Убеждение справедливо, если выполняются все условия истинности.

Вопросы(term) могут быть сложные, состоящие из нескольких терм. Лучше использовать конъюкцию и дизъюнкцию. Использование переменных – формулировки. Переменные используются для передачи значений в пространстве (в разных кусках программы) и времени. Переменные в прологе используются особым способом. Значения-утверждения изначально не определены. Именованные переменные могут быть связаны со значениями, есть особенность этих переменных в том, что значение могут потеряться. Анонимные переменные не могут передать значения из одной области программы в другую. Анонимная переменная уникальна всегда.

Постановка переменной term

Термы, в которые не входят переменные – основные.

A(x1,x2,…,xn)

{xi = ti}

Tetta = {x1=t1,x2=t2,…,xn=tn}

B = A\*tetta

определение подстановкой называется множество пар вида {xi = ti}, где ti это терм, не содержащий переменных. Если А некоторый терм, то подстановка, которая к нему применена, то в результате получим пример исходного терма А, то есть частный случай(более общее утверждение)

Применить подстановку внутри терма заключается в замене каждого вхождения xi на ti. Получили составной терм с конкретными значениями.

Терм С является общим примеров А и В, если существуют такие подставки t1 и t2, что С = А\*tetta1, C = B\*tetta2.

Для того, чтобы система дала конкретный ответ, она должна подобрать значение переменной, поэтому должна строить такие примеры с термами.

Логические программы и процедуры

Знание находится в заголовке правила. В заголовке хранится терм, название главного пунктура. Система подбирает с помощью алгоритмов знания, которые помогут ответить «да» на заданный вопрос.

Логическая программа это бесконечное множество правил, в заголовке содержат арность или место (кол-во аргументов) и знания. Отношения определяемых процедур называются предикатами.

Если есть два пунткрура с разным кол-вом аргументов, то будет считаться, что они из разных программ.

Декларативная семантика программ

Синтаксис языка определяет грамматику утверждений. С точки зрения интерактивных языков семантика определяет реакцию системы на уровне аппаратуры, семантикой формул в логике предикатов является процесс связывания формулы базы знаний с логическим значением.

Пролог является и логическим языком и к нему применима императивная семантика (пошагово). То есть программа может быть определена и процедурной семантикой, и императивной.

С позиции декларативной семантики, особый вид (формулы первого порядка) формул может быть представлен в прологе. Оказывается важен порядок в базе знаний и фиксация условий истинности конкретных правил.

Система должна подбирать общие примеры. Унификация - операция, которая позволяет (что-то…, а можно помедленнее???), в результате происходит двунаправленная передача параметров процедурам, неразрушающая конкретизация переменных и проверка условий.

T1 T2

Конктретизированные термы унифицируются только в том случае, если они совпадают

Если Т1 не конкретизированная переменная или свободная(нес связанная со значением), а Т2 составной терм, не содержащий в качестве аргумента Т1, то унификация успешна

Если т1 и т2 не конкретизированные переменные, то их унификация всегда завершается успехом, а т1 и т2 становятся сцепленными, то есть двумя именами одного значения. Если одна из переменных потом получит значение, то и вторая его получит.

Если т1 и т2 составные термы, то они успешно унифицируются, если А) у т1 и т2 одинаковые глав пунктуры, Б) имеют одинаковые арности, В) успешно унифицируются каждая пара их соответствующей компоненты. Система многократно запускает алгоритм унификации.

Алгоритм унификации

Терма S называется более общим, чем Т, если Т является примером S, а S не является примером Т.

Терм S называется наиболее общим примером двух термов Т1 и Т2, если S такой их общий пример, который является более общим по отношению к любому другому их примеру.

Унификатором двух термов называется подстановка, которая, будучи применена к каждому терму, даст одинаковый результат.

Наиболее общим унификатором двух термов называется унификатор, соответствующий наиболее общему примеру термов.

При наличии переменных система должна построить пример утверждения.

Если два терма унифицируемы, то существует единственный с точностью до переименования переменных наиболее общий унификатор.

Система запускает алгоритм унификации многократно. Рабочая область, которая хранит равенство, результирующая ячейка памяти. Алгоритм представляет собой цикл, который обрабатывает термы. Цикл завершается, если возникает (…что-то…) или успех.

Начало

Занести Т1 = Т2

Положить неудача = 0

Пока стол не пуст

Считать S = T

Обработать:

А) S != T

B) S = T

V) если S не сод Т, то отыскать в стеке при результирующе ячейке все вхождения S и заменить их на Т, добавить в результирующую ячейку S = T

Д) если S и Т сост термы с разными пунктурами или арностью, то неудача = 1.

Е)если S и T составные термы с одинаковыми пунктурами и одной арностью, очистить раб поле в конец цикла, то есть переход на следующий шаг цикла.

Если произошел выход из цикла с неудача = 1, то унификация невозможна, иначе она успешна, а в результирующей ячейке находится подстановка, то есть наиболее общий унификатор.

Большие буквы – переменные, маленькие – константы.

t(X, p(X,Y)) = t(q(W),p(q(x),b))

