i_2 ведет себя в точности как пользователь i_1. Если $D_{s,t}=0$ для $s=1$, $t=1$, пользователь i_1 более экспансивен и склонен выставлять оценку на 1 балл выше. С другой стороны, если $D_{s,t}=0$ для $s=1$, $t=-1$, более экспансивен пользователь i_2. Наконец, если $D_{s,t}=0$ для $s=-1$, $t=v+1$, поведение пользователя i_2 полностью обратно поведению пользователя i_1. Однако, во всех описанных случаях пользователь i_2 надежно предсказывает поведение пользователя i_1.

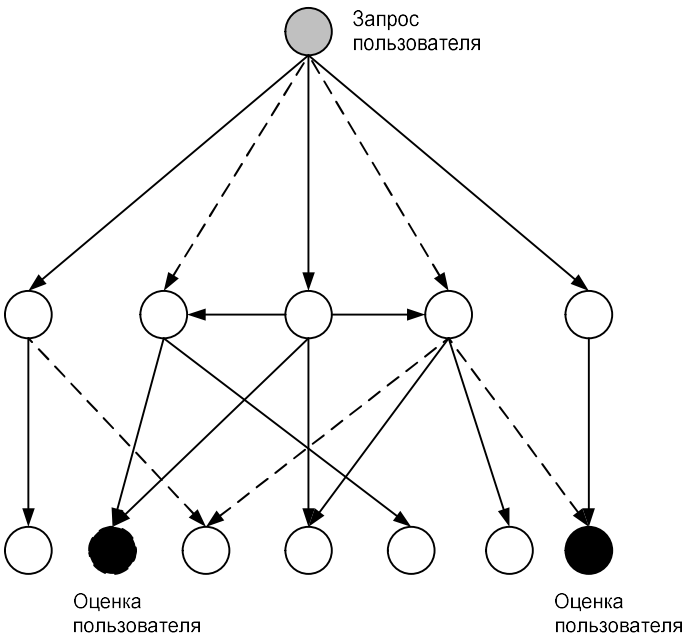

Рис.3.4. Метод хортинга

Данный алгоритм удобно реализовать на ориентированном графе (рис. 3.4). Вершины ориентированного графа представляют пользователей, а ребра между ними – *предсказуемость* как степень близости между двумя пользователями. Для того чтобы вычислить

оценку конкретного объекта для пользователя i_n нужно найти кратчайший путь из вершины i_n к вершине i_m , представляющей пользователя, который уже дал оценку этому объекту. Когда путь найден, по значениям *s* и *t* для каждой дуги графа обратным проходом вычисляется значение оценки для пользователя i_n.

В отличие от многих других подходов к совместной фильтрации, хортинг исследует транзитивные отношения: граф может быть пройден через пользователей, которые еще не давали оценки объекта. Кроме того, метод учитывает такую характеристику пользователя, как эмоциональность, а также принимает во внимание пользователей с противоположными оценками.

Главным недостатком методов совместной фильтрации является необходимость сбора большого числа оценок, что затруднительно на первых этапах работы системы. При этом актуальной является проблема масштабируемости алгоритмов – производительность методов и точность оценок снижается с ростом числа объектов оценки и пользователей.

За предпочтение объекта другим объектам может приниматься его покупка. Функция предпочтения пользователя может строиться на основе списка документа, к которым он обращался непосредственно перед совершением покупки. Однако сбор не выраженных явно оценок в полной мере не решает проблемы, поскольку ложноположительная рекомендация может в итоге привести к разочарованию клиента.

В работе [Linden и др., 2003] предложен алгоритм, который вместо схожих пользователей ищет схожие объекты. Для этого на основе данных о двух и более объектах в пользовательской корзине предварительно строится матрица близости объектов. Далее алгоритм ищет объекты, релевантные текущему набору пользователя, производит их агрегацию и рекомендует наиболее коррелируемые из них. Основная часть вычислений производится в офлайне, что позволяет решить проблему

масштабируемости, сохраняя при этом качество рекомендаций. Данный метод персонализации используется в системе Amazon.com.