# СЕМАНТИЧЕСКАЯ МОДЕЛЬ ИНТЕЛЛЕКТУАЛЬНЫХ РЕШАТЕЛЕЙ ЗАДАЧ

*К.А. Уваров*

Белорусский государственный университет информатики и радиоэлектроники,
кафедра интеллектуальных информационных технологий
П.Бровки, 6, г. Минск, Республика Беларусь
+375-017-293-80-92; e-mail: Konstantin.Uvarov@gmail.com

**Рассматривается модель обработки информации, используемая в технологии проектирования машин обработки знаний. Технология основана на применении агентного подхода и универсального семантического способа кодирования знаний.**

**Ключевые слова – активная программа, машина обработки знаний, операция, семантическая сеть.**

## 1 ВВЕДЕНИЕ

Любая интеллектуальная система обладает знаниями и умениями. Знания интеллектуальной системы хранятся в базе знаний, а умения определяют то, насколько эффективно система может пользоваться знаниями при решении задач. Интеллектуальность системы определяется соотношением между знаниями и навыками системы: чем больше классов задач способна решать система на основе одних и тех же знаний, тем она выше. Интеллектуальный решатель задач является формальным уточнением того, какими навыками обладает интеллектуальная система для решения задач. Поэтому интеллектуальным решателем задач будем называть машину обработки знаний, т.к. решение задач требует обработки знаний, хранимых в системе.

## 2 ОПЕРАЦИИ ИНТЕЛЛЕКТУАЛЬНОГО РЕШАТЕЛЯ ЗАДАЧ

В основе теории машин обработки знаний лежит теория абстрактных графодинамических ассоциативных машин [1]. Для обработки знаний, представленных с использованием семантического кода, в работе [2] предложен специальный класс абстрактных машин – абстрактная sc-машина.

Абстрактная sc-машина определяется внутренним языком (язык на котором в памяти машины представляется обрабатываемая информация), множеством операций, обрабатывающих память, и исходным состоянием памяти. Абстрактная sc-машина представляет собой абстрактную графодинамическую параллельную информационную машину, у которой внутренним языком является графовый язык, основанный на семантическом коде, а в ее графодинамической памяти хранится полная информация, требуемые для операций абстрактной sc-машины. Все вспомогательные информационные конструкции, необходимые для функционирования операций sc-машины тоже представлены в ее памяти. Операции абстрактной sc-машины будем называть sc-операциями или операциями над sc-памятью.

Разные абстрактные sc-машины имеют разные системы(наборы) операций. Следовательно, понятие абстрактной sc-машины определяет целый класс (семейство) абстрактных машин, имеющих одинаковую организацию памяти, но разные системы операций.

Каждой sc-операции соответствует программа, написанная на языке, предназначенном для обработки баз знаний. Она определяет операционную семантику этой sc-операции, устанавливая то, какое преобразование эта операция осуществляет над sc-памятью. Программы операций абстрактной sc-машины являются демоническими и обладают следующими свойствами: автономность, активность, самоинициируемость. Они способны вызывать другие программы, являющиеся пассивными, т.е. инициируемыми только в результате явного обращения к ним из других программ либо, как в случае рекурсивной программы, из нее же самой. Таким образом, множество программ абстрактной sc-машины может быть разбито на множества демонических и пассивных программ. Преобразования, выполняемые пассивными программами абстрактной машины, в результате их вызова демоническими программами, будем условно называть базовыми преобразованиями этой машины.

Для записи программ операций sc-машины в большинстве случаев используется процедурный язык программирования SCP (Semantic Code Programming). Он относится к классу графовых языков программирования. Его особенностью является то, что не только данные, но и сами программы представляются в виде sc-конструкций. Данный язык является языком параллельного и асинхронного программирования. Это необходимо для того, чтобы обеспечить адекватную интерпретацию не только последовательных, но и параллельных формальных моделей [1].

Кроме языка SCP в интеллектуальных системах могут быть использованы и другие языки программирования, по отношению к которым язык SCP будет выполнять роль базового языка. С помощью языка SCP осуществляется компиляция или интерпретация программ других языков, так же он рассматривается как ассемблер специализированного компьютера ориентированного на обработку баз знаний.

Каждую sc-операцию можно представить в виде авто-

номного агента. Взаимодействие между операциями осуществляется через общую память средствами специализированного языка, предназначенного для обеспечения коммуникаций между агентами. Данный язык основан на использовании семантического кода и является открытым. С его помощью можно сформировать описание задачной ситуации и указать факт успешного или безуспешного выполнения sc-операции.

Обязательным требованием к реализованной sc-операции является наличие ее спецификации, что обуславливается необходимостью осуществлять поиск подходящей sc-операции. Спецификация sc-операции содержит набор ключевых узлов, описание условий применения и программу, обрабатывающую sc-конструкции, если условия применения удовлетворены. Для записи исходных текстов sc-операций может использоваться язык высокого уровня SCPH (SCP high-level), на котором программа записывается более компактно. Существует псевдо-естественный способ записи программ этого языка он называется SCPHn (SCP high-level natural). Он является легким для чтения и понимания неподготовленными разработчиками машин обработки знаний. Описание условия применения sc-операции записывается с помощью специализированного языка. Данный язык позволяет описать множество ситуаций в sc-памяти (набор sc-конструкций), появление которых приводит к запуску sc-операции.

## 3 КЛАССИФИКАЦИЯ ОПЕРАЦИЙ ИНТЕЛЛЕКТУАЛЬНОГО РЕШАТЕЛЯ ЗАДАЧ

Приведем классификацию sc-операций по ряду признаков.

На основе признака принадлежности ключевых узлов sc-операции предметной области, выделяют:
- теоретико-множественные sc-операции;
- логические sc-операции;
- реляционные sc-операции;
- арифметические sc-операции;
- навигационно-поисковые sc-операции.

Основное действие, выполняемое sc-операцией, определят три основных класса sc-операций:
- sc-операции поиска;
- sc-операции генерации;
- sc-операции удаления.

Локализация событий, инициирующих выполнение sc-операции, и результатов выполнения sc-операции определяют следующие классы:
- рецепторные sc-операции (sc-операции ввода) – событие, инициирующее выполнение операции, происходит вне системы. В результате выполнения sc-операции происходят изменения в памяти системы;
- акцепторные sc-операции (sc-операции вывода) – инициируются при изменении памяти, а результаты работ таких операций направлены вовне системы;
- внутренние sc-операции (sc-операции память-память) – изменение памяти приводит к запуску sc-операции, результаты выполнения которой тоже меняют состояние памяти.

В дальнейшем sc-операцией будем называть внутренние sc-операции, т.к. это наиболее широкий класс sc-операций, в свою очередь рецепторные и акцепторные sc-операции в основном используются для реализации пользовательского интерфейса.

Тип действия, инициирующего выполнение sc-операции, позволяет разделить sc-операции на следующие классы:
- sc-операции, инициируемые генерацией выходящей sc-дуги из указанного элемента;
- sc-операции, инициируемые генерацией входящей sc-дуги в указанный элемент;
- sc-операции, инициируемые удалением указанного sc-элемента [2, 3].

Момент срабатывания sc-операции позволяет разделить все sc-операции на два класса:
- "пред" sc-операции. Операции данного класса запускаются после инициирования, но до осуществления некоторого действия;
- "пост" sc-операции. Такие sc-операции запускаются после осуществления некоторого действия.

В результате работы "пред" sc-операции, принимается решение: осуществлять инициированное действие или нет. По своей сути такие sc-операции очень близки к триггерам, используемым в современных реляционных базах данных для поддержания информационной целостности; "пост" sc-операции, используются для внесения необходимых изменений в соответствии с изменившимся контекстом [2, 3].

Семантическая классификация sc-операций определяется смыслом решаемых операцией задач:
- sc-операции поиска ответов на вопросы. Благодаря им система умеет "вспоминать" информацию и отвечать на вопросы пользователя;
- дедуктивные sc-операции. Их применение позволяет системе при отсутствии явно представленного в базе знаний ответа сгенерировать его, используя логические правила и знания о закономерностях[5];
- sc-операции интерпретации программ. С помощью операций этого класса система интерпретирует как процедурные, так и непроцедурные программы (в том числе арифметические формулы, химические уравнения, нейросетевые модели, генетические алгоритмы и т.д.);
- sc-операции организации процесса решения задачи. К этому классу sc-операций относятся операции определяющие стратегию решения задачи. Например, операции сведение задачи к подзадачам; операции решения задачи путём пополнения контекста задачи; операции, реализующие их возможные комбинации. Операции этого класса вместе с дедуктивными sc-операциями образуют ядро интеллектуального решателя задач[6];
- sc-операции сборки мусора. Они предназначены для поддержания целостности и актуальности базы знаний, и направлены на обеспечение эффективности работы системы. Под информационным мусором понимаются синонимичные или временные sc-конструкции. Примерами sc-операций сборки мусора являются: sc-

операция, удаляющая кратные пары принадлежности, *кратные связки* отношений; sc-операция, склеивающая sc-узлы с одинаковым содержимым и т.д. Информационным мусором также может являться полученный и сообщенный пользователю ответ, в таком случае sc-операции этого класса принимают решение: хранить полученный ответ или эффективнее для системы не сохранять его, считая мусором, и повторно вычислять при необходимости.

## 4 ПРИМЕРЫ ОПЕРАЦИЙ

Для иллюстрации возможного способа организации процесса решения задачи с использованием sc-операций рассмотрим следующий пример. Пользователем формируется предикатный вопрос к системе "Является ли указанный объект t1 многоугольником?". Формулировка вопроса в виде scg-текста представлена на рисунке 1. Для записи формулировки вопроса используются:

1) **нечеткая sc-дуга, указывающая на то, что пользователь не знает, является ли t1 элементом множества многоугольников**

2) **временная дуга из ключевого узла "установить позитивность или негативность", определяющего класс вопросов.**


Рис.1. Формулировка вопроса

В базе знаний хранится информация о том, что t является треугольником, а треугольники являются подмножеством многоугольников. Указанное состояние базы знаний представлено на рисунке 2.

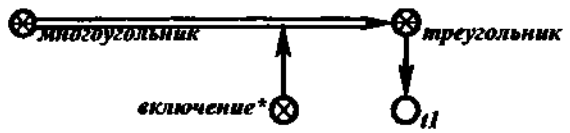


Рис.2. Исходное состояние базы знаний

После того как система получила от пользователя формулировку вопроса инициируется sc-операция, генерирующая на основе вопроса формулировку задачи на поиск/проверку. Данная формулировка приводится на рисунке 3.

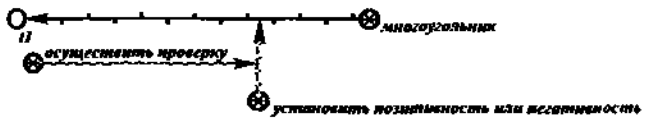


Рис.3. Формулировка задачи проверки принадлежности

В ней используется ключевой узел "осуществить проверку". Факт наличия временной позитивной sc-дуги выходящей из него, инициирует sc-операцию, которая проверяет наличие пары принадлежности или не принадлежности кратной нечеткой sc-дуге. Условием инициирования данной операции является факт проведения временной пары принадлежности из ключевого узла "осуществить проверку". Т.к. в базе знаний нет пары принадлежности и пары непринадлежности между sc-узлами "многоугольник" и t1, то вышеназванная sc-операция выполнится безуспешно. Для указания этого факта используется ключевой sc-узел с идентификатором "безуспешный", что показано на рисунке 4. По принадлежности ключевых узлов sc-операции к предметной области эта операция относится к операциям обработки вопросов и является внутренней операцией поиска.


Рис.4. Результаты работы sc-операции проверяющей принадлежность

Таким образом, для получения ответа на поставленный вопрос недостаточно применения sc-операций проверки, но исходя из факта включения треугольников во множество многоугольников и того, что t1 является треугольником, мы можем утверждать, что t1 также входит во множество многоугольников. Для получения системой аналогичного вывода используется sc-операция, которая инициируется после безуспешного выполнения операции проверки, и в результате ее работы генерирует ответ, добавляя t1 во множество многоугольников. Эта sc-операция относится к классу дедуктивных операций.

Последовательность выполнения sc-операций определяется их приоритетами, которые задает разработчик машин обработки знаний. В нашем случае, операция осуществляющая проверку имеет больший приоритет, чем sc-операция поиска на основе включений.

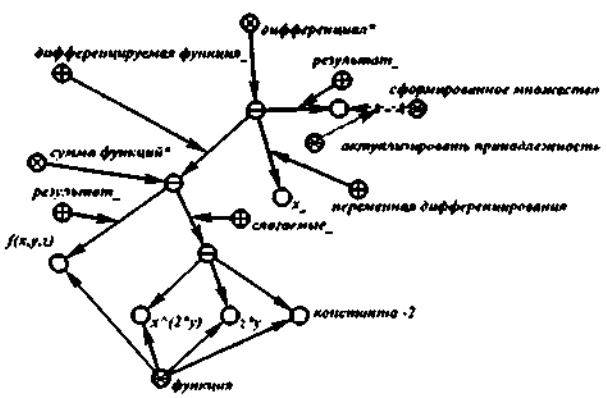


Рис.5. Формулировка задачи на вычисление дифференциала суммы функций

Когда ответ на поставленный вопрос получен, инициируется sc-операция сборки мусора, которая удаляет временные конструкции и нечеткую sc-дугу.

Рассмотрим sc-операции интерпретации текстов языков программирования. Операции этого класса имеют широкое применение в различных предметных областях – в математических и естественнонаучных дисциплинах. Ниже будет приведен пример работы sc-операции интерпретации правил дифференцирования функций. Эта опе-

рация применяется при дифференцировании любых функций от любого числа аргументов.

Рассмотрим пример использования данной операции. Задача состоит в вычислении производной по переменной x для функции $f(x,y,z)=x^{2y}+zy-2$. На рисунке 5 представлен фрагмент задачной ситуации, содержащий условие запуска для вышеназванной sc-операции и связку функционала "сумма функций*" соединяющую три функции: $x^{2y}$, $zy$ и константную функцию, единственным значением которой является -2, в функцию, являющуюся их суммой.

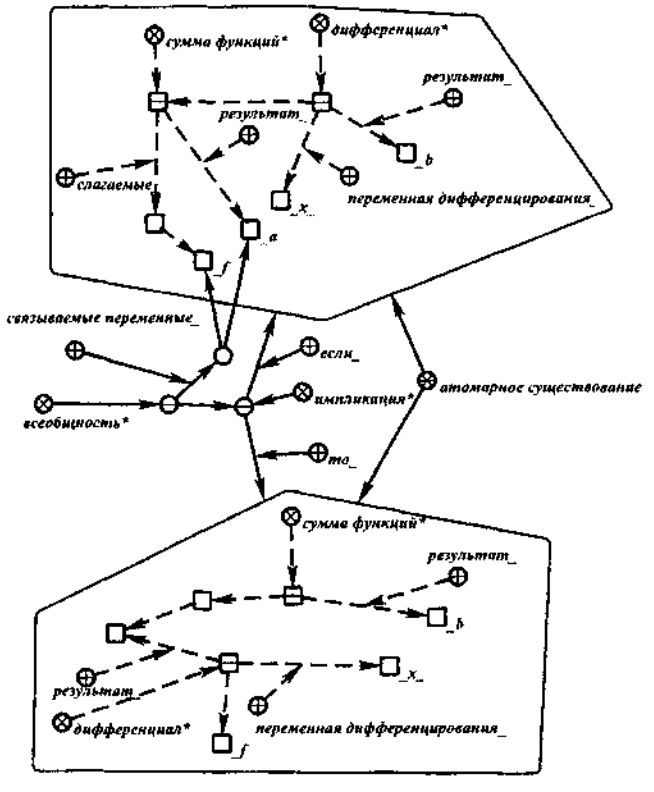


Рис.6. Правило вычисления дифференциала суммы функцией

Инициированная sc-операция определяет вид дифференцируемой функции. На основе этой информации выбирается соответствующее правило дифференцирования. В приведенном примере дифференцируемая функция является суммой функций. Для вычисления ее производной будет использоваться правило, представленное на рисунке 6.

В математической записи это правило имеет вид: $(u+v+w+...)'=u'+v'+w'+...$ Оно записывается с использованием языка SCL (Semantic code logic), предназначенного для представления логических фактов и закономерностей на основе логики предикатов первого порядка. Данное правило выступает в роли интерпретируемой программы для приведенной sc-операции. В результате ее выполнения генерируются результирующая функция и запросы на вычисление дифференциалов для ее подфункций. В нашем случае в качестве результата получим: сумму производных по x, функций $x^{2y}$, $zy$ и константы -2, т.е. сгенерируется три формулировки задачи на вычисление дифференциала. Таким образом, исходная функция выступает в роли данных для описанного правила, а инициирование sc-операции происходит рекурсивно до тех пор, пока не будут найдены константные функции, производные которых равны нулю. Кроме описанной sc-операции, к операциям данного класса относятся операции интерпретации логических закономерностей, операции формирования реляционных структур, операции описания способов построения геометрических чертежей и другие.

## ЗАКЛЮЧЕНИЕ

Описанная модель является ключевым компонентом семантической технолонии проектирования интеллектуальных решателей задач. Особенностями и достоинствами предложенной модели являются:

- унифицированное представление обрабатываемых знаний и умений, обеспечивающих решение задач;
- обеспечение совместимости различных способов решения задач;
- направленность на обеспечение независимости sc-операций друг от друга, приводящая к максимизации повторного использования операций и к уменьшению сроков разработки интеллектуальных систем.

## ЛИТЕРАТУРА


[1] Программирование в ассоциативных машинах / В.В. Голенков [и др.]; под ред. В.В. Голенкова. – Минск : БГУИР, 2001.

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

[3] Ивашенко, В.П., Применение мультиагентного подхода для реализации виртуальной графодинамической среды. / В.П. Ивашенко, А.Л. Кондратенко // Четвёртая международная летняя школа-семинар по искусственному интеллекту для студентов и аспирантов: сб. науч. тр. / БГУИР. - Минск, 2000. - С.154-156.

[4] Достоверный и правдоподобный вывод в интеллектуальных системах / Вагин В.Н. [и др.]; – М. : ФИЗМАТЛИТ, 2008.

[5] Нильсон Н., Искусственный интеллект: методы поиска решений / Н. Нильсон. – М. : Мир, 1973.

[6] Поспелов Д. А. Моделирование рассуждений. Опыт анализа мыслительных актов / Д. А. Поспелов. – М. : Радио и связь, 1989.