10.7256/2305-6061.2013.01.7

В.В. Голенков, Д.В. Шункевич, И.Т. Давыденко

СЕМАНТИЧЕСКАЯ ТЕХНОЛОГИЯ ПРОЕКТИРОВАНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ РЕШАТЕЛЕЙ ЗАДАЧ НА ОСНОВЕ АГЕНТНО-ОРИЕНТИРОВАННОГО ПОДХОДА

Аннотация. *В работе приводится описание открытой семантической технологии проектирования интеллектуальных решателей задач. Отдельное внимание уделяется методике проектирования решателей и операций, являющихся составными частями таких решателей. Технология представляет собой многоуровневую систему, каждый уровень которой отвечает за выполнение определенных функций в процессе решения конкретной задачи. На каждом из уровней функционируют агенты определенного назначения, за оптимизацию деятельности которых отвечает присутствующий на каждом из уровней агент-супервизор. В работе приводятся описания алгоритмов работы агентов-супервизоров, а также принципы организации системы подобного рода в целом.*
Ключевые слова: *Программное обеспечение, интеллектуальная система, интеллектуальный решатель задач, логический вывод, семантика, агент, супервизор, иерархическая структура, алгоритм, технология*

Введение

В настоящее время большую актуальность имеет переход от ориентирования проектировщиков интеллектуальных систем с навязываемой (предлагаемой) машины обработки знаний на проектирование решателей задач из готовых компонентов [1]. В основе этого подхода лежат общие принципы организации машин обработки знаний, позволяющих осуществить интегрирование различных моделей решения задач, как существующих, так и новых. Это предоставляет возможность реализации данной технологии на основе любых моделей решения задач.

В работе рассматривается методика проектирования интеллектуальных решателей задач, которая входит в состав комплексной открытой технологии проектирования интеллектуальных систем OSTIS [2].

1. Цель и задачи предлагаемой технологии

В основе данной работы лежит тезис о том, что различных интеллектуальных системах могут применяться различные модели решения задач. Гипотетически возможно

10.7256/2305-6061.2013.01.7

существование универсального решателя задач, объединяющего в себе все известные способы и методы решения задач. Однако использование такого решателя в прикладных целях не является реальным и тем более целесообразным. Таким образом, наиболее приемлемым вариантом становится создание библиотеки совместимых между собой компонентов, из которых впоследствии может быть скомпилирован решатель, удовлетворяющий необходимым требованиям.

Основным требованием, предъявляемым при таком подходе к реализации моделей решения задач, является представление их в соответствующем формальном виде – как многоагентной системы, работающей над общей памятью. Приведение существующих моделей к такому формализму является нетривиальной задачей, хоть и не привносит радикальных изменений в текущее состояние теории решения задач в целом.

На основании сказанного выше, можно выдвинуть следующую гипотезу: любые модели решения задач могут быть представлены в описанном ранее формализме с целью максимального упрощения интеграции данных моделей между собой.

Целью данной работы является создание открытой семантической технологии компонентного проектирования интеллектуальных решателей задач, позволяющей осуществить компиляцию готового решателя из имеющихся компонентов.

К рассматриваемой технологии выдвинуты следующие требования:

- открытость (технология является частью открытого проекта OSTIS);
- расширяемость (возможность свободного добавления новых компонентов);
- гибкость (пользователь может использовать лишь часть компонентов из предложенной библиотеки для обеспечения требуемого функционала разрабатываемой интеллектуальной системы);
- доступность (сравнительно неподготовленный пользователь может использовать технологию для создания интеллектуального решателя задач по интересующей его предметной области);
- универсальность (технология может быть использована в любых предметных областях, обеспечивая при этом полный функционал в случае необходимости);
- модульность (технология включает структурированную библиотеку компонентов обработки знаний);
- полнота (технология должна обеспечивать решение как можно большего набора задач из заданной предметной области за конечное время).
- асинхронность и параллельность (технология должна обеспечивать возможность параллельной асинхронной работы операций и, как следствие, возможность параллельного решения ряда задач)

Таким образом, при достижении поставленной цели возникают следующие проблемы:

- обеспечение предметной независимости и универсальности компонентов технологии и всей технологии в целом;
- обеспечение синхронизируемости отдельных компонентов технологии между собой;
- обеспечение синхронизируемости всей технологии в целом с другими направлениями проекта OSTIS, в частности с технологией проектирования баз знаний;

10.7256/2305-6061.2013.01.7

• обеспечение самодостаточности компонентов (или групп компонентов) технологии, т.е. способности их функционировать отдельно от других компонентов без утраты целесообразности их использования;

• обеспечение антропоморфности, и как следствие, доступности широкому кругу пользователей принципов и методов, рассматриваемых в рамках технологии при решении задач;

• возрастание времени решения задачи при расширении функционала интеллектуального решателя;

• возрастание времени решения задачи при расширении базы знаний системы.

Рассмотрим задачи, детализирующие подход к преодолению описанных проблем:

• разработка четкой системы правил проектирования операций интеллектуального решателя;

• уточнение языка вопросов, используемого при взаимодействии операций с другими компонентами технологии, в том числе и с другими операциями;

• разработка систем операций, обеспечивающих поиск решения задачи по одной из известных стратегий решения или их комбинации;

• разработка систем операций, обеспечивающих поиск решения задачи по одному из известных правил логического вывода или их комбинации, в рамках некоторой стратегии решения;

• реализация разработанных операций и их отладка;

• разработка классификации операций решателя;

Более подробно приведенные задачи описаны ниже.

2. **Состав технологии проектирования интеллектуальных решателей задач**

В состав технологии входят следующие компоненты:

• модель интеллектуального решателя задач;

• библиотека ip-компонентов (intelligent property components – компонентов интеллектуальной собственности) решателя;

• система автоматизации проектирования;

• методика проектирования интеллектуальных решателей задач;

• help-системы поддержки проектирования интеллектуальных решателей задач;

• система управления коллективным проектированием интеллектуальных решателей задач.

Одним из достоинств семантической технологии компонентного проектирования интеллектуальных решателей задач является предметная независимость системы операций, используемых решателем, что позволяет не привязываться к конкретной предметной области.

Далеко не все задачи возможно решить путем применения классической дедуктивной логики. В связи с этим технология предполагает возможность применения различных логических подходов к решению задач в различных предметных областях.

10.7256/2305-6061.2013.01.7

Некоторые логические подходы, которые используются в технологии:

- классическая дедуктивная логика; [3]
- методы индуктивного вывода;
- абдуктивный вывод;
- нечеткие логики;
- правдоподобные рассуждения;
- логика умолчаний;
- темпоральная логика.

Различные логические подходы позволяют проектировать решатели задач для интеллектуальных систем в разных предметных областях, учитывая их специфику.

3. **Модель интеллектуального решателя задач**

В предлагаемом подходе к преодолению приведенных проблем решатель задач рассматривается в неклассическом варианте. В данном случае решатель задач представляет собой графодинамическую sc-машину (память в качестве модели представления знаний использует семантическую сеть), состоящую из двух частей:

- графодинамической sc-памяти;
- систему sc-операций [4].

Система операций является агентно-ориентированной и представляет собой набор sc-операций, условием инициирования которых является появление в памяти системы некоторой определенной конструкции. При этом операции взаимодействуют между собой через память системы посредством генерации конструкций, являющихся условиями инициирования для другой операции. При таком подходе становится возможным обеспечить гибкость и расширяемость решателя путем добавления или удаления из его состава некоторого набора операций. Более подробно процесс проектирования операций и предъявляемые к ним требования рассмотрены в соответствующем разделе.

Отличительной особенностью решателя задач как многоагентной системы в рамках данного подхода является принцип взаимодействия операций-агентов. Агенты обмениваются сообщениями исключительно через общую память путем использования соответствующего языка взаимодействия (языка вопросов-ответов), в отличие от большинства классических многоагентных систем, в которых агенты обмениваются сообщениями непосредственно друг с другом [5]. В рассматриваемом подходе каждый агент, формулируя вопросную конструкцию в памяти, априори не знает, какой из агентов будет обрабатывать указанную конструкцию, а лишь дожидается появления в памяти факта окончания обработки вопроса. При этом в решении поставленной таким образом задачи может принимать участие целый коллектив агентов. Аналогичным образом, реагируя на появление некоторой конструкции в памяти, агент в общем случае не знает, кто из его коллег поставил данный вопрос, а лишь может проверить соответствие сгенерированной конструкции своему условию инициирования. В случае наличия такого соответствия, агент нач-

10.7256/2305-6061.2013.01.7

нет обработку указанного вопроса (решение поставленной задачи), и в результате работы сгенерирует некоторый ответ на поставленный вопрос.

Проверка соответствия сгенерированного вопроса условиям инициирования агентов происходит следующим образом: автору вопроса после его формулирования необходимо инициировать данный вопрос (включить его во множество инициированных вопросов). После инициирования вопроса каждый из агентов, работающих в памяти, переходит в активное состояние и начинает проверку условия инициирования. При этом проверка начинается с наиболее уникальных фрагментов условия (например, типа вопроса) с целью оптимизации данного процесса. В случае установления факта изоморфности вопросной конструкции и условия инициирования агент начинает решение поставленной задачи, в противном случае агент переходит в состояние пассивного ожидания.

Описанная модель взаимодействия агентов в общей памяти позволяет обеспечить максимальную расширяемость системы агентов и предельно упростить процесс добавления новых агентов в уже имеющийся коллектив.

4. **Библиотека универсальных совместимых ip-компонентов решателя**

Центральным элементом семантической технологии проектирования интеллектуальных решателей задач является библиотека совместимых ip-компонентов.

Опишем структуру этой библиотеки.

- Библиотека готовых решателей задач:
  - решатель задач для экспертных систем, построенных на основе продукционной модели представления знаний;
  - решатель задач по геометрии;
  - решатель задач по теории графов;
  - и другие.

- Библиотека стратегий решения задач:
  - разбиение задачи на подзадачи:
    - поиск критерия разбиения задачи;
    - разбиение задачи по заданному критерию;
    - соединение решений, полученных в результате разбиения;
    - удаление дублирующихся знаний;
    - и другие;
  - решение задачи с конца (стратегия обратного вывода)
    - унификация;
    - проверка противоречивости базы знаний;
    - соединение И-подцелей;
    - соединение ИЛИ-подцелей;
    - получение решения;

10.7256/2305-6061.2013.01.7

••• удаление промежуточных утверждений;
••• и другие;
•• упрощение задачи (переход от формулировки в терминах предметной области к формулировке на логическом языке):
••• операция обобщения;
••• вывод обобщенного логического высказывания;
••• фаззификация;
••• дефаззификация;
••• и другие;
•• случайный поиск и метод проб и ошибок (применимо в тех случаях, когда известно, что задача имеет небольшое число путей решения):
••• определения числа возможных путей решения;
••• применения пути решения;
••• оценка эффективности пути решения;
••• получения решения из всех путей;
••• и другие;
•• использование правил для решения типовых задач:
••• поиск подходящего класса задач для данной задачи;
••• решения задачи методами найденного класса задач;
••• поиск наиболее близкого класса задач для данной задачи;
••• и другие;
•• метод деления пополам:
••• разделение множества предполагаемых решений пополам;
••• принятие решения об отказе от одной из половин;
••• восстановление решения;
••• и другие;
•• применение аналогии:
••• генерация логического утверждения по аналогии;
••• восстановление решения после применения аналогии;
••• генерация фактов по аналогии;
••• и другие;
•• метод поиска в глубину:
••• операция поиска в глубину;
••• генерация знаний на заданной глубине поиска;
••• попытка решения задачи после генерации знаний;
••• восстановление шагов решения после применения поиска;
••• и другие;
•• метод поиска в ширину:
••• операция поиска в ширину;
••• построение дерева решений поиска в ширину;
••• нахождения наиболее короткого решения (за наименьшее число шагов);
••• поддержка очереди фактов;

10.7256/2305-6061.2013.01.7

••• и другие;
•• перебор вариантов решения:
••• генерация варианта решения;
••• проверка варианта решения;
••• выполнение «отката»;
••• применение решения;
••• восстановление решения;
••• и другие;
•• генерация всех возможных следствий (прямой логический вывод):
••• поиск всех возможных правил для применения;
••• применение найденных правил;
••• проверка новых фактов на то, что они отсутствуют в базе знаний;
••• дополнение базы знаний сгенерированными фактами;
••• восстановление решения;
••• и другие;

• Библиотека операций:
•• логического вывода:
••• генерация знаний на основании определения (эквиваленции);
••• генерация знаний на основании продукции (импликации);
••• генерация определения на основании двух импликаций;
••• генерация более частного импликативного высказывания на основании более общего;
••• интерпретация арифметического выражения;
••• вывод обобщенного высказывания;
••• и другие;
•• поисковые операции:
••• операция поиска значения;
••• операция поиска формулы для нахождения значения искомой характеристики;
••• операция поиска доказательства;
••• операции других машин интеллектуального поиска;
••• и другие;
•• интерпретации хранимых способов решения задач:
••• алгоритмов;
••• процедурных программ;
••• логических программ;
••• нейронных сетей;
••• генетических алгоритмов;
••• методических указаний к решению задач;
••• и других;
•• «сборки мусора»:
••• удаления всех шаблонов, по которым осуществлялся поиск;
••• удаления всех сгенерированных промежуточных логических утверждений;

10.7256/2305-6061.2013.01.7

••• и другие;
•• мониторинга качества базы знаний:
••• устранение избыточности;
••• устранение противоречивости;
••• проверка полноты;
••• и другие.

• Библиотека базовых преобразований:
•• поиск изоморфоной конструкции по образцу;
•• генерация изоморфной конструкции по образцу;
•• поиск всех выходящих из узла дуг;
•• генерация тройки (узел-дуга-узел);
•• генерация пятерки (узел-дуга с атрибутом-узел);
•• поиск пятерки (узел-дуга с атрибутом-узел);
•• и другие.

• Библиотека программ, реализованных на различных языках программирования и на различных платформах

5. **Средства автоматизации проектирования интеллектуальных решателей задач**

В состав семантической технологии так же входят и инструментальные средства, которые позволяют автоматизировать процесс создания интеллектуальных решателей задач. Эти средства позволяют работать на более высоком уровне абстракции, чем текст программы на языке SCP [6]. Перечислим основные возможности, которые имеет данное средство автоматизации:

• возможность включить (исключить) операцию из проектируемой системы операций интеллектуального решателя;
• верификация спроектированного набора операций;
• проверка операций на предметную независимость, универсальность;
• возможность отладки системы операций на конкретной базе знаний;
• профайлер производительности (отслеживание процессорного времени работы операций);
• профайлер памяти (отслеживание текущего состояния sc-памяти);
• просмотр стека вызовов (последовательность сгенерированных в памяти вопросов, операций, обрабатывающих данные вопросы и сгенерированные ответы на данные вопросы);

Таким образом, средства автоматизации позволяют разработчику интеллектуального решателя задач создавать наборы операций, которые реализовывают различные подходы к решению задач в рамках различных логических подходов.

10.7256/2305-6061.2013.01.7

6. **Методика применения технологии при проектировании конкретных решателей задач**

Технология проектирования интеллектуальных решателей задач основана на задачно-ориентированной методологии. В связи с этим проектирование системы операций состоит из четырех основных этапов:

- создание тестового сборника задач, которые решаются в рамках исследуемой предметной области;
- определение набора предметно независимых операций, которые будут использоваться при решении задач из тестового сборника;
- уточнение семантической спецификации каждой из указанных операций;
- реализация и отладка операций.

В общем случае можно выделить следующие предметно независимые классы задач [7]

- Задачи синтеза доказательства
- Задачи верификации
- Задачи синтеза способа (алгоритма) решения
- Задачи анализа
- Задачи классификации

В качестве примера предметной области рассмотрим геометрию Евклида. Тогда классификация задач тестового сборника будет выглядеть следующим образом:

- по способу решения:
  - вычислительные задачи;
  - задачи на доказательство;
  - задачи на построение;
  - задачи на уточнение;
  - комбинированные задачи;
- по объекту решения:
  - геометрические точки;
  - прямые и отрезки;
  - треугольники;
  - многоугольники;
  - окружности;
  - и другие;
- по размерности пространства:
  - планиметрические;
  - стереометрические.

7. **Методика проектирования операций**

Рассмотрим ряд принципов, соблюдение которых необходимо для корректности работы разрабатываемого решателя задач.

10.7256/2305-6061.2013.01.7

Каждая операция должна быть предметно независимой, т.е. в секции констант данной операции не должны быть описаны константы, имеющие отношение непосредственно к рассматриваемой предметной области. Исключение составляют понятия, которые могут использоваться в различных предметных областях (например, отношения ***«включение*»*** и ***«часть-целое*»***). Данное правило может также быть нарушено в случае, если операция является вспомогательной и ориентирована на обработку какого-либо конкретного класса объектов (например, арифметические операции могут напрямую работать с конкретными отношениями ***«сложение*»*** и ***«умножение*»*** и т.п.). Всю необходимую для решения задачи информацию операция должна извлекать из семантической окрестности запроса.

Операция должна по возможности меньше ориентироваться на фиксированную форму представления фрагментов базы знаний, на работу с которыми она ориентирована. Степень глубины формализации и другие аспекты базы знаний определяются инженером по знаниям и не должны влиять на корректность работы операции. При обнаружении некорректности в представлении рассматриваемого фрагмента базы знаний операция, как и вся система не должна терять управления и, по возможности, сообщить пользователю о некорректности приведенных знаний.

Одна операция может состоять из ряда подпрограмм на выбранном языке программирования. Не стоит путать понятия ***«операция»*** и ***«программа»*** (***«подпрограмма»***). Подпрограмма не является агентом и должна вызываться другой подпрограммой. Операция представляет собой как минимум одну подпрограмму, которая имеет особый формат входных и выходных данных, автоматически реагирует на состояние sc-памяти и при необходимости запускает другие подпрограммы.

При проектировании подпрограмм следует учитывать возможность использования различными операциями одних и тех подпрограмм. Таким образом, появляется необходимость говорить не только о библиотеке sc-операций, но и библиотеках подпрограмм на различных языках программирования, например библиотеке scp-подпрограмм.

Каждая операция должна самостоятельно проверять полноту соответствия условия инициирования конструкции, имеющейся в памяти системы на данный момент. В процессе решения задачи может возникнуть ситуация, когда на появление одной и той же конструкции среагировали несколько операций. В таком случае выполнение продолжает только та операция, условие инициирования которой полностью соответствует сложившейся ситуации. Остальные операции обязаны в данном случае прекратить выполнение.

Выполнение предыдущего пункта достигается за счет тщательного уточнения спецификаций разрабатываемых операций. В общем случае условия инициирования у нескольких операций может совпадать, однако такая ситуация является очень нежелательной и может быть реализована в том случае, если операции не вносят критических изменений в ту область памяти, с которой работают остальные операции.

При проектировании систем операций интеллектуального решателя рекомендуется по возможности использовать операции, уже имеющиеся в библиотеке операций [1]. При необходимости реализации новой операции следует проектировать ее по возможности более общей, однако необходимо выделять в отдельные операции

10.7256/2305-6061.2013.01.7

фрагменты рассуждений, которые могут быть использованы отдельно при решении другого класса задач.

Если в процессе работы операция генерирует в памяти какие-либо временные конструкции, то при завершении работы она обязана удалять всю информацию, использование которой в системе более нецелесообразно. Исключение составляют ситуации, когда подобная информация необходима нескольким операциям для решения одной задачи, однако после решения задачи информация становится бесполезной или избыточной и требует удаления. В данном случае ни одна из операций может оказаться не в состоянии удалить информационный мусор. В таком случае возникает необходимость говорить о специализированных вспомогательных операциях, задачей которых является уничтожение информационного мусора.

Операции необходимо объединять в группы для решения многоходовых задач, т.е. таких задач, для решения которых недостаточно всего одной операции. Очевидно, что под данное определение попадает большинство задач из практически любой предметной области. Группа операций является в некотором смысле самостоятельной подсистемой в рамках целостной системы операций.

При объединении операций в группы рекомендуется проектировать операции таким образом, чтобы они могли быть использованы не только в рассматриваемой группе. В случае, если это не представляется возможным и некоторые операции, будучи отделенными от группы, теряют смысл, необходимо указать данный факт при документировании рассматриваемых операций.

Инициатором запуска операции может бать как непосредственно пользователь системы, так и другая операция. При этом это никак не должно отражаться в работе самой операции. Необходимость вывода (трансляции) какого-либо фрагмента памяти пользователя отслеживается компонентами пользовательского интерфейса.

Язык взаимодействия операций через sc-память представляет собой подмножество языка вопросов. При расширении языка вопросов для введения какой-либо новой операции необходимо максимально сокращать sc-конструкцию, представляющую собой вопрос. В вопросе должна указываться только критически важная информация, все остальные знания операция должна находить самостоятельно в семантической окрестности вопроса. Это позволяет уменьшить сложность восприятия интерфейса операции потенциальным пользователем и унифицировать форматы вопросов у большого числа различных операций.

При выделении в алгоритме решения задачи отдельных операций необходимо учитывать два фактора:

- операции должны быть как можно более универсальными, т.е. использоваться при решении как можно большего числа задач, что позволит избежать повторной реализации одних и тех же фрагментов рассуждений и уменьшит избыточность знаний в системе;
- операции должны быть по возможности антропоморфными, т.е. одна операция должна моделировать некий единый законченный акт мыслительной деятельности человека. Не следует искусственно увязывать ряд действий в одну операцию и наоборот, расчленять одно самодостаточное действие на поддействия. Это вызовет сложности вос-

10.7256/2305-6061.2013.01.7

приятия принципов работы операции разработчиками и не позволит использовать операцию в ряде систем (например, в обучающих системах, которые должны объяснять ход решения пользователю);

Таким образом, в процессе разработке системы операций можно выделить следующие этапы:

- определение необходимого набора операций (с учетом уже имеющихся в библиотеке);
- определение ключевых узлов языка вопросов для связи операций через графодинамичекую память;
- составление спецификаций каждой из операций;
- реализация и тестирование операций;

Заключение

Таким образом, семантическая технология проектирования интеллектуальных решателей задач позволяет проектировать такие системы предметно независимых операций, которые способны решать унифицированным образом множество различных задач одного класса. Каждая из операций представляет собой ip-компонент, который может быть использован в других прикладных системах. Многоагентная модель позволяет легко осуществлять интеграцию компонентов машин обработки знаний при условии корректной интеграции баз знаний. При этом никаких дополнительных действий по интеграции машин обработки знаний не требуется, т.к. грамотно разработанная операция многоагентной модели самостоятельно контролирует условие инициирования и текущее состояние памяти.

Основная цель подобной работы – позволить даже относительно неподготовленному (в области проектирования интеллектуальных систем) человеку создать интеллектуальную справочную систему по интересующей его предметной области, обладающую гибким функционалом, который определяется разработчиком на стадии проектирования.

Список литературы:

1. *Гаврилова Т.А., Хорошевский В.Ф.* Базы знаний интеллектуальных систем. Учебник / *Гаврилова Т.А.*. [и др.]; – СПб. : Изд-во «Питер», 2001.
2. Проект OSTIS [Электронный ресурс]. Минск, 2011. – Режим доступа: http://ostis.net/. – Дата доступа: 01.10.2012.
3. Достоверный и правдоподобный вывод в интеллектуальных системах / *Вагин В.Н.* [и др.]; – М. : ФИЗМАТЛИТ, 2008.
4. *Голенков, В.В.* Представление и обработка знаний в графодинамических ассоциативных машинах / *Голенков В.В.* [и др.]; под ред. В.В. Голенкова – Минск, 2001.
5. *Тарасов, В.Б.* От многоагентных систем к интеллектуальным организациям / *В.Б. Тарасов*; – М. :Изд-во УРСС, 2002.

6. Программирование в ассоциативных машинах / *Голенков В. В.* [и др.]; под ред. *В. В. Голенкова* – Минск, 2001.
7. *Пойя Д.* Как решать задачу. / *Д. Пойя*; — М.: Либроком, 2010.

Библиография:

1. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем. Учебник / Гаврилова Т.А.. [и др.]; – СПб. : Изд-во «Питер», 2001.
2. Голенков, В.В. Представление и обработка знаний в графодинамических ассоциативных машинах / Голенков В.В. [и др.]; под ред. В.В. Голенкова – Минск, 2001.
3. Достоверный и правдоподобный вывод в интеллектуальных системах / Вагин В.Н. [и др.]; – М. : ФИЗМАТЛИТ, 2008.
4. Пойя Д. Как решать задачу. / Д. Пойя; — М.: Либроком, 2010.
5. Программирование в ассоциативных машинах / Голенков В. В. [и др.]; под ред. В. В. Голенкова – Минск, 2001.
6. Проект OSTIS [Электронный ресурс]. Минск, 2011. – Режим доступа: http://ostis.net/. – Дата доступа: 01.10.2012.
7. Тарасов, В.Б. От многоагентных систем к интеллектуальным организациям / В.Б. Тарасов; – М. :Изд-во УРСС, 2002.

References (transliteration):

1. Gavrilova T.A., Khoroshevskiy V.F. Bazy znaniy intellektual’nykh sistem. Uchebnik / Gavrilova T.A.. [i dr.]; – SPb. : Izd-vo «Piter», 2001.
2. Golenkov, V.V. Predstavlenie i obrabotka znaniy v grafodinamicheskikh assotsiativnykh mashinakh / Golenkov V.V. [i dr.]; pod red. V.V. Golenkova – Minsk, 2001.
3. Dostovernyy i pravdopodobnyy vyvod v intellektual’nykh sistemakh / Vagin V.N. [i dr.]; – M. : FIZMATLIT, 2008.
4. Poyya D. Kak reshat’ zadachu. / D. Poyya; — M.: Librokom, 2010.
5. Programmirovanie v assotsiativnykh mashinakh / Golenkov V. V. [i dr.]; pod red. V. V. Golenkova – Minsk, 2001.
6. Proekt OSTIS [Elektronnyy resurs]. Minsk, 2011. – Rezhim dostupa: http://ostis.net/. – Data dostupa: 01.10.2012.
7. Tarasov, V.B. Ot mnogoagentnykh sistem k intellektual’nym organizatsiyam / V.B. Tarasov; – M. :Izd-vo URSS, 2002.