Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

МОДЕЛИ РЕШЕНИЯ ЗАДАЧ В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ

Рекомендовано УМО по образованию в области информатики и радиоэлектроники в качестве пособия по выполнению курсового проекта для специальности 1-40 03 01 «Искусственный интеллект»

Минск БГУИР 2015

УДК 004.8:164(076)
ББК 32.813я7+22.12я7
М74

Авторы:
В. В. Голенков, Н. А. Гулякина, Н. В. Гракова, И. Т. Давыденко,
И. И. Жуков, Д. В. Шункевич

Рецензенты:
кафедра интеллектуальных систем Белорусского государственного университета (протокол №11 от 01.04.2014);

старший научный сотрудник государственного научного учреждения «Объединенный институт проблем информатики Национальной академии наук Беларуси», кандидат технических наук, доцент Н. А. Кириенко

Модели решения задач в интеллектуальных системах : пособие / М74 В. В. Голенков [и др.]. – Минск : БГУИР, 2015. – 70 с. : ил.
ISBN 978-985-543-096-5.

Сформулированы основные положения, касающиеся выполнения курсового проектирования, даны рекомендации по решению типовых задач в рамках курсового проекта, а также рассмотрены типичные ошибки и проблемы, возникающие в процессе выполнения курсового проекта.

УДК 004.8:164(076)
ББК 32.813я7+22.12я7

ISBN 978-985-543-096-5

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

При изучении дисциплины «Модели решения задач» студенты знакомятся с проблемами и основными понятиями методов построения моделей решения задач и обработки знаний, а также углубляют и систематизируют знания и навыки в области компьютерных архитектур для распределенной сетевой обработки знаний.

Целью выполнения курсового проекта по данной дисциплине является закрепление следующих навыков:

- использование на практике моделей параллельной обработки знаний;
- использование на практике конкретных архитектур для распределенной обработки знаний при решении реальных прикладных задач;
- применение различных методов решения задач в интеллектуальных системах.

Для достижения цели необходимо решить следующие задачи:

1) проанализировать предметную область и выделить наиболее характерные для нее задачи вычислительного и невычислительного характера;
2) для выделенных задач разработать фрагменты базы знаний, описывающие исходные данные и необходимые для решения логические утверждения;
3) протестировать возможность решения выделенных задач при помощи предлагаемого универсального решателя задач;
4) доработать предложенный универсальный решатель задач для расширения перечня задач, решаемых в рамках конкретной предметной области.

Пособие включает в себя:

1) описание необходимого программного обеспечения;
2) описание предлагаемого универсального решателя задач;
3) примеры решения вычислительной и невычислительной задачи на основе предлагаемого решателя;
4) описание наиболее типичных ошибок, возникающих в процессе работы.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Интеллектуальный решатель задач, как правило, включающий также информационно-поисковую машину и набор служебных операций обработки знаний (операции сборки мусора, выявления противоречий в базе знаний и т. д.), является важнейшей частью любой интеллектуальной системы, т. к. именно возможностями решателя задач определяется функционал системы в целом, возможность давать ответы на нетривиальные вопросы пользователя и способность решать различные задачи.

В настоящее время большую актуальность имеет переход от ориентирования проектировщиков интеллектуальных систем с навязываемого (предлагаемого) решателя задач на проектирование решателей задач из готовых компонентов. В связи с этим становится важным формирование общих принципов организации решателей задач, позволяющих осуществить интегрирование различных моделей решения задач, как существующих, так и новых. Это предоставляет возможность использовать в конкретной интеллектуальной системе любые модели решения задач.

В рамках данного курсового проекта основной задачей является выделение и формализация наиболее характерных задач, решаемых в какой-либо предметной области, а также доработка набора операций (или агентов), входящих в состав предложенного универсального решателя задач.

Любой агент представляет собой открытую систему, помещенную в некоторую среду, причем эта система обладает собственным поведением, удовлетворяющим некоторым экстремальным принципам. Таким образом, агент считается способным воспринимать информацию из внешней среды с ограниченным разрешением, обрабатывать ее на основе собственных ресурсов, взаимодействовать с другими агентами и действовать на среду в течение некоторого времени, преследуя свои собственные цели.

В рамках предлагаемого в курсовом проекте подхода любая машина обработки знаний, в том числе интеллектуальный решатель задач, представляет собой графодинамическую sc-машину (память в качестве модели представления знаний использует семантическую сеть), состоящую из двух частей:

- графодинамической sc-памяти;
- системы sc-операций.

Условием инициирования каждой из операций является появление в памяти системы некоторой определенной конструкции [1, 2, 3].

В качестве модели представления знаний в интеллектуальных системах , разрабатываемых по предлагаемой технологии, используется семантическая сеть.

Семантическая сеть – это ориентированный граф, вершины которого – понятия, а дуги – отношения между ними [4].

В процессе выполнения курсового проекта необходимо формализовать в базе знаний условия выделенных в рамках выбранной предметной области задач, а также набор логических утверждений, необходимый для их решения.

База знаний – совокупность знаний предметной области, записанная на машинный носитель в форме, понятной эксперту и пользователю (обычно на некотором языке, приближенном к естественному). Параллельно такому «человеческому» представлению существует база знаний во внутреннем «машинном» представлении [4].

Принципы проектирования баз знаний на основе предлагаемых средств описаны в работах [1, 5, 6].

В разрабатываемой системе понятие программного агента в целом конкретизируется до понятия абстрактного sc-агента. Под абстрактным sc-агентом понимается некоторый класс функционально эквивалентных sc-агентов, разные экземпляры (т. е. представители) которого могут быть реализованы по-разному.

Каждый абстрактный sc-агент имеет соответствующую ему спецификацию. В спецификацию каждого абстрактного sc-агента входит:

- указание ключевых sc-элементов этого sc-агента, т. е. тех sc-элементов, хранимых в sc-памяти, которые для данного sc-агента являются «точками опоры»;
- формальное описание условий инициирования данного sc-агента, т. е. тех ситуация в sc-памяти, которые инициируют деятельность данного sc-агента;
- формальное описание первичного условия инициирования данного sc-агента, т. е. такой ситуации в sc-памяти, которая побуждает sc-агента перейти в активное состояние и начать проверку наличия своего полного условия инициирования;
- строгое, полное, однозначно понимаемое описание деятельности данного sc-агента, оформленное при помощи каких-либо понятных, общепринятых средств, не требующих специального изучения, например на естественном языке;
- описание результатов выполнения данного sc-агента.

Абстрактные sc-агенты делятся на неатомарные и атомарные абстрактные sc-агенты. Под неатомарным абстрактным sc-агентом понимается абстрактный sc-агент, который декомпозируется на коллектив более простых абстрактных sc-агентов, каждый из которых в свою очередь может быть как атомарным абстрактным sc-агентом, так и неатомарным абстрактным sc-агентом.

Рассмотрим ряд принципов, соблюдать которые рекомендуется при проектировании операций решателя задач в произвольной предметной области.

Каждая операция должна быть по возможности предметно независимой, т. е. по возможности меньше опираться на константы, имеющие отношение непосредственно к рассматриваемой предметной области. Исключение составляют понятия, которые могут использоваться в различных предметных

областях. 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.). Полный перечень указанных отношений представлен в прил. 1.

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 сопоставляется с как минимум одной подпрограммой, 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 же подпрограмм.

Каждая операция должна самостоятельно проверять полноту соответствия условия инициирования конструкции, имеющейся в памяти системы на данный момент. В процессе работы системы 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.

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 общей задачи, однако позже информация становится бесполезной или избыточной и требует удаления. В данном случае ни одна из операций не может оказаться в состоянии удалить информационный мусор. В таком случае возникает необходимость говорить о специализированных вспомогательных операциях, задачей которых является уничтожение информационного мусора.

При необходимости операции можно объединять в группы для решения более сложных задач. Очевидно, что такого рода задачи могут рассматриваться в рамках практически любой предметной области. Подобная группа операций является в некотором смысле самостоятельной подсистемой в рамках целостной системы операций. При объединении операций в группы

рекомендуется проектировать их таким образом, чтобы они могли быть использованы не только в рассматриваемой группе и, будучи отделенными от группы, не теряли смысл.

Инициатором запуска какой-либо операции может быть как непосредственно пользователь системы, так и другая операция. При этом за счет организации взаимодействия через общую память это никак не отражается в алгоритме работы самой операции. Необходимость вывода (трансляции) какого-либо фрагмента памяти конечному пользователю отслеживается компонентами пользовательского интерфейса самостоятельно.

Язык взаимодействия операций через sc-память описан в [7]. В вопросе должна указываться только критически важная информация, т. е. параметры запроса, все остальные знания операция способна найти самостоятельно, анализируя память в семантической окрестности вопроса. Это позволяет уменьшить сложность восприятия принципов работы с операцией потенциальным пользователем и унифицировать форматы вопросов у большого числа различных операций.

При разработке алгоритмов работы проектируемых операций необходимо учитывать ряд факторов:

- 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 количество хранимых в системе знаний;
- 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 восприятия 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;
- определение ключевых узлов языка вопросов для связи операций посредством графодинамической памяти;
- разработка алгоритма работы каждой из операций;
- реализация и тестирование коллектива операций.

1. НАСТРОЙКА ПРОГРАММНЫХ СРЕДСТВ В ОПЕРАЦИОННОЙ СИСТЕМЕ WINDOWS

Поэтапно необходимо выполнить следующие шаги:

1. Скачать Python 2.6. Это можно сделать по следующей ссылке: https://www.dropbox.com/s/ic9wgcorhzhcn9p/Python26.rar

2. Добавить путь к скачанной версии Python в Переменные среды, для этого необходимо выполнить следующие действия:

2.1. Зайти в Пуск→Панель управления→Система и безопасность→Система (либо правой кнопкой мыши щелкнуть по ярлыку «Мой компьютер», в контекстном меню выбираем Свойства, либо сочетание клавиш Window+Pause+Break).

2.2. Выбрать Дополнительные параметры системы (рис. 1.1).

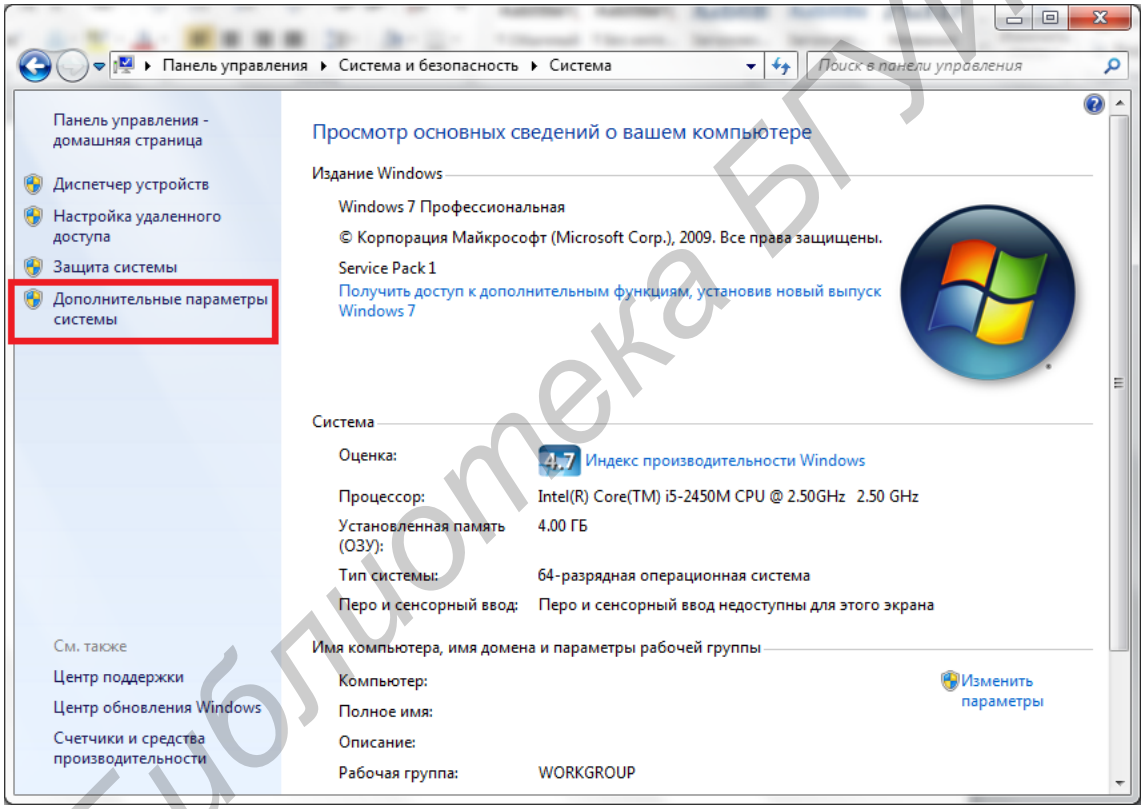

Рис. 1.1. Система Windows. Основные сведения о компьютере

2.3. В появившемся окне выбрать «Переменные среды…» (рис. 1.2).

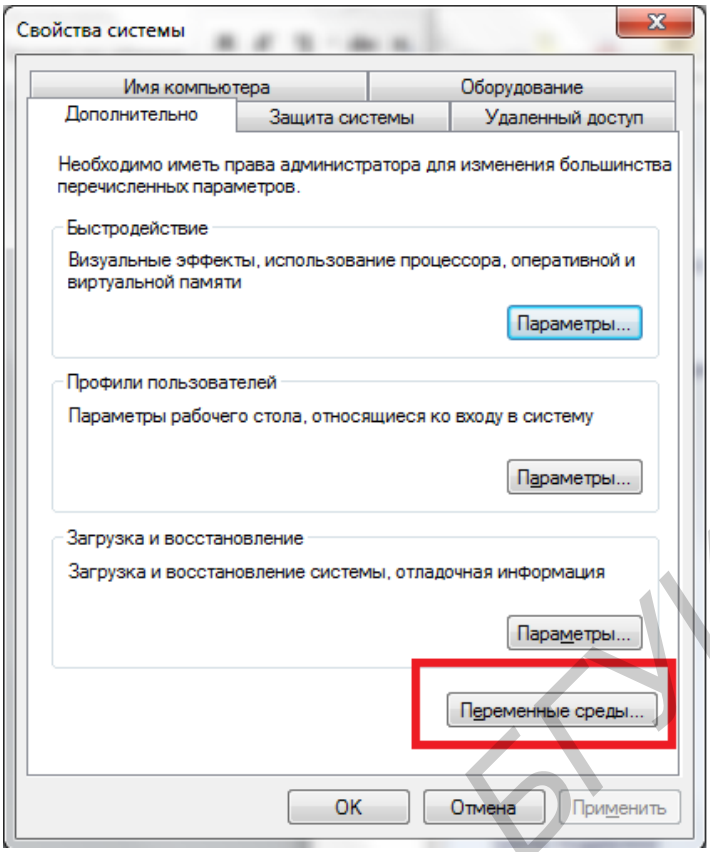

Рис. 1.2. Свойства системы

2.4. В открывшемся окне в области «Системные переменные» найти переменную Path и нажать Изменить (рис. 1.3).

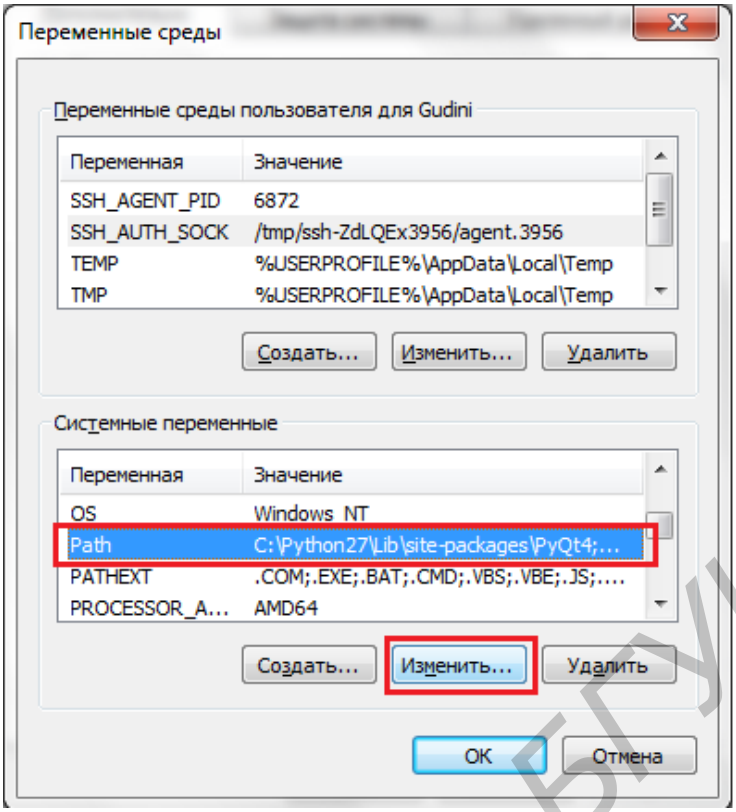

Рис. 1.3. Переменные среды

2.5. В окне в области Значение переменной нужно вписать путь к скаченному ранее Python 2.6.

Примечание. В области Значение переменной уже есть пути к переменным, поэтому необходимо, перед тем как вписать путь к Python 2.6, поставить точку с запятой «;» (рис. 1.4).

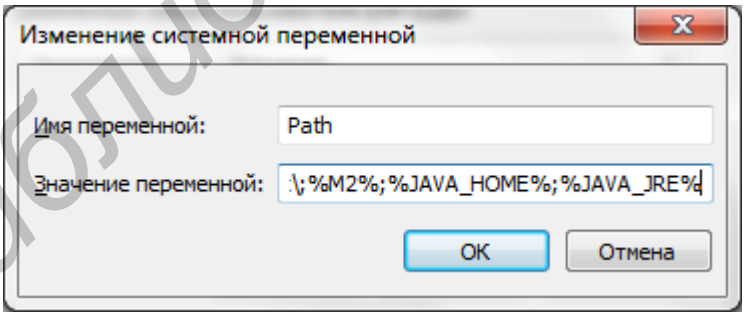

Рис. 1.4. Изменение системной переменной

2.6. Для проверки корректной установки Python запустить Командную строку (Пуск→Все программы→Стандартные→Командная строка). В командной строке прописать следующий тест:

python –V

В консоль должна вывестись версия языка python (рис. 1.5).

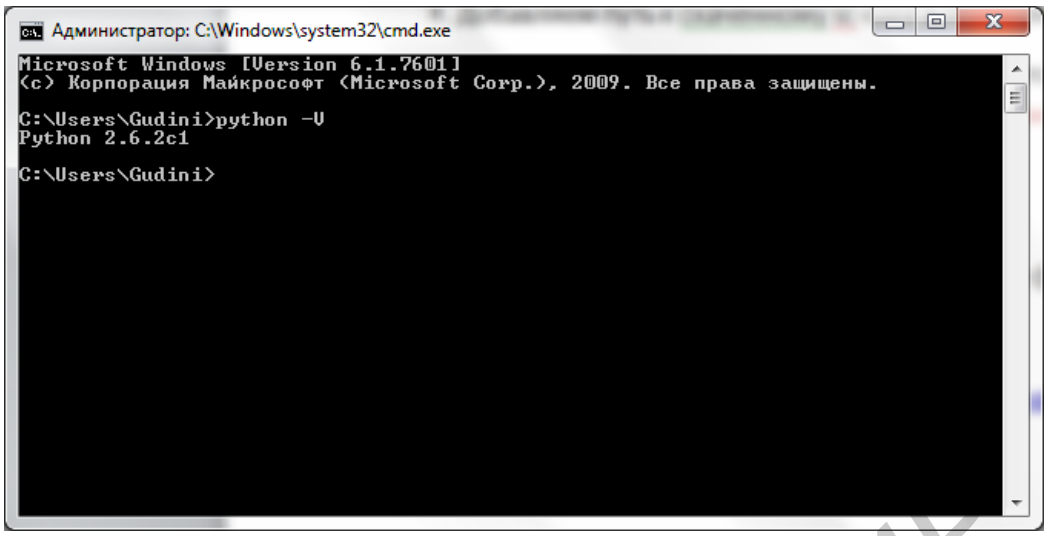

Рис. 1.5. Консоль Windows

3. Скачать sc-core. Это можно сделать по следующим ссылкам:

– для 32-разрядной Windows:

https://www.dropbox.com/sh/7zi37thai67badz/fFZBNe6a4w

– для 64-разрядной Windows:

https://www.dropbox.com/sh/xojxciamrxcpi4v/BiU7arw5l1

4. Добавить путь к скачанному sc-core в переменные среды, для этого в окне Переменные среды (как открыть это окно, смотрите подробно пп. 2.1–2.4) в области Системные переменные нужно нажать кнопку Создать:

– имя переменной: SC_CORE_HOME

– значение переменной: путь к sc-core (к примеру: c:\sc-core\ )

5. Скачать py_ui по ссылке

https://www.dropbox.com/s/r2yfwi37lxn54rq/py_ui.rar

2. ОПИСАНИЕ ПРОЦЕССА РАБОТЫ С РЕШАТЕЛЕМ ЗАДАЧ В КОНСОЛЬНОМ РЕЖИМЕ

1. Исходные тексты базы знаний (БЗ) в виде GWF поместить в папку:

py_ui/ \repo\fs_repo_src\seb\planimetry_src

2. Отредактировать один из файлов temp*.m4scp *py_ui\repo\fs_repo_src\operation*). Рекомендуется использовать temp.m4scp для расчетных задач, temp3.m4scp для остальных.

Для того чтобы это сделать, необходимо открыть файл в KBE, либо с помощью Блокнота или Notepad. В открывшемся файле найти

соответствующие фрагменты и исправить нужные значения. Подробности в комментариях к исходному коду.

Для расчетных задач (файл temp.m4scp):

```
program(temp,
[[
       value="/seb/planimetry/S"; // Вместо S необходимо указать
имя узла, которое является результатом
]],

…

genElStr3([
     1_: fixed_: q_var_value, //Запрос значения величины – для
расчетных задач
  2_: assign_: const_: pos_: arc,
     3_: assign_: node_: const_: temp
])

genElStr3([
     1_: fixed_: temp,
     2_: assign_: const_: pos_: arc,
     3_: fixed_: value //Узел – знак величины, значение которой
надо узнать
])

genElStr3([
     1_: fixed_: question, //Указание того факта, что узел temp –
знак вопроса. Необязательно
     2_: assign_: const_: pos_: arc,
     3_: fixed_: temp
])

genElStr3([
     1_: fixed_: q_initiated, //Инициирование вопроса в памяти, в
последнюю очередь
     2_: assign_: const_: pos_: arc,
     3_: fixed_: temp
])
```

Для нерасчетных задач (файл temp3.m4scp):

```
genElStr3([
     1_: fixed_: q_validity, //Запрос истинности – для нерасчетных
задач
  2_: assign_: arc,
     3_: assign_: const_: node_: temp
])

genElStr3([
     1_: fixed_: temp,
     2_: assign_: arc,
```

```
     3_: fixed_: link //Узел - знак высказывания, истинность
которого хотим установить(либо атомарное, либо импликативное)
])

genElStr3([
     1_: fixed_: q_initiated, //Инициирование вопроса в памяти, в
последнюю очередь
     2_: assign_: arc,
     3_: fixed_: temp
])
```

3. Отредактировать файл preprocessing.m4scp, строка 45

```
callReturn([
          1_: fixed_: "/operation/temp3/temp3", //Указываем здесь
     тот файл, который правили в предыдущем пункте
          2_: fixed_:
          {[
          ]}
])
```

4. Сборка БЗ. Для этого запустить файл *py_ui \repo\ rule_builder.py*

Примечание. Каждый раз, когда изменяются gwf-файлы, необходимо пересобрать БЗ.

5. В папке sc-core отредактировать Start.bat (путь: sc-core/bin):

*start-pm --fsrepo **D:\BSUIR\OSTIS\repo\fs_repo** 1>out.txt*

Здесь указывается путь к репозиторию у вас на компьютере.

6. Запустить Start.bat и дождаться завершения работы системы. В зависимости от сложности поставленной задачи и мощности компьютера решение может занимать от нескольких секунд до нескольких минут.

7. Файл out.txt содержит информацию о результатах работы решателя.

В случае успешного завершения решения расчетной задачи ответ выглядит так:	В случае успешного завершения решения нерасчетной задачи ответ выглядит так:
REQUESTED VALUE IS: **18.0**	**REQUESTED STATEMENT WAS PROVED!**

3. ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ О РАБОТЕ С СИСТЕМОЙ

Информация, приведенная в данном разделе, может помочь при отлаживании задачи.

1. Использованные в решении утверждения (можно найти по слову «was»).

EQ was genered for statement:	**IMPL was genered for statement:**
printEl: node|const:/seb/planimetry/Неравенство треугольника	printEl: node|const:/seb/planimetry/Формула Герона

Указываются утверждения, которые были успешно применены в ходе решения задачи. Порядок записей в логе соответствует порядку использования утверждений. Если какое-либо утверждение должно было быть использовано, но свидетельств этому нет, то ошибка кроется, как правило, в том, что в памяти нет конструкции, которая соответствует посылке утверждения. То есть либо ошибка в самом утверждении, либо ранее сгенерирована несоответствующая конструкция.

2. Просмотренные в процессе решения объекты (можно найти по слову «pts»).

PTS VALUE
printEl: node|const:/seb/planimetry/Отр(ТчкВ1;ТчкС1)

Указываются объекты, которые просмотрел решатель в процессе работы. Ниже указываются классы, которым принадлежит данный объект.

CLASS
printEl: node|const:/seb/planimetry/отрезок

Если указанная системой ситуация вам непонятна, то ошибка скорее всего кроется в описании исходных данных задачи (фактографическая часть знаний).

4. ПРИМЕРЫ РЕШЕНИЯ ЗАДАЧ

4.1. Пример расчетной задачи

Условие. Определить полную поверхность конуса, если высота h=21, L=29.

Решение.

1. Необходимо указать значения величин, используемых в задаче (рис. 4.1).

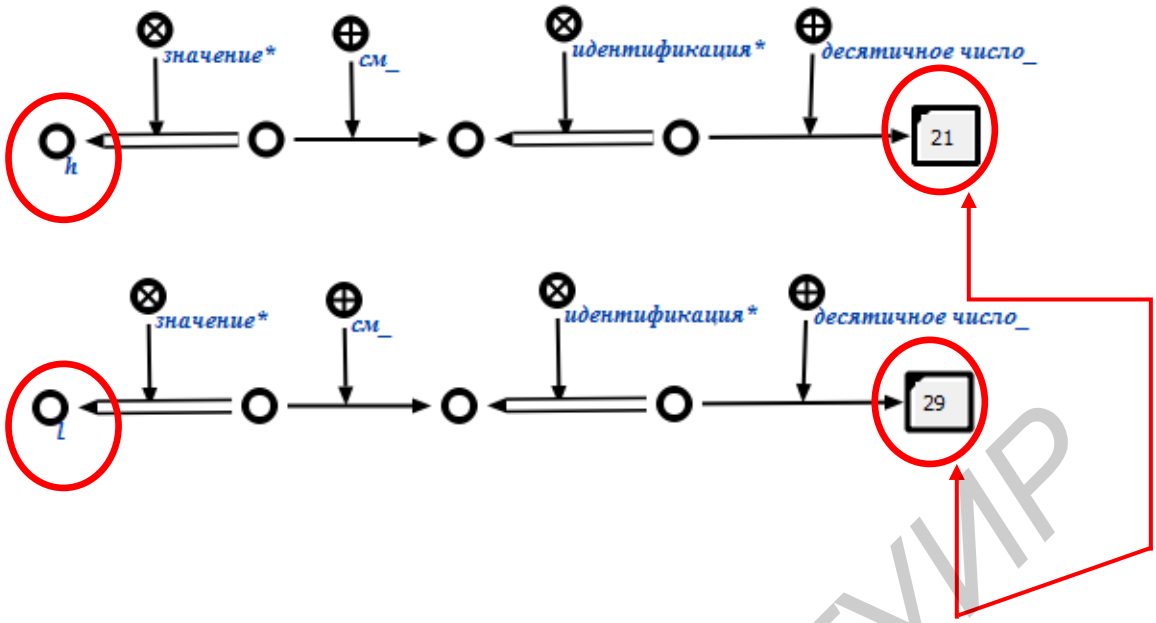

Рис. 4.1. Указание основных значений величин

Обратите внимание на то, что система работает с этими данными.

Здесь представлены основные значения. Теперь необходимо указать вспомогательные, которые понадобятся при расчетах – это значение PI и возведение в квадрат, т. е. цифра 2 (рис. 4.2).

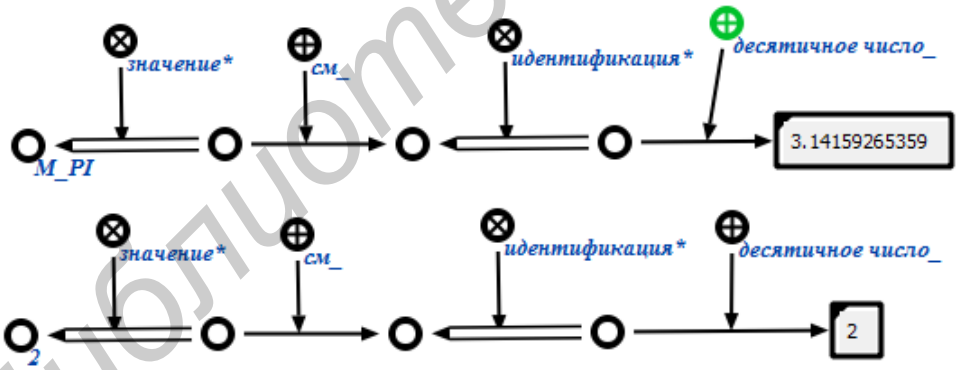

Рис. 4.2. Указание дополнительных значений величин

Итак, значения составлены. Для удобства лучше всего поместить их в отдельный файл под названием Значения.gwf.

Столь длинная цепочка используется для того, чтобы полно описать значение, которое поступает в систему: формат числа (десятичное, двоичное и т. д.), единицы измерения (см, дм, м).

2. Теперь перейдем к условию задачи и формализуем те данные, которые нам даны (для удобства храним условие в файле Условие задачи.gwf). Здесь необходимо указать, что дан конус, у него есть образующая, высота и радиус.

Также в условии можно указать все утверждения, которые связаны с решением данной задачи (более детально будет рассмотрено далее).

На рис. 4.3 формализованы данные о конусе. Как говорилось ранее, указываем, что у конуса есть высота, образующая и радиус. Также указываем, что все они имеют значения. Обратите внимание, что название узла h соответствует тому, что было указано в Значения.gwf. То есть в файле Значения.gwf указано название узла с маленькой буквы h, следовательно, и здесь необходимо указать с маленькой (см. выделенные окружностями и стрелками моменты). Обратите внимание, как именован узел с образующей.

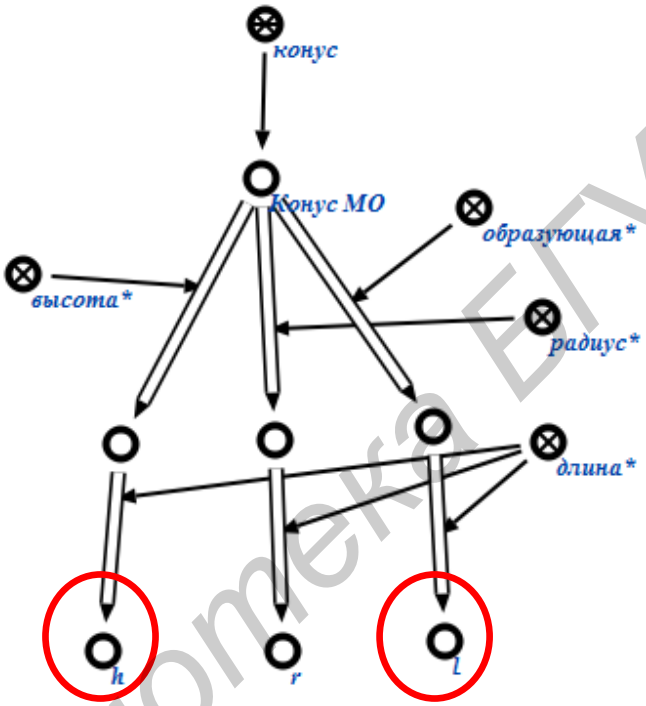

Рис. 4.3. Представление конуса в формализованном виде

На рис. 4.4 показаны все утверждения, которые будут использоваться при решении задачи. Обратите внимание на написание «утверждения о классе*». Каждое утверждение соотносится с тем классом объекта, с которым оно будет использоваться в утверждении. То есть, к примеру, в утверждении на рис. 4.5 говорится о площади конуса, следовательно, данное утверждение соотносится с классом конуса, как и показано на рис. 4.4.

Рис. 4.4. Утверждения о классе

3. Более подробно рассмотрим каждое из утверждений.

3.1. Формула вычисления площади полной поверхности конуса (рис. 4.5).

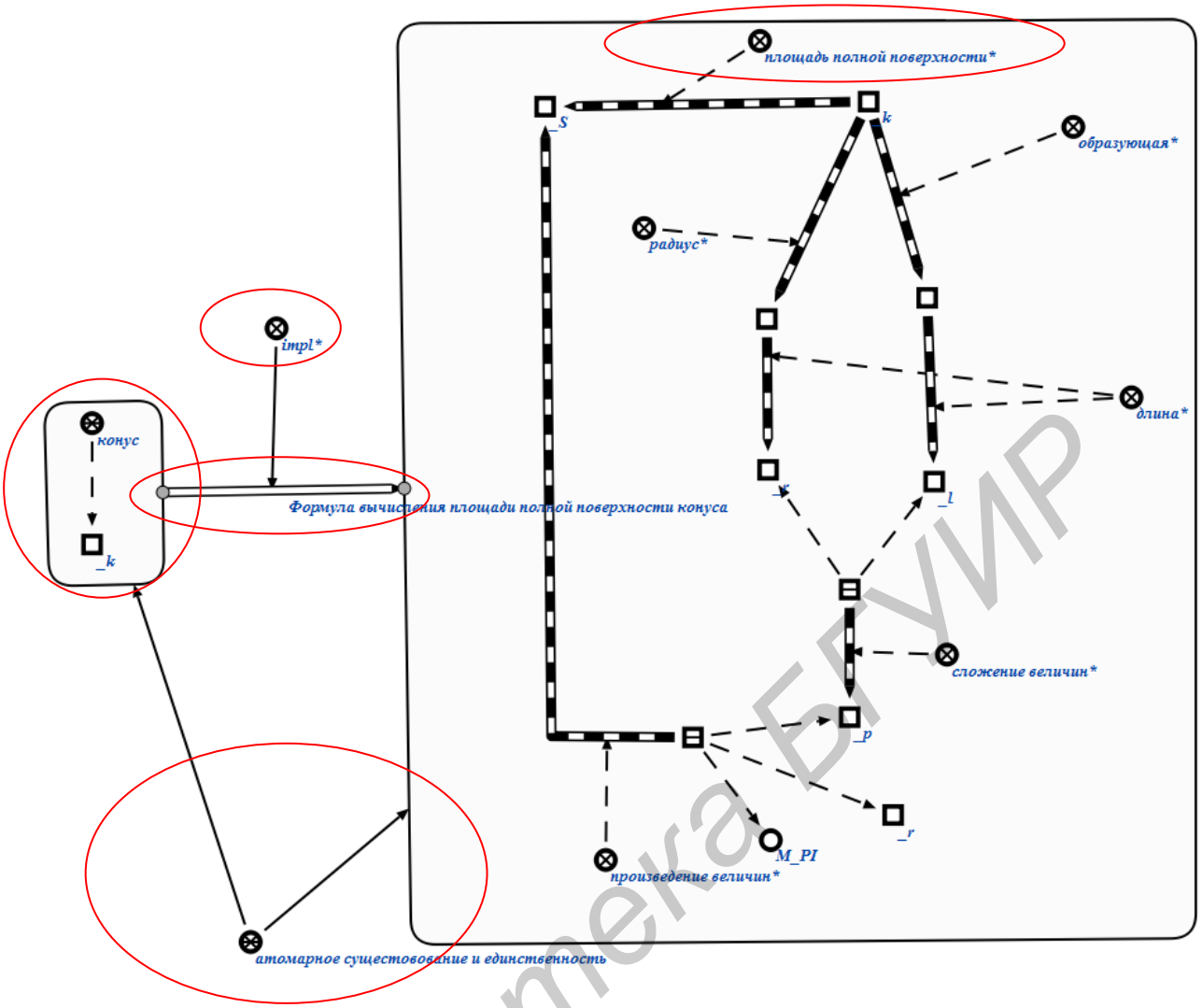

Рис. 4.5. Формула вычисления площади полной поверхности конуса

Обратите внимание на выделенные моменты. Не забывайте именовать бинарную дугу как «Формула вычисления площади полной поверхности конуса», т. е. название должно соответствовать тому, как написано в утверждении, иначе оно не будет найдено. Из утверждения видно, что для вычисления площади полной поверхности необходимо знать радиус основания конуса. Чтобы его найти, нужно установить соответствие между образующей и высотой (рис. 4.6).

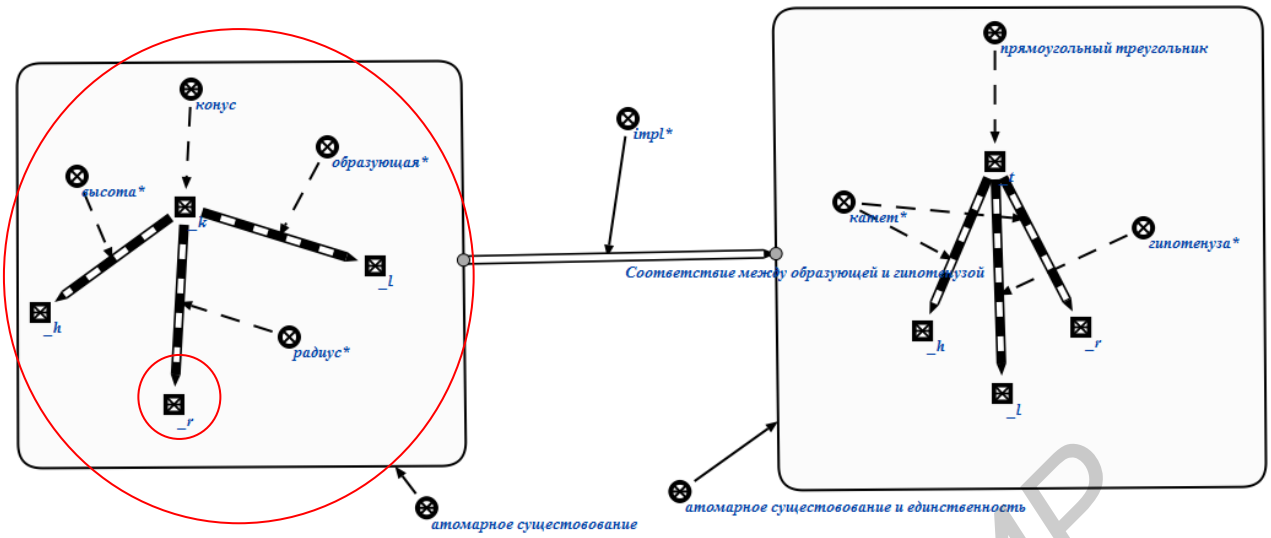

Рис. 4.6. Соответствие между образующей и гипотенузой

Из рисунка видно, что высота, образующая и радиус конуса образуют прямоугольный треугольник. Следовательно, необходимо формализовать утверждение о теореме Пифагора, откуда и будет найден радиус. Теорема Пифагора представлена на рис. 4.7.

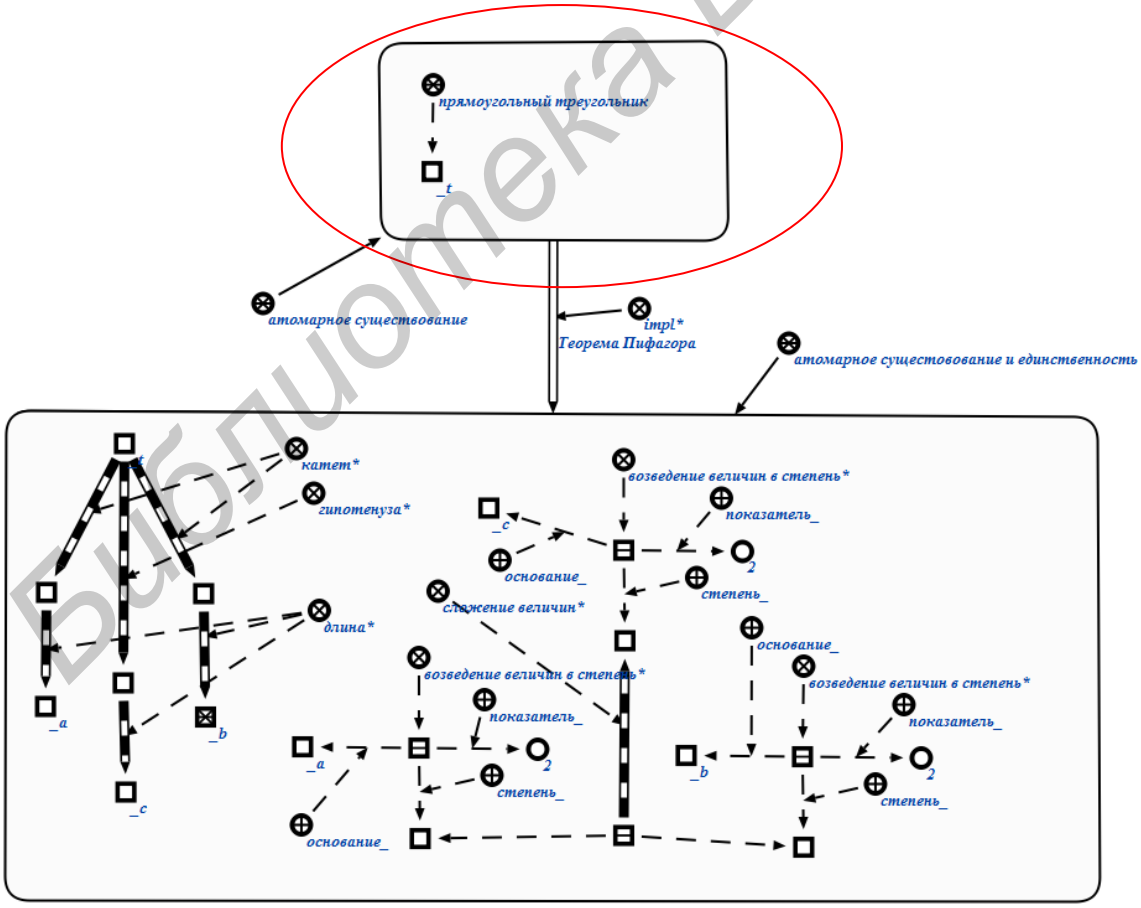

Рис. 4.7. Теорема Пифагора

Из данного утверждения система сможет найти значение радиуса и подставить в основное утверждение для нахождения площади полной поверхности конуса.

Рекомендация. Обратите внимание, что утверждения записываются в максимально общем виде (см. рис. 4.5 левая часть утверждения, выделенные моменты). То есть в левой части, к примеру, указывается, что это конус или прямоугольный треугольник, без лишней информации. Но не во всех случаях рекомендуется так делать.

Из рисунка 4.6 видно, что в левой части неизвестен радиус основания конуса. В данном случае радиус будет являться катетом прямоугольного треугольника. Поэтому на рис. 4.7 подробно было описано, что будет катетами, а что гипотенузой в прямоугольном треугольнике.

Обратите внимание, что решатель может выражать недостающие переменные в формуле и считать их значение. К примеру, по теореме Пифагора можно вычислить любой из катетов или гипотенузу, используя только одну формализованную формулу, что и произошло на рис. 4.7. Был известен катет и гипотенуза, а уже сам решатель, используя формализованное выражение теоремы Пифагора, смог найти значение недостающего катета.

В утверждениях только правая часть помечается как «атомарное существование и единственность». Это делается в том случае, если при указанной левой части существует только одна правая. Во всех остальных случаях просто ставится «атомарное существование».

Теперь необходимо сформировать файл с вопросом (вопрос.gwf) (рис. 4.8).

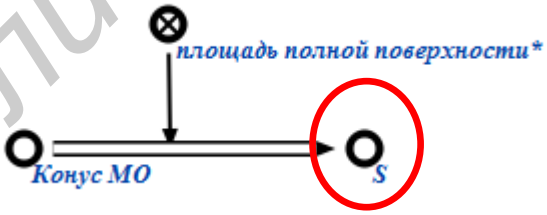

Рис. 4.8. Вопрос системе

В итоге, должны получиться следующие файлы:

1) вопрос.gwf;

2) значения.gwf;

3) условие задачи.gwf;

4) площадь полной поверхности конуса.gwf;

5) соответствие между образующей и гипотенузой.gwf;

6) теорема Пифагора.gwf.

Первых три файла являются обязательными для всех расчетных задач. Остальные варьируются в зависимости от задачи.

Данные файлы необходимо поместить в папку:

…py_ui \repo\fs_repo_src\seb\planimetry_src\

Если в папке уже есть какие-то файлы, то можно их удалить, чтобы избежать различного рода конфликтов и ошибок.

Теперь необходимо собрать БЗ. Для этого запустить на выполнение файл rule_builder.py.

Для удобства можно создать файл с расширением .bat и туда поместить следующий текст:

```
rule_builder.py

PAUSE
```

В дальнейшем, для того чтобы собрать БЗ, просто запускать этот файл.

В случае успешного завершения будет выведено сообщение: Find 0 errors (рис. 4.9).

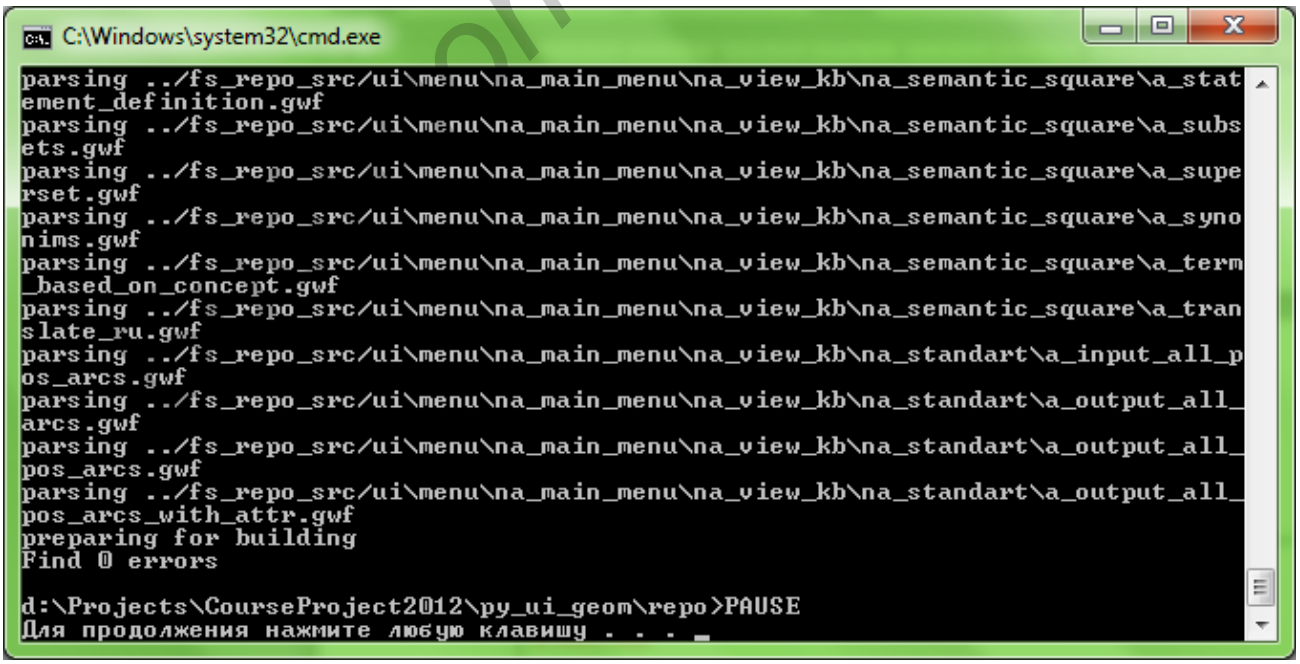

Рис. 4.9. Консоль. Сборка БЗ завершена

Если в сборке будут найдены какие-то ошибки, то проблема скорее всего в файлах SCP, которые менялись. К примеру, пропущена запятая либо она лишняя.

В файле py_ui_geom\repo\fs_repo_src\operation\temp.m4scp необходимо изменить переменную *value*:

```
program(temp,
[[
     sabc="/seb/planimetry/VM";
     //#abc="/seb/planimetry/TR";
     //#new="/seb/planimetry/New";
     value="/seb/planimetry/R1";//вместо R1 необходимо написать S
     value1="/seb/planimetry/S_OR";
     value2="/seb/planimetry/BA1";
     node=nrel_multiplication;
//#  here=/"\n\nHandler setted!\n\n"/; ]],
```

Обратите внимание, что здесь указывается имя узла, которое было в вопросе (вопрос.gwf).

То есть **value="/seb/planimetry/S";**

Это необходимо для того, чтобы система нашла то, что нужно посчитать.

Теперь можно запускать систему для решения задачи. Для этого запускаем sc-core\bin\Start.bat (рис. 4.10).

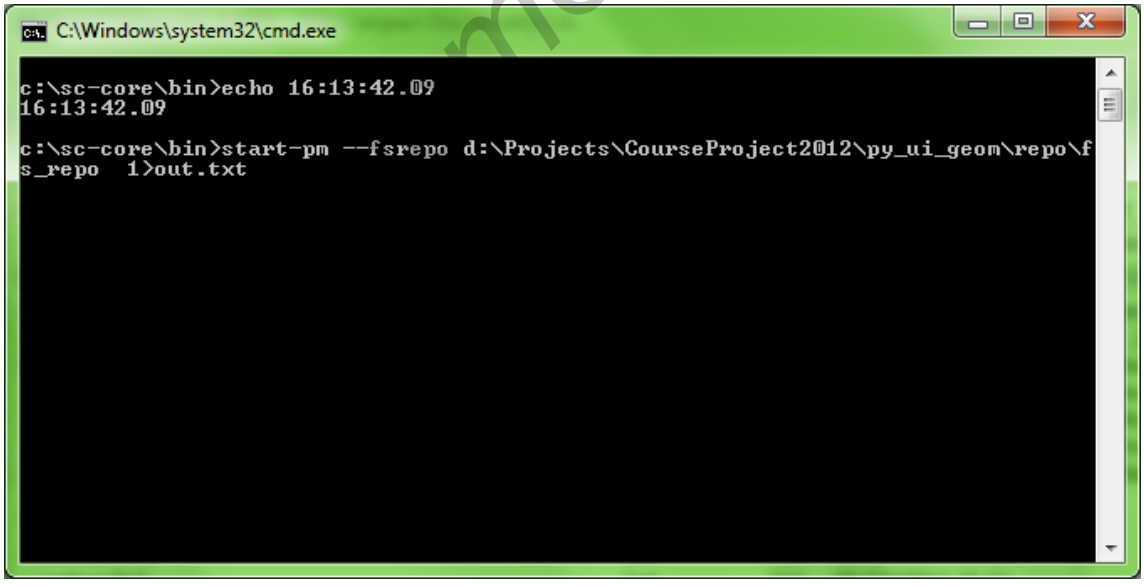

Рис. 4.10. Консоль. Процесс решение задачи

После успешного завершения работы решателя появляется окно, изображенное на рис. 4.11.

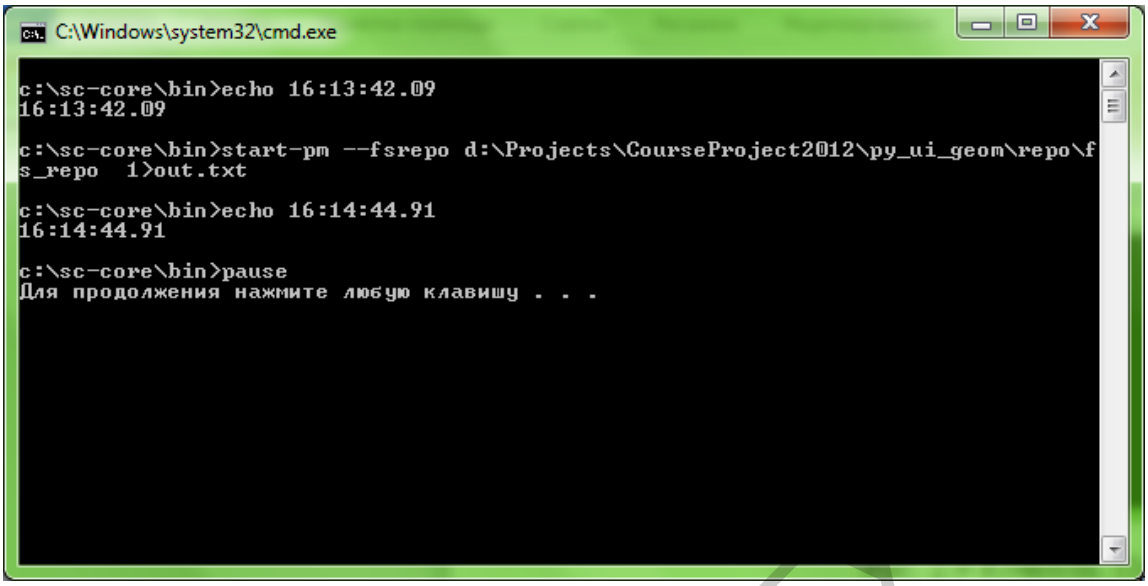

Рис. 4.11. Консоль. Окончилось решение задачи

Итог решения находится в sc-core\bin\out.txt. Файл содержит весь ход решения задачи. В конце файла находится ответ (рис. 4.12).

```
6064
6065
6066  I have exited POSTORDER TREE SEARCH MANAGER
6067
6068
6069  REQUESTED VALUE IS:
6070
6071  3078.76
6072
6073
6074  I have exited FIND VALUE
6075
```

Рис. 4.12. Результат решения задачи

4.2. Пример задачи на доказательство

Условие. Доказать, что Джон примерный студент, если он сдал математику, химию, физику. Сдать данные предметы означает сдать сессию и получать стипендию.

Решение

Доказательство будет таковым, что если студент сдал математику, химию, физику, тогда он сдал сессию. Если студент сдал сессию, то он примерный студент и он получает стипендию.

1. Сформулируем условие задачи (условие.gwf) (рис. 4.13).

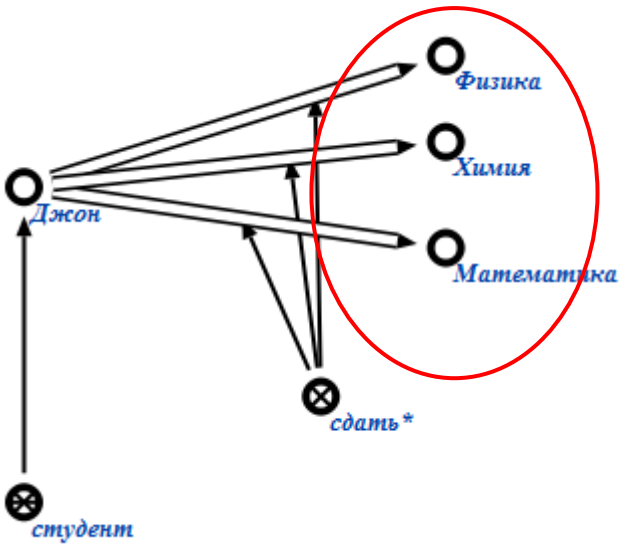

Рис. 4.13. Условие задачи

Здесь показано, что студент Джон сдал физику, химию, математику.

2. Укажем все утверждения, которые будут использоваться в задаче (рис. 4.14).

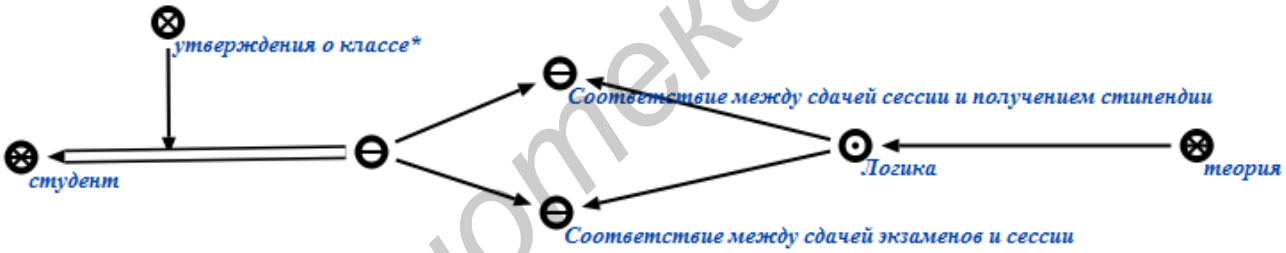

Рис. 4.14. Утверждение о классе «студент»

В данной задаче будут использоваться только два утверждения. Обратите внимание, что утверждения относятся к разделу «Логика».

3. Утверждения

3.1. «Соответствие между сдачей экзаменов и сессией» (gwf-файл имеет такое же название).

Здесь указывается то, что если студент сдал предметы (физику, химию и математику), тогда он сдал сессию. Данное утверждение представлено в формализованном виде на рис. 4.15.

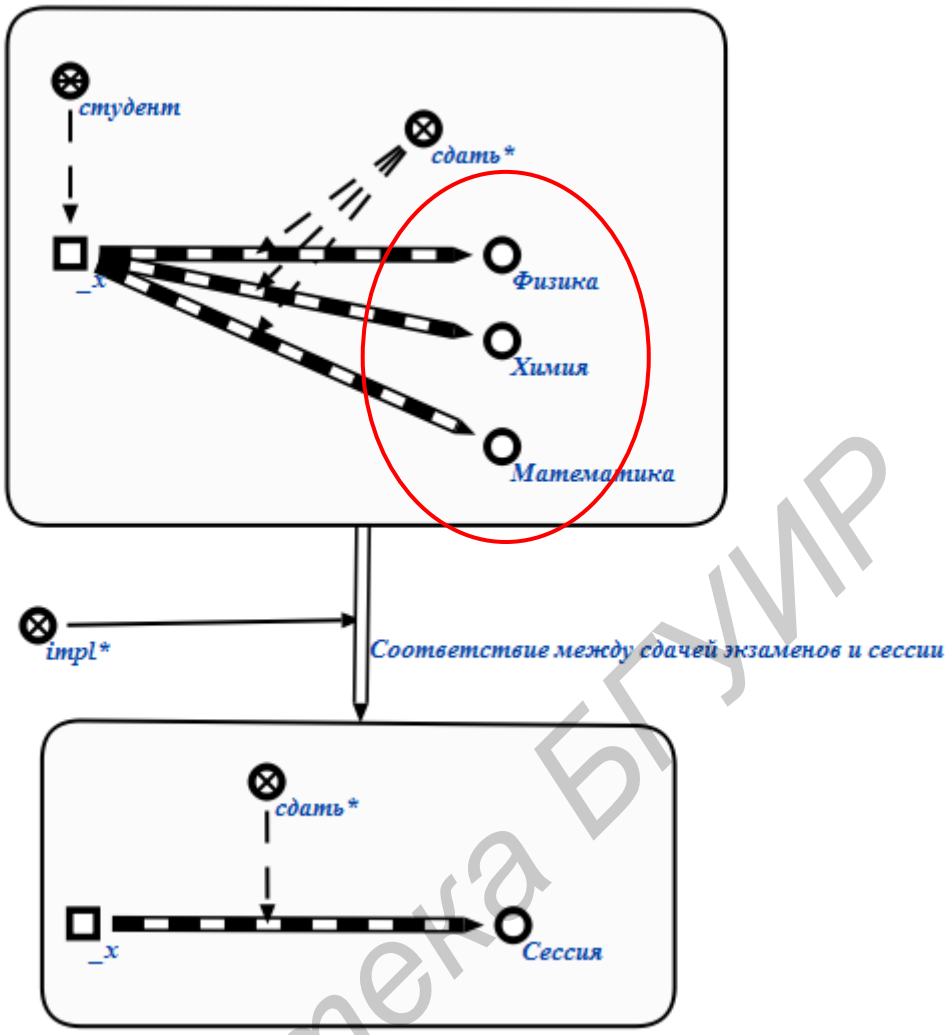

Рис. 4.15. Соответствие между сдачей экзаменов и сессии

3.2. «Соответствие между сдачей сессии и получением стипендии» (gwf-файл имеет такое же название).

Здесь показывается то, что если студент сдал сессию, то он примерный студент и он получает стипендию. Данное утверждение представлено в формализованном виде на рис. 4.16.

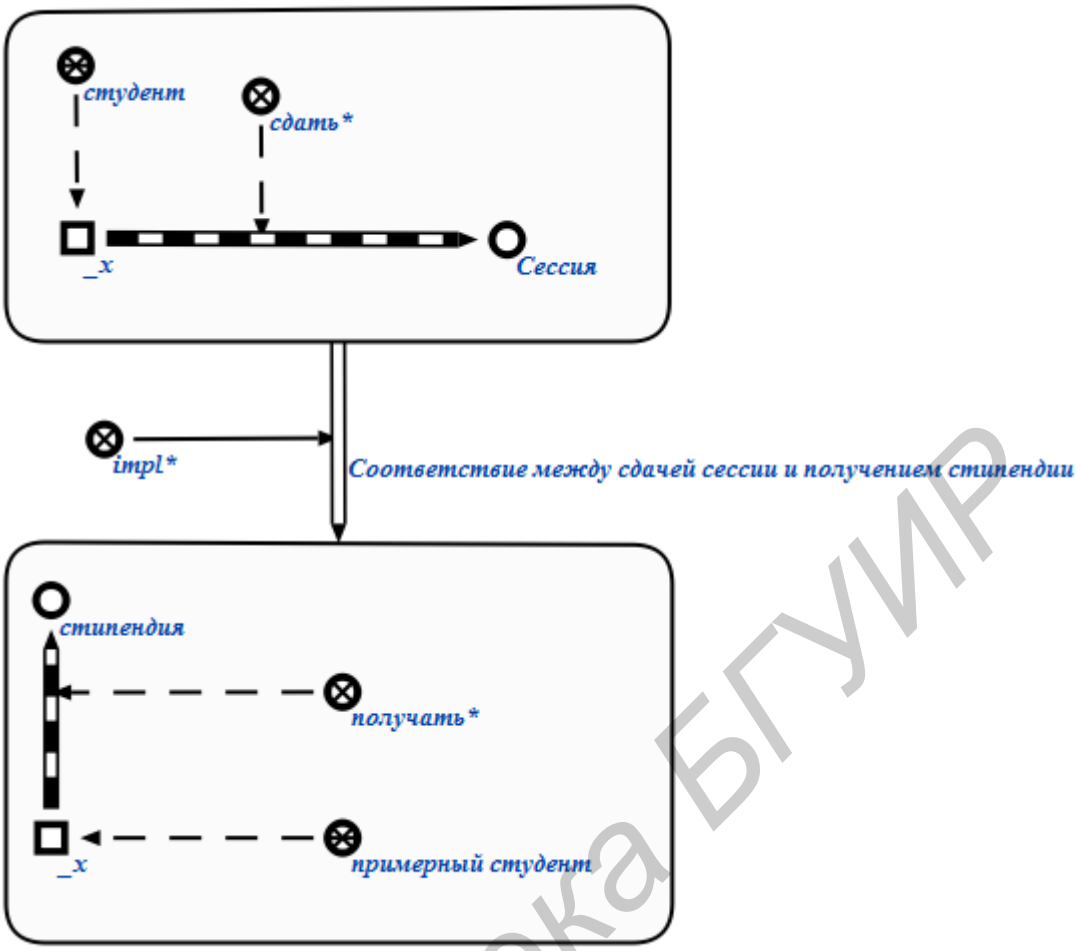

Рис. 4.16. Соответствие между сдачей сессии и получением стипендии

4. Сформулируем вопрос системе (вопрос.gwf) (рис. 4.17):

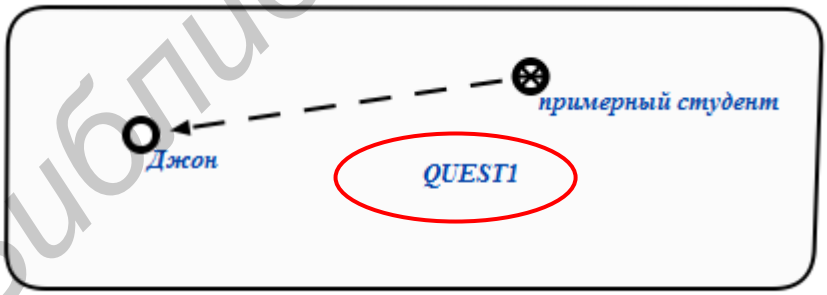

Рис. 4.17. Вопрос системе

В вопросе указывается, что Джон является примерным студентом. Система должна доказать, что Джон примерный студент, исходя из того, что он сдал предметы (химию, математику и физику).

Обратите внимание, что вопрос помещен в контур, который имеет название «QUEST1».

Обратите внимание на все рисунки задачи на доказательство. В контуре переменными узлами указывается информация, которой в системе пока нет (т. е. собственно то, что необходимо доказать), константами – та информация, которая уже есть.

5. В итоге должны получиться следующие файлы:

– вопрос.gwf;

– условие.gwf;

– соответствие между сдачей сессии и получением стипендии.gwf;

– соответствие между сдачей экзаменов и сессии.gwf.

Все файлы необходимо поместить по следующему пути:

.. \py_ui_geom\repo\fs_repo_src\seb\planimetry_src\

6. Редактируем файл temp3.m4scp, который используется для задач на доказательство (рис. 4.18).

Путь к файлу: ..\py_ui_geom\repo\fs_repo_src\operation\

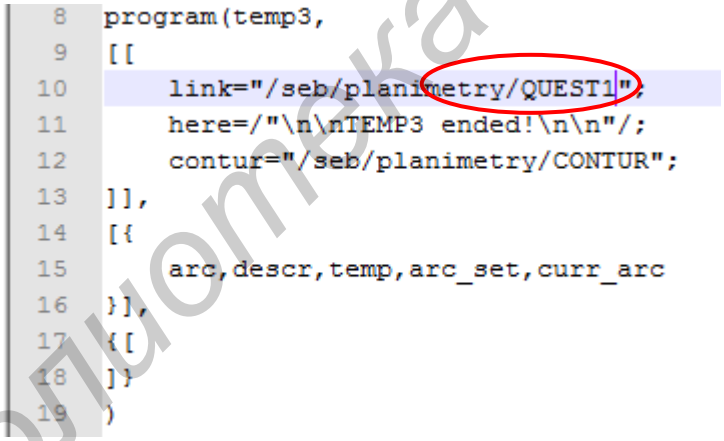

Рис. 4.18. Редактирование файла temp3.m4scp

Обратите внимание на выделенный момент на рис. 4.18. Здесь необходимо указать имя контура, в который был заключен вопрос на доказательство. Так как в нашем примере контур назывался «QUEST1», то на рис. 4.18 мы видим соответствующее название в переменной link.

7. Редактируем файл preprocessing.m4scp (рис. 4.19). Здесь указывается файл, который будет использован temp.m4scp или temp3.m4scp (подробно см. «Инструкцию по работе с ИРЗ в консольном режиме»):

```
44
45  callReturn([1_: fixed_: "/operation/temp3/temp3",
46          2_: fixed_:
47          {[
48          ]}
49  ])
```

Рис. 4.19. Редактирование файла preprocessing.m4scp

Указываем, что система будет решать задачу на доказательство.

8. Собираем БЗ. Для этого надо запустить на выполнение файл rule_builder.py

В случае успешного завершения будет выведено сообщение: Find 0 errors (рис. 4.20).

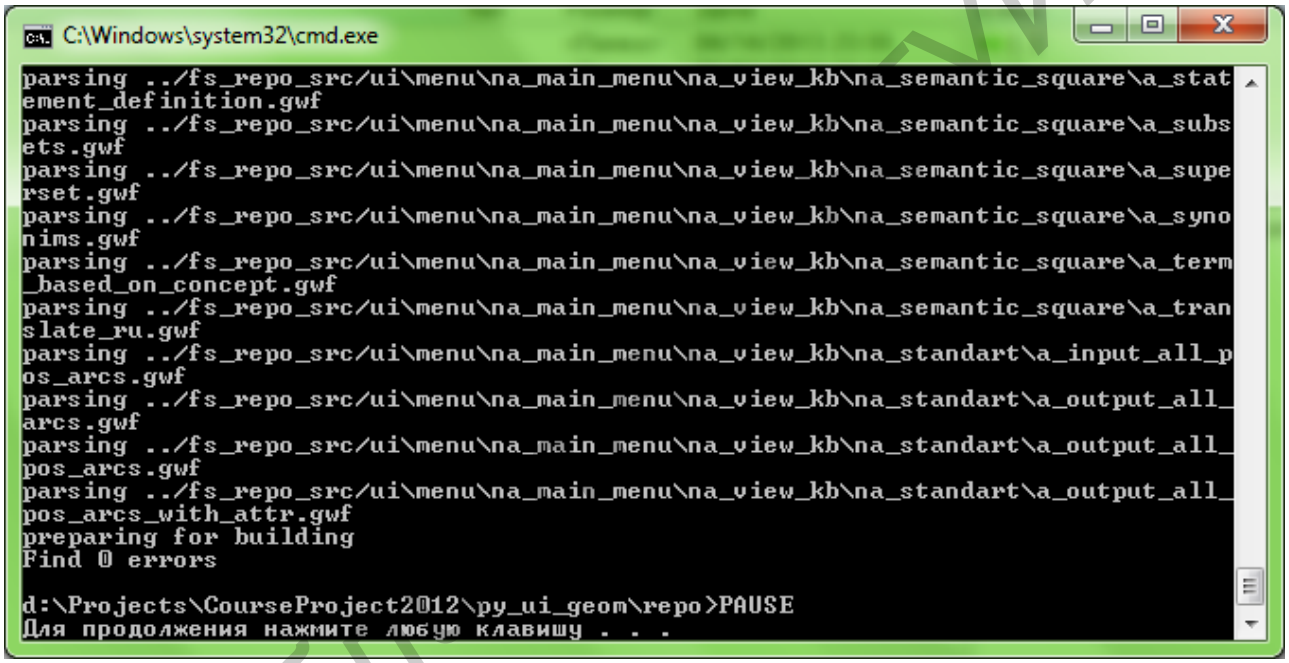

Рис. 4.20. Сборка БЗ завершена. Система не нашла ошибок

Теперь можно запускать систему для решения задачи. Для этого необходимо запустить sc-core\bin\Start.bat.

В итоге в файле sc-core\bit\out.txt можно увидеть результат доказательства задачи (рис. 4.21).

```
1907
1908  Operation : Check atomic statement validity
1909  State : Good result
1910  REQUESTED STATEMENT WAS PROVED!
1911
```

Рис. 4.21. Результат решения задачи

Как видно, результат задачи успешен. Задача доказана.

Процесс решения задачи будет выглядеть примерно так:

1. Система инициализирует вопрос.

2. Будет выполнено утверждение «Соответствие между сдачей экзаменов и сессии». Системой будет установлено, что студент сдал сессию.

3. Будет выполнено утверждение «Соответствие между сдачей сессии и получением стипендии». Отсюда система установит, что студент является примерным.

4. Так как условие доказано, система выдаст положительный ответ.

К сведению:

Решение расчетной задачи заканчивается, как только у нужной величины (той, которая была указана в вопросе) появилось значение, решение других задач – когда для контура, который был вопросом, находится полностью изоморфная конструкция.

Изоморфные конструкции – это конструкции, между которыми можно установить взаимно-однозначное соответствие.

5. ПРИМЕР РАБОТЫ СИСТЕМЫ ОПЕРАЦИЙ

Рассмотрим подробнее процесс решения задачи системой операций решателя задач в рамках интеллектуальной справочной системы по геометрии Евклида.

Пример условия задачи (рис. 5.1):

Исходные данные:

- в треугольнике *Треугк(ТчкА; ТчкВ; ТчкС)* заданы три биссектрисы *От р(ТчкА; ТчкА1)*, *От р(ТчкВ; ТчкВ1)* и *От р(ТчкС; ТчкС1)*;
- в треугольник *Треугк(ТчкА; ТчкВ; ТчкС)* вписана окружность *Окр(ТчкО; ТчкА2)*;
- окружность *Окр(ТчкО; ТчкА2)* и треугольник *Треугк(ТчкА; ТчкВ; ТчкС)* имеют общие точки *ТчкА2*, *ТчкВ2*, *ТчкС2*;
- длина отрезка *От р(ТчкА; ТчкВ)* равна 12 см;
- длина отрезка *От р(ТчкА; ТчкС)* равна 10 см;
- длина отрезка *От р(ТчкА1; ТчкС)* равна 5 см.

Задача:

Определить длину радиуса окружности *Окр(ТчкО; ТчкА2)*.

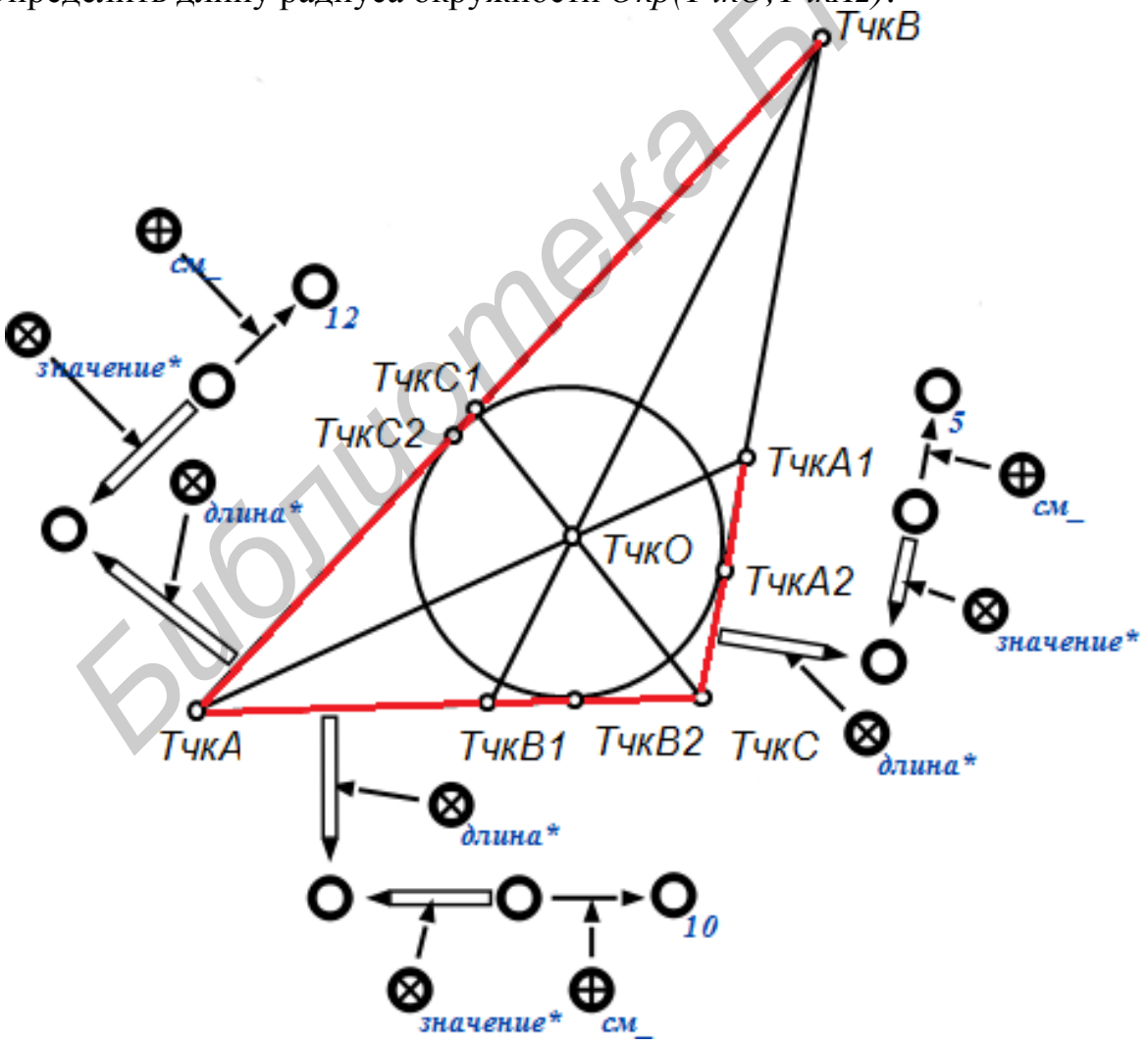

Рис. 5.1. Иллюстрация к задаче

Содержимое базы знаний системы (контекст решения задачи):

– теорема о биссектрисе (биссектриса внутреннего угла треугольника делит противоположную сторону в отношении, равном отношению двух прилежащих сторон);
– формула вычисления длины отрезка как суммы длин двух отрезков, его составляющих;
– формула вычисления периметра треугольника как суммы длин трех его сторон;
– формула Герона для вычисления площади треугольника по длинам трех его сторон (рис. 5.5);
– формула вычисления радиуса вписанной в треугольник окружности как отношения площади треугольника к его полупериметру.

Далее представлено формальное описание условия задачи (рис. 5.2, 5.3), а также формально описание некоторых фрагментов базы знаний. Полное описание всех фрагментов базы знаний можно найти на сайте OSTIS. Типология поддерживаемых логических утверждений представлена в прил. 2.

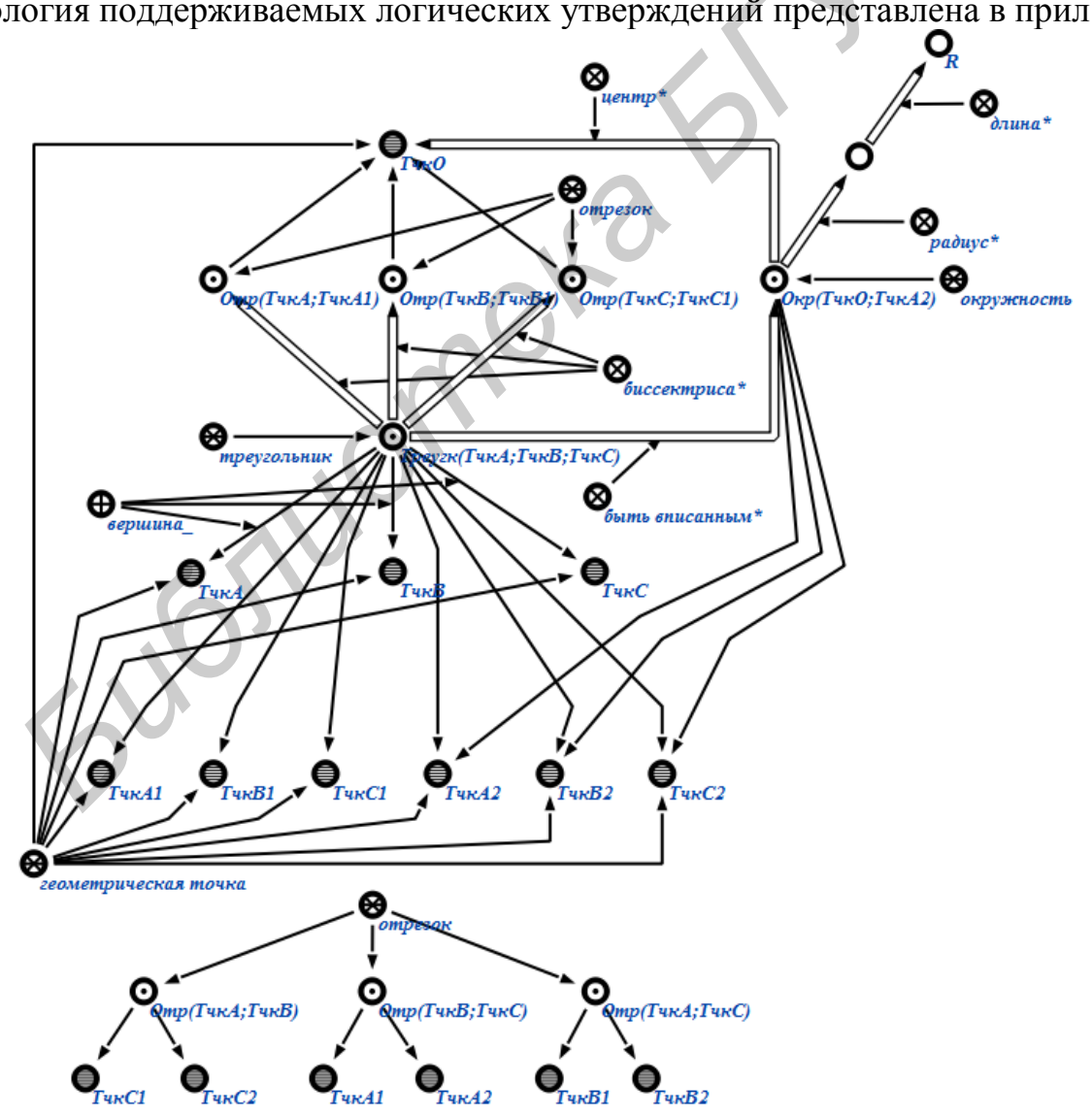

Рис. 5.2. Формальное описание условия задачи (фрагмент 1)

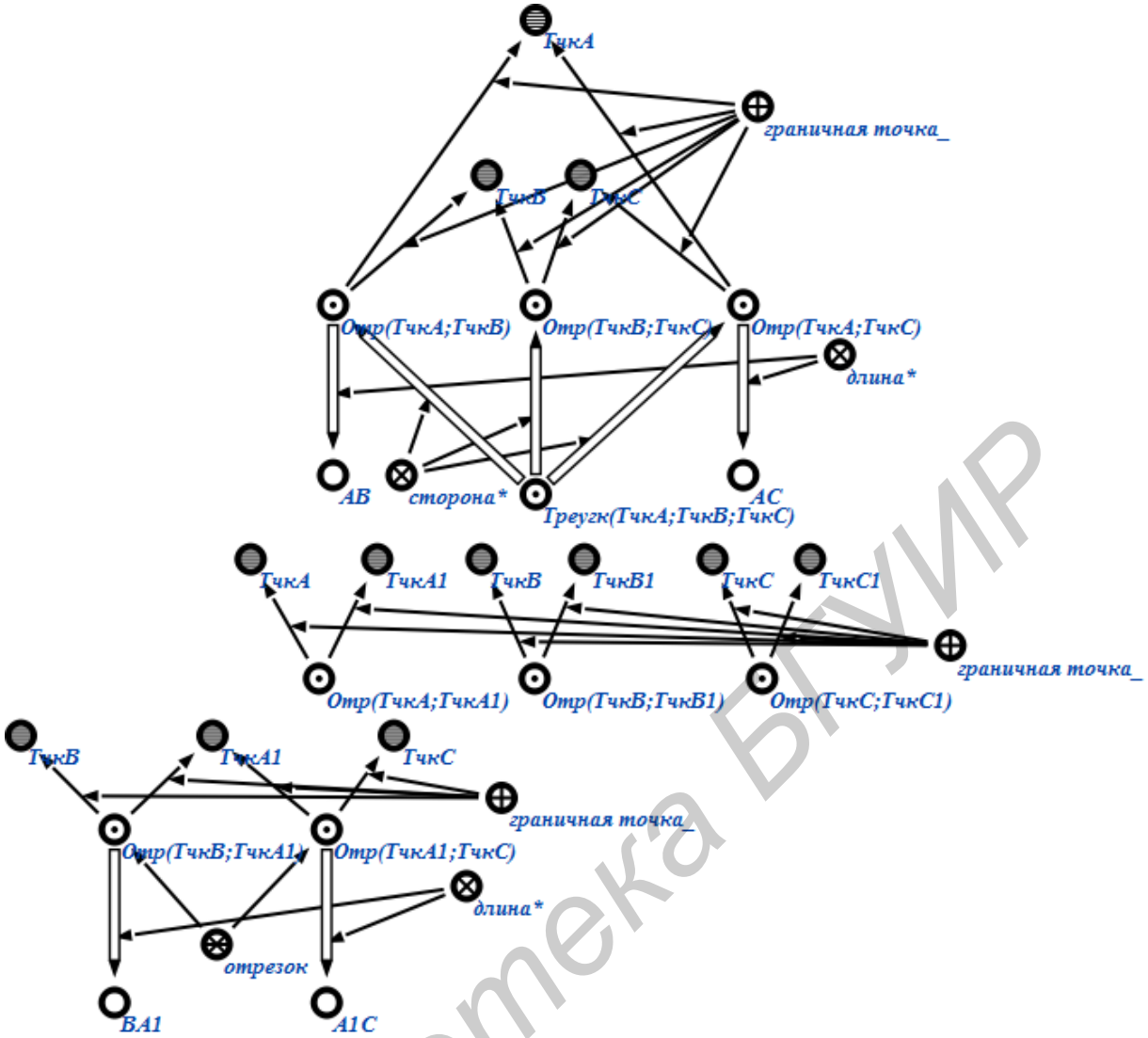

Рис. 5.3. Формальное описание условия задачи (фрагмент 2)

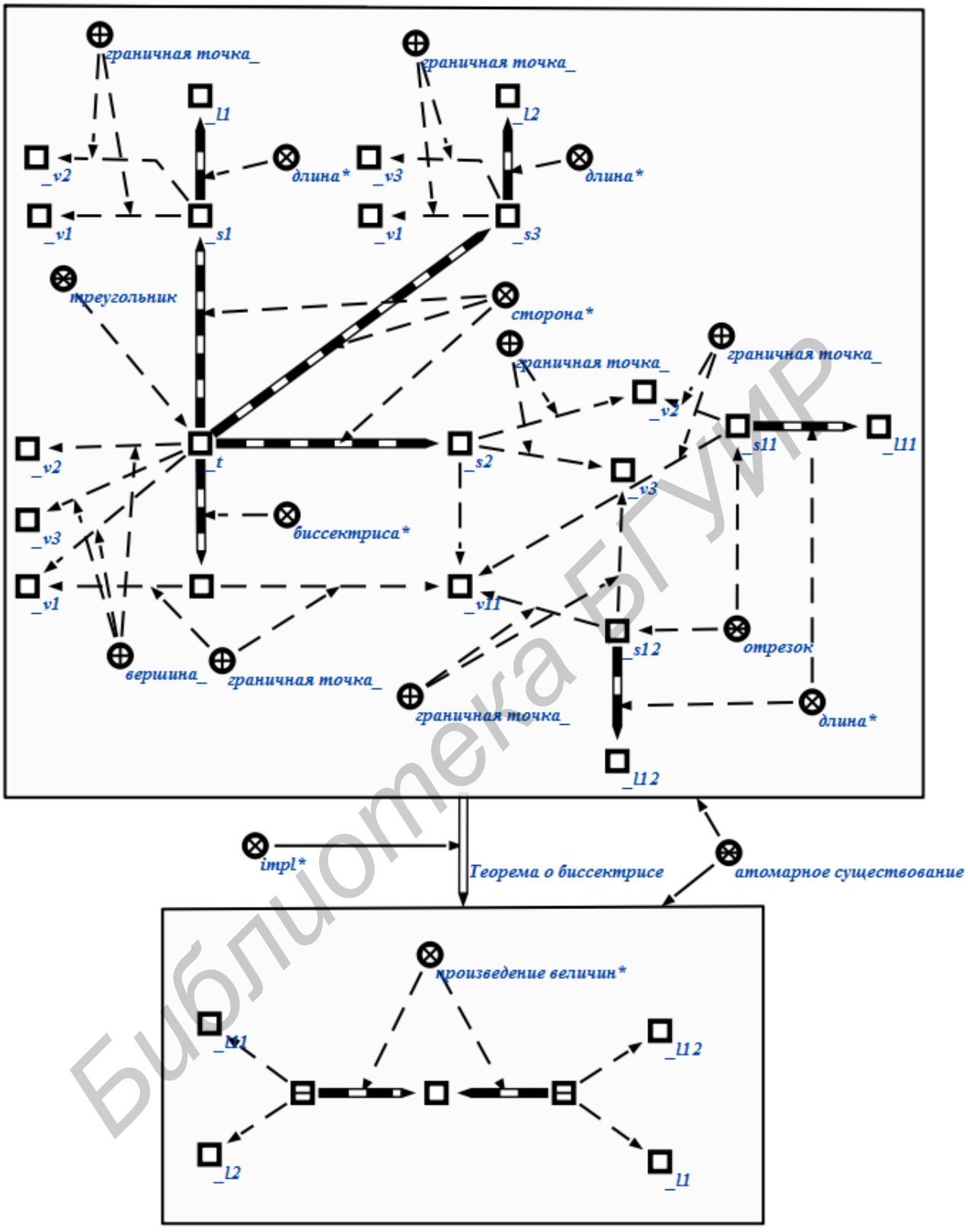

Рис. 5.4. Теорема о биссектрисе

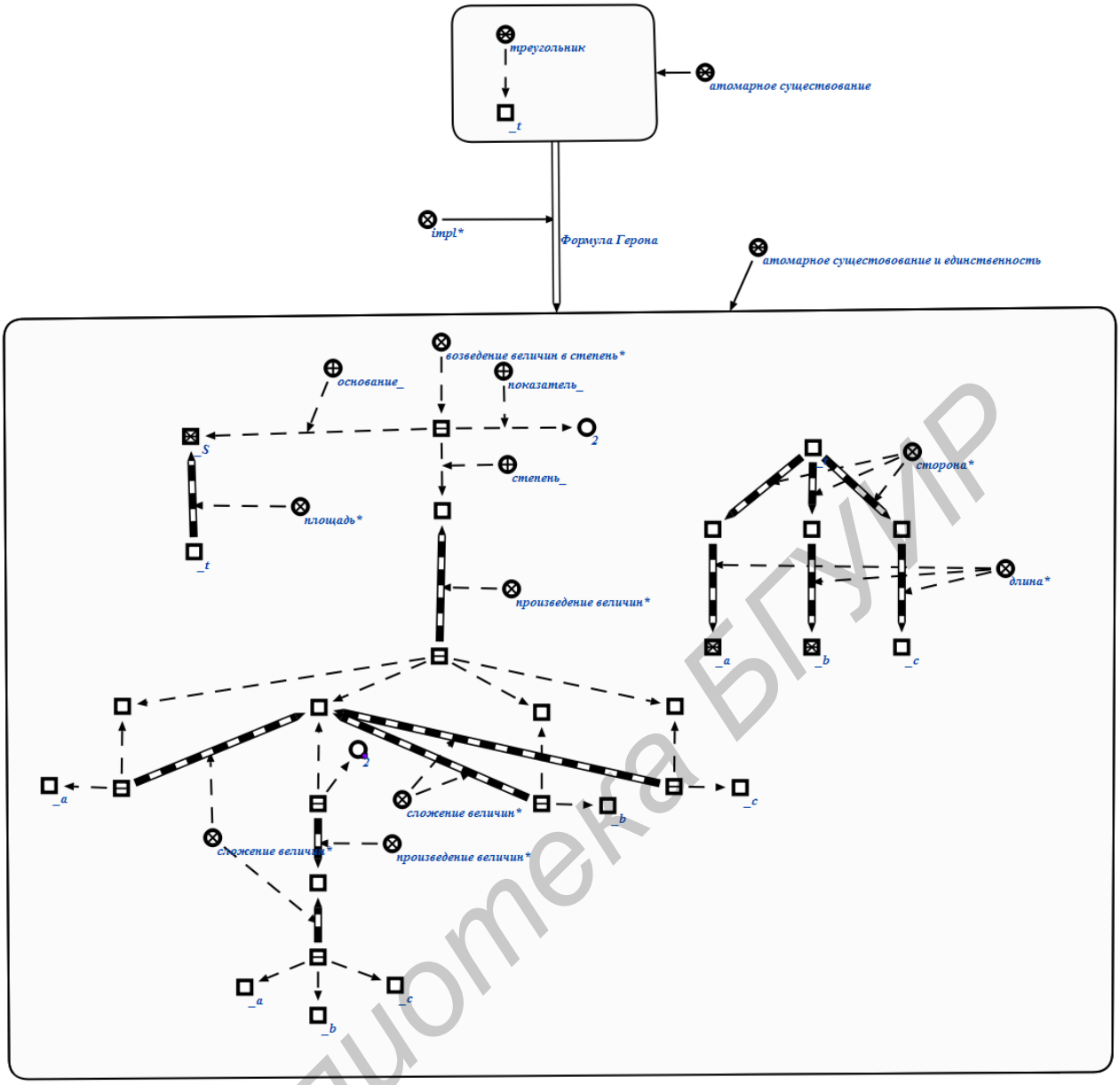

Рис. 5.5. Формула Герона

Для инициирования требуемого набора операций необходимо создать в памяти вопросную ситуацию (рис. 5.6):

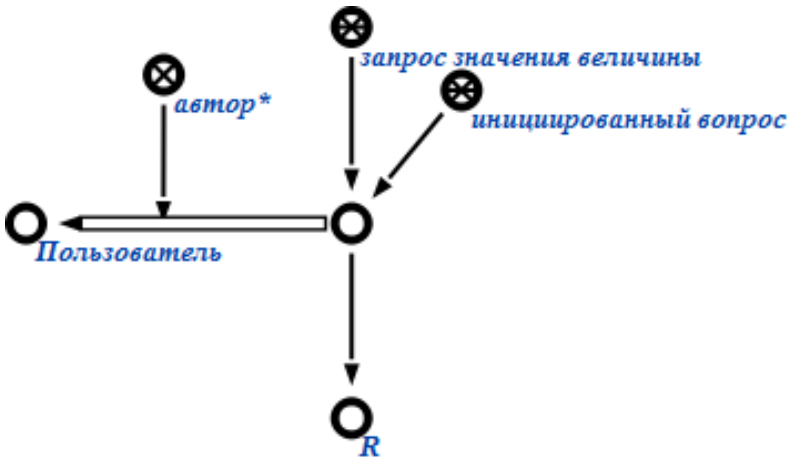

Рис. 5.6. Формат вопроса

Опишем краткий протокол решения задачи по шагам. Полную версию протокола решения данной задачи можно найти на сайте OSTIS.

Шаг 1

Используемая операция

Операция поиска значения (find_value).

Пояснение

Операция пытается найти уже имеющееся значение требуемой величины.

Требуемое значение отсутствует.

Условие инициализации (рис. 5.7):

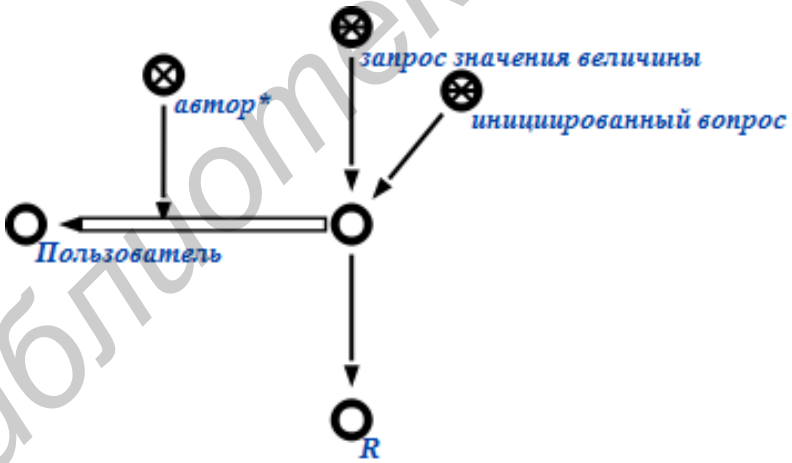

Рис. 5.7. Условие инициализации операции на шаге 1

Результат работы (рис. 5.8):

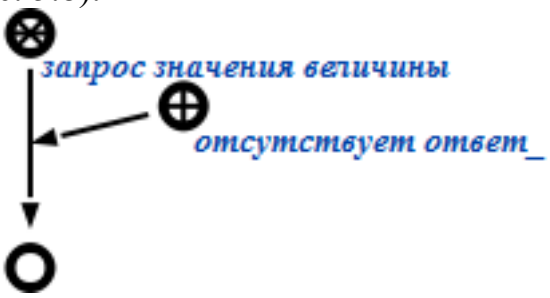

Рис. 5.8. Результат работы операции на шаге 1

Шаг 2

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Пояснение

Операция организует запуск рекурсивной операции поиска в глубину.

Для этого на рассматриваемый объект устанавливается запрос поиска в глубину.

Условие инициализации (рис. 5.9):

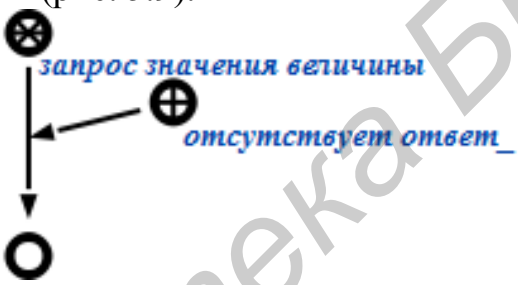

Рис. 5.9. Условие инициализации операции на шаге 2

Результат работы (рис. 5.10):

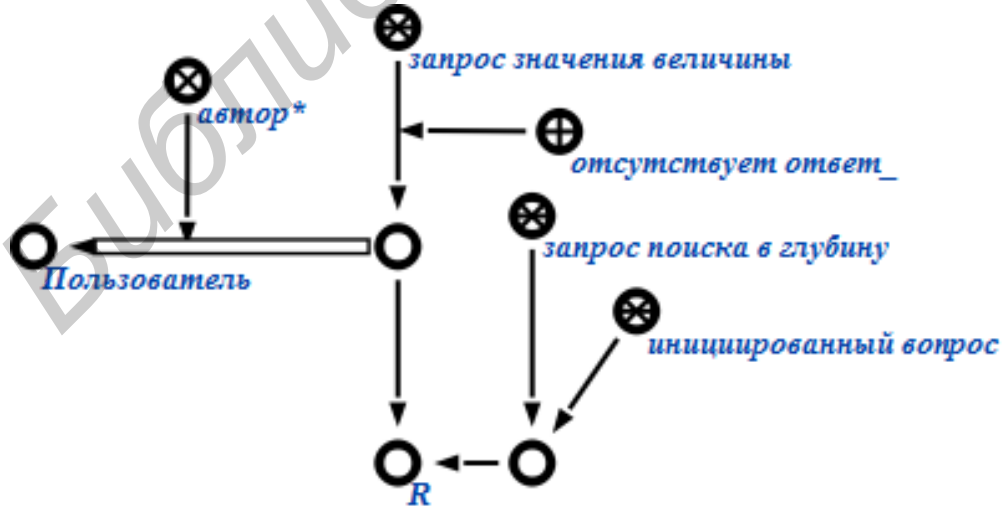

Рис. 5.10. Результат работы операции на шаге 2

Шаг 3

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

Операция просматривает все объекты, связанные с исходным объектом, и пытается сгенерировать новые знания. Если знания сгенерировать не удалось, запрос поиска в глубину устанавливается на объекты, связанные с исходным объектов. Просмотренные узлы добавляются в множество просмотренных узлов. В данном случае новые знания генерируются для объекта *Треугк(ТчкА;ТчкВ;ТчкС)*. При этом используется утверждение «*Теорема о биссектрисе*».

Условие инициализации (рис. 5.11):

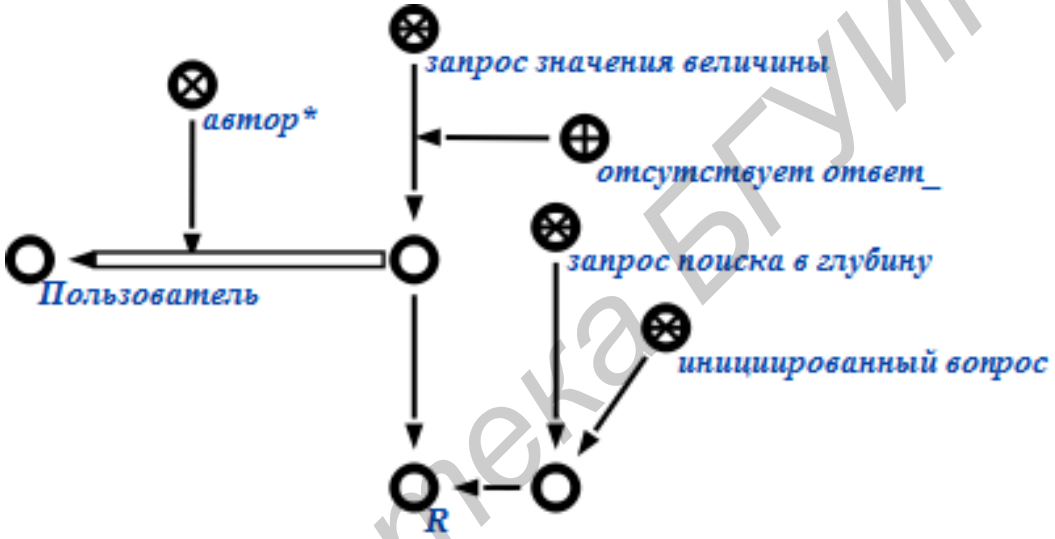

Рис. 5.11. Условие инициализации операции на шаге 3

Результат работы (рис. 5.12):

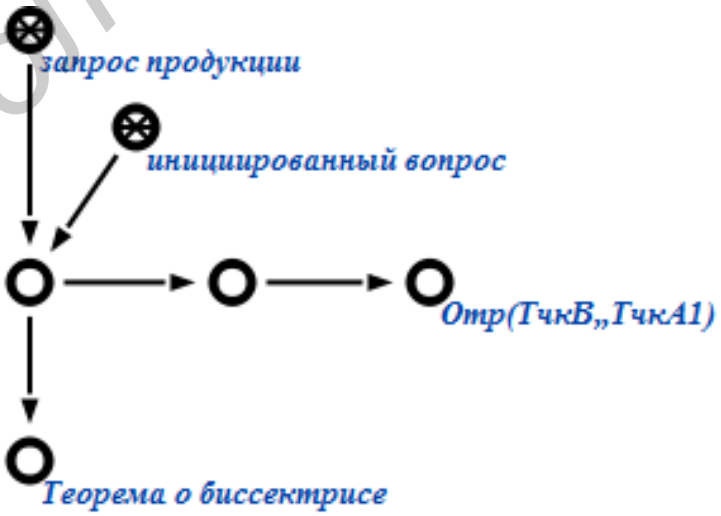

Рис. 5.12. Результат работы операции на шаге 3

Шаг 4

Используемая операция

Операция получения значения продукции (find_value_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

На основании теоремы о биссектрисе вычисляется длина отрезка *Отр(ТчкВ;ТчкА1)* – 6 см.

Условие инициализации (рис. 5.13):

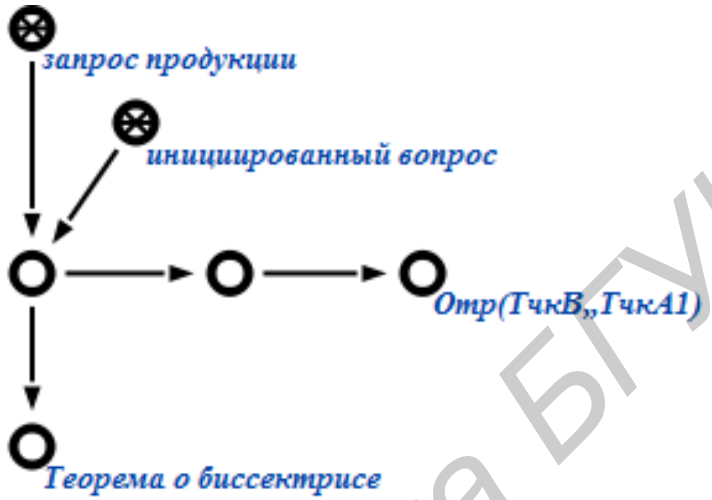

Рис. 5.13. Условие инициализации операции на шаге 4

Результат работы (рис. 5.14):

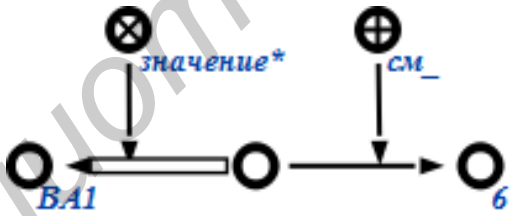

Рис. 5.14. Результат работы операции на шаге 4

Шаг 5 аналогичен шагу 1

Используемая операция

Операция поиска значения (find_value).

Шаг 6 аналогичен шагу 2

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 7 аналогичен шагу 3

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Отр(ТчкВ;ТчкС)*. При этом используется утверждение «*Формула вычисления длины отрезка*».

Шаг 8

Используемая операция

Операция получения значения продукции (find_value_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

Вычисляется длина отрезка *Отр(ТчкВ;ТчкС)* как сумма длин отрезков *Отр(ТчкВ;ТчкА1) и Отр(ТчкА1;ТчкС)* – 11 см.

Результат работы (рис. 5.15):

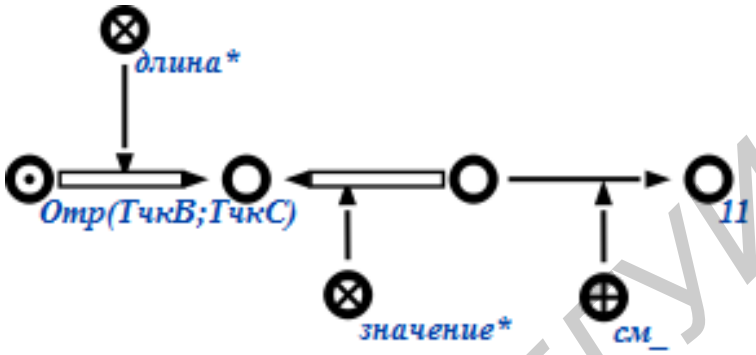

Рис. 5.15. Результат работы операции на шаге 8

Шаг 9 аналогичен шагу 1

Используемая операция

Операция поиска значения (find_value).

Шаг 10 аналогичен шагу 2

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 11 аналогичен шагу 3

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Треугк(ТчкА;ТчкВ;ТчкС)*. При этом используется утверждение «*Формула вычисления периметра треугольника*».

Шаг 12

Используемая операция

Операция получения значения продукции (find_value_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

Вычисляется периметр треугольника *Треугк(ТчкА;ТчкВ;ТчкС)* – 33 см.

Результат работы (рис. 5.16):

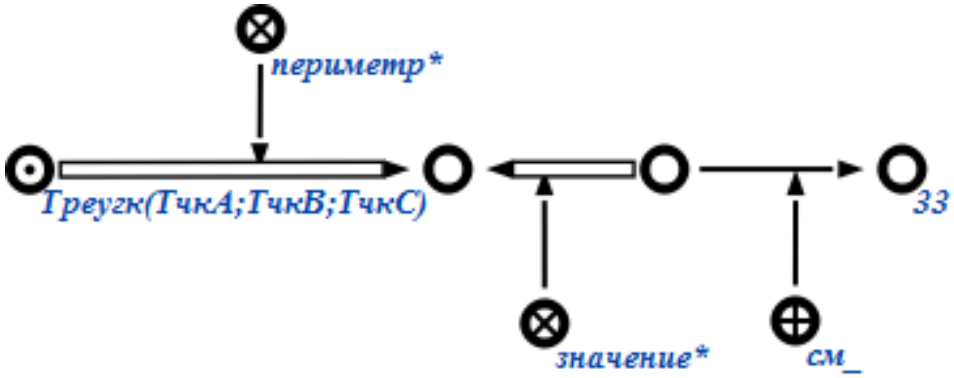

Рис. 5.16. Результат работы операции на шаге 12

Шаг 13 аналогичен шагу 1

Используемая операция

Операция поиска значения (find_value).

Шаг 14 аналогичен шагу 2

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 15 аналогичен шагу 3

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Треугк(ТчкА;ТчкВ;ТчкС)*. При этом используется утверждение «*Формула Герона*».

Шаг 16

Используемая операция

Операция получения значения продукции (find_value_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

Вычисляется площадь треугольника *Треугк(ТчкА;ТчкВ;ТчкС)* – 51,52 см^2.

Результат работы (рис. 5.17):

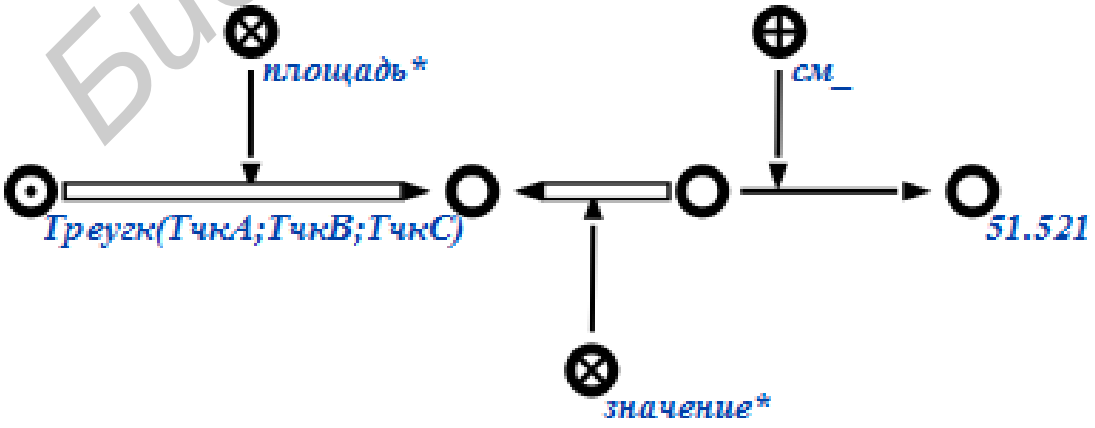

Рис. 5.17. Результат работы операции на шаге 16

Шаг 17 аналогичен шагу 1

Используемая операция

Операция поиска значения (find_value).

Шаг 18 аналогичен шагу 2

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 19 аналогичен шагу 3

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Окр(ТчкО;ТчкА2)*. При этом используется утверждение «*Соотношение площади треугольника и радиуса вписанной окружности*».

Шаг 20

Используемая операция

Операция получения значения продукции (find_value_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

Вычисляется радиус окружности *Окр(ТчкО;ТчкА2)* – 3,1225 см.

Результат работы (рис. 5.18):

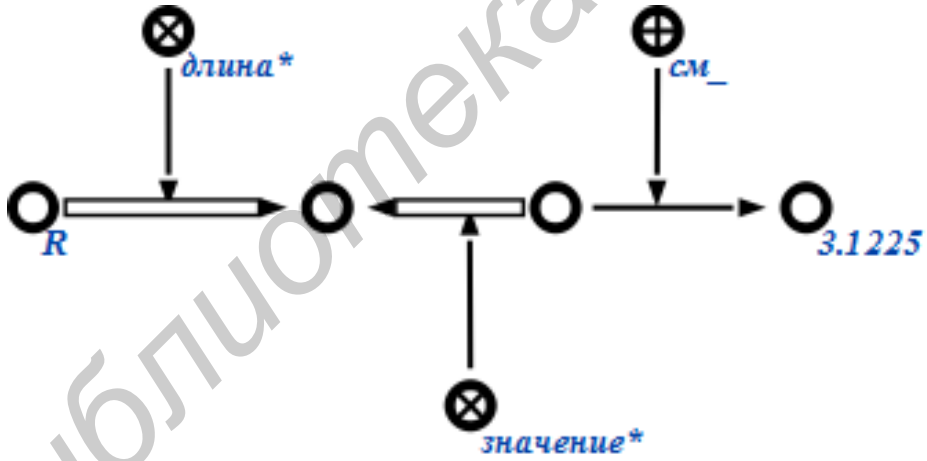

Рис. 5.18. Результат работы операции на шаге 20

Шаг 21

Используемая операция

Операция поиска значения (find_value).

Пояснение

Операция пытается найти уже имеющееся значение требуемой величины.

Требуемое значение присутствует.

Результат работы (рис. 5.19):

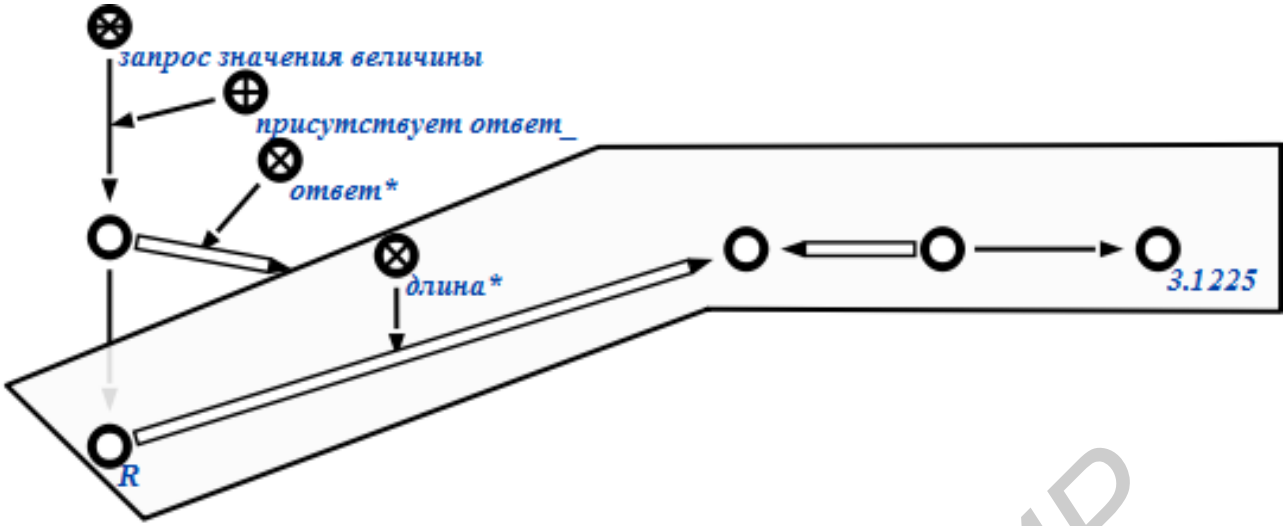

Рис. 5.19. Результат работы операции на шаге 21

Рассмотрим еще один пример решения задачи в рамках той же предметной области.

Пример условия задачи:

Исходные данные:

– на окружности O1 заданы точки A, B и C;
– длина хорды AB равна 6 см;
– хорда AC проходит через центр O1;
– длина радиуса O1 равна 5 см.

Задача:

Определить площадь треугольника, образованного точками A, B и C (рис. 5.20).

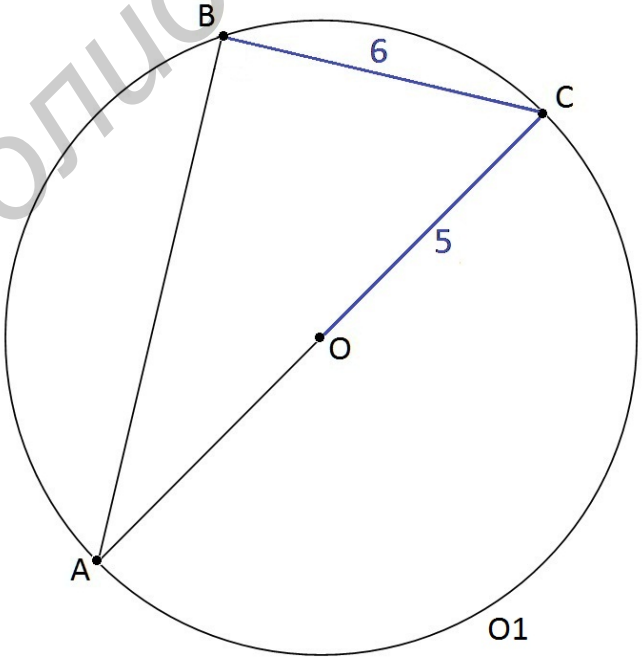

Рис. 5.20. Иллюстрация к задаче

Содержимое базы знаний системы (контекст решения задачи):

- определение диаметра окружности: хорда, проходящая через центр окружности, является диаметром данной окружности;
- соотношение длин диаметра и радиуса: длина диаметра окружности вдвое больше длины радиуса окружности;
- теорема о величине вписанного угла, опирающегося на диаметр некоторой окружности: если одна из сторон треугольника является диаметром окружности, а противолежащая вершина принадлежит данной окружности, то мера угла при данной вершине равна 90^{o};
- определение прямоугольного треугольника через градусную меру одного из углов: треугольник, у которого градусная мера одного из углов равна 90, называется прямоугольным;
- формула вычисления площади прямоугольного треугольника по заданным длинам катетов: площадь прямоугольного треугольника равна полупроизведению длин катетов данного треугольника;
- теорема Пифагора: квадрат длины гипотенузы прямоугольного треугольника равен сумме квадратов длин катетов данного прямоугольного треугольника (рис. 5.22).

Далее представлено формальное описание условия задачи (рис. 5.21), а также формальное описание некоторых фрагментов базы знаний.

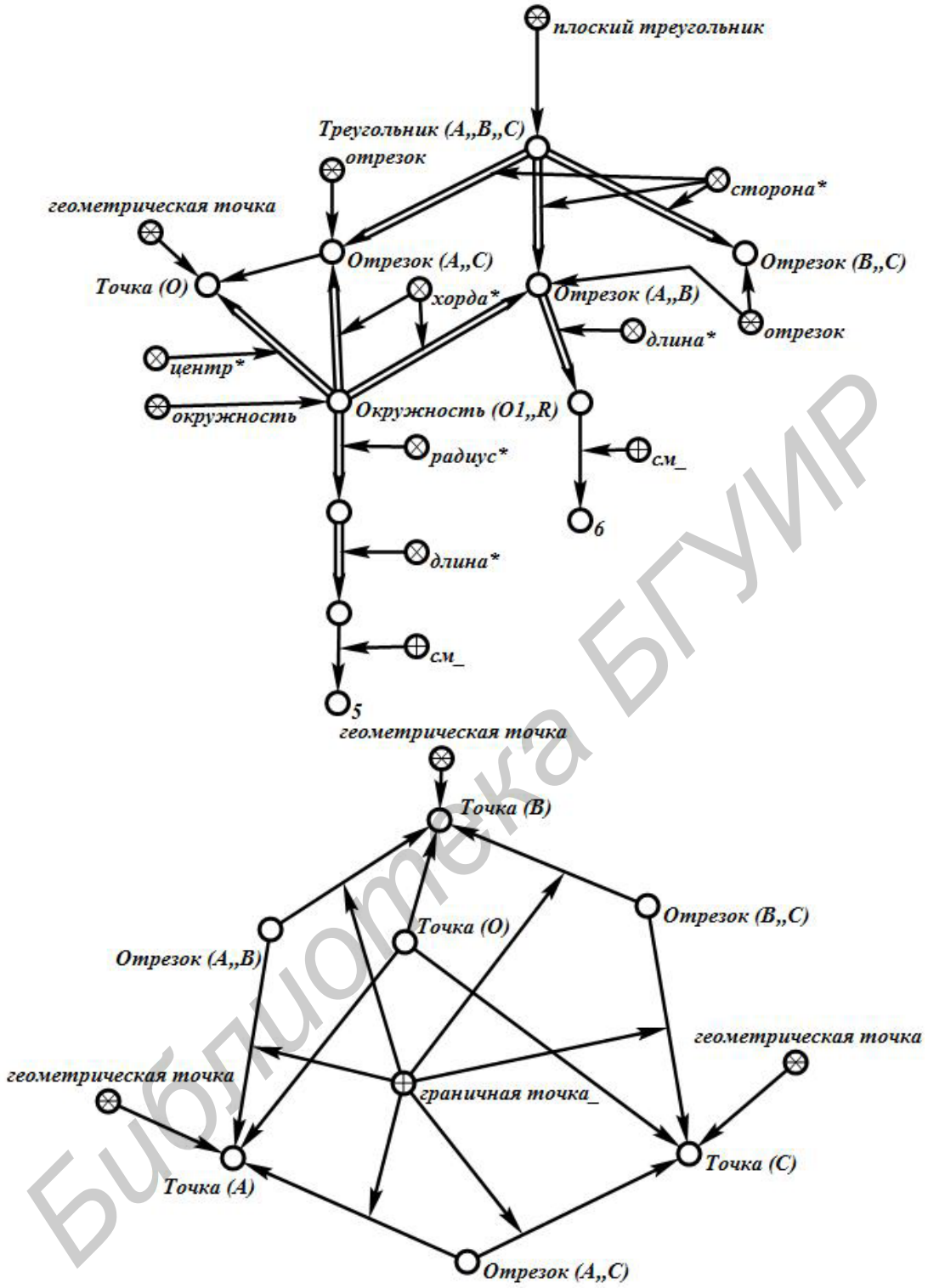

Рис. 5.21. Формальное описание условия задачи

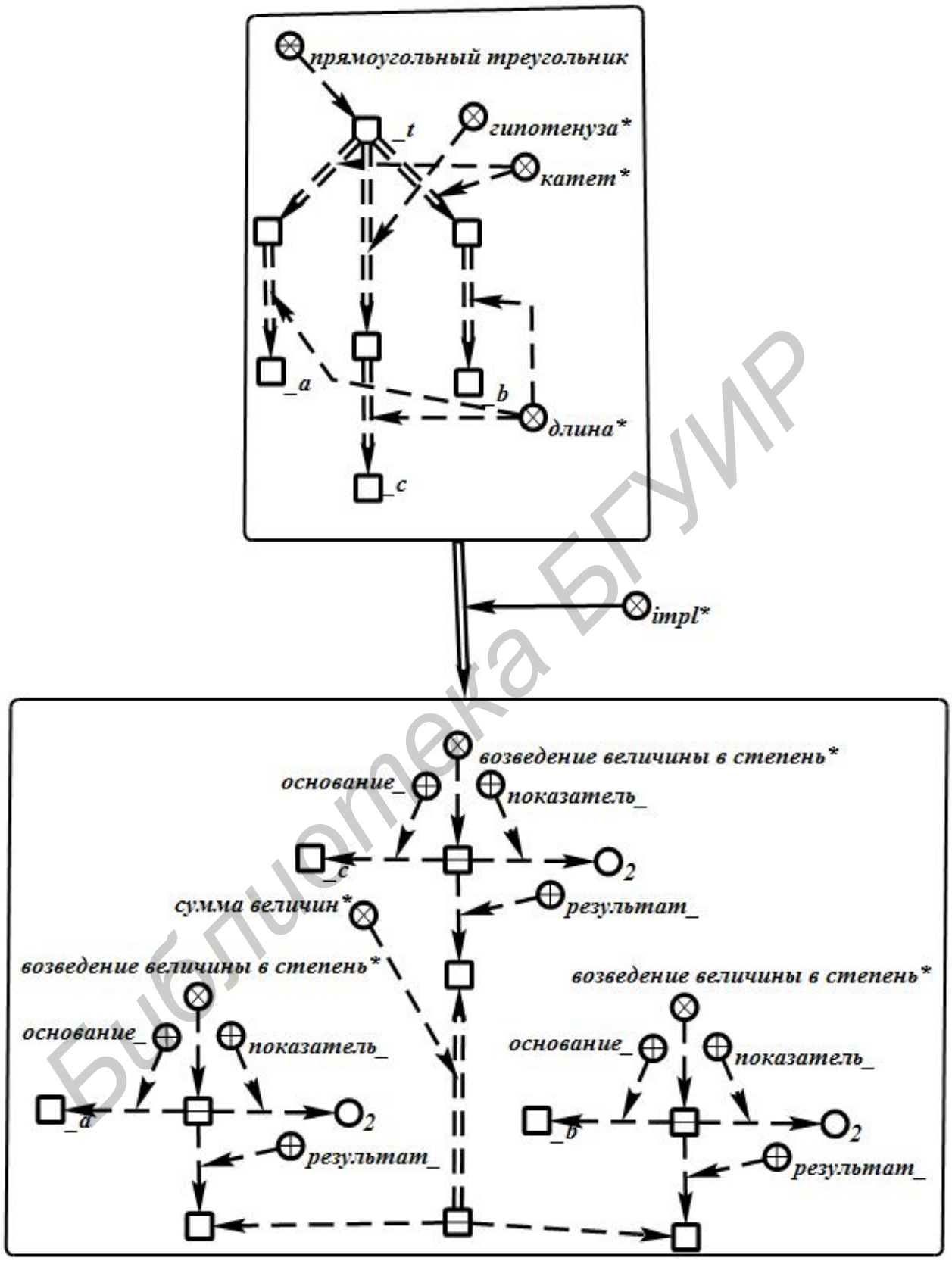

Рис. 5.22. Теорема Пифагора

Для инициирования требуемого набора операций необходимо создать в памяти вопросную ситуацию (рис. 5.23):

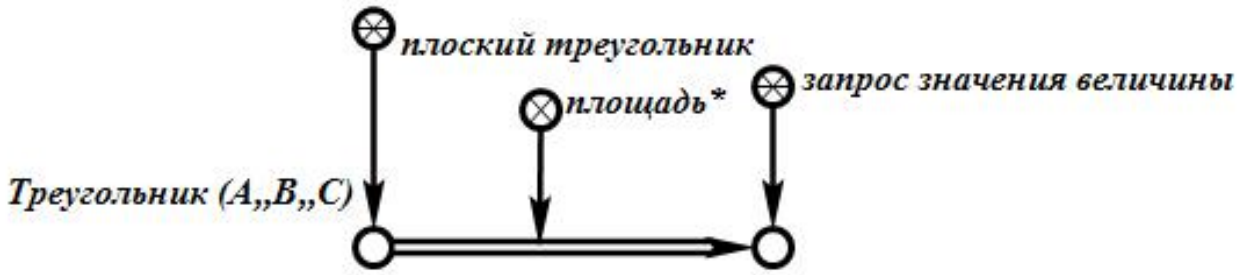

Рис. 5.23. Формат вопроса

Опишем краткий протокол решения задачи по шагам. Полную версию протокола решения данной задачи можно найти на сайте OSTIS.

Шаг 1

Используемая операция

Операция поиска значения (find_value).

Пояснение

Операция пытается найти уже имеющееся значение требуемой величины.

Требуемое значение отсутствует.

Условие инициализации (рис. 5.24):

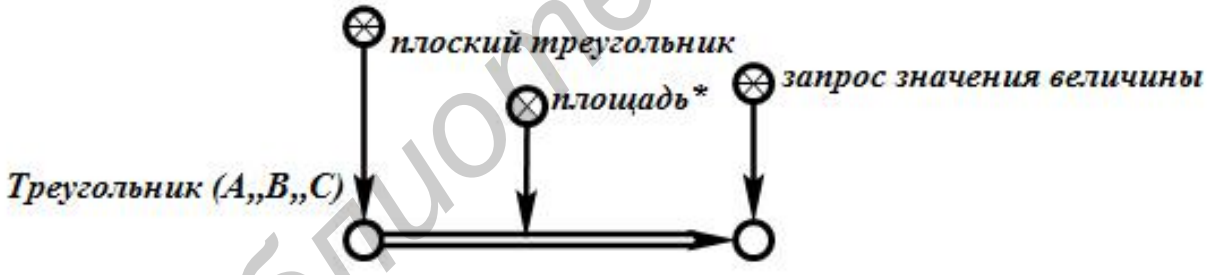

Рис. 5.24. Условие инициализации операции на шаге 1

Результат работы (рис. 5.25):

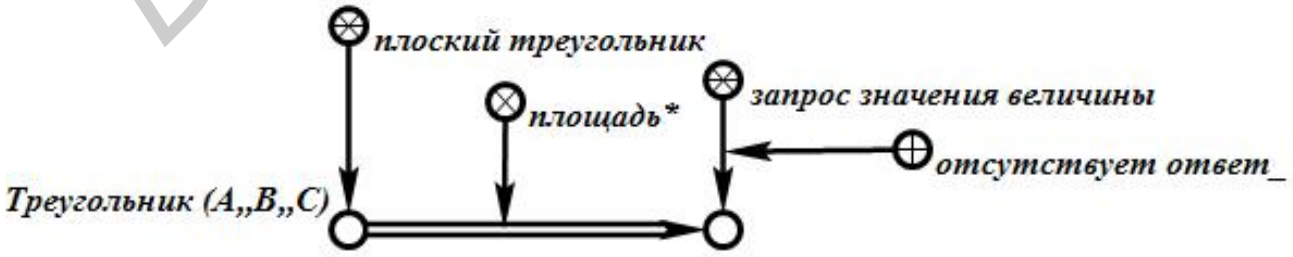

Рис. 5.25. Результат работы операции на шаге 1

Шаг 2

Используемая операция

Операция поиска формулы (find_formula).

Пояснение

Операция пытается найти готовую формулу для вычисления требуемого значения величины.

Требуемая формула отсутствует.

Условие инициализации (рис. 5.26):

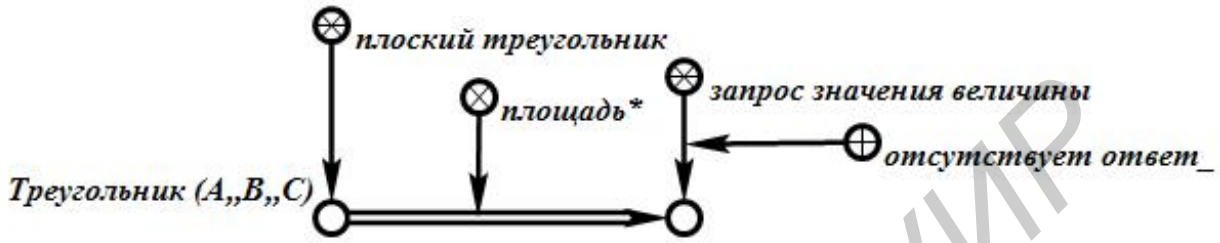

Рис. 5.26. Условие инициализации операции на шаге 2

Результат работы (рис. 5.27):

Рис. 5.27. Результат работы операции на шаге 2

Шаг 3

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Пояснение

Операция организует запуск рекурсивной операции поиска в глубину.

Для этого на рассматриваемый объект устанавливается запрос поиска в глубину.

Условие инициализации (рис. 5.28):

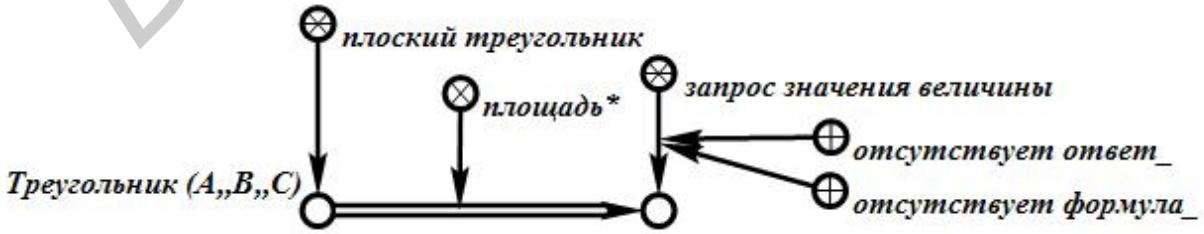

Рис. 5.28. Условие инициализации операции на шаге 3

Результат работы (рис. 5.29):

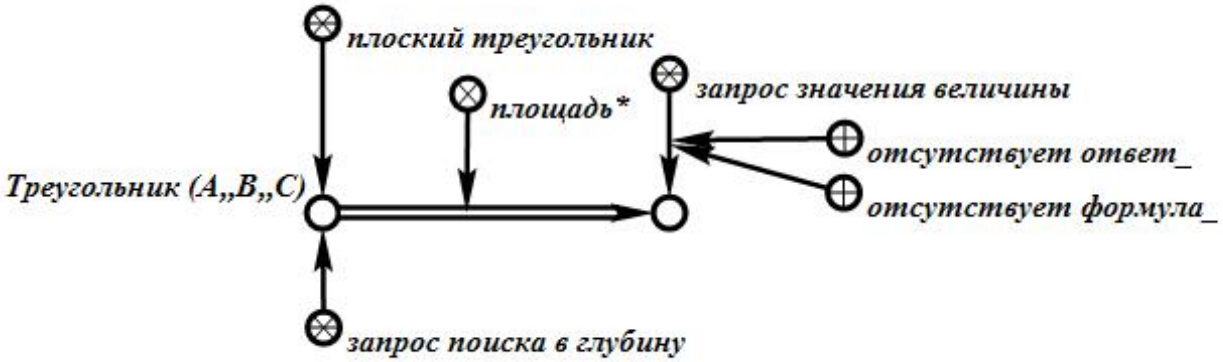

Рис. 5.29 . Результат работы операции на шаге 3

Шаг 4

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

Операция просматривает все объекты, связанные с исходным объектом, и пытается сгенерировать новые знания. Если знания сгенерировать не удалось, запрос поиска в глубину устанавливается на объекты, связанные с данным. Просмотренные узлы добавляются в множество просмотренных узлов. В данном случае новые знания генерируются для объекта Отр(A,,C).

Условие инициализации (рис. 5.30):

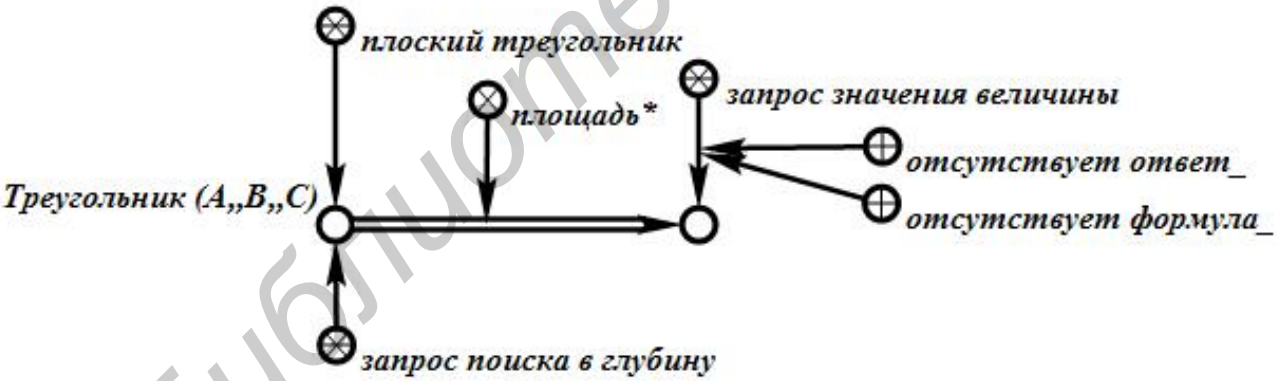

Рис. 5.30. Условие инициализации операции на шаге 4

Результат работы (рис. 5.31):

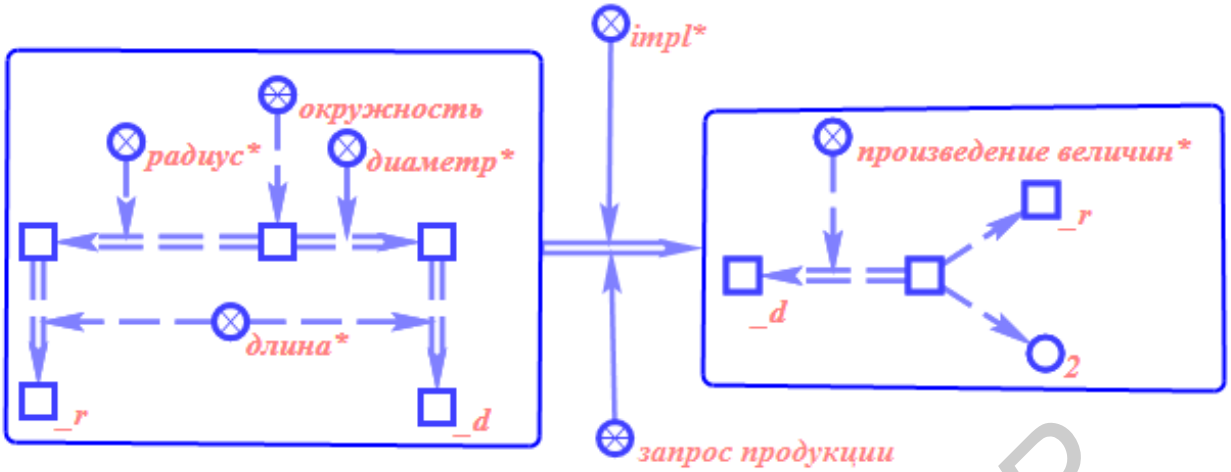

Рис. 5.31. Результат работы операции на шаге 4

Шаг 5

Используемая операция

Операция получения значения продукции (perform_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

На основании определения диаметра окружности и факта прохождения хорды AC через центр окружности O1 делается вывод о том, что хорда AC является диаметром O1.

Условие инициализации (рис. 5.32):

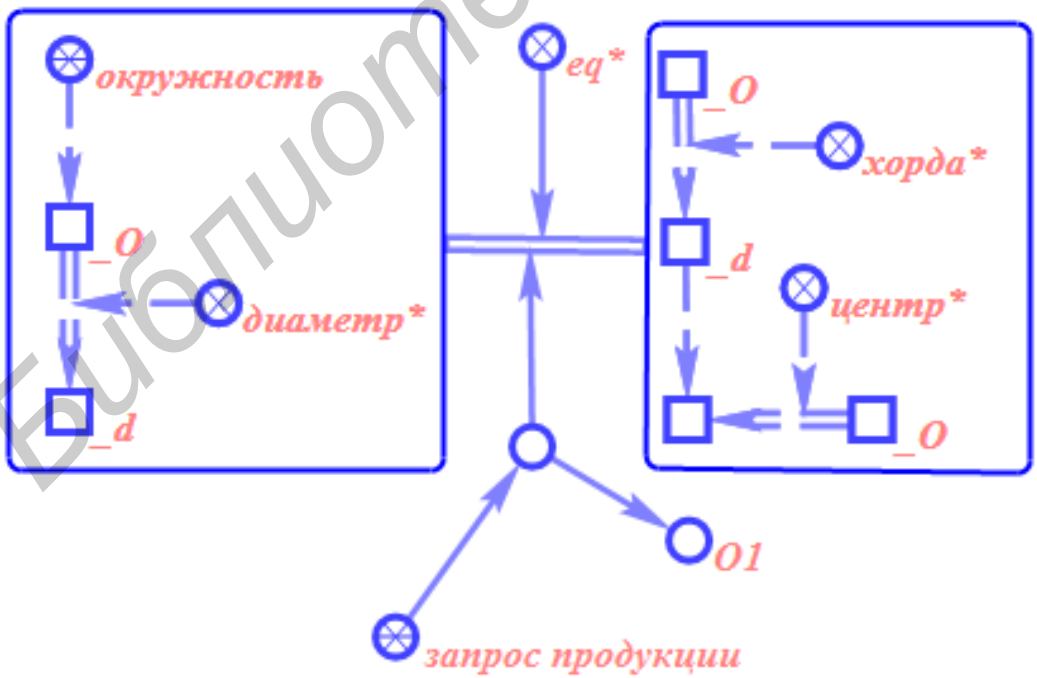

Рис. 5.32. Условие инициализации операции на шаге 5

Результат работы (рис. 5.33):

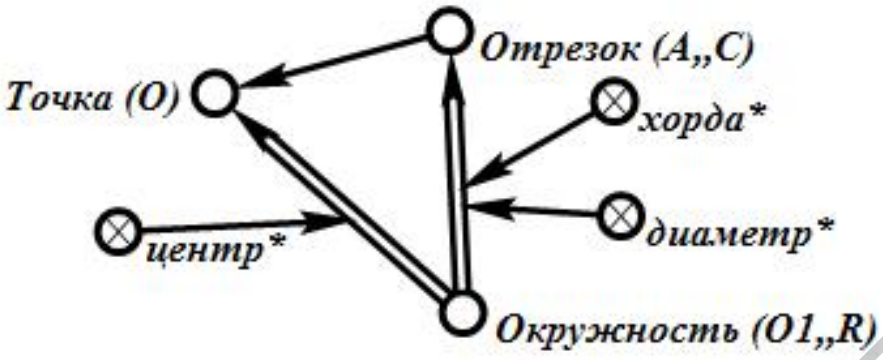

Рис. 5.33. Результат работы операции на шаге 5

Шаг 6 аналогичен шагу 1
Используемая операция
Операция поиска значения (find_value).
Шаг 7 аналогичен шагу 2
Используемая операция
Операция поиска формулы (find_formula).
Шаг 8 аналогичен шагу 3
Используемая операция
Операция, организующая поиск в глубину (postorder_tree_search_manager).
Шаг 9 аналогичен шагу 4
Используемая операция
Операция поиска в глубину (postorder_tree_search).
Пояснение
В данном случае новые знания генерируются для объекта *От p(A,,C)*.
Шаг 10
Используемая операция
Операция получения значения продукции (perform_production);
Операция интерпретации арифметического выражения (calculation).
Пояснение
Вычисляется длина стороны АС как удвоенная длина радиуса окружности О1.
Шаг 11 аналогичен шагу 1
Используемая операция
Операция поиска значения (find_value).
Шаг 12 аналогичен шагу 2
Используемая операция
Операция поиска формулы (find_formula).
Шаг 13 аналогичен шагу 3
Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 14 аналогичен шагу 4

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Треугк(A,,B,,C)*.

Шаг 15

Используемая операция

Операция получения значения продукции (perform_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

Делается вывод о том, что градусная мера внутреннего угла треугольника ABC между сторонами AB и BC равна 90°.

Шаг 16 аналогичен шагу 1

Используемая операция

Операция поиска значения (find_value).

Шаг 17 аналогичен шагу 2

Используемая операция

Операция поиска формулы (find_formula).

Шаг 18 аналогичен шагу 3

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 19 аналогичен шагу 4

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Треугк(A,,B,,C)*.

Шаг 20

Используемая операция

Операция получения значения продукции (perform_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

Делается вывод о том, что треугольник ABC является прямоугольным, т. к. мера угла ABC равна 90°. При этом сторона AC является гипотенузой, а стороны BC и AB – катетами.

Шаг 21 аналогичен шагу 1

Используемая операция

Операция поиска значения (find_value).

Шаг 22 аналогичен шагу 2

Используемая операция

Операция поиска формулы (find_formula).

Шаг 23 аналогичен шагу 3

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 24 аналогичен шагу 4

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Треугк(A,,B,,C).*

Шаг 25

Используемая операция

Операция получения значения продукции (perform_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

По теореме Пифагора определяется неизвестная длина стороны BC – 8 см.

Шаг 26 аналогичен шагу 1

Используемая операция

Операция поиска значения (find_value).

Шаг 27 аналогичен шагу 2

Используемая операция

Операция поиска формулы (find_formula).

Шаг 28 аналогичен шагу 3

Используемая операция

Операция, организующая поиск в глубину (postorder_tree_search_manager).

Шаг 29 аналогичен шагу 4

Используемая операция

Операция поиска в глубину (postorder_tree_search).

Пояснение

В данном случае новые знания генерируются для объекта *Треугк(A,,B,,C).*

Шаг 30

Используемая операция

Операция получения значения продукции (perform_production);

Операция интерпретации арифметического выражения (calculation).

Пояснение

По формуле вычисления площади прямоугольного треугольника вычисляется искомое значение площади.

Выбор операций, необходимых для решения задач в каждой конкретной прикладной интеллектуальной системе, определяется разработчиком интеллектуального решателя. В связи с этим некоторые операции, необходимые в одной предметной области, будут избыточными в другой.

Например, операции нечеткого и правдоподобного вывода будут очень полезны в системах, где имеется много критериев для принятия решения, анализируется множество характеристик, которые просто невозможно описать с точки зрения однозначной истинности или ложности.

Эти же операции в геометрии Евклида напротив будут избыточны, т. к. решение задач осуществляется только по правилам классического вывода (дедуктивного, обратного и т. д.). Общие рекомендации по составлению задач предлагаются в прил. 3.

6. СЕМАНТИЧЕСКИЙ УНИФИЦИРОВАННЫЙ ЯЗЫК ОПИСАНИЯ ВОПРОСОВ

Язык описания вопросов (или просто язык вопросов) предназначен для формулирования вопросов системе сторонними субъектами, каковыми могут быть как пользователи системы, так и сторонние интеллектуальные системы. Также и в рамках самой системы операции обработки знаний могут формулировать вопросы, предназначенные для других подобных операций.

Каждый вопрос в памяти системы описывается некоторой вопросной конструкцией, содержащей сам узел вопроса и дополнительную информацию, необходимую для поиска ответа на заданный вопрос – тип вопроса, аргументы и т. д.

Ниже приводятся описания ключевых узлов языка описания вопросов в рамках технологии, предлагаемой для решения задач, поставленных в рамках курсового проекта.

- Ключевой узел ***вопрос*** – знак множества вопросов произвольного вида без уточнения конкретной семантики вопроса.

Рис. 6.1. Вопрос

- Семейство знаков множеств частных вопросов.

Примерами знаков множеств частных вопросов могут служить ***запрос значения величины, запрос истинности*** и т. д.

Принадлежность какому-либо классу частных вопросов уточняет семантику данного конкретного вопроса. Классы вопросов можно разделить на предметно-зависимые (запрос яркости небесного тела) и предметно-независимые (запрос значения величины). Однако такое деление достаточно условно, потому как, например, запрос температуры объекта или запрос скорости объекта могут быть использованы далеко не только в системе по физике.

Любой класс частных вопросов является подмножеством множества вопросов (рис. 6.2).

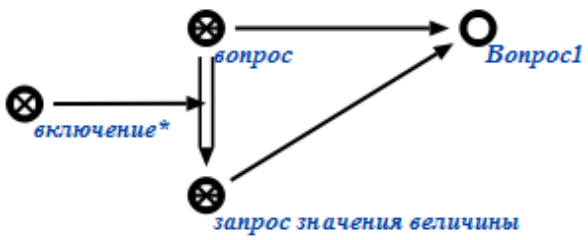

Рис. 6.2. Частный вопрос

• Аргументы вопроса.

Аргументами вопроса могут быть любые элементы базы знаний системы. В зависимости от конкретного класса вопроса их количество может варьироваться от 0 до 10. Аргумент вопроса с теоретико-множественной точки зрения является его элементом, т. е. любой вопрос представляет собой множество аргументов (иногда – пустое множество) (рис. 6.3).

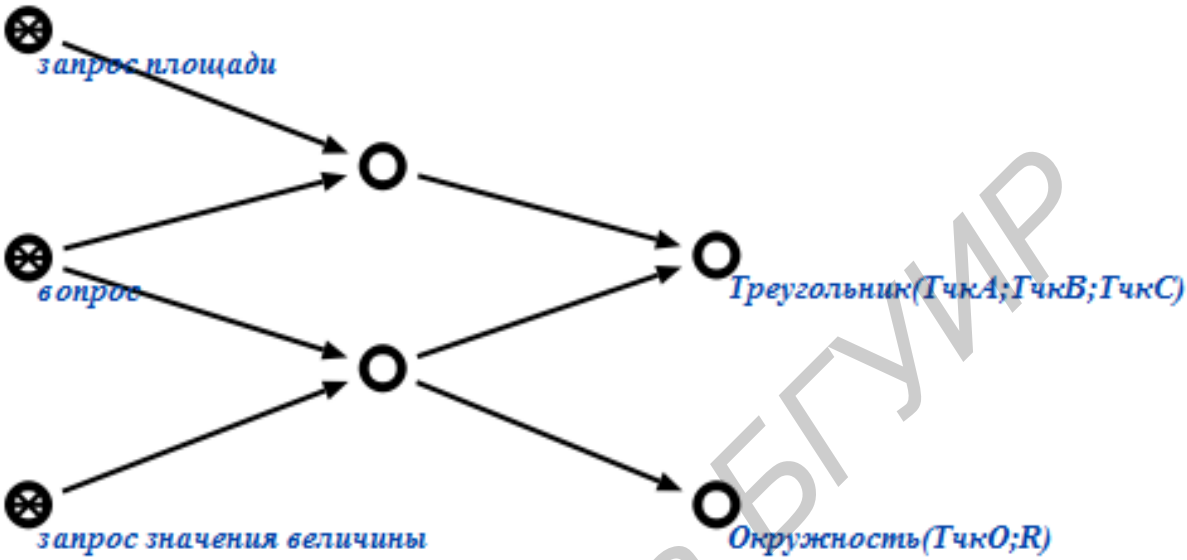

Рис. 6.3. Аргументы вопроса

При необходимости роль каждого аргумента может уточняться при помощи соответствующих ролевых отношений (рис. 6.4):

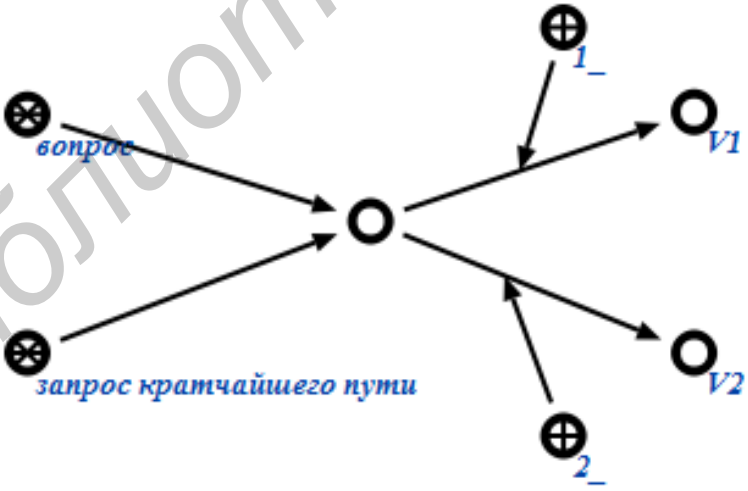

Рис. 6.4. Роль аргументов вопроса

• Ключевой узел ***автор****.

Является знаком отношения, связывающего вопрос с его автором, т. е. субъектом, сгенерировавшем в памяти соответствующую вопросную конструкцию (рис. 6.5).

Указание автора является необходимым в ряде случаев, например, когда вопрос задан пользователем через интерфейс и ответ также должен быть выдан в нужное окно пользовательского интерфейса.

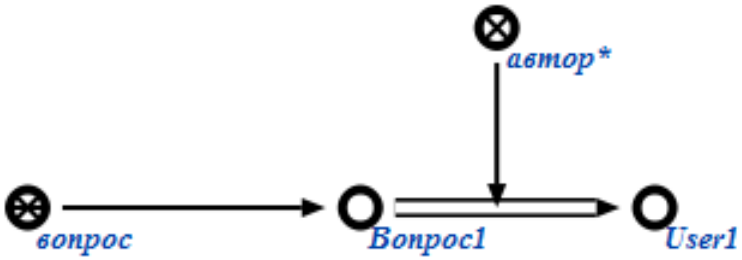

Рис. 6.5. Автор вопроса

- Ключевой узел ***ответ****.

Является знаком отношения, связывающего конкретный вопрос и некоторую конструкцию, являющуюся ответом на данный вопрос.

Вид этой конструкции определяется конкретным классом вопроса. Наличие ответа на вопрос (конечно, при условии возможности его получения) является необходимым независимо от автора, класса вопроса и прочих параметров (рис. 6.6).

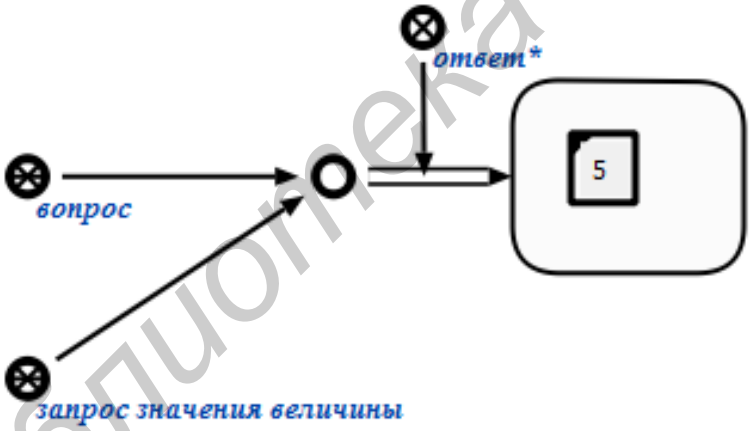

Рис. 6.6. Ответ на вопрос

- Ключевой узел ***решение**** (рис. 6.7).

Является знаком отношения, связывающего конкретный вопрос с решением. Решение представляет собой набор связок, каждая из которых содержит информацию о том, какое утверждение было использовано на данном шаге решения и для какого объекта, а также какая конструкция получена в итоге.

На указанных связках задано отношение строгого порядка, обеспечивающее возможность определения последовательности шагов решения конкретной задачи.

Даже в случае успешного ответа на вопрос решение может отсутствовать, к примеру, когда решение задачи ограничено простым поиском.

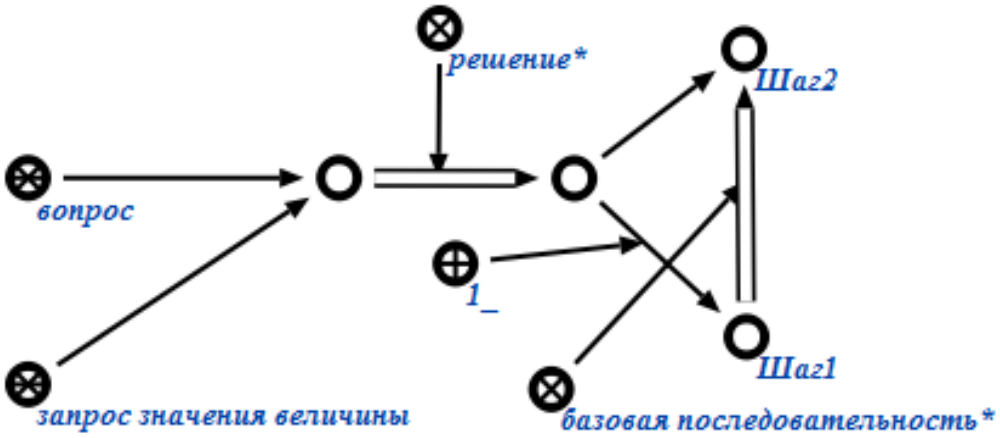

Рис. 6.7. Решение задачи

- Ключевой узел ***инициированный вопрос*** (рис. 6.8).

Является знаком множества инициированных вопросов. Факт инициирования вопроса свидетельствует о том, что вопросная конструкция полностью сформирована и машина обработки знаний может приступать к поиску ответа на поставленный вопрос. Таким образом, генерация дуги принадлежности вопроса множеству инициированных вопросов должна являться завершающим шагом построения любой вопросной конструкции.

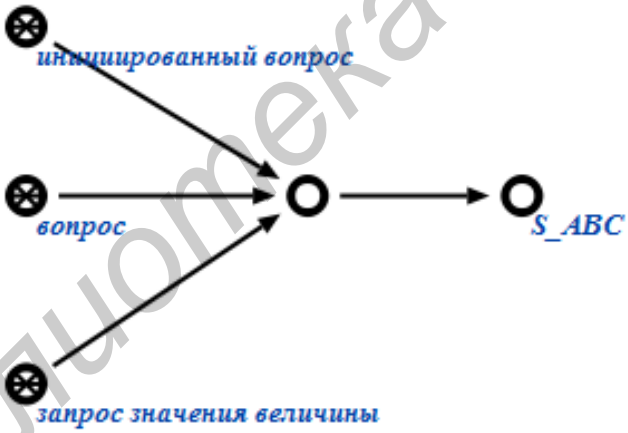

Рис. 6.8. Инициированный вопрос

- Ролевое отношение ***присутствует ответ'*** (рис. 6.9).

Используется для указания того факта, что системе удалось найти ответ на поставленный вопрос. Роль указывается для вопроса в рамках некоторого класса частных вопросов.

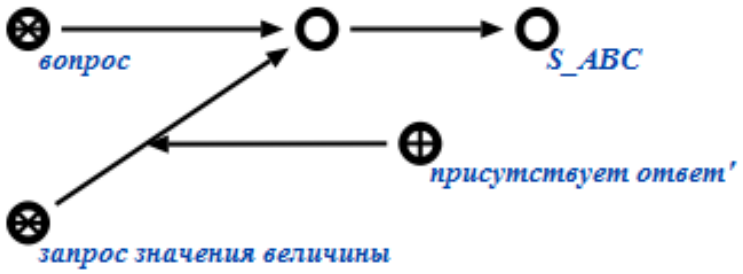

Рис. 6.9. Присутствие ответа на вопрос

Предполагается, что факт присутствия ответа указывается после того, как ответная конструкция полностью сформирована и сгенерирована связка отношения ***ответ****.

- Ролевое отношение ***отсутствует ответ'*** (рис. 6.10).

Используется для указания того факта, что ответ на поставленный вопрос на настоящий момент найден быть не может. Роль указывается для вопроса в рамках некоторого класса частных вопросов.

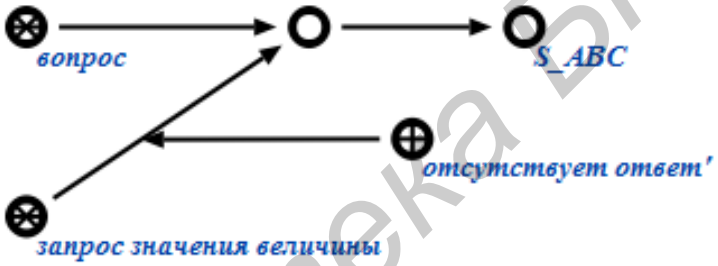

Рис. 6.10. Отсутствие ответа на вопрос

Для дополнительной синхронизации между агентами машины обработки знаний факт отсутствия ответа может уточняться более частными ролевыми отношениями, например, ***ответ отсутствует в явном виде'***.

7. СОДЕРЖАНИЕ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Пояснительная записка содержит описание процесса проектирования интеллектуальной справочной системы, построенной с помощью семантических технологий проектирования интеллектуальных систем либо фрагмента такой системы.

Пояснительная записка должна содержать следующие разделы проекта интеллектуальной справочной системы:

Введение

Освещаются основные направления в области разработки и внедрения интеллектуальных систем. Обосновываются актуальность темы и степень новизны, формулируются цель и задачи курсового проекта.

Технико-экономическое обоснование проектируемой интеллектуальной справочной системы

Обосновывается актуальность разработки интеллектуальной справочной системы по выбранной предметной области; приводится описание пользователя системы (указание категории пользователей системы – исследователь, преподаватель, студент, экспериментатор, школьник, эксперт, турист и т. д.; указание функциональных возможностей, предоставляемых системой, конкретным категориям пользователей); приводятся аналоги систем, решающих проблемы в заданной предметной области, их сравнение с разрабатываемой системой по различным критериям (функциональные возможности, многообразие поддерживаемых видов знаний, обработка знаний и т. д.); указываются преимущества (достоинства) разрабатываемой прикладной системы по отношению к аналогам; обосновывается использование семантической технологии проектирования интеллектуальных систем; оцениваются трудозатраты по реализации (изготовлению) системы.

База знаний интеллектуальной справочной системы

Описываются фрагменты базы знаний, разработанные для отладки реализованных операций машины обработки знаний проектируемой системы. Сюда входят собственно исходные тексты базы знаний проектируемой интеллектуальной системы, оформленные на каком-либо из внешних вариантов записи внутреннего языка представления знаний.

Машина обработки знаний интеллектуальной справочной системы

Описывается собственно разработанная в рамках курсового проекта система операций машины обработки знаний интеллектуальной системы. В данном разделе можно выделить ряд подразделов.

Спецификация машины обработки знаний интеллектуальной справочной системы

Выделяются предметные задачи, которые должна уметь решать проектируемая интеллектуальная справочная система. Предметные задачи необходимо разбить на некоторые типы, т. е. провести классификацию предметных задач. «Интеллект» системы для данной версии определяется многообразием предметных задач и их нетривиальностью решения (т. е. решение задач, для которых явно отсутствуют алгоритмы решения).

Далее приводится список используемых в операциях ip-компонентов, а также перечень уже реализованных компонентов, необходимых для решения предметных задач интеллектуальной справочной системы. Здесь необходимо отметить как сами операции, так и программы, их реализующие и включенные в библиотеки ip-компонентов.

В заключение подраздела приводится декомпозиция операций на подпрограммы и содержательная структура библиотеки программ специфицированных операций. Декомпозиция операций на подпрограммы позволяет перейти на уровень программирования операций. В итоге необходимо указать перечень необходимых подпрограмм для реализации операций и определить структуру библиотеки операций.

Алгоритмы и исходные тексты программ, реализующие операции машины обработки знаний

Приводятся разработанные алгоритмы и тексты scp-программ и шаблоны для информационного поиска, описываются способы их вызовов.

Верификация и отладка программ специфицированных операций

Приводятся тестовые наборы, на которых тестировались разработанные операции. В случае выявленных ошибок приводится протокол тестирования с указанием типа ошибки, фрагмента и другой служебной информации.

Пользовательский интерфейс интеллектуальной справочной системы

Рассматривается взаимодействие между пользователем системы и самой интеллектуальной справочной системой. Пользовательский интерфейс представляет собой специфическую интеллектуальную систему, построенную по семантическим технологиям, которая включает базу знаний и машину обработки знаний пользовательского интерфейса, проектируемых в соответствии с методикой проектирования этих компонентов.

Интеграция разработанной системы с другими системами

В результате такой интеграции может получиться новое качество интегрированной системы, когда на новые вопросы пользователей проинтегрированная система дает ответ, а две прикладные системы в отдельности – нет.

Направления дальнейшего развития разработанной системы

Указываются направления дальнейшего развития прикладной интеллектуальной системы по различным направлениям: добавление новых видов знаний, наполнение базы знаний новыми знаниями, разработка операций обработки знаний с учетом специфики предметной области, разработка пользовательских интерфейсов.

СПИСОК ВОЗМОЖНЫХ ВАРИАНТОВ ТЕМ КУРСОВОГО ПРОЕКТА

1. Интеллектуальный решатель задач по геометрии.
2. Интеллектуальный решатель задач по физике.
3. Интеллектуальный решатель задач по астрономии.
4. Интеллектуальный решатель задач по теории графов.
5. Интеллектуальный решатель задач по лингвистике.
6. Интеллектуальный решатель задач по теории множеств.
7. Интеллектуальный решатель задач по географии.
8. Интеллектуальный решатель задач по числовым моделям.
9. Интеллектуальный решатель задач по фармакологии.
10. Интеллектуальный решатель задач по музыке.
11. Интеллектуальный решатель задач по химии.
12. Интеллектуальный решатель задач по истории.
13. Интеллектуальный решатель задач по изобразительному искусству.
14. Интеллектуальный решатель задач по автодиагностике.
15. Интеллектуальный решатель задач по кулинарии.
16. Интеллектуальный решатель задач по правилам дорожного движения.

ПРИЛОЖЕНИЕ 1
АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ, ПОДДЕРЖИВАЮЩИЕСЯ В СИСТЕМЕ

Отношение **сложение величин*** (рис. П.1.1, П.1.2):

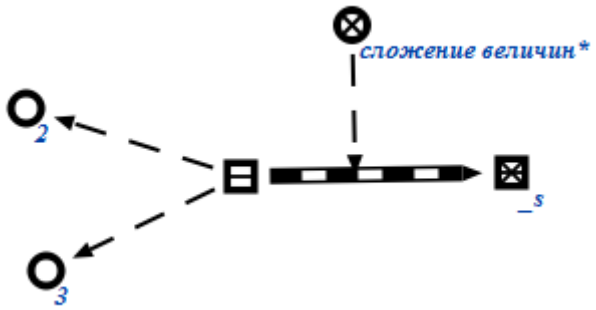

Рис. П.1.1. Сложение двух величин

Обратите внимание, что складывать можно несколько величин.

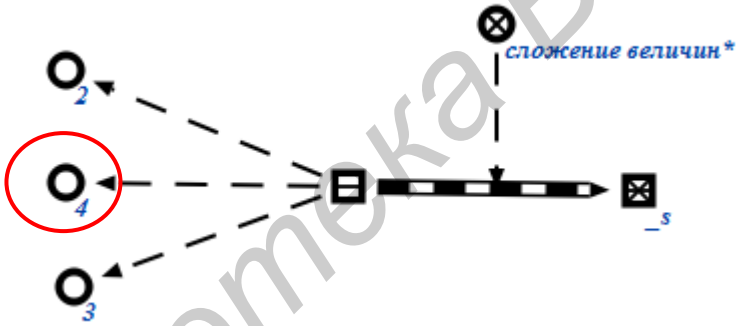

Рис. П.1.2. Сложение нескольких величин

Отношение **произведение величин***:

В рис. П.1.1 следует заменить **сложение величин*** на **произведение величин***.

Отношение **возведение в степень*** (рис. П.1.3):

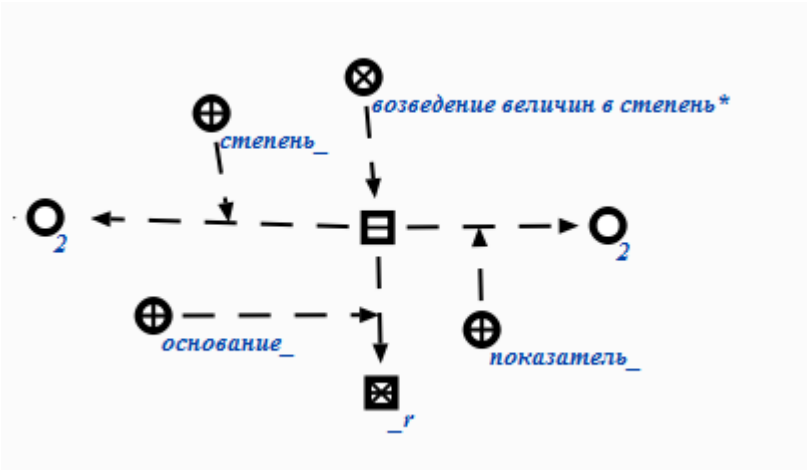

Рис. П.1.3. Возведение в степень

На рис. П.1.3 изображено возведение числа 2 в степень 2 (т. е. 2^2). Обратите внимание, что необходимо не забывать указывать атрибуты «степень_», «основание_», «показатель_».

Отношение **больше*** (рис. П.1.4):

Рис. П.1.4. Сравнение суммы чисел с другим значением

На рис. П.1.4 изображено отношение **больше***. Здесь показано, что сумма двух значение **a** и **b** больше, чем **c**.

Отношение **равенство*** (рис. П.1.5):

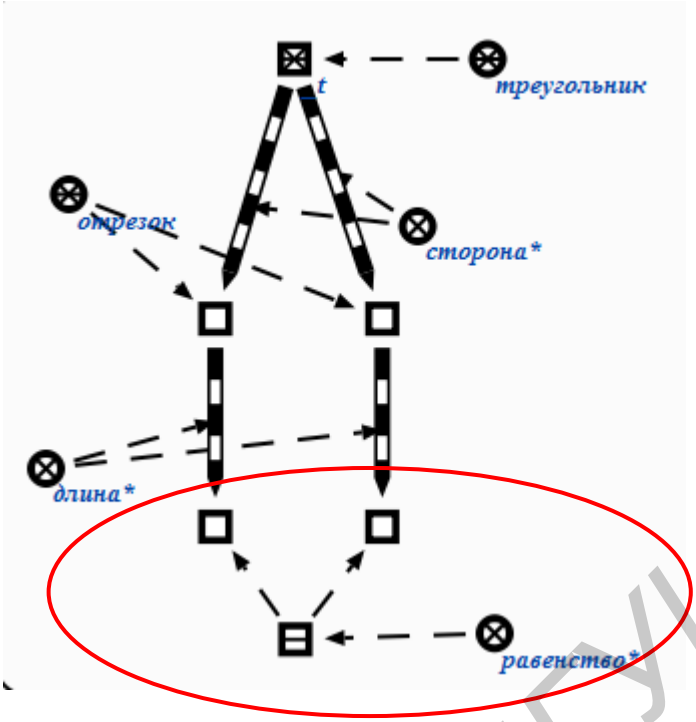

Рис. П.1.5. Отношение **равенство***

На рис. П.1.5 показано, что две стороны треугольника равны. Обратите внимание, как в выделенном элементе описывается отношение **равенство***.

ПРИЛОЖЕНИЕ 2
УТВЕРЖДЕНИЯ, ПОДДЕРЖИВАЮЩИЕСЯ В СИСТЕМЕ

Импликация (**impl***) (рис. П.2.1):

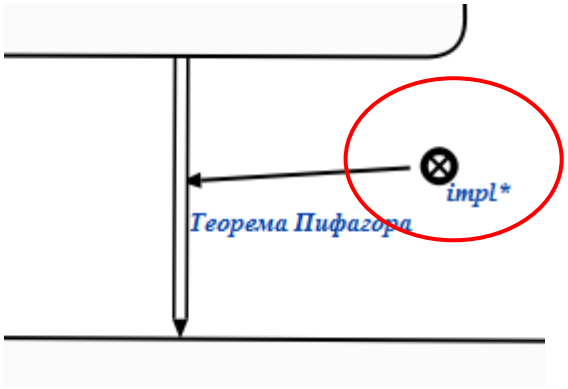

Рис. П.2.1. Импликация

Эквиваленция(**eq***) (рис. П.2.2):

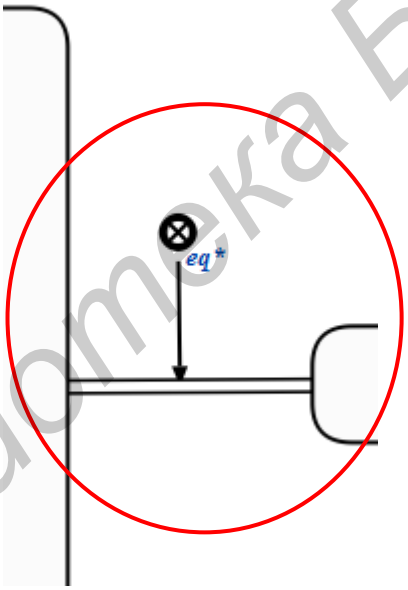

Рис. П.2.2. Эквиваленция

Обратите внимание, как выглядит эквиваленция по сравнению с импликацией.

ПРИЛОЖЕНИЕ 3
ОБЩИЕ РЕКОМЕНДАЦИИ ПО СОСТАВЛЕНИЮ ЗАДАЧ

1. При составлении решения задачи результат применения одного утверждения должен давать возможность использовать следующее, чтобы задача могла решаться по шагам. Подробно рассмотрите пример расчетной задачи. Из него видно, что для того, чтобы найти неизвестное значение радиуса основания, необходимо для начала установить, что радиус, образующая и высота образуют прямоугольный треугольник. А уже из прямоугольного треугольника искать неизвестное значение. Как видим, система это делает по шагам.

2. Ничего очевидного для системы нет. Например, то, что сумма двух частей отрезка дает его длину полностью, для системы не очевидно. Данное утверждение необходимо тоже формализовать, для чего максимально подробно описывать все данные о задаче, учитывать все, чтобы системе было очевидно, что вы хотите получить в итоге. Старайтесь это по максимуму записать в утверждения.

3. Обратите внимание на именование «атомарное существование и единственность» (рис. П.3.1).

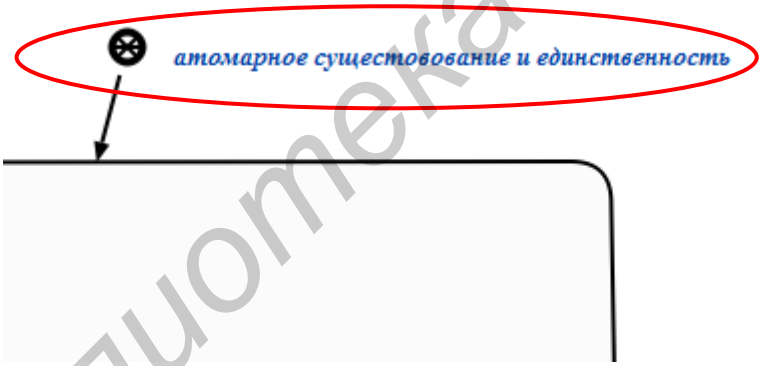

Рис. П.3.1. Ошибка в системе

Данная ошибка находится на стадии решения, но обращаем внимание, что на данный момент рекомендуется писать так, чтобы система решала или доказывала ваши задачи.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Представление и обработка знаний в графодинамических ассоциативных машинах / В. В. Голенков [и др.]; под ред. В. В. Голенкова. – Минск, 2001.

2. Голенков, В. В. Семантическая технология проектирования интеллектуальных решателей задач на основе агентно-ориентированного подхода / В. В. Голенков, Д. В. Шункевич, И. Т. Давыденко // Программные системы и вычислительные методы. – 2013. – №1.

3. Тарасов, В. Б. От многоагентных систем к интеллектуальным организациям / В. Б. Тарасов. – М. : Изд-во УРСС, 2002.

4. Базы знаний интеллектуальных систем: учебник / Т. А. Гаврилова [и др.]. – СПб. : Питер, 2001.

5. Гулякина, Н. А. Методика проектирования семантической модели интеллектуальной справочной системы, основанная на семантических сетях / Н. А. Гулякина, И. Т. Давыденко, Д. В. Шункевич // Программные системы и вычислительные методы. – 2013. – №1.

6. Давыденко, И. Т. Технология компонентного проектирования баз знаний на основе унифицированных семантических сетей / И. Т. Давыденко // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2013) : материалы междунар. науч.-техн. конф., Минск, 21–23 февраля 2013 г. – Минск : БГУИР, 2013. – С. 185–190.

7. Шункевич, Д. В. Модели и средства компонентного проектирования машин обработки знаний на основе семантических сетей / Д. В. Шункевич // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2013): материалы междунар. науч.-техн. конф., Минск, 21–23 февраля 2013 г. – Минск : БГУИР, 2013. – С. 269–280.

Св. план 2014, поз. 28

Учебное издание

Голенков Владимир Васильевич
Гулякина Наталья Анатольевна
Гракова Наталья Викторовна и др.

МОДЕЛИ РЕШЕНИЯ ЗАДАЧ В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ

ПОСОБИЕ

Редактор *М. А. Зайцева*
Корректор *Е. Н. Батурчик*
Компьютерная правка, оригинал-макет *В. М. Задоля*

Подписано в печать 10.04.2015. Формат 60x84 1/16. Бумага офсетная. Гарнитура «Таймс».
Отпечатано на ризографе. Усл. печ. л. 4,3. Уч.-изд. л. 4,1. Тираж 150 экз. Заказ 148.

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий №1/238 от 24.03.2014,
№2/113 от 07.04.2014, №3/615 от 07.04.2014.
ЛП №02330/264 от 14.04.2014.
220013, Минск, П. Бровки, 6