

Интеллектуальная многоагентная система дедуктивного вывода на основе сетевой организации

Аннотация. В статье рассматриваются вопросы практической реализации методов теории многоагентных систем в организации интеллектуальной прикладной системы, которая в качестве основного механизма обработки знаний использует процедуру дедуктивного вывода. Исследуются задачи создания модели агентов, а также организации многоагентной системы. Разрабатывается механизм опосредованной централизованной коммуникации интеллектуальных агентов. Обсуждаются архитектура и принципы функционирования интеллектуальной прикладной многоагентной системы.

Ключевые слова: интеллектуальные прикладные системы, многоагентные системы, инженерия знаний, объектно-ориентированная методология, теория множеств.

Введение

В настоящее время процедуры дедуктивного вывода все еще продолжают оставаться сильным средством при манипулировании знаниями. Их использование в интеллектуальных прикладных системах получило весьма широкое распространение вследствие простоты реализации, строгости и общности используемой логики. Тесная кооперация с человеком позволяет интеллектуальным прикладным системам успешно решать задачи в прикладных предметных и/или проблемных областях. Яркими примером подобных интеллектуальных прикладных систем являются экспертные системы, системы поддержки принятия решений, нашедшие широкое применение в таких предметных областях, как медицина, геология, управление производством, транспортом, энергетическими комплексами и т.п. Однако процедуры дедуктивного вывода выполняют лишь формальные манипулирования знаниями, в то время как на долю человека возлагаются задачи представления сложных задач в виде совокупности простых, поиска нужных эвристик для сокращения пространства поиска. К тому же, при применении подобных интеллектуальных систем в прикладных предметных областях особое значение приобретает проблема эффективности и

производительности. Одним из способов решения этой проблемы является использование специальной архитектуры интеллектуальной системы, которая организовывается таким образом, что решение сложной проблемы производится на разных вычислительных уровнях системы.

Представление сложной задачи в виде совокупности простых, позволяет использовать процедуру дедуктивного вывода для подзадач, обладающих исключительно сложной логикой при явном задании вариантов. Тогда как оставшиеся вычислительные части задачи передаются специально сконструированным для этого вычислительным модулям, именуемым интеллектуальными агентами [1, 2]. Однако в большинстве случаев архитектура системы организовывается на принципах классической теории организации по Ф.В. Тейлору применительно к многоагентной системе. Жесткая организационно-управленческая иерархия тейлоровской многоагентной системы не способна эффективно решать задачи практической степени сложности в прикладных предметных областях. Решением этой задачи является создание специальной архитектуры интеллектуальной прикладной многоагентной системы организованной на принципах сетевых посттейлоровских организаций смешанного вертикально-горизонтального типа [1].

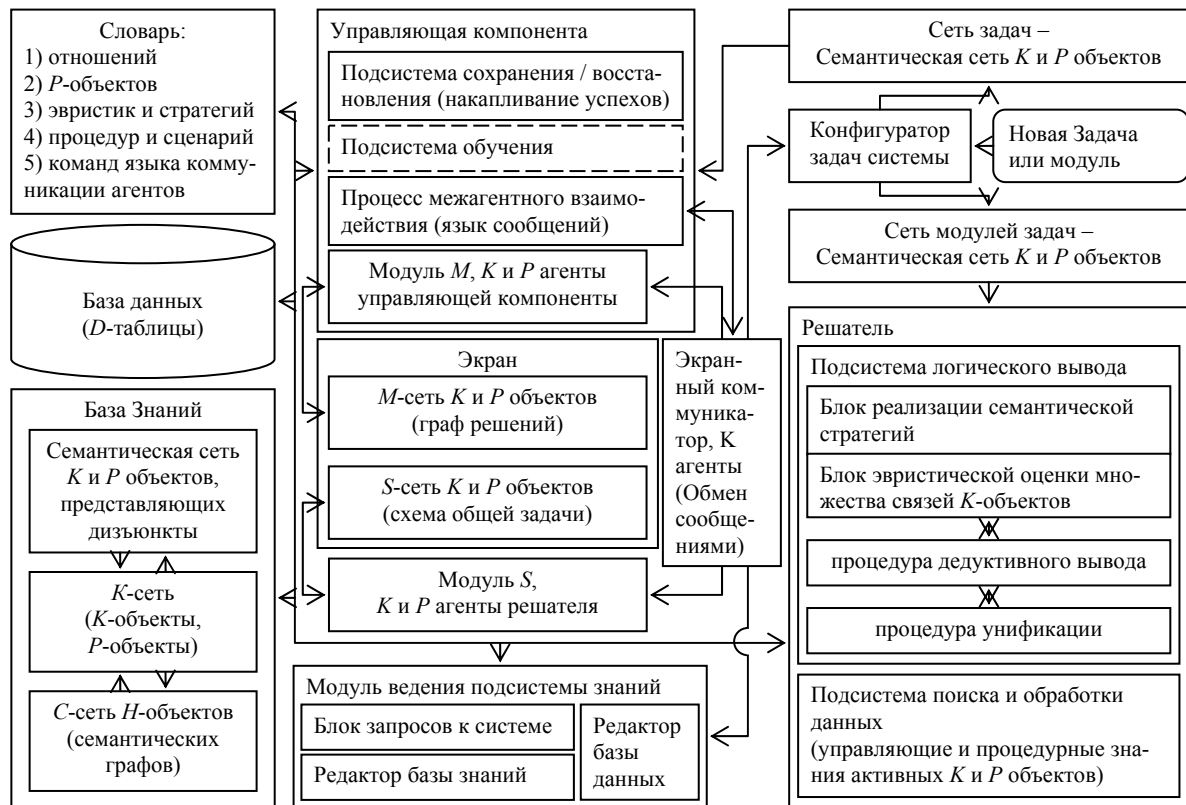


Рис. 1. Архитектура интеллектуальной прикладной многоагентной системы

1. Архитектура интеллектуальной прикладной многоагентной системы

На Рис.1 представлена архитектура проблемно-ориентированной интеллектуальной многоагентной системы, предназначенной для эффективного и производительного решения задач практической степени сложности.

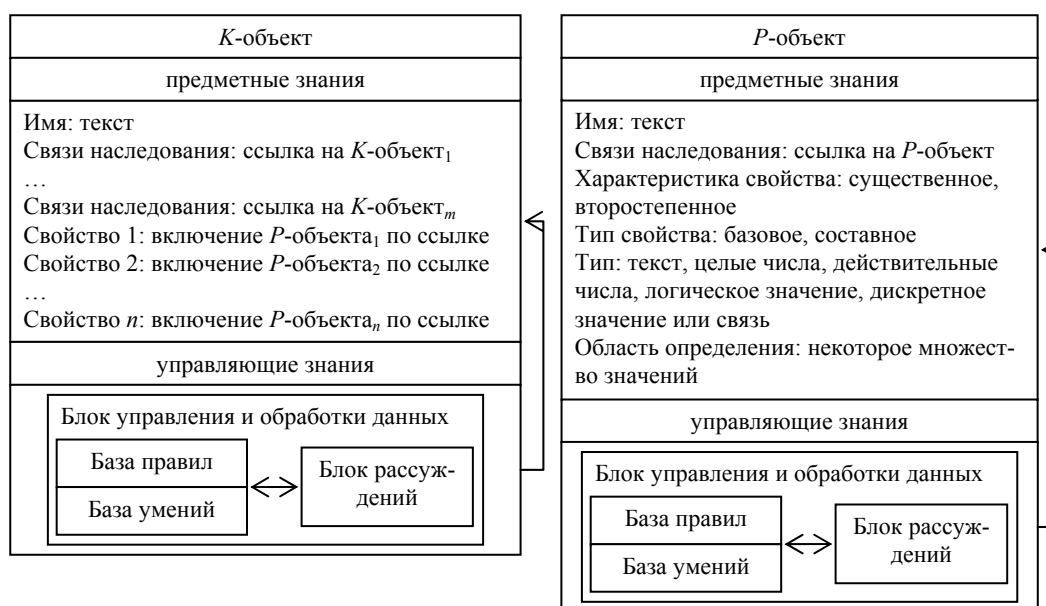
Решение конкретных задач системой реализуется отдельными модулями поиска решений, за исполнение и интерпретацию которых отвечает решатель (исполняющая компонента) и управляющая компонента. В интеллектуальной системе используется логический дедуктивный вывод и подсистема поиска и обработки данных на уровне концептуальных объектов. Механизм логического дедуктивного вывода производит вывод на семантической сети концептуальных объектов [3]. Сокращение пространства поиска и выбор связей объектов производится эвристической функцией и семантической стратегией, которая использует значимую информацию об объектах. Поиск и обработка соответствующей информации, а также решение задач производится интеллектуальными агентами решателя.

Дополнительно интеллектуальная система включает ряд вспомогательных инструментальных средств, а именно редакторы базы знаний и базы данных, конфигуратор задач системы, с помощью которых, обеспечивается настройка, конфигурирование и управление информационной системой.

2. Представление знаний в интеллектуальной системе

База знаний интеллектуальной прикладной системы содержит знания о прикладной предметной области, представленные концептуальной моделью системы [3]. Знания, описываемые в концептуальной модели системы, разделяются на части, обусловленные как различной внутренней структурой знаний, так и потребностью в различных частях знаний на разных этапах их дальнейшей обработки [4, 5].

На первом уровне концептуальной модели задается структурная модель прикладной предметной области. Знания, описываемые в структурной модели, представляются следующими составляющими: К-сеть и D-таблицы. К-сеть содержит

Рис. 2 Структура *K*-объекта и *P*-объекта

конкретные факты, события или состояния, а также взаимосвязи между этими единицами, которые присутствуют в виде отношений между конкретными объектами в моделируемой предметной области. Значения, характеризующие объекты *K*-сети, хранятся в *D*-таблицах.

Для того чтобы модель представления знаний позволяла алгоритмам поиска решать задачи практической степени сложности, в её описание включены две наиболее значимые категории знаний: предметные (декларативные и экстенциональные, процедурные и интенциональные, глубинные и поверхностные) и управляющие. Эти знания моделируются с помощью объектно-ориентированного и агентно-ориентированного подхода.

В качестве информационных единиц модели представления знаний рассматриваются два типа концептуальных объекта, описывающих триаду “объект-свойство-отношение”: *K*-объект – основной объект модели, *P*-объект – свойство объекта модели, а составное свойство объекта – отношение (Рис. 2).

На втором уровне концептуальной модели задается формальная модель прикладной предметной области, которая представлена множеством дизъюнктов. Исходное множество дизъюнктов описывает в формальном виде знания прикладной предметной области, представленные на первом уровне. Так как множество дизъюнктов является объектом работы процедуры дедуктивного вывода, то выбор способа

их представления является крайне важным для создания эффективной и производительной процедуры дедуктивного вывода. В качестве метода представления данных в логике предикатов первого порядка используется процедура дедуктивного вывода на семантической сети концептуальных объектов [3].

На третьем уровне концептуальной модели задается семантическое содержание, характеризующее объекты *K*-сети и объекты формальной модели. Семантическое описание объектов моделируется семантическими графами, которые представлены блочными структурами Хендрикса [4]. Для удобства пользования различными вычислительными механизмами системы, блочные структуры, описывающие семантические графы, представлены в базе знаний отдельной семантической *C*-сетью концептуальных *H*-объектов. Внутреннее описание *H*-объектов имеет сходную с другими концептуальными объектами структуру [3].

3. Представление задач в интеллектуальной системе

База знаний интеллектуальной прикладной системы также содержит знания, описывающие решаемые системой задачи и исполнительные модули задач, которые реализуют решение этих задач. Эти знания представлены знаниями концептуальной модели системы в форме семанти-

ческих сетей концептуальных К-объектов и Р-объектов [3].

Одна семантическая сеть содержит описание задач, решаемых системой. Для описания задачи вводится класс *Задача*, имеющий атрибуты “Имя задачи”, “Параметры задачи” и “Имя операции”. Параметры задачи представлены множеством пар вида $\langle O_c, S_o \rangle$, где O_c – классы объектов базы знаний, S_o – множество объектов классов O_c , для которых должна быть решена данная задача. Имя операции определяет имя производимой на объектах операции.

Другая семантическая сеть содержит описание задач, решаемых системой агентов и различными вычислительными механизмами. Для описания данных задач вводятся классы *Центральный модуль* и *Периферийный модуль*. С помощью периферийных модулей реализуются решения центральных модулей. Класс *Центральный модуль* имеет следующие атрибуты: “Имя модуля”, “Параметры модуля”. Параметры модуля представлены множеством пар вида $\langle M_c, M_o \rangle$, где M_c – классы объектов M модуля и/или S модуля, M_o – множество объектов классов M_c , для которых должна быть решена данная задача.

На семантической сети задач определены отношения “Подзадача”, “Вызывает”, “Исполняет”. Отношение “Подзадача” связывает некоторую задачу с другими задачами (ее подзадачами), решение которых требуется для решения данной задачи. Отношение “Вызывает” определяет потенциальную возможность порождения одной задачи от другой. Отношение “Исполняет” связывает задачу с центральными модулями, решение которой они обеспечивают. На семантической сети модулей также определено отношение “Исполняет”, которое связывает периферийный модуль с центральным модулем, решение которого он обеспечивает.

Класс *Периферийный модуль* имеет следующие параметры: “Имя периферийного модуля”, “Входные объекты”, “Выходные объекты”, “Исполняющие подпрограммы”. Значением параметра “Входные объекты” является множество объектов и/или классов объектов базы знаний, которые необходимы для функционирования периферийного модуля. Значением параметра “Выходные объекты” является множество объектов и/или классов объектов базы знаний, которые могут быть созданы периферийным модулем в качестве результатов ее

исполнения. Значением параметра “Исполняющие подпрограммы” являются процедуры, функции и/или сценарии, с помощью которых происходит решение задач.

При необходимости набор задач, а также периферийных и центральных модулей может быть расширен с помощью конфигуратора системы, который позволяет регистрировать новые задачи и модули и включать их в базу знаний.

Пример 1. Рассмотрим описание классов *Задача*, *Центральный модуль* и *Периферийный модуль*. Класс *Задача*: O_c – Р-объект, $S_o = \{t_1, t_2\}$, где t_1 и t_2 – термы, представленные в виде размеченного ациклического орграфа и записанные в виде массива [3]. Имя операции – Унификация. Класс *Центральный модуль*: O_c – Р-объект, $S_o = \{P_1, P_2\}$, где P_1 и P_2 – Р-объекты модуля S . Класс *Периферийный модуль*: объекты $\{t_1, t_2\}$ записаны в P_1 центрального модуля, объект t_n записан в P_2 центрального модуля, процедура УНИФИКАЦИЯ ().

4. Модель агентов

Моделью решения задачи управления поиском является разбиение процесса дедуктивного вывода на множественные параллельные процессы. Каждый процесс, производя вывод и исполняя какой-либо план поиска, строит свое множество данных [2]. В дополнение к дедукции, параллельный поиск должен управлять не только взаимодействиями процессов, но и разбиением задач на подзадачи. Это вызывает дополнительную проблему, которая заключается в том, что подсистема вывода должна иметь возможность гибко изменять управление поиском при решении сложных прикладных задач. Другой задачей, возникающей при этом, является задача выполнения вывода с достаточной эффективностью и постоянной производительностью, которая не должна снижаться при увеличении множества обрабатываемых данных.

Каждому процессу назначается свой исполнитель, именуемый интеллектуальным агентом. Каждый интеллектуальный агент имеет в своем распоряжении все необходимые знания и механизмы по манипулированию данными. Однако, такая схема организации системы агентов, именуемая в литературе тейлоровской организацией многоагентной системы [1], представляет собой структуру классической древовидной иерархии, в которой присутствует лишь один по-

настоящему интеллектуальный агент, а все остальные агенты оказываются либо рефлекторными, либо реактивными. Отношения между уровнями в такой организации жестко фиксированы, задачи передаются сверху вниз, а результаты двигаются обратно. При этом верхний интеллектуальный агент обычно самостоятельно распределяет все задачи между заранее определенными агентами-исполнителями. Подобная организация многоагентной системы целесообразна в случае статической, однозначно определенной и стабильной предметной области, хорошей разложимости исходной задачи на подзадачи, которые агенты-исполнители решают независимо друг от друга.

Таким образом, если интеллектуальная прикладная многоагентная система, построена для решения задач практической степени сложности в прикладных предметных областях, характеризующихся некоторым уровнем неопределенности, то для более эффективной организации основных её компонент (решатель, управляющая компонента) требуется применение сетевых посттейлоровских организаций смешанного вертикально-горизонтального типа [1]. В основе организации сетевой многоагентной системы лежит переход от жесткой организации к “плоским” и гетерархическим структурам, переход от единоначалия к кооперации и координации, где отношение коммуникации играет одну из центральных ролей.

Модель интеллектуальной прикладной многоагентной системы, архитектура которой изображена на рисунке 1, состоит из следующих компонент: экран, обеспечивающий опосредованную, но централизованную коммуникацию между агентами; множество интеллектуальных агентов, которые представляют и управляющую компоненту, и исполнительную компоненту (решатель). Базовыми типами агентов являются интеллектуальный K -агент и интеллектуальный P -агент.

Модель интеллектуальных агентов построена на принципах сетевой расширенной организации [1, 4]. Она не является монолитной и имеет открытую, неоднородную и переменную структуру. Центральным узлом модели является интеллектуальный K -агент, в котором сосредоточены важнейшие стратегические ресурсы, знания и процессы решения задач, составляющих общую задачу системы. Остальные процессы и наиболее рутинные работы, состав-

ляющие тактические ресурсы, которые обеспечивают реализацию стратегий и выполнение действий, выведены наружу и доверены внешним периферийным узлам. Таким образом, периферийные узлы модели работают, в основном используя ресурсы и опыт, поставляемые центральным узлом. Периферийными узлами в модели интеллектуальных агентов являются интеллектуальные P -агенты. Каждый P -агент включен в K -агент по ссылке, и, являясь свойством интеллектуального K -агента, является исполнителем по отношению к своему базовому агенту.

Сетевая организация позволяет создать гибкую организацию системы агентов, с переменной, настраиваемой под конкретную предметную область структурой, состоящей из множества интеллектуальных агентов, число и свойство которых зависит от решаемых задач. Относительная простота периферийных агентов и “высокая интеллектуальность” центрального агента определяют архитектурные и функциональные различия между блоками управления и обработки данных интеллектуальных K и P агентов. Эти различия также отображаются и в методах представления задач для интеллектуальных K и P агентов.

5. Устройство агентов

На Рис. 3 представлено устройство интеллектуальных K и P агентов.

Основное отличие интеллектуальных K и P агентов состоит в способе решения задачи. Заложенный в P -агент метод рассуждений основан на принципе продукционной системы.

Блок управления и обработки данных, являясь источником знаний, состоит из трех частей: X_i , Y_i , Z_i , где X_i – множество условий, при выполнении которых i -й P -агент активируется, Y_i – множество действий, которые совершаются i -м P -агентом. P -агент выполняет действия над объектами заданных слотов панели экрана при помощи своей базы умений Z_i , в которой представлены процедуры и сценарии обработки и манипулирования объектами базы знаний системы.

Инициирование того или иного действия происходит на основании собственных рассуждений P -агента, которые представлены множеством правил. Каждое правило есть множество пар “ситуация – действие” имеющее вид $T_i \rightarrow D_i$, где T_i есть множество условий описывающие

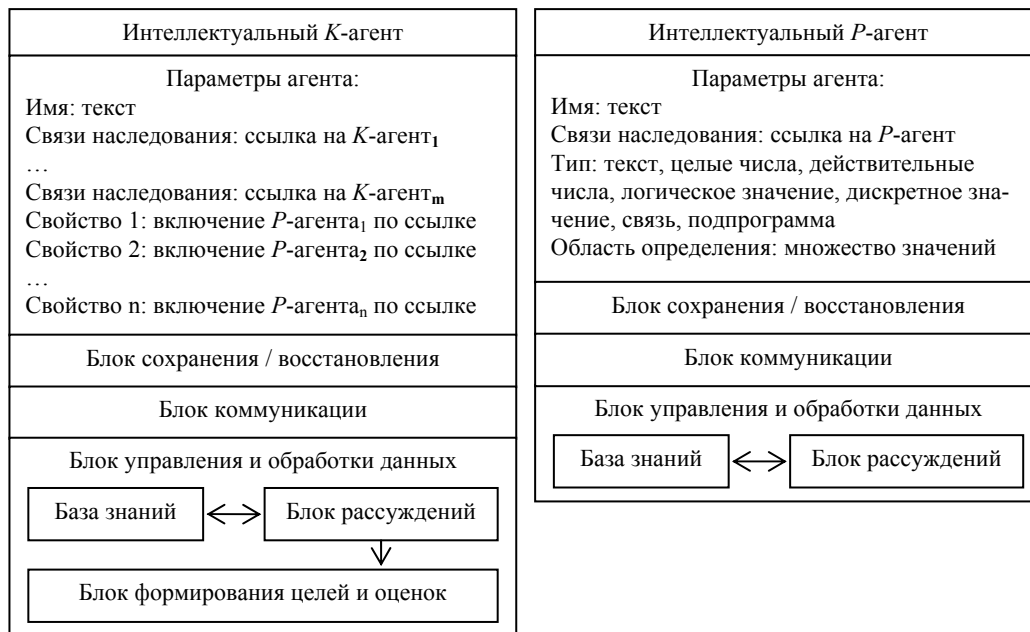


Рис. 3. Структура интеллектуального K-агента и интеллектуального P-агента

ситуацию I , а D_i есть список действий выполняющиеся при возникновении ситуации i . Каждое правило рассматривается как бинарный набор K -объектов концептуальной модели системы с установленным между ними бинарным причинно-следственным отношением. То есть знание P -агента представлено в виде соотношения $R(K_i, K_j)$, где R – бинарное причинно-следственное отношение (*caus*), а K_i и K_j – концептуальные объекты.

Интеллектуальный K -агент, являясь базовым типом интеллектуальных агентов в предлагаемой модели многоагентной системы, имеет более сложный по устройству механизм логических рассуждений. Логический вывод, исполняемый блоком рассуждений K -агента, реализуется за счет применения механизма наследования свойств и обеспечивает правильное разрешение семантических конфликтов, возникающих вследствие множественности наследования и наличия исключений [6, 4]. Знания, представленные в базе знаний K -объекта, описываются в виде концептуальных объектов (K -объект – основной объект модели, а P -объект – описывает свойство объекта и отношения) концептуальной модели интеллектуальной системы. База знаний K -агента, организованная по принципу иерархической организации знаний, представляет собой набор структурированных концептуальных объектов (классов и их экземпляров), связанных отношениями наследования.

Системы с иерархической организацией знаний обладают многими преимуществами, среди которых выделяются следующие: исключение избыточности представления; ускорение процесса логического вывода; простота оценки целостного образа знаний [5, 6].

Механизм наследования свойств, используемый в системах с иерархической организацией знаний, позволяет получить из родового понятия отсутствующую информацию о некотором свойстве видового понятия или экземпляра. Системы наследования изображаются с помощью ациклического направленного графа, именуемым графом наследования.

Блок рассуждений K -агента, используя механизм наследования, строит граф наследования, вершины которого соединяются дугами двух типов: “является”, “не является”. Вершинами графа наследования, с одной стороны, являются K -объекты базы знаний агента. С другой стороны – P -объекты, информация об имени и области определения которых, преобразована в предикатную литеру. Это позволяет абстрагироваться от другой описательной информации P -объекта и сконцентрироваться на поиске по графу истинности или ложности предиката. Полученный граф наследования описывается в рамках некоторого формализма и на основе отношения наследования устанавливаются пути графа, вдоль которых разрешено наследование свойств. Таким образом, граф со-

стоит из статической и динамической частей. Структура статической части определяется на основе содержимого слотов наследования K -объектов. Динамическая часть строится с учетом контекста, в котором произошел вызов механизма наследования, в зависимости от того, наличие какого свойства K -объекта необходимо установить.

Предложенный в [7] аппарат неоднородных семантических сетей, объектно-ориентированный подход [4, 6] и модель рассуждений на основе законов алгебры множеств [8] лежат в основе поиска на графе наследования путей наследования. Процесс построения из множества исходных предложений множества заключительных предложений организован на основе некоторых суждений и правил вывода. Каждое суждение состоит из упорядоченной пары $X \Rightarrow Y$ объектов поиска. Математическая формулировка отношения наследования в виде комбинации свойств и матриц совместности позволяет записать пару $X \Rightarrow Y$ в виде $X \subseteq Y$ на концептуальных K -объектах, поскольку экстенциональное представление типа X есть часть экстенционального представления типа Y .

Введем следующие обозначения объектов поиска:

X, Y, Z – вершины обозначающие некоторые классы;

x_i, y_i, z_i – вершины обозначающие некоторые экземпляры;

P_i – предикатная литера, обозначающая P -объект I ;

$\bar{x}_i$ – отрицание объекта поиска;

$\bar{P}_i$ – отрицание нахождения свойства в объекте поиска.

Выразим все известные типы суждений с использованием упорядоченной пары $X \Rightarrow Y$ в графе наследования:

1. $X \Rightarrow Y$ – “все X есть Y ”, равнозначно “ X является подклассом Y ”

2. $x \Rightarrow Y$ – “некоторые X есть Y ”, равнозначно “ x является экземпляром Y ”

3. $X \Rightarrow \bar{Y}$ – “все X не есть Y ”, равнозначно “ X не является Y ”

Приведем правила вывода, в основе которых лежит семантика данных суждений.

1. Правило транзитивности гласит, что из двух суждений $X \Rightarrow Y$ и $Y \Rightarrow Z$ следует $X \Rightarrow Z$.

2. Правило контрапозиции гласит, что из $X \Rightarrow Y$ следует $\bar{Y} \Rightarrow \bar{X}$.

Применение математической формулировки отношения наследования, описанной в [7], позволяет рассматривать последовательность концептуальных объектов на графе наследования как путь наследования объектов поиска.

Определение 1. Путь наследования это последовательность вершин графа наследования $x_1 \Rightarrow X_1 \Rightarrow X_2 \Rightarrow \dots \Rightarrow X_n$, которая определяет, что свойства высших по иерархии типов концептуальных объектов автоматически переносятся на предшествующие типы по *isa*-иерархии.

Определение 2. Последовательность $x_1 \Rightarrow X_1 \Rightarrow X_2 \Rightarrow \dots \Rightarrow X_n \Rightarrow p^s$, образованную объединением пар $x_1 \Rightarrow X_1$, $X_1 \Rightarrow X_2$, ..., $X_{n-1} \Rightarrow X_n$, $X_n \Rightarrow p^s$, где символ s определяет присутствие или отсутствие объекта поиска на пути, назовем путем наследования объектов поиска.

На пути наследования имеются следующие ограничения: экземпляры могут являться только начальными объектами пути, и отрицательные дуги графа, удовлетворяющие третьему суждению, относятся исключительно к P -объектам и могут находиться только в самом конце пути. Пути наследования дают возможность получения заключений относительно их начальных объектов поиска.

Пример 2. Путь $x_1 \Rightarrow X_1 \Rightarrow X_2 \Rightarrow \dots \Rightarrow X_n$ соответствует заключению $X_n(x_1)$ и означает, что x_1 является экземпляром класса X_n . Путь $x_1 \Rightarrow X_1 \Rightarrow X_2 \Rightarrow \dots \Rightarrow X_n \Rightarrow \bar{p}$ соответствует заключению $\neg p(x_1)$ и означает, что p не является свойством x_1 .

Определение пути наследования между двумя объектами аналогично нахождению некоторого логического вывода, который обосновывает заключение, касающееся этих объектов. Однако в связи с наличием множественности наследования и исключений, не все пути наследования приводят к заключениям. Некоторый объект или класс может наследовать информацию не только от одного своего класса, но и от множества предшественников. В системах множественного наследования граф наследования изображается решеткой, поскольку каждая вершина не обязательно имеет единственного предшественника. Такая организация связей между объектами становится конфликтной, в случае, если информация, поступающая от различных источников наследования, противоречива [6]. В работе [6] подробно про-

анализированы теоретические вопросы построения сетей с наследованием, а также предложен алгоритм формирования суждений при наличии исключений.

Определение 3. Альтернативным путем наследования объектов поиска является путь $Y_1 \Rightarrow \dots \Rightarrow Y_m \Rightarrow \overline{p_i}(\overline{p_i})$, где $Y_1 = X_i \dots Y_m = X_{i+1}$, который принадлежит основному пути наследования $X_1 \Rightarrow X_2 \Rightarrow \dots \Rightarrow X_n \Rightarrow \overline{p_i}(\overline{p_i})$, и несет противоречие или свойству $\overline{P}$ или свойству $\overline{P}$.

Определение 4. Выборка объектов происходит по тому пути наследования, в котором присутствует свойство P . В случае обнаружения альтернативного пути, и при отсутствии содержания альтернативного пути наследования в основном пути, в рассмотрение берутся объекты обоих путей наследования.

6. Механизм коммуникации и управления в модели интеллектуальных агентов

В интеллектуальной многоагентной системе все взаимодействия интеллектуальных агентов осуществляются через специальный механизм опосредованной централизованной коммуникации, реализуемой на принципах “доски объявлений”. Основная идея опосредованной коммуникации состоит в воспроизведении взаимодействия агентов посредством доски объявлений, на которой записаны решаемые задачи и вся известная информация. Доска объявлений рассматривается как промежуточный модуль для осуществления взаимодействий между агентами, а агенты в зависимости от их опыта пытаются заполнить доску объявлений [1, 4, 5].

Поскольку архитектура интеллектуальной прикладной многоагентной системы базируется на принципах сетевой организации, то основной механизм опосредованной коммуникации агентов являются семантические сети концептуальных объектов, а в качестве модели кооперации агентов используется модель контрактных сетей [1]. Механизм опосредованной коммуникации, именуемый экраном, представлен на Рис. 4.

Экран состоит из M -сети и S -сети, а также двух модулей системы, которые непосредственно относятся к формированию и использованию данных сетей – модуль системы M

(управляющей компоненты) и модуль системы S (решателя). Модуль системы S представлен множеством интеллектуальных агентов решателя. Каждый K -агент и/или P -агент решает поставленную перед ним задачу на каждом из этапов решения общей задачи системы. Все решения задач агенты выполняют в рамках модуля S .

При работе многоагентной системы агенты решателя порождают, обновляют и оценивают различные варианты решений и коммуникаций. Для реализации в рамках экрана данной функции многоагентной системы используется специальный механизм, именуемый S -сетью. S -сеть представлена семантической сетью концептуальных объектов [3]. По аналогии с доской объявлений каждый концептуальный K -объект S -сети рассматривается в качестве панели экрана, на которой представлена информация о знаниях, возбужденных в процессе решения поставленной перед агентом задачи. К этим знаниям относятся знания о последовательности состояний интеллектуальных агентов решателя в процессе решения, знания о прикладной предметной области и знания о K и P задачах этой области, языке и командах коммуникации агентов, которые были активированы в процессе решения общей задачи системы. Каждый K -объект S -сети хранит частное решение задачи. В любой момент времени S -сеть экрана содержит все задачи, выполняемые интеллектуальными агентами решателя, а все множество K -объектов, образующие сеть, в общем виде описывает рассматриваемую интеллектуальной системой общую задачу.

Состояние любой панели экрана (т.е. K -объекта S -сети) разделяется на информационные уровни, определяемые структурой решаемой интеллектуальным агентом задачи. Такими уровнями являются уровень процедуры дедуктивного вывода, уровень процедуры унификации, уровень семантической стратегии, уровень обработки данных концептуальными объектами, уровень сообщений межагентной коммуникации, уровень логического вывода самого агента, и т.п. Каждый информационный уровень представлен множеством слотов, а каждый слот уровня представлен множеством базовых элементов, которые используются для описания задачи на данном уровне. В качестве базовых элементов выступают K -объекты, P -объекты, термы, H -объекты, значимая информация,

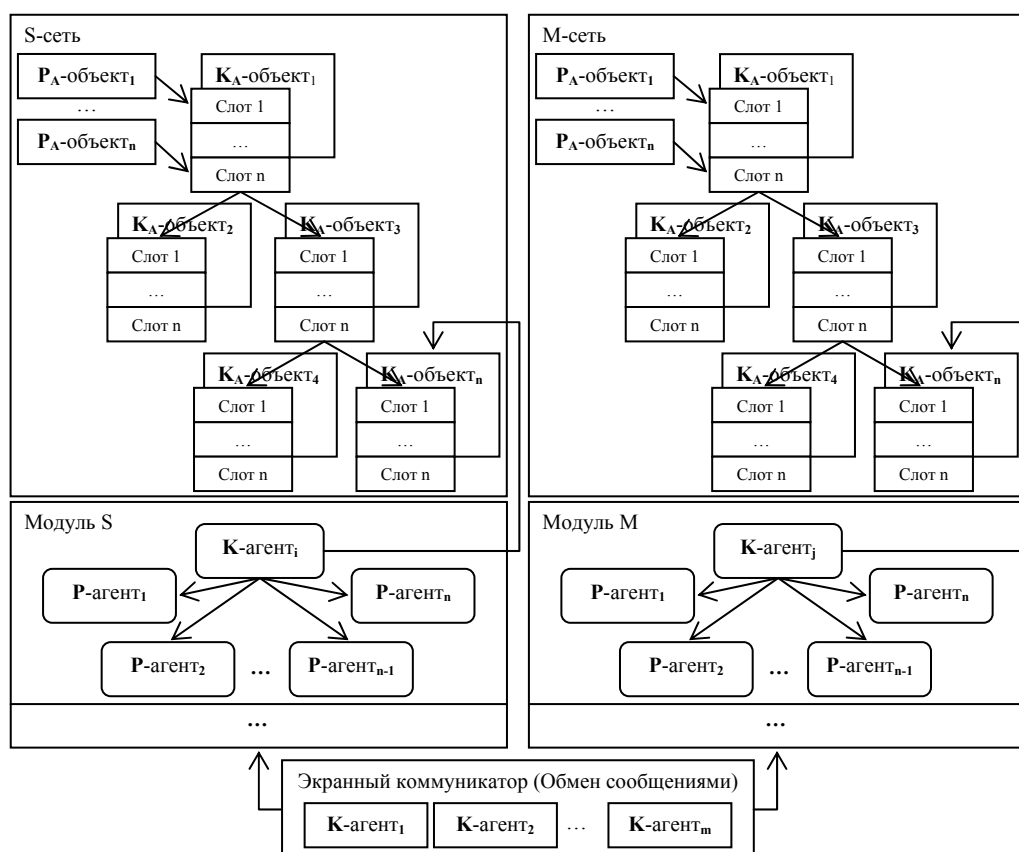


Рис. 4. Архитектура Экрана

представленная в виде типов данных и области определения, команды языка коммуникации агентов, и т.п. Уровни любой панели экрана образуют иерархическую структуру, базовые элементы в которых, описываются в терминах базовых элементов нижних уровней. Поскольку в качестве панелей экрана выступают K -объекты S -сети, то P -объекты, являясь свойствами K -объектов, описывают слоты панелей экрана.

В процессе работы подсистемы решения задач на каждом шаге процесса существует множество альтернатив, по которым процесс может следовать. Множеству интеллектуальных агентов в процессе решения необходимо осуществлять как выбор между альтернативными стратегиями поиска решения, так и выбор между объектами знаний, активирующих поиск. Вдобавок, любая задача, решаемая интеллектуальным агентом на любом этапе решения, может не быть непосредственно разрешена в связи с тем, что для разрешения задачи требуется запуск ряда процедур и/или подпрограмм по разрешению подзадач входящих в нее. К тому же знания могут быть представлены неявно или в общих утверждениях, а процедура дедуктивно-

го вывода активизируется для вывода явных представлений. Для решения этих проблем, а также для обеспечения контроля всего процесса решения задач используется модуль системы M , который состоит из множества интеллектуальных агентов управляющей компоненты. M -сеть, представляющая семантическую сеть концептуальных объектов, по своему описанию и функциональному назначению повторяет аналогичную S -сеть. Каждый K -объект M -сети, также описывающий панель экрана, содержит частные решения и коммуникации агентов модуля системы M . Отличие состоит лишь в том, что каждая задача интеллектуальных агентов модуля системы M является задачей управления ходом решения задач, который производится агентами модуля системы S . Таким образом, в любой момент времени M -сеть экрана содержит все задачи управления, выполняемые интеллектуальными агентами управляющей компоненты, а все множество K -объектов, образующие сеть, в общем виде описывает граф решений общей задачи.

K -агенты управляющей компоненты строят граф решений каждая вершина которого, пред-

ставляет некоторое информационное окружение, включающее множество задач, подзадач и коммуникаций агентов. Все действия подсистемы решения (логического вывода, поиска решений подзадач и т.п.), на каждом из многочисленных этапов, совершаются в рамках определенного окружения этого графа. Любое решение агентов управляющей компоненты делается в некотором окружении X_l^i . Окружение X_l^i может представляться как одним K -объектом, так и некоторым подмножеством K -объектов M -сети. Такой выбор зависит и от сложности задачи, определяемой интеллектуальным агентам решателя, и от возможного порождения подзадач при решении искомой задачи. Типичным примером решения подзадачи, который вызывает создание нового окружения, является нахождение резольвенты двух K -объектов формальной модели прикладной предметной области. Такая задача может противоречить другим решениям или ограничить возможности логического вывода. Поэтому рассмотрение данной задачи представляется отдельно от других задач.

Для представления результата решения искомой задачи создается окружение X_{l+1}^i , которое является потомком X_l^i . То есть каждая новая задача решается в своем информационном окружении, а результат решений формирует новое окружение, которое наследуется от предыдущего. Окружение потомок наследует все знания окружения родителя, и на каждом шаге решения общей задачи, процесс управления перемещается от одних объектов к другим. Таким образом, отсутствует единый центр управления, а процесс управления искомой задачи происходит в рамках определенных информационных окружений графа решений.

Если рассматривать M -сеть в графической интерпретации, то K -объекты родители (X_l^i) и K -объекты потомки (X_{l+1}^i) описываются на разных информационных уровнях, поскольку между данными объектами устанавливаются отношения наследования, а уровни выделяются в соответствии с наследованием свойств. Все подзадачи, которые порождаются во время решения, описываются K -объектами на одном информационном уровне с основным K -объектом. Каждый такой K -объект, описываю-

щий подзадачу, находится в отношениях “*Вызывает*” или “*Подзадача*” с основным K -объектом. Весь процесс решения общей задачи начинается с единственной вершины графа, имеющей название “*начало*”, а заканчивается вершиной – “*окончание*”. Конечная вершина всегда является потомком вершины, которая описывает результирующий ответ на общую задачу.

Формирование задач происходит на основании состояния экрана (т.е. процесса исполнения и управления общей задачи) и поступающей входной информации. Только после того как решены все подзадачи, агенты управляющей компоненты с помощью блока *формирования целей и оценок* и механизма логического вывода анализируют ответы и определяют состояние и дальнейшее развитие процесса решения общей задачи. Для более эффективного решения задачи анализа ответов, в блок *формирования целей и оценок* и механизм логического вывода K -агентов включены концептуальные объекты, описывающие специальные эвристические правила. Ответами на искомые задачи являются: *да* – решение найдено, процесс завершен; *нет* – решение не найдено, процесс завершен; *неизвестно* – решение не найдено, процесс не завершен; *если требуется информация* – требуется дополнительная информация (объекты базы знаний) для разрешения поступившей задачи; *если требуются ресурсы* – требуется дополнительное время для завершения решения поступившей задачи; *если вызывает подзадачу* – поступившая задача имеет отношение “*Подзадача*”, определяющая потенциальную возможность порождения одной задачи другой. В случае порождения ответа *нет*, агент управляющей компоненты формирует информационное сообщение и удаляет в модуле S агенты и объекты, на которых производилось это решение.

Поскольку при работе интеллектуальной многоагентной системы интеллектуальные агенты M -сети и S -сети независимо порождают, обновляют и оценивают решения задач, относящиеся к прикладной предметной области, то для обеспечения механизма коммуникации агентов используется отдельный модуль, именуемый экраным коммуникатором. Экраный коммуникатор состоит из интеллектуальных K -агентов. Каждый K -агент обладает знаниями о структуре и состоянии процессов межагентного взаимодействия. При возникновении потребности коммуникации между агентами модулей

системы M и/или S каждый агент формирует в выходном и/или входном слоте концептуально-го объекта M или S -сети лишь тело и адресат сообщения. Вся организация взаимодействия между агентами модулей системы M и/или S перекладывается на K -агенты экранного коммуникатора. В качестве языка коммуникации агентов используется подмножество языка $KQML$ (Knowledge Query Manipulation Language) [1, 4]. Языковые конструкции языка $KQML$ позволяют использовать язык разметки страниц XML для создания сообщений меж-агентного взаимодействия.

Пример 3. Приведем пример команды со стороны K -агента модуля M K -агенту модуля S на создание K -объекта в S -сети. Сообщение записано на языке XML .

```
<Message Name="0017"> – имя сообщения
  <Receiver Name="K-agent4 (Module-S)" /> –
    уникальное имя адресата
  <Sender Name="K-agent4 (Module-M)" /> –
    уникальное имя отправителя
  <Type Level="1"> – приоритет сообщения
    (высший)
  <Valid /> <Content> CREATE (K-object)
    LOCATE (S-NET) </Content> – содержание со-
    общения </Message> – окончание сообщения.
```

7. Пример исполнения системы агентов

Рассмотрим процесс функционирования системы агентов на примере нахождения резольвенты двух K -объектов. Исходное множество дизъюнктов, представлено семантической сетью концептуальных объектов [3].

В процессе резольвирования двух K -объектов используются следующие процедуры: унификация, нахождения упорядоченного фактора, удаление K -объекта тавтологии, удаления чистого K -объекта, нахождения поглощенного K -объекта. Для получения резольвенты используется механизм наследования K -объектов. Задача k описывает задачу нахождения резольвенты двух K -объектов. Задачами для агентов модуля M являются: контроль процесса решения задачи K -агентом модуля S , определение схемы панели экрана (K -объектов и P -объектов S -сети и M -сети), а также оценка полученных результатов.

K -объект₆ описывает дизъюнкт $L(a,b)$, а K -объект₇ описывает дизъюнкт $\neg H(y) \vee \neg L(a,y)$.

P -объект₁₀ описывает предикатную литеру $L(a,b)$, а P -объект₁₁ описывает предикатную литеру $\neg L(a,y)$ [2, 3].

Шаг 1. Предоставляется K -агенту _{$i-1$} модуля M процессорное время. (K -агент _{$i-1$} связан с предыдущей вершиной графа решений, т.е. K -объектом _{$i-1$} M -сети.)

Шаг 2. Активируется *Блок управления и обработки данных* K -агента _{$i-1$} . Запускается процедура наследования K -агента _{$i-1$} и создается новый K -агент _{i} модуля M . Запускается процедура наследования K -объекта _{$i-1$} и создается новый K -объект _{i} M -сети. Запускается процедура, создающая связь K -агента _{i} с K -объектом _{i} . В P -объект _{$p+1,i$} (входной слот вершины графа решений) записывается описание задачи k .

Шаг 3. Предоставляется K -агенту _{i} модуля M процессорное время. Активируется *Блок управления и обработки данных* K -агента _{i} . Запускается процедура поиска на семантической сети модулей задач исполняющих модулей. (Исполняющие модули задач (центральные и периферийные) считаются известными системе, если их описание представлено в семантической сети модулей задач, либо их можно вывести методом логического вывода.) *Блок рассуждений* методом логического вывода на иерархии наследования, для поступившей K -агенту _{i} задачи k , порождает исполняемую конфигурацию исполнителей K -агента _{i} . Запускается процедура установления по свойствам агента связей со всеми его P -агентами _{p,i} исполнителями. Запускается процедура, устанавливающая связи каждого P -агента _{p,i} с некоторым подмножеством P -объектов M -сети. (Первый P -объект данного подмножества предназначен для хранения входной информации, поступающей от других агентов, а второй P -объект – для хранения выходной информации.)

Шаг 4. Запускается процедура наследования K -агента _{$i-1$} и создается новый K -агент _{i} экранного коммуникатора. (K -агент _{i} (с помощью своих P -агентов исполнителей) формирует тексты сообщений, в которых записаны команды создания нового K -объекта _{j} S -сети и нового K -агента _{j} модуля S . Адресатом этих сообщений является K -агент _{$j-1$} модуля S . K -агент _{i} формирует сообщения и отправляет их адресату.)

Шаг 5. Предоставляется K -агенту _{$j-1$} модуля S процессорное время. Активируется *Блок управления и обработки данных* K -агента _{$j-1$} .

Запускается процедура наследования K -агента $_{j-1}$ и создается новый K -агент $_j$ модуля S . Запускается процедура наследования K -объекта $_{j-1}$ и создается новый K -объект $_j$ S -сети. Запускается процедура, создающая связь K -агента $_j$ с K -объектом $_j$.

Шаг 6. Предоставляется K -агенту $_j$ процессорное время. Активируется *Блок управления и обработки данных* K -агента $_j$. Запускается процедура поиска на семантической сети модулей задач исполняющих периферийных модулей. *Блок рассуждений* методом логического вывода на иерархии наследования, для поступившей K -агенту $_j$ задачи k , порождает исполняемую конфигурацию исполнителей K -агента $_j$. Запускается процедура установления по свойствам агента связей со всеми его P -агентами-исполнителями. Запускается процедура, устанавливающая связи каждого P -агента $_{pj}$ с некоторым подмножеством P -объектов S -сети.

Шаг 7. Каждый P -агент $_{pj}$ модуля S активирует свой *Блок управления и обработки данных* и запускает процедуры разрешения записанной в каждом P -объекте $_{pj}$ задачи. (Первый P -агент решает задачу унификации двух термов, второй P -агент решает задачу получения резольвенты, третий P -агент решает задачу нахождения упорядоченного фактора и т.д. согласно процедуре вывода. Результаты решения задач P -агенты записывают в соответствующие P -объекты $_{pj}$ S -сети, отвечающие за хранение выходной информации. В первую ячейку этих P -объектов записываются итоговые ответы P -агентов. Для рассматриваемой задачи ответом является зарезервированное слово *да*.)

Шаг 8. Активируется *Блок управления и обработки данных* K -агента $_j$. Запускается процедура формирующая оценку решения поступившей задачи k на основании полученных ответов от P -агентов $_{pj}$. Запускается процедура формирующая текст сообщения, в котором записаны итоговые результаты и ответ на задачу k . Адресатом этого сообщения является K -агент $_i$ модуля M . Предоставляется K -агенту $_i$ процессорное время. Активируется *Блок управления и обработки данных* K -агента $_i$. Запускаются процедуры формирующие сообщение и отправляющие сообщение адресату. (Сообщение записывается в P -объект $_{p+2,i}$ M -сети.)

Шаг 9. Предоставляется K -агенту $_i$ модуля M процессорное время. Активируется *Блок управления и обработки данных* K -агента $_i$. Запуска-

ется процедура, формирующая оценку ответа и поступивших от K -агента $_j$ результатов решения задачи k .

Шаг 10. Запускается процедура сортирующая список задач и выбирающая на основании эвристической функции следующую задачу $k+1$. Переход к шагу 1.

Заключение

Рассмотрены вопросы использования много-агентного подхода в организации интеллектуальной прикладной системы, использующей в качестве основного механизма обработки знаний процедуру логического дедуктивного вывода. В качестве модели представления знаний рассматривается специально разработанная концептуальная модель, описывающая сложноструктурированную предметную область. В качестве формализма представления данных в логике предикатов первого порядка выбран метод дедуктивного вывода на семантических сетях концептуальных объектов.

Предложенная модель интеллектуальных агентов построена на принципах сетевой расширенной организации. Центральным узлом модели является интеллектуальный K -агент, а периферийными узлами – интеллектуальные P -агенты. Разработаны структуры K и P агентов, а также модель функционирования механизмов рассуждений агентов. Исследована задача разработки организации многоагентной системы, в основе которой предложено использовать сетевые посттейлоровские организации смешанного вертикально-горизонтального типа. Разработан механизм опосредованной централизованной коммуникации агентов, реализуемый на принципах “доски объявлений”. Полученный механизм позволяет разделить основной процесс поиска на непосредственно процессы решений и управления, взаимодействия между которыми, совершаются посредством подсистемы общения агентов. Приведен алгоритм функционирования системы агентов при решении конкретной задачи.

Представлена архитектура интеллектуальной прикладной многоагентной системы. В рамках данной системы удалось достичь достаточно естественной интеграции различных по своему назначению и возможностям средств и методов моделирования с агентной технологией. В виду того что вводимая в базу знаний

модель прикладной предметной области может быть подвержена как структурным, так и качественным изменениям, поддерживается настройка системы на предметную область и типы решаемых задач.

Применение модели агентов, механизма опосредованной коммуникации и сетевой организации многоагентной системы позволяет повысить эффективность и производительность интеллектуальных прикладных систем, которые используются при решении задач практической степени сложности.

Литература

1. Тарасов В.Б. От многоагентных систем к интеллектуальным организациям: философия, психология, информатика. – М.: Эдиториал УРСС, 2002. – 352 с.
2. Достоверный и правдоподобный вывод в интеллектуальных системах / В.Н. Вагин, Е.Ю. Головина, А.А. Загорянская, М.В. Фомина; Под ред. В.Н. Вагина, Д.А. Поспелова. – М.: ФИЗМАТЛИТ, 2004. – 704 с.
3. Раговский А.П. Метод дедуктивного вывода на семантических сетях концептуальных объектов // Программные продукты и системы. – 2011. – № 2. – С. 19-25.
4. Рассел С., Норвиг П. Искусственный интеллект: современный подход: Пер. с англ. // 2-е изд. – М.: Вильямс, 2006. – 1408 с.
5. Попов Э.В. Экспертные системы. Решение неформализованных задач в диалоге с ЭВМ. – М.: Наука, 1987. – 288 с.
6. Jackson P. Introduction to Expert Systems // 3-d ed. – West Group, Rochester, New York: Addison Wesley, 1998
7. Осипов Г.С. Приобретение знаний интеллектуальными системами. – М.: Наука – Физматлит, 1997. – 112 с.
8. Кулик Б.А. Моделирование рассуждений на основе законов алгебры множеств // Пятая национальная конференция по искусственному интеллекту с международным участием КИИ-1996: Труды конференции. – СПб.: Аврора, 1996. – С. 59-62.

Раговский Антон Павлович. Соискатель ученой степени кандидата технических наук в Московском государственном университете приборостроения и информатики. В 2001 году окончил Московскую государственную академию приборостроения и информатики. Область научных интересов: интеллектуальные прикладные системы, дедуктивный и абдуктивный вывод, прикладная теория графов, инженерия программного обеспечения и методы объектно-ориентированного проектирования. E-mail: anton_ragovskiy@rambler.ru