К лаб.13

!!!! В лабораторных работах будем использовать только конъюнкцию (термы в теле разделяются запятыми). И простой вопрос. Не усложняйте программы.

Утверждения программы — это предикаты. Фактически, структура предиката – это структура знания, отраженного в заголовке правил, входящих в процедуру (правил может быть несколько в одной процедуре). Для описания логики и структуры знания используются домены. Домен на формальном уровне обозначает природу (смысл) объекта, участвующего в формулировании знания. Порядок перечисления объектов, участвующих в формулировании знания, с логической точки зрения не важен, но будучи один раз установлен, он становится фиксированным (зафиксирована природа соответствующего аргумента). Структура предиката описывается в разделе: **PREDICATES.** При описании указывается домен, к которому относится конкретный аргумент (возможно, это не конкретизированная переменная) заголовка процедуры. Домен может быть использован: стандартный или описанный пользователем. Отнесение аргумента к конкретному домену не связано с типизацией и дальнейшим распределением памяти для объекта или для переменной (например X), а связано с тем, что в процессе унификации система должна «понять»: X – это обозначение (например) фамилии? или автомобиля?, несмотря на то, что и фамилия, и марка автомобиля – это строка. И этот анализ система выполняет с учетом домена аргумента.

Напомним, что выполнение программы заключается в попытке методом проб и ошибок ответить «Да» (логическое программирование) на поставленный вопрос. Это основная цель. И для этого необходимо подобрать (найти) соответствующее знание. А знание сформулировано в заголовке правила. Иначе – ответ «Нет».

В процессе выполнения программы — система, используя встроенный алгоритм унификации, пытается обосновать возможность истинности вопроса, строя подстановки и примеры термов (вопроса и формулировки

знания), используя базу знаний, и найти такие значения переменных, при которых это удается, а значит, на поставленный вопрос можно дать ответ «Да». Возможно, система «ошибается» в своих обоснованиях и возникает тупиковая ситуация, или, ответив на вопрос, пытается найти другой способ доказательства. Тогда включается механизм отката (отказа от последнего заключения (какого?) и последних действий, сделанных системой) и выполняется ре- конкретизация переменных, конкретизация которых была выполнена на последнем шаге.

Предложения программы на Prolog (знания) могут содержать переменные. На момент фиксации в программе, переменные обозначают некоторый (любой), неизвестный объект предметной области (вспомните кванторы). Значения переменных подбираются системой в процессе унификации (сопоставления) двух составных термов из программы на Prolog (для этого алгоритм унификации автоматически и многократно запускается системой). Однако, для обоснования истинности вопроса, не всегда бывает возможным подобрать нужное знание в принципе, или не всегда это можно выполнить однозначно. Кроме этого, не очевиден принцип и алгоритм (последовательность перебора) выбора знания для использования. Часто при выборе знания и человеку, и системе необходимо оценить содержательный смысл – семантику утверждения, что на формальном уровне однозначно выполнить сложно. Следует заметить, что недетерминизм поиска ответа – это базовый принцип работы естественного, искусственного интеллекта и системы Prolog. Таким образом, в сложных ситуациях – при высокой степени неопределенности, и человек и система Prolog совершают ошибки. Для того, чтобы такая ошибка выбора знания для системы Prolog была не критична, в систему встроен механизм отката (на момент последнего сделанного выбора) и отмена принятого решения (выполненной конкретизации).

Фактически, формально, работа алгоритма унификации заключается в попарном сопоставлении термов и попытке построить для них общий пример. Например, для поиска ответа на вопрос система должна найти

подходящее знание. А знание зафиксировано в заголовке правила. Т.е. система должна подобрать подходящее правило (подходящий заголовок) (это назначение алгоритма унификации). И система должна «понять» формально, что заголовок подходит. Для этого она строит унификатор – подстановку (побочный эффект работы алгоритма унификации). В результате применения этой подстановки некоторые переменные конкретизируются значениями, которые (значения) могут далее быть использованы при доказательстве истинности тела выбранного правила (так же с использованием алгоритма унификации). Т.е. значения переменных переходят на следующий шаг доказательства. Таким образом, с помощью алгоритма унификации происходит двунаправленная передача параметров процедурам. Например, из внешнего мира в программу для дальнейшего использования или из программы во внешний мир – значения интересующего нас параметра. Но основной результат работы программы – это «Да» или «Нет» на поставленный вопрос.