РАЗРАБОТКА МАТЕМАТИЧЕСКОГО АППАРАТА ЛОГИЧЕСКОГО ПРОГРАММИРОВАНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ АГЕНТОВ ИНТЕРНЕТ

А. А. Морозов
научный сотрудник Института Радиотехники и Электроники РАН
morozov@mail.cplire.ru

Ю. В. Обухов
зам. дир. Института Радиотехники и Электроники РАН
obukhov@mail.cplire.ru

Аннотация. В статье рассматривается проблема разработки математического аппарата логического программирования интеллектуальных агентов, осуществляющих поиск, распознавание, извлечение и анализ информации в Интернет. Обсуждается проблема обеспечения корректности и полноты логического вывода в динамической среде Интернет, в условиях постоянного изменения и пополнения информации. Рассмотрен разработанный авторами математический аппарат логического программирования агентов Интернет, основанный на принципе повторного доказательства подцелей.

Abstract. In this article we considered the problem of development of mathematical apparatus for logic programming of intelligent agents for searching, recognizing, collecting and analyzing information in the Internet. The problem of providing soundness and completeness of logical inference in the dynamic environment of Internet, under the conditions of permanent update and revision of information is discussed. We proposed a new mathematical apparatus based on the principle of repeated proving subgoals for logic programming of the Internet agents.

Введение

Интеллектуальными агентами Интернет называют программы, автоматизирующие поиск, распознавание, извлечение и анализ информации во всемирной Сети, ориентированные на нужды конкретного пользователя (или группы пользователей). Интеллектуальные агенты отличаются от широко используемых в настоящее время поисковых систем Интернет тем что:

1. Они способны работать самостоятельно в течение длительных промежутков времени (дни, недели и более), выполняя задание, порученное пользователем.
2. Как любая программа, однажды созданный агент может быть использован в будущем любое количество раз, в то время как запрос, посланный универсальной поисковой системе, вызывает однократную операцию сбора информации.

В настоящее время не существует общепринятого определения агентов. Однако агентами принято называть программы, демонстрирующие свойства автономности, реактивности (т.е. реагирующие на внешние стимулы), проактивности (например, способные планировать свои действия) и (в случае многоагентных систем) обладающие социальным поведением [1,2].

Одним из наиболее интересных и перспективных подходов к программированию интеллектуальных агентов Интернет является логическое программирование агентов [1,3]. Актуальность этого подхода определяется, в частности, соответствием идеологии и принципов логического программирования задачам поиска, распознавания и анализа неструктурированной, плохо структурированной, а также гипертекстовой информации [7,9]. В последнее десятилетие было разработано большое количество подходов и методов логического программирования агентов Интернет, основанных на различных модификациях и расширениях языка Пролог, а также неклассических логиках (линейной, модальной, F-Logic и др.) [1,3,7]. Тем не менее, до сих пор не был создан математический аппарат, который обеспечивал бы свойства корректности и полноты логических программ (агентов), работающих в динамическом внешнем окружении (т.е. в условиях постоянного изменения и пополнения исходной информации в Интернет). Для решения этой проблемы мы разработали математический аппарат, основанный на принципе повторного доказательства подцелей.

1. Идея математического аппарата

Созданный математический аппарат логического программирования агентов Интернет включает:

1. Модель интеллектуальных агентов, работающих в динамическом внешнем окружении.
2. Декларативную семантику агентов.
3. Стратегии исполнения агентов Интернет, корректные и (при определённых условиях) полные относительно декларативной семантики этих агентов.

В рамках разработанной модели интеллектуальных агентов, агент Интернет (а также набор взаимодействующих агентов Интернет) описывается с помощью логической программы, исполняемой под управлением специальной стратегии, реализующей так называемые повторные доказательства подцелей.

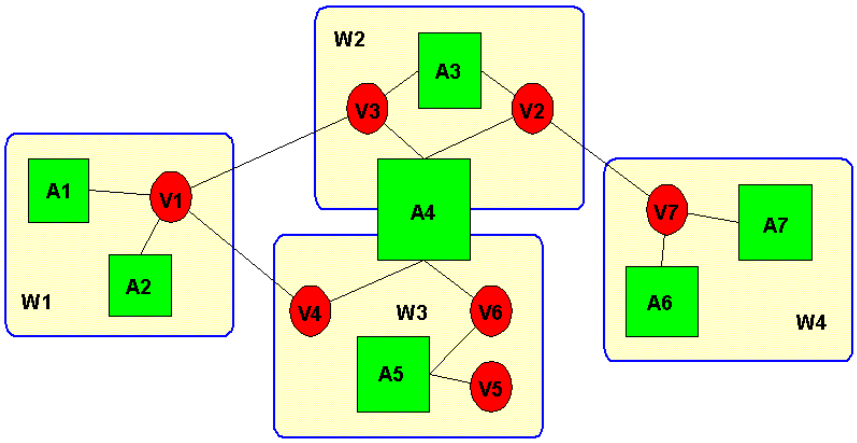

Рис. 1. Логические акторы, связанные общими переменными.

Идея повторных доказательств состоит в том, что программа разделяется на подцели (названные «логическими акторами») [4-9], обладающие следующими свойствами (см. рис. 1):

1. Единственным каналом обмена информацией между акторами A_1–A_n являются общие переменные V_1–V_m.
2. Доказательство акторов может осуществляться в произвольном порядке, независимо друг от друга.
3. Кроме того, результаты доказательства любого актора можно отменить, оставив без изменения другие подцели программы.

После отмены результатов доказательства актора, оно может быть осуществлено повторно. Тем самым, можно реализовать модификацию проведённых рассуждений. Результаты вычислений и сам ход рассуждений могут быть частично изменены во время и после окончания логического вывода. Это позволяет, в частности, приводить логические рассуждения в соответствие с новой информацией, поступающей извне.

Заметим, что для структурирования пространства поиска логической программы мы используем разработанные ранее объектно-ориентированные синтаксические средства (классы и наследование) [4,5,6,9], поэтому в нашей модели различным акторам могут соответствовать различные пространства поиска (W_1–W_k). Например, на рисунке 1 подцели актора A_4 доказываются в некоторых пространствах W_2 и W_3.

2. Доказательство логического актора в динамической среде Интернет

Рассмотрим отдельный логический актор, в ходе исполнения которого используется информация, доступная в Интернет. Исполнение каждого отдельно взятого актора осуществляется под управлением стандартной стратегии Пролога («поиск слева направо в глубину с возвратом»), которую можно изобразить графически в виде обхода соответствующего И-ИЛИ дерева программы (рис. 2).

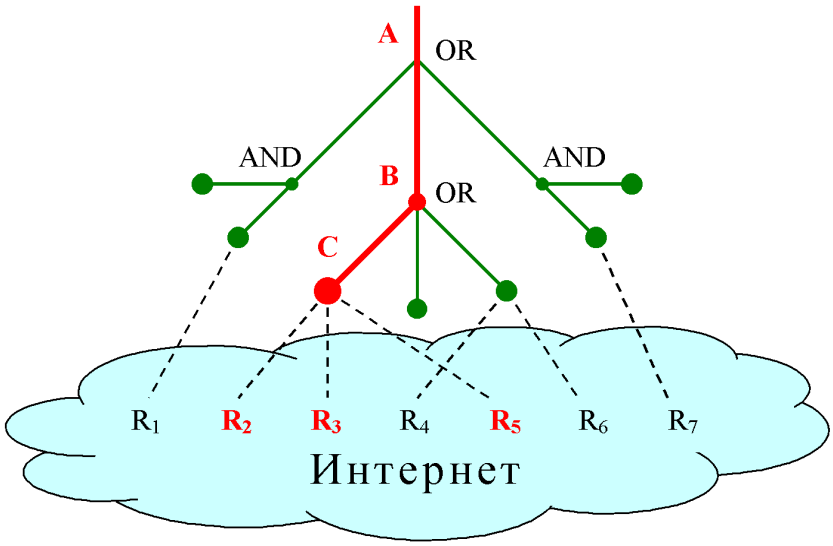

Рис. 2. И-ИЛИ дерево логического актора, доказываемого в Интернет.

В ходе доказательства логический актор обращается к ресурсам Интернет R_1–R_x. Предположим, что доказательство этой подцели закончилось успехом, а именно, в ходе логического вывода была успешно пройдена ветвь A-B-C. Это означает, что построенное доказательство зависит от ресурсов Интернет R_2, R_3, R_5, использованных при прохождении ветви A-B-C, и не зависит от других ресурсов Интернет, к которым

программа, возможно, обращалась при прохождении иных ветвей И-ИЛИ дерева. Следовательно, повторное доказательство рассматриваемого актора необходимо произвести (только) при изменении какого-либо из ресурсов R_2, R_3, R_5, а все остальные ресурсы можно не учитывать.

Предположим теперь, что доказательство актора завершилось неудачей. В этом случае значительно труднее определить, какие из использованных ресурсов Интернет оказали решающее влияние на результат доказательства, потому что при прохождении различных ветвей ИЛИ дерева использовались разные ресурсы, и, кроме того, сочетание данных из различных ресурсов Интернет могло повлиять на негативный результат при прохождении отдельных ветвей ИЛИ дерева. В общем случае, каждый из ресурсов Интернет R_1–R_x мог повлиять на (неудачный) исход логического вывода.

Таким образом, для осуществления корректного управления повторными доказательствами, в ходе доказательства логического актора необходимо запоминать:

1. Полный список ресурсов Интернет, к которым актор обращался во время обхода всех пройденных им ветвей И-ИЛИ дерева.
2. Список ресурсов Интернет, к которым актор обращался во время прохождения текущей ветви ИЛИ дерева.

Если текущая ветвь ИЛИ дерева приведёт к успеху, список ресурсов (1) можно забыть, а вот состояние ресурсов из списка (2) в дальнейшем необходимо отслеживать, чтобы, при изменении любого из этих ресурсов, осуществить повторное доказательство актора. В случае неудачи доказательства актора, для корректного управления повторным доказательством актора достаточно (хотя не всегда необходимо) отслеживать состояние всех ресурсов из списка (1).

3. Системы логических акторов и декларативная семантика агентов

Для описания логических акторов мы используем только дефинитные логические правила (без отрицания not), поэтому каждый отдельный актор, так же как система взаимодействующих акторов, всегда имеют классическую теоретико-модельную семантику. Таким образом, отдельно взятый агент Интернет, так же как система взаимодействующих агентов, в рамках нашей модели представляются в виде дефинитной логической программы, целевым утверждением которой является конъюнкция соответствующих логических акторов.

Определение. Теоретико-модельной семантикой агента Интернет (системы агентов Интернет) является теоретико-модельная семантика соответствующей логической программы.

Наиболее сложной и интересной задачей, возникающей при таком представлении интеллектуальных агентов, является разработка стратегий управления, поддерживающих повторные доказательства акторов и при этом обладающих математическими свойствами корректности и (желательно) полноты. Мы разработали несколько стратегий управления исполнением логических программ, поддерживающих повторные доказательства.

В качестве одной из первых была создана стратегия последовательного исполнения логических программ с логическими акторами. Эта стратегия подробно описана в [5], там же доказаны теоремы о корректности этой стратегии и её полноте при условии отсутствия зацикливаний логической программы. На основе разработанной стратегии

последовательного исполнения логических программ были созданы методы и экспериментальные средства логического анализа SADT диаграмм, а также методы визуального программирования на основе SADT и средства управления графическим пользовательским интерфейсом на основе SADT нотации [5]. Однако дальнейшие эксперименты с визуальным логическим программированием интеллектуальных агентов показали целесообразность разработки более сложных, параллельных стратегий управления.

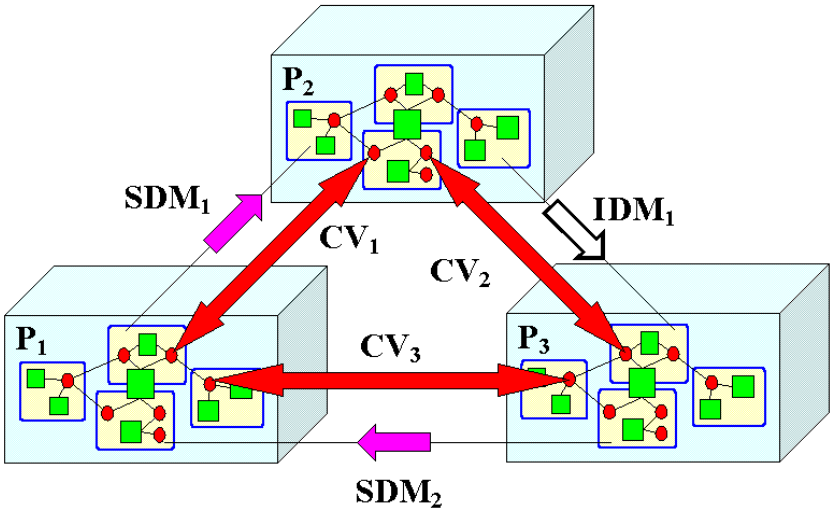

Рис. 3. Система взаимодействующих процессов.

В настоящее время мы расширили нашу модель, введя в неё параллельные процессы, взаимодействующие с помощью асинхронных сообщений различных видов (рис. 3). Каждый процесс (P_1–P_q) является системой логических акторов. В рамках расширенной модели мы рассматриваем асинхронные сообщения двух видов [9]:

1. Так называемые прямые сообщения (интуитивно соответствующие асинхронному вызову предикатов из одних процессов в других).
2. Потоковые сообщения (соответствующие передаче данных через общие переменные процессов CV_1–CV_r).

В свою очередь, прямые сообщения также подразделяются на два вида:

1. «Информационные» прямые сообщения (IDM).
2. «Переключающие» прямые сообщения (SDM).

Отличие между этими двумя разновидностями прямых сообщений состоит в правилах обработки сообщений принимающим процессом. А именно, в случае неуспешного завершения вычислений, вызванных сообщением в получающем процессе, информационное сообщение просто игнорируются, в то время как переключающее сообщение переводит процесс в особое состояние «неудачный», в котором приостанавливается обработка поступающих информационных прямых сообщений. Вывести процесс из состояния «неудачный» может только успешное завершение обработки потокового или переключающего прямого сообщения, пришедшего позже.

Использование различных видов прямых и потоковых сообщений позволяет описывать сложное поведение агентов (систем взаимодействующих агентов), не вводя

в модель средства синхронизации процессов. Подробное описание методов асинхронного обмена информацией и состояний процессов можно найти в [4,9].

В рамках расширенной модели все агенты, так же как и системы взаимодействующих агентов, обладают классической теоретико-модельной семантикой. При этом, однако, стратегия исполнения параллельных процессов, обменивающихся сообщениями, в общем случае, уже не обладает полнотой относительно декларативной семантики программ. В [9] приведено строгое определение теоретико-модельной семантики системы параллельных взаимодействующих агентов и теорема о корректности стратегии параллельного исполнения агентов.

Отдельной темой для обсуждения являются условия, при которых параллельная логическая программа обладает не только корректностью, но и полнотой относительно своей декларативной семантики. Очевидно, что гарантировать наличие этого свойства можно только при выполнении достаточно жёстких ограничений на свойства отдельных процессов и системы процессов в целом:

1. Каждый отдельный процесс (как отдельная логическая программа, вычисляющая некоторые данные) должен обладать полнотой относительно своей декларативной семантики. В частности, должно быть гарантировано отсутствие зацикливаний процессов.
2. Процедуры, вычисляющие данные, передаваемые из одних процессов в другие, должны быть детерминированными, чтобы полнота логического вывода не была потеряна из-за невозможности распространения отката из принимающего процесса в передающий.
3. Передача информации между процессами должна быть строго однонаправленной. Можно потребовать, чтобы для передачи информации использовались только потоковые сообщения. В системе взаимодействующих процессов не должно быть обратных связей.

Системы взаимодействующих процессов, удовлетворяющих таким жёстким условиям, имеют важное практическое применение, так как описывают конвейерную обработку данных, принимаемых системой интеллектуальных агентов извне. Наличие свойств корректности и полноты позволяет реализовать исчерпывающий перебор, то есть гарантировать нахождение всех решений задачи, удовлетворяющих заданным логическим условиям.

4. Реализация и практическое использование математического аппарата

В большинстве случаев задачи сбора и обработки информации, требующие автоматизации с помощью интеллектуальных агентов, носят индивидуальный характер и определяются конкретными пользователями. Поэтому для достижения реального экономического эффекта от использования агентов необходимо создание технологий программирования агентов, максимально облегчающих и упрощающих разработку таких программ конечными пользователями, не имеющими специальных навыков программирования. В идеале, создание агента Интернет должно быть таким же простым, как составление запроса к универсальной поисковой системе. Разработанные методы и средства логического программирования обеспечивают основу для решения этой задачи. На основе созданного математического аппарата разработан параллельный

объектно-ориентированный логический язык Акторный Пролог (определение языка, включающее все новые средства, см. на сайте [4]). В новых версиях языка мы ввели специальные средства, поддерживающие разработку агентов Интернет, в частности:

1. Предопределённые классы для получения информации по HTTP и FTP протоколам Интернет.
2. Средства визуального программирования, включающие транслятор SADT диаграмм в параллельный Акторный Пролог и систему управления пользовательским интерфейсом на основе SADT диаграмм [9].
3. Пакеты и др. синтаксические средства, необходимые для реализации компонентного визуального программирования.

В настоящее время реализован действующий прототип системы логического программирования интеллектуальных агентов Интернет на основе разработанного математического аппарата и языка Акторный Пролог. Разработанная система позволяет создавать агенты для сбора и анализа информации в Интернет. Использование объектно-ориентированного логического подхода существенно упрощает создание и последующую модификацию агентов Интернет.

5. Отличие от других подходов

Основным отличием нашей модели и методов программирования интеллектуальных агентов является использование стратегий управления, поддерживающих повторные доказательства подцелей. Метод повторных доказательств обеспечивает (классическую) корректность стратегий исполнения логических программ в динамическом внешнем окружении, чего не могут обеспечить существующие подходы, основанные на расширениях стандартного Пролога (LogicWeb, Minerva, DLP), логическом программировании с ограничениями (язык Oz и др.) и неклассических логиках (W-ACE, Lygon). В сущности, разработанный метод повторных доказательств можно обобщить и использовать также для исполнения логических программ с ограничениями и неклассических логических программ. Однако неклассические логики и ограничения не обеспечивают наличие классической теоретико-модельной семантики агентов, исполняемых в динамическом внешнем окружении.

Фактически, нам удалось реализовать модифицируемые рассуждения с помощью специальных стратегий логического вывода, оставаясь в рамках классической логики первого порядка. Заметим, что разработанный подход свободен от недостатков немонотонных логических систем, таких как необщезначимость (в классическом смысле) выводимых формул, зависимость результата от порядка применения правил вывода и повышенная вероятность зацикливания логических программ.

Заключение

Разработан математический аппарат логического программирования интеллектуальных агентов, осуществляющих поиск и распознавание информации в сложно структурированной, динамической среде Интернет. В основу разработанного математического аппарата положен принцип повторного доказательства подцелей, позволяющий модифицировать логические рассуждения во время и после окончания

исполнения логической программы, приводя их в соответствие с новой информацией, поступающей извне.

На основе разработанного аппарата модифицируемых рассуждений создан параллельный объектно-ориентированный логический язык Акторный Пролог, обеспечивающий корректность логических программ (интеллектуальных агентов), функционирующих в условиях постоянного изменения и пополнения информации.

Разработанные средства позволяют создавать персональные системы (агенты) сбора и анализа информации в Интернет. Использование объектно-ориентированного логического подхода существенно упрощает создание и последующую модификацию агентов Интернет. Разработанный подход поддерживает визуальное и компонентно-ориентированное программирование агентов Интернет.

Работа выполнена при поддержке РФФИ (грант 0001-00560).

Литература

1. **Sadri F., Toni F.** Computational Logic and Multiagent Systems: a Roadmap. – 1999. (http://citeseer.nj.nec.com/sadri99computational.html)
2. **Huang Z., Eliëns A., van Ballegooij A., de Bra P.** A Taxonomy of Web Agents // Proc. of DEXA Workshop. – 2000. – pp. 765-769. (http://citeseer.nj.nec.com/408821.html)
3. **Davison A.** Logic Programming Languages for the Internet // A. Kakas, F. Sadri (editors), Invited Submission for Computational Logic: From Logic Programming into the Future. – Springer Verlag, 2001. (http://fivedots.coe.psu.ac.th/~ad/papers/summBob.ps.gz)
4. **Морозов А.А., Обухов Ю.В.** Акторный Пролог. Определение языка программирования. – Москва, 1996. – Препринт ИРЭ РАН 2(613). – 57 с. (Новая версия определения языка опубликована на нашем сайте http://www.cplire.ru/Lab144/index.html)
5. **Морозов А.А.** Логический анализ функциональных диаграмм в процессе интерактивного проектирования информационных систем: Диссертация на соискание учёной степени кандидата физико-математических наук. – М., 1998. – 199 с. (http://www.cplire.ru/Lab144/auto.html)
6. **Morozov A.A.** Actor Prolog: an Object-Oriented Language with the Classical Declarative Semantics // Proc. of IDL'99 workshop. – Paris, 1999. (http://www.cplire.ru/Lab144/paris.pdf)
7. **Morozov A.A., Obukhov Yu.V. Gulyaev Yu.V.** On The Problem of Using Logic Object-Oriented Programming in the World Wide Web // Proc. of the Special Russian Session "The Internet Developments in Russia". First IEEE/Popov Workshop on Internet Technologies and Services. – Moscow, 1999. – pp. 54-59. (http://www.cplire.ru/Lab144/internet.pdf)
8. **Morozov A.A., Obukhov Yu.V.** On the Problem of Logical Recognition in the Dynamic Internet Environment // Pattern Recognition and Image Analysis. – 2001. – Vol. 11. – No. 2. – pp. 454-457. (http://www.cplire.ru/Lab144/pria5.pdf)
9. **Morozov A.A., Obukhov Yu.V.** An Approach to Logic Programming of Intelligent Agents for Searching and Recognizing Information on the Internet // Pattern Recognition and Image Analysis. – 2001. – Vol. 11. – No. 3. – pp. 570-582. (http://www.cplire.ru/Lab144/pria570m.pdf)