УДК 004.3

О.В. Морозова, аспирант
А.В. Григорьев, канд. техн. наук, доцент,
Донецкий национальный технический университет, г. Донецк, Украина
olmalyavka@gmail.com, grigorie@r5.dgtu.donetsk.ua

Анализ методов построения экспертных систем в продукционных инструментальных оболочках

Работа посвящена анализу типичных инструментальных оболочек для создания экспертных систем. Выделены достоинства и недостатки этих оболочек с точки зрения применяемых методов построения системы продукционных правил и механизма вывода. Сделаны выводы о достоинствах и недостатках применения данных систем для создания интеллектуальных САПР с адаптацией на требуемую предметную область

Ключевые слова: Clips, Prolog, Nexpert Object, Exys Corvid, Jess, G2, продукционная модель, механизм логического вывода

Введение

В настоящее время активно идет процесс повышения уровня интеллектуальности самых разнообразных САПР, что чаше выражается в их построению как гибридных.

Гибридная САПР включает в себя как обычные средства и методы построения моделей объектов проектирования, так и – интеллектуальные методы аналогичного назначения. Чаще всего гибридизация достигается путем включения в проблемно-ориентированный САПР экспертных систем (ЭС), автоматизирующих процессы проектирования. Т.о, в САПР надстраивается база знаний (БЗ), которая реализует методику проектирования в данной предметной области (ПрОб) [1]. Построить такую интеллектуальную САПР означает, с одной стороны, выбрать методы построения ЭС, а с другой стороны, выбрать методы построения интерфейса «САПР – ЭС».

Следует отметить, что выбор методов построения ЭС проектирования должен отвечать следующим специфическим требованиям:

- способность обеспечить передачу знаний в ЭС от проектировщика, играющего роль инженера по знаниям;
- способность учесть уровень квалификации, подготовки проектировщика, что выражается в его способности связно и целостно изложить методику проектирования, а так же – свое видение модели объекта проектирования;
- применение некоторого общего подхода ко всем ПрОб, используя некоторые универсальные принципы построения САПР;
- обеспечение предметной адаптации ЭС, т.е. отражения в ней как специфики модели проектирования, так и – специфики методики проектирования в данной ПрОб.

Выбрать метод автоматизации построения таких ЭС, в частности, означает выбрать:

- метод представления знаний;
- метод организации вывода требуемого решения;
- метод задания технического задания на требуемое решение и т.д.

Следует отметить, что данная задача в настоящее время не имеет полноценного решения. Это делает актуальным задачу исследования разнообразных методов подходов к автоматизации построения ЭС, применяемых в настоящее время, с целью построения комплекса средств решения поставленной задачи.

Современное развитие систем искусственного интеллекта привело к созданию большого количества универсальных программных средств для построения ЭС различного назначения, так называемых «оболочек». В основе большинства таких продуктов лежит механизм накопления знаний о ПрОб и способов их обработки. Большинство существующих инструментальных оболочек (ИО) являются оболочками общего назначения. К ним можно отнести классические «универсальные» системы, такие как Clips, Prolog, Nexpert Object, Exys Corvid, Jess и т.д. Названные системы, хотя и обладаю мощными возможностями в представлении знаний и организации вывода, но не обладают достаточно развитыми средствами предметной адаптации с точки зрения САПР. С другой стороны, имеются оболочки нового поколения типа G2 или СПРУТ, имеющие более суженые возможности в представлении знаний и

организации вывода, уже ориентированные на некоторую ПрОб, но одновременно обладающие способностью адаптироваться в рамках данной ПрОб на специфику задачи.

Анализ как тех, так и других ИО интересен с точки зрения поставленной задачи, т.е. – выбора соответствующих средств и методов построения ЭС, адаптируемой на любую техническую ПрОб в САПР.

Данные программные комплексы используют разнообразные специфические способы представления и обработки знаний, основанные на тех или иных стандартных подходах. В частности, для представления знаний используются фреймы, семантические сети, системы продукций, нечеткие множества, нейронные сети, генетические алгоритмы и др.

Однако, в рамках данных инструментальных средствах нет возможности создания полноценных интеллектуальных САПР, так как большинство из них не способны в достаточной степени адаптироваться на техническую ПрОб.

ИО требуемого типа должна использовать принципы построения оболочек названных типов, одновременно будучи ориентированной на создание интеллектуальной САПР для избранной технической ПрОб.

В частном случае, оболочка такого типа может обладать средствами адаптации на ПрОб в рамках концепции «умного эксперта», т.е. профессионального проектировщика, обладающего большим опытом проектирования в данной РоОб и способного задать методику проектирования устройств данного типа как продукционную базу знаний в рамках выбранной ПрОб без привлечения инженера по знаниям [2].

Кроме того, сузим задачу, рассматривая только ИО создания ЭС продукционного типа, предполагая, что данные системы в наибольшей степени соответствуют концепции «умного эксперта».

Целью предлагаемой работы является анализ методов построения различных оболочек продукционного типа для выявления их достоинств и недостатков с последующим использованием результатов анализа при решении задачи построения оболочки общего назначения, ориентированной на технические ПрОб САПР, функционирующей в рамках концепции «умного эксперта» в ПрОб.

Направление анализа инструментальных оболочек

В настоящее время уже имеются работы, посвященные анализу и классификации такого рода ИО, например, [3]. В данных классификациях отмечают общие характеристики, такие как:

- наличие графического интерфейса работы с базой знаний (БЗ),
- наличие модифицированных механизмов логического вывода и др.

Однако представляется важным другой аспект подобного анализа, а именно, применимость данных оболочек для создания интеллектуальных САПР различного назначения, то есть наличие в данных ИО средств предметной адаптации. Интерес представляют следующие аспекты анализа:

- возможность построения интегрированных САПР, объединяющих готовую ЭС с проблемно-ориентированными САПР, эффективность работы которых может быть повышена за счет включения в них БЗ;
- наличие интерфейса с проблемно-ориентированными САПР для решения задачи получения описания некоторых апробированных решений с целью создания БЗ путем обучения на прецедентах;
- возможность отразить специфику представления моделей объектов проектирования в САПР, т.е. - возможность отражения физической семантики ПрОб, включая потоки, потенциалы, координаты взаимодействия (емкость, индуктивность и т.д.), возможность описания моделей пространства и времени, отражения законов сохранения и т.д.;
- анализ уровней абстракции представления объектов в САПР, которые способны поддерживать данные оболочки, включая структурный, системный, функционально–логический и количественный уровни абстракции;
- полнота реализации моделей того или иного имеющегося абстрактного уровня по составу отношений, используемых в САПР на данном уровне абстракций;
- возможность отразить те или иные методы проектирования, существующие в «обычных» САПР различных ПрОб, как продукционную БЗ;
- наличие модульности в БЗ и соответствие модулей структурным или функциональным типам блоков в модели объекта проектирования;
- возможность адаптироваться на квалификацию проектировщика в инженерии знаний, предполагая, что он не является профессиональным инженером по знаниям, но является квалифицированным, опытным специалистом в проектировании.

Проделаем анализ перечисленных выше ИО с данной точки зрения.

Оболочка G2

Одной из наиболее универсальных платформ для разработки и внедрения экспертных приложений является платформа G2 компании Gensym [4].

ЭС, создаваемые в G2, можно отнести к системам оперативного управления бизнес-процессами. Данная платформа может решать несколько типов задач, такие как:

- моделирование;
- реализацию расписаний и планирований;
- диагностика;
- мониторинг в реальном времени;
- системы проектирования.

На рис. 1 представлена схема взаимодействия всех компонентов платформы G2.

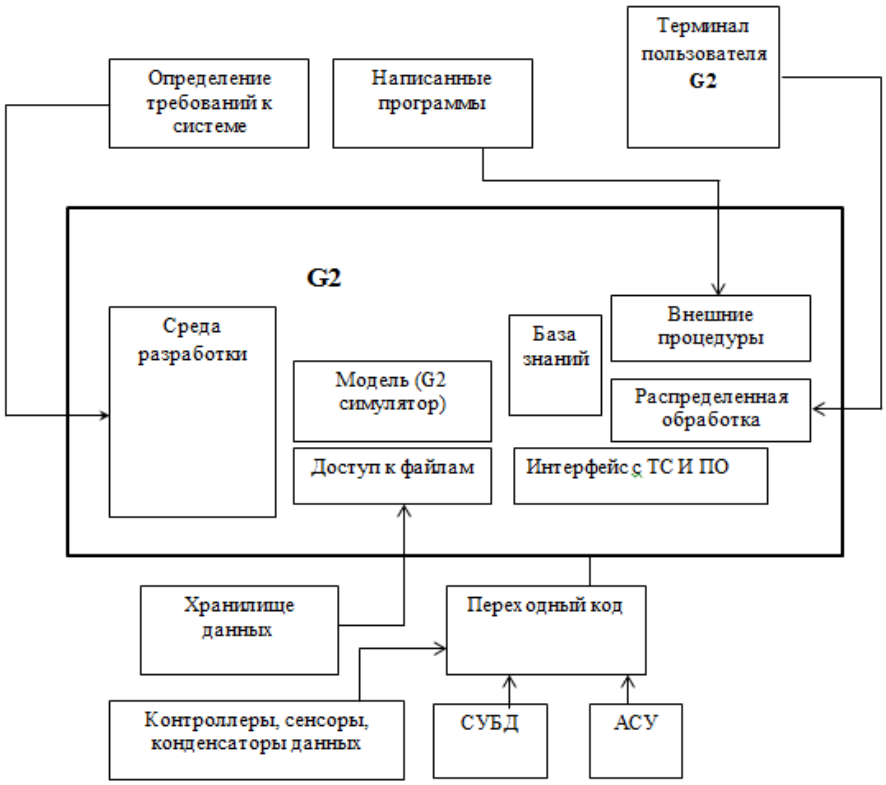

Рисунок 1 – Схема платформы G2

Основу экспертной компоненты G2 составляет система продукционных правил. Правила в системе G2 могут быть разного типа и назначения, например:

- обобщенные, касающиеся целого класса объектов;
- специфические, относящиеся к конкретному экземпляру объекта.

В посылке правила могут указываться условные выражения и директивы, определяющие порядок выполнения утверждений, следствия - могут быть последовательные или параллельные. Важной особенностью механизма логического вывода G2 является и большой набор способов выполнения и активации правил. Правило в G2 может быть активировано в одном из случаев:

1. Данные, входящие в посылку правила изменились (прямой вывод).
2. Правило присваивает переменной значение, которое требуется другому правилу или процедуре (обратный вывод).
3. Через временной промежуток, который определен для данного правила.
4. Явная или неявная активация другим правилом, которое применяет операцию фокусирования.
5. Переменной, находящейся в посылке, присваивается значение, независимо от того, изменилось оно или нет.
6. Всегда, когда запускается некоторое приложение.
7. Определенный объект перемещен либо пользователем, либо другим правилом.
8. Определенное отношение между объектами установлено или уничтожено.
9. В результате обращения к своему источнику данных переменная не получила значение.

Разберем перечисленные пронумерованные случаи работы и активации правил в системе G2.

Первые два способа в G2 поддерживают обратную цепочку рассуждений (дедуктивную, поиск цели), прямую цепочку рассуждений (индуктивную, на основе событий), а также смешанные заключения. Одно и то же правило может использоваться как в прямой, так и в обратной цепочке рассуждений. Правила могут быть отнесены к различным категориям (например, «одобрение кредита с высокой степенью риска») и ассоциированы с классами объектов в модели. Эти механизмы позволяют разработчику целенаправленно объединять правила в цепочки для достижения любой необходимой степени эффективности.

Следующий, 3-й метод механизм есть механизм запуска процедур-демонов.

4-й метод при работе приложений в любой момент времени позволяет сфокусироваться на тех правилах, которые необходимы для определенного объекта, или заранее введенные пользователем.

Методы с 5 по 9 являются методами обработки данных на базе управления событиями. Для их реализации в G2 определен специальный тип правил, начинающихся с ключевого слова «как только». Правилам с предписанием «как только» будет присвоен наибольший приоритет и они должны срабатывать в первую очередь. Такой путь обеспечивает работу системы в реальном времени, ориентируясь на изменение внешнего мира [4].

Рассмотрим синтаксис правил, который представлен на рис. 2.

Т.о, в системе G2 форма представления продукционных правил обеспечивают как быстродействие их срабатывания, так и - отслеживания их влияний на внешнее окружение объекта.

Система G2 использует процедурный подход в представлении БЗ. Для процедурного подхода в системе G2 был разработан собственный язык программирования, близкий с Паскалем. В данный язык добавлены элементы для работы процедуры в реальном времени, а именно:

- отслеживание наступления события;
- разрешение прерывать другим задачам данную процедуру;

```
<правило> ::= {[<префикс for>]
{<правило if> | <правило unconditionally>
| <правило when> | <правило whenever>}
| <правило initially>}
<префикс for> ::= for {any | the } <item> ...
<правило if> ::= if <логическое выражение> then <список действий>
<правило unconditionally> ::= unconditionally <список действий>
<правило when> ::= when <логическое выражение> then <список действий>
<правило whenever> ::= whenever <описание события>
[or <описание события>] [and when <логическое выражение>] then <список действий>
<правило initially> ::= initially
[if <логическое выражение> then] <список действий>
```

Рисунок 2 - Правила в форме Бекуса-Наура

- директивы, которые определяют порядок выполнения операций (последовательных или параллельных);
- итераторы, которые могут организовывать цикл над множеством экземпляров класса.

Данный язык позволяет одновременно обрабатывать множеству процедур для множества различных объектов.

Несмотря на сложность и богатство синтаксических конструкций G2 для описания знаний, их применение упрощается за счет естественно-языкового подхода.

G2 работает с разными ПрОб, такими как телекоммуникации, химическая, нефтяная и газовая промышленность, аэрокосмическая и др. В G2 встроен механизм адаптации на ПрОб. Для обеспечения участия специалиста из той или иной ПрОб в процессе разработки приложений, в G2 поддерживаются возможность задания правил как на естественном языке, так и - на графическом. Собственно правила могут быть организованы различными способами, такими как таблицы, древовидные диаграммы, и иерархические рабочие пространства. Правила и порядок их выполнения могут представлены в виде графических структур, которые отображают взаимодействие, интеграцию и последовательность для моделей процессов.

Оболочка Clips

Наиболее распространённым универсальным языком представления знаний в настоящее время является Clips. Программная среда Clips предназначена для создания ЭС. Более 80% экспертных систем строятся именно на его основе [5]. Структурная схема работы среды Clips представлена на рис. 3.

Язык Clips основан на системе правил. Эти правила являются порождающими, то есть срабатывание правила добавляет данные по предложению-образцу. Т.к. Clips оперирует известной продукционной моделью представления знаний, то, соответственно, рассматриваются 3 основных элемента:

- база фактов;
- база правил;
- блок логического вывода.

Основными компонентами языка описания правил являются база фактов и база правил. На них возлагаются следующие функции:

- в базе фактов хранятся исходные состояния ПрОб;
- в базе правил хранится информация об операциях, которые необходимо выполнить для решения поставленной задачи.

Правила в Clips имеет вид "Если (условие), то (действие)". Под "условием" понимается некоторое предложение-образец, по которому осуществляется поиск в базе данных, а под "действием" – действия, выполняемые при успешном исходе поиска.

Блок логического вывода Clips анализирует факты с точки зрения различных правил и решает, какие из правил можно отправить на обработку. Данный блок работает циклически, в котором каждый цикл состоит из нескольких шагов:

1. Сравнивание фактов и правил;
2. Выбор нужного правила отправленного на выполнение;
3. Выполнение операций, записанных в правиле.

При запуске приложения Clips пользователю предоставляется приглашение и сообщение о том, что он работает в режиме интерпретатора [6].

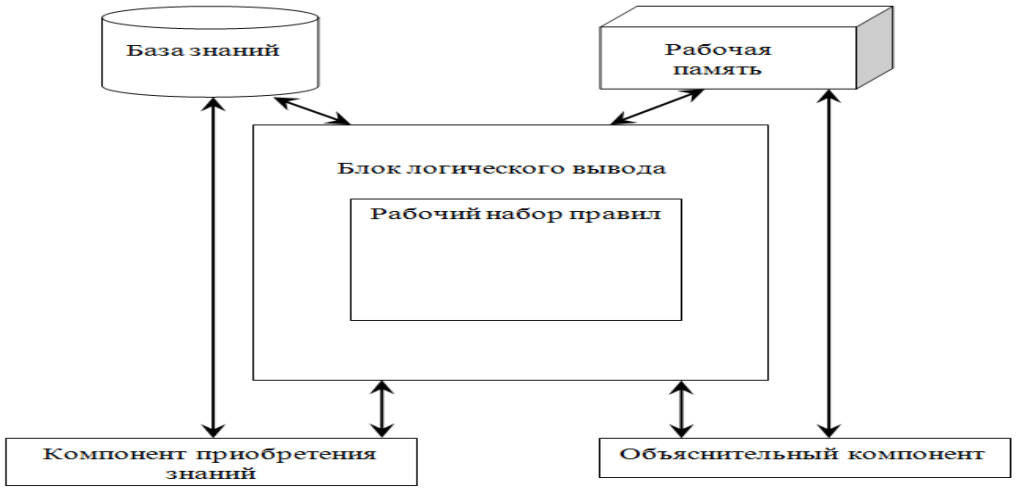

Рисунок 3 – Схема работы среды Clips

В данном режиме у пользователя появляется возможность работы с множеством команд. При этом пользователь может самостоятельно создавать факты (используя командную строку) и записывать их в базу фактов. Для определения фактов можно использовать не только списочные структуры, но и шаблоны, которые напоминают простые записи.

В языке Clips также можно определять функции, они имеют схожий вид и синтаксис с функциями языка LIPS. В записи функции должен быть записан префикс «?». Функция возвращает как результат последнее выражение в списке.

Clips реализует прямой логический вывод, что обеспечивает высокую степень возможности прогнозирования. При этом, если посылки двух правил одинаковы, то необходимо задавать еще один параметр с определенным числовым значением, трактуемым как приоритет выполнения правила.

Оболочка Java Expert System Shell (Jess)

Jess – это оболочка для разработки ЭС, написанная Эрнестом Фридманом-Хиллом (Sandia National Laboratories in Livermore, CA) полностью на языке Java компании Sun Microsystems. Jess изначально являлся производной языка CLIPS, но вскоре вырос в полную, отдельную, динамическую среду [7]. Используя Jess, можно построить Java-приложение с возможностью обработки данных на основе знаний, представленных в виде

правил. На данный момент, Jess - один из наиболее легких и быстрых оболочек для создания ЭС. Структура Jess представлена на рис. 4.

Система Jess как и Clips, основана на правилах. Jess является декларативным языком, то есть программа состоит из набора инструкций. Так же можно считать, что Jess является интерактивным интерпретаторам. Система может работать в режиме реального времени и – в пакетном режиме. Пакетный режим предполагает, что один и несколько файлов могут работать и выполняться одновременно. Скриптовый язык Jess все еще совместим с Clips, т.е. большинство скриптов Jess будут работать в Clips и наоборот. Jess также включает в себя возможность создания, управления и вызова методов Java объектов [8].

Представление знаний и накопление баз знаний в системе Jess происходит с помощью правил и фактов. Рассмотрим вкратце их синтаксис. Синтаксис факта приведен на рис. 5. Синтаксис и вид записи правила в Jess праведен на рис. 6. Фактически в этом случае записывается факт и соответствующая последовательность выполнения функций.

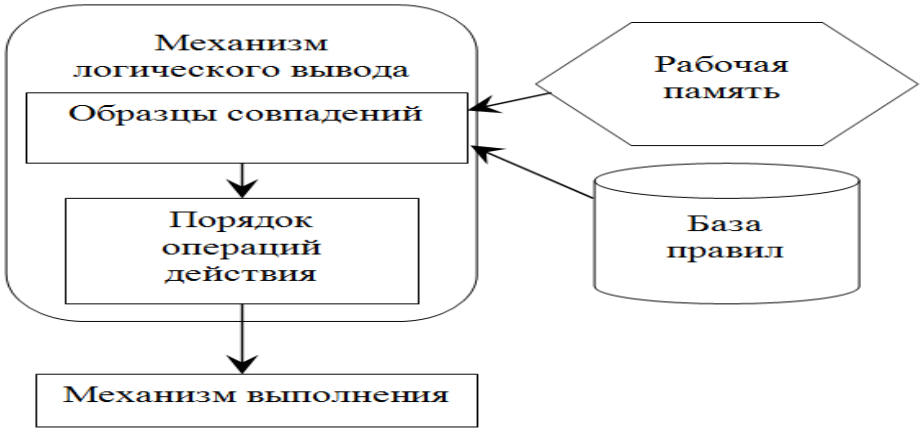

Рисунок 4 – Схема работы системы Jess

```
(deftemplate <name> [<comment>] (slot| multislot <slotname>
(<slotoptions>)*)*)
```

Рисунок 5 - Запись факта на Jess

```
(defrule <name> [<comment>]<fact>* => <function>*)
```

Рисунок 6 - Запись правила на Jess

Jess реализует прямой логический вывод. В основе механизма вывода Jess лежит Rete-алгоритм, который повышает эффективность и быстродействие на задачах множественного сравнения. Rete-алгоритм представляет собой очень быстрое средство сравнения с шаблонами. Алгоритм запоминает результат последнего тестирования знаний и заново проверяет только вновь появившиеся факты.

Данный алгоритм реализован с помощью построения системы узлов, каждый из которых представляет одно или несколько проверок фактов для каждого правила. Факт, который добавляется или удаляется из базы знаний, обрабатывается этой системой узлов. В основе этой системы лежат конкретные действия. В Rete-алгоритме воплощены два эмпирических наблюдения, на основании которых была предложена структура данных, лежащая в его основе:

1) Временная избыточность: изменения, возникающие в результате запуска одного из правил, обычно затрагивают лишь несколько фактов, а каждое из этих изменений влияет только на несколько правил.

2) Структурное подобие: один и тот же шаблон часто обнаруживается в левой части больше чем одного правила.

Оболочка Prolog

Еще одной «классической» средой разработки ЭС является Prolog. Его особенность и отличие от других языков состоит в том, что он является декларативным языком или языком исчисления предикатов. Предикат – это логическая формула от одного или нескольких аргументов. Можно сказать, что предикат – это функция, имеющая бинарное множество значений {ложно, истинно}. Prolog предполагает наличие набора фактов и правил, которые обеспечивают нахождение решений на основе этих фактов. То есть решение Prolog получает путем логического вывода на основе введенных ранее известных тезисов. Работа инструментальной оболочки представлена на рис. 7.

Основной формой представления данных в Prolog являются факты. Факты в Prolog изображают объекты и отношения, правила предполагают проверку истинности этих отношений. Отношение между объектами и есть факт [9]. Если в естественном языке отношения устанавливаются в предложении, то в Prolog (в логике предикатов), отношения соответствуют простой фразе, состоящей из имени отношения и объекта, заключенных в круглые скобки. Факты так же выражают свойства объекта. Синтаксис факта в Prolog:

Имя(Объект).

Более сложное понятие в Prolog - это правила. Они позволяют вывести один факт из других фактов. Правило является истинным заключением, если было найдено другое истинное заключение или факт является истинным.

Синтаксис правил в Прологе имеет две части: заголовок и тело:

Заголовок: — <Подцель>, <Подцель>, ... , <Подцель>.

Рисунок 7 – Структурная схема работы системы Prolog

Тело правила состоит из одной или более подцелей. Подцели разделяются запятыми, что соответствует конъюнкции, за последней подцелью правила следует точка.

При наличии записи нескольких фактов в Prolog возникает вопрос, какие же отношения существуют между этими фактами. Исходя из этого, в системе Prolog создаются запросы, то есть вопросы о наличии тех или иных отношений между фактами. Запрос отрабатывается перебором всех фактов, начиная с первого, вплоть до исчерпания фактов.

Синтаксис Prolog разработан для того, чтобы отображать знания о свойствах и взаимосвязях. Типичная программа на Prolog состоит из четырех основных программных разделов, к которым относятся:

- раздел предложений;
- раздел предикатов;
- раздел доменов;
- раздел целей.

Раздел предложений — это основа Prolog-программы; именно в этот раздел записываются факты и правила, которыми будет оперировать Prolog, пытаясь разрешить цель программы.

Раздел предикатов — это тот, в котором объявляются предикаты и типы их аргументов.

Раздел доменов служит для объявления доменов, не являющихся стандартными доменами Prolog.

В раздел целей помещается цель Prolog-программы.

Механизм логического вывода Prolog основан на сравнении фактов и является обратным логическим выводом. С помощью подбора ответов на запросы он извлекает хранящуюся (известную) информацию. Он пытается проверить истинность предположения, запрашивая для этого информацию, о которой уже известно, что она истинна. На этом основан принцип резолюции, который требует представления формул исчисления предикатов в виде набора дизъюнктов, связанных операцией конъюнкции. Метод резолюции представляет собой прохождение дерева решений в глубину. Он так же может выбирать из альтернатив и находить возможные решения. Механизм логического вывода Prolog может возвращаться назад и просматривать более одного "пути" при решении всех составляющих задачу частей.

Оболочка Exsys Corvid

Система-оболочка Exsys Corvid представляет собой инструментальное средство, которое может быть использовано для разработки ЭС в любой ПрОб. Она относится к системам дедуктивного продукционного типа, которая поддерживает различные режимы конструирования ЭС и способна обрабатывать неопределенности [10]. Эта система ориентирована на различные классы пользователей в зависимости от их подготовки в области искусственного интеллекта и программирования, имеет развитый интерфейс с современными СУБД и электронными таблицами, средства сбора статистики и т. д.

Знания в системе представлены в виде продукционных правил. Основной частью системы является база знаний, которая наполняется по мере работы системы. В базе знаний хранятся правила. Правила могут быть двух типов с одним заключением или с альтернативным заключением. Синтаксис правил:

1 тип IF (условие) THEN (заключение);

2 тип IF (условие) THEN (заключение 1) ELSE (заключение 2)

Также в правилах задается коэффициент уверенности, который определяет вероятность выполнения данного правила. Данный коэффициент задает эксперт. В Exsys Corvid коэффициент уверенности оценивается по трем шкалам:

Коэффициенты уверенности в первой шкале бинарные, т.е.: 0 – ложь, 1 – истина.

Вторая шкала имеет десятеричные значения, в которых 0 – абсолютная ложь, 10 – абсолютная истина, значения между ними показывают различные степени уверенности.

Третья шкала является более подробной по отношению ко второй и имеет значения от -100 – абсолютная ложь, +100 – абсолютная истина, значения между ними трактуются так же, как и во второй шкале.

Exsys Corvid позволяет использовать квалификаторы для создания правил. Квалификатор – это специальная текстовая переменная, которая имеет имя и может принимать одно из нескольких значений. Квалификаторы используются для формирования условий [11]. При работе система позволяет вносить изменения для правил, квалификаторов их текстовую информацию и вариант значения.

Одной из особенностей системы Exsys Corvid является возможность вывода информации из других правил, что позволяет разбивать сложные задачи на подзадачи.

В системе Exsys Corvid реализованы два механизма логического вывода - прямой и обратный. В Exsys Corvid имеется возможность объяснения полученных результатов. При необходимости можно выяснить весь ход вычислений, вплоть до исходных данных. Система Exsys Corvid в процессе поиска решения способна выполнять ряд вычислительных задач, но некоторые вычислительные задачи выходят за рамки его возможностей. В Exsys Corvid имеется возможность вызова внешних программ для выполнения вычислений. Внешние программы могут передавать данные Exsys Corvid через БД.

Оболочка Nexpert Object

Система Nexpert Object фирмы NEURON DATA - это многоплатформенное средство с набором библиотек с элементами управления баз знаний и оригинальным механизмом двунаправленного вывода с обработкой исключающих ситуаций путем различия "сильных" и "слабых" связей.

Nexpert представляет собой гибкую инструментальную среду для разработки приложений и ЭС. В основе системы лежат продукции и объектно-ориентированные схемы.

Данная ИО имеет «открытую» архитектуру, которая позволяет ей взаимодействовать с внешними модулями и внешними программами. Внешние модули и программы могут работать с этой системой с помощью правил и объектов [12].

Особенностями Nexpert Object являются прямые и обратные цепочки рассуждений. Основные возможности Nexpert Object:

- способность использовать симметричный формат правил;
- автоматически генерировать цели;
- сопоставлять с образцом;

- интерпретировать;
- динамически создавать объекты, классы, свойства, методы, множественное и пользовательское наследование;
- немонотонное рассуждение.

Nexpert включает в себя графический интерфейс, который позволяет разработчикам и экспертам в этой ПрОб изменить правила, а также отображать обзор правил, с использованием динамического и графического механизма просмотра. Базы знаний могут быть разработаны на разных платформах, а после перенесены на конечное место обработки. Nexpert имеет интерфейс прикладного программирования (API), который предоставляет разработчику доступ к функциям Nexpert-библиотеки. Nexpert интегрирован с стандартными реляционными базами данных и электронными таблицами двух классов.

Первый класс использует известные встроенные процедуры передачи информации с таблицами таких «классических» форматов как NXP (Nexpert собственный формат), SYLK (Microsoft Excel), и WKS (Lotus), а также следующие базы данных: DBASE III, DBASE III Plus (DBF формата), FoxBase, NXPDB (собственные собственной базе данных в NEXPERT), SylkDB (Excel), и WKSDB (Lotus).

Второй класс интеграции использует отдельные процедуры для подключения к Nexpert основных реляционных баз данных, разрабатываемых на современном рынке (например, Oracle, Sybase, Ingres, Informix). Любой запрос SQL может быть вызван из Nexpert и передан в СУБД.

Существует взаимно однозначное соответствие между ячейками таблиц, записями и полями баз данных, с одной стороны, и - классами, объектами и свойствами в базах знаний, с другой стороны.

Такое соответствие обеспечивает целостную структуру представления фактов в системе Nexpert Object.

Мета-эвристическая оболочка

Суть подхода состоит в формировании на основе множества текстовых описаний моделей прототипов в форме И-ИЛИ-дерева и формирования ряда продукций над ИЛИ-узлами (синтермами), связывающими имеющий смысл в данном наборе прототипов комбинации признаков. Подход работает в пакетном режиме.

Пакетные продукции впрямую реализуют перечисленные выше зависимости и этапы синтеза внутренней структуры блока на основе И-ИЛИ-деревьев, косвенно задаваемых в продукциях [13].

Для передачи системе информации о порядке выбора продукций описываются правила вывода. Грамматика языка описания правил вывода имеет такие особенности. Каждое правило вывода имеет свое уникальное имя. В правой части правила описывается выражение, аргументами которого являются имена продукционных правил, а так же имена самих правил. В правой части могут быть использованы скобки и отношения "и" и "или". Пример формата правила приведен на рис. 8.

В выражении позволяется использовать все имена продукций, описанных в проекте. Правилом цели является правило, описанное в начале. Правила перед отработкой переводятся в полиз, а далее по полизу строится "и-или" дерево вывода. В вершинах дерева находятся имена продукционных правил.

Формат продукции, принятый в системе следующий:

Имя Продукции: "Условная часть" Тип Продукции "ТО-часть".

Имя продукции - это строка символов, необходимая для управления порядком отработки продукций. "Условная" и "ТО " часть продукции имеют идентичный формат. Как "условная" так и "ТО" часть состоят из элементов. Элементы представляют собой нисходящий уровень описания детализации структуры проектируемого объекта по формату: {Название_библиотеки.Тип.Массив.Номер.Свойства.}. В процессе отработки продукции для "условной" части проверяется наличие элементов в БД. Если "условная" часть истинна, то элементы "ТО"-части создаются в БД.

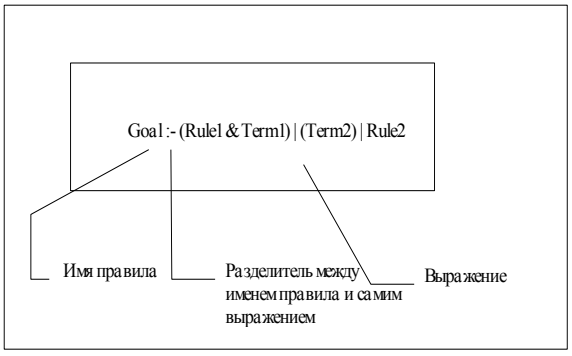

Рисунок 8 - Пример формата правила

Продукции могут быть определенными и недоопределенными. Если для одного из элементов продукции не существует ни одного известного решения, то на место имени элемента ставиться зарезервированное слово New. В этом случае значение элемента будет определятся в диалоге с пользователем в процессе отработки продукции.

Использование зарезервированного слова NIL предполагает выбор пользователем в диалоге любого значения из числа возможных для данного элемента.

Между "условной" и "ТО" частями стоит символ, определяющий тип продукции. Возможные варианты:

1. "->" - порождает элемент со своими свойствами;

2. "=>" - порождает связи в проектируемой модели;

3. Безусловная продукция. Она выполнятся всегда и в ее теле отсутствует "условная" часть.

В продукциях на месте значения элемента [свойства] (после знака "=") могут стоять: произвольные строки; действительные числа; переменные (их значения вычисляются в процессе разбора); математические функции или выражения; скобки. Переменные, математические функции и выражения должны стоять внутри символов "<" ">". Переменные могут описываться отдельно от продукций между символами # # и им может присваиваться требуемое значение с помощью знака ":=". Переменные могут описываться и в самих продукциях, перед значением элемента [свойства], после знака "=". Комментарии записываются между "(*" и "*)".

Метод взаимного отображения вариантов предполагает взаимно-однозначное соответствие одинаковых номеров в случае одинакового числа вариантов и полное покрытие большего числа вариантов меньшим путем повторения, в случае их разного количества в посылке и действии. В случае, если при отработке продукций отрабатывает более чем одно условие посылок, то формируется дерево вывода, приводящее к порождению ряда решений. Дерево вывода - представляет собой динамическую структуру данных. Особенности дерева вывода таковы: дерево вывода - симметричное, бинарное (каждая вершина дерева имеет одинаковое число исходящих дуг, равное двум), несбалансированное (в нем могут быть не заполненные уровни). Для построения дерева используется механизм полиза.

Заключение

Исходя из проведенного анализа типичных оболочек, в качестве итога можно построить таблицу сравнения их характеристик (табл. 1). «Плюсами» отмечены наличие наиболее сильных механизмов в системах.

Анализ существующих инструментальных средств позволил выявить следующие их достоинства и недостатки.

Недостатками рассмотренных систем, от которых следует уйти, являются:

- для всех систем, в основе, которых лежит прямой и обратный логический вывод, недостатком является большие затраты времени на их выполнение; это касается систем Clips, Prolog, Exys, Jess;

- сравнение по образцам и шаблонам, предполагает необходимость нахождения значения в базе знаний, но так как они не ориентированы на определенную предметную область, система только усложняется для понимания пользователя; это касается систем Jess, Clips;

- сложный для понимания механизм вывода, который предполагает 1) возвраты в определенные узлы, 2) сохранение позиций нахождения правила в базе знаний, 3) комбинирование нескольких алгоритмов прохождения по дереву решений; это касается систем Clips, Prolog, Nexpert Object, Exys Corvid, Jess, G2;

- недостатком Rete-алгоритма является высокая потребность в оперативной памяти в системе; это касается систем - Jess;

- несогласованность между организацией поиска в глубину и деревом поиска, которое включают повторяющиеся состояния и бесконечные пути в методе резолюции; это касается систем - Prolog;

- обратный механизм логического вывода с поиском в глубину сталкивается с проблемами, обусловленными излишними вычислениями; общее количество этапов логического вывода может определяться экспоненциальной зависимостью от количества базовых фактов, вырабатываемых в процессе вывода; это касается систем - Prolog;

- привязанность ЭС к определенной архитектуре рабочего терминала, это касается систем: G2, Nexpert Object.

Достоинства, которые желательно сохранить:

- Clips предлагает аппарат для работы с нечёткими множествами, что может быть полезно для математиков, ведущих работу в области прогнозирования и оценки качества;

- мета-эвристическая оболочка способна адаптироваться на предметную область САПР, так же обладает способностью работать с нечеткими множествами;

- G2 строит модели, функционирующие в реальном времени, и обеспечивает работу с динамическими системами;

- Prolog, используя метод резолюций, позволяет быстро искать решения в дереве вывода;

- Exys дает пользователю возможность задавать наиболее удобный и правильный коэффициент уверенности выполнения правила;

- Jess реализует Rete-алгоритм, который повышает быстродействие процесса сравнения с шаблонами;

- Nexpert Object имеет удобный графический интерфейс редактирования правил.

Таблица 1 - Сравнение параметров существующих систем

Системы / Параметры	Clips	Prolog	Nexpert Object	Jess	Exys Corvid	G2	ДеСиМ
База фактов	+			+			
База правил	+			+			
Правила в виде продукций	+	+	+	+	+	+	+
Прямой логический вывод	+		+	+	+	+	
Обратный логический вывод		+	+		+	+	
Модификация алгоритмов вывода		+		+		+	+
Ориентация на квалификацию пользователя			+		+	+	+
Графический интерфейс оболочки	+	+	+	+	+	+	+
Адаптация на предметную область					+	+	+

Как общий вывод по анализу, можно отметить, что:

1) Язык ДеСиМ (мета-эвристическая оболочка) является наиболее развитой системой, методы представления знаний в которой, а так же методы организации вывода и построения интерфейса «САПР – ЭС» в наибольшей степени соответствуют поставленной задаче;

2) Данная система обладает рядом недостатков с точки зрения возможностей других систем.

Главными идеями, которые рождаются после проделанного анализа, являются:

1) Необходимо ввести шкалу квалификации эксперта в ПрОб, выполняющего роль инженера по знаниям, именно в области инженерии знаний; на одном конце шкалы – возможности ДеСиМ, упрощенные до предела, но понятные проектировщику, а на другом – сложные, эффективные, но не совсем прозрачные методы представления знаний и организации вывода, применяющиеся в наиболее развитых системах; это позволит адаптировать ИО на растущую квалификацию проектировщика;

2) Исходя из этой шкалы квалификации, необходимо построить шкалу доступных средств представления знаний, организации вывода и – обеспечения интерфейса «САПР – ЭС»;

3) Такой путь позволит строить системы типа G2, но ориентированные на конкретные ПрОб в САПР (например, проектирование в энергетике).

Т.о., перспективной задачей является создание на основе ДеСиМ нового комплексного подхода, включая:

1) метод пополнения базы знаний, удобный и понятный для «умного» эксперта в ПрОб, играющего роль инженера познаниям, но имеющего разную квалификацию;

2) механизмы логического вывода, лишенные недостатков с точки зрения «умного» эксперта в ПрОб САПР той или иной квалификации.

Список литературы

1. Григорьев А.В. Построение двухсторонних трансляторов в задаче создания интеллектуальных надстроек над проблемно-ориентированными САПР / А.В. Григорьев, О.В. Морозова // Сборник трудов XI международной научной конференции им. Т.А. Таран. - Киев: Просвита, 2011. - С. 68-75.

2. Григорьев А.В. Анализ существующих способов создания интерфейса «языки формальных спецификаций — проблемно-ориентированные языки» / А.В. Григорьев, О.В. Морозова // Сборник научных трудов донецкого национального технического университета. – 2011. – № 14. – С. 270 – 275.

3. Тельнов Ю.Ф. Интеллектуальные информационные системы: учебное пособие / Ю.Ф. Тельнов. - М.: Московский международный институт эконометрики, информатики, финансов и права, 2003. – 118 с.

4. Тенденции развития систем искусственного интеллекта [Электронный ресурс]. – Режим доступа: http://www.mari-el.ru/mmlab/home/AI/12/index.html

5. Riley Gary CLIPS online documentation [Электронный ресурс] / Gary Riley. – Режим доступа: http://clipsrules.sourceforge.net/OnlineDocs.html

6. Частиков А. П. Разработка экспертных систем. Среда CLIPS / А.П. Частиков, Т.А. Гаврилова, Д.Л. Белов. – СПб.: БХВ-Петербург, 2003. – 608 с.

7. Управление знаниями [Электронный ресурс]. – Режим доступа: http://ryk-kypc2.narod.ru/lec.html

8. Martin Strauss Jess. The Java Expert System Shell / Martin Strauss // Seminar "AI Tools". – 2007. – 33с.

9. Братко И. Алгоритмы искусственного интеллекта на языке Prolog / И. Братко; пер. с англ. — [3-е изд.]. - М.: Издательский дом "Вильямс", 2004. — 640 с. : ил. — Парал. Тит. англ.

10. Knowledge Automation Expert System [Электронный ресурс]. – Режим доступа: http://exsyscorvid.com/

11. Рокотян И.С. Разработка баз знаний на основе экспертной системы EXSYS / И.С. Рокотян, Е.А. Хачатурова. – М.: Издательство МЭИ, 1998. – 28 с.

12. Nexpert Object Development System [Электронный ресурс]. – Режим доступа: http://cseweb.ucsd.edu/users/little/OldSites/CSE_Uptime/v2.8/nexpert.html

13. Григорьев А.В. Методы решения задачи структурного синтеза в интеллектуальных САПР, построенных на основе семиотической модели структур / А.В. Григорьев // Наукові праці Донецького національного технічного університету. Серія: "Обчислювальна техніка та автоматизація". - 2010. - Випуск 19. — С. 128-141.

Надійшла до редакції 05.09.2012

О.В. МОРОЗОВА, О.В. ГРИГОР'ЄВ
Донецький національний технічний університет

Аналіз методів побудови експертних систем у продукційних інструментальних оболонках

Робота присвячена аналізу типових інструментальних оболонок для створення експертних систем. Виявлені достоїнства і недоліки цих оболонок з точки зору застосовуваних методів побудови системи продукційних правил і механізму виведення. Зроблено висновки про переваги і недоліки застосування даних систем для створення інтелектуальних САПР з адаптацією на потрібну предметну область.

Ключові слова: Clips, Prolog, Nexpert Object, Exys Corvid, Jess, G2, продукційна модель, механізм логічного висновку

O.V. MOROZOVA, A.V. GRIGORIEV
Donetsk National Technical University

Analysis of Expert Systems Methods in Production Tool Shells

The paper analyzes the typical tool shells of creating expert systems. We considered the advantages and disadvantages of these shells in terms of the methods of building a system of production rules and inference mechanism. The conclusions about the advantages and disadvantages of these systems usage for developing intelligent CAD systems with adaptation to the desired subject domain are made.

Keywords: Clips, Prolog, Nexpert Object, Exys Corvid, Jess, G2, production model, inference engine