|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Лабораторная работа №12*

*По предмету: «Функциональное и логическое программирование»*

Преподаватели: Толпинская Н. Б.,

Строганов Ю. В.,

Студент: Мирзоян С.А.,

Группа: ИУ7-65Б

Москва, 2020 г.

Теоретическая часть

Prolog — это декларативный язык программирования, при использовании которого решение задачи получается путем логического вывода из ранее известных положений. Программа на Prolog не является определением последовательности действий, она представляет собой набор фактов и правил, обеспечивающих получение заключений на основе этих утверждений. Prolog базируется на предложениях Хорна, являющихся подмножеством формальной системы, которая называется логикой предикатов.

Программа на Prolog состоит из двух частей: базы знанийи вопроса. С помощью подбора ответов на запросы Prolog извлекает известную в программе информацию. База знаний содержит факты, на основе которой программа дает ответ на вопрос. Одной из особенностей Prolog является то, что при поиске ответов на вопрос, он рассматривает альтернативные варианты и находит все возможные решения, при которых на поставленный вопрос можно ответить – «yes».

База знаний состоит из предложений – **CLAUSES,** фактов и правил. Каждое предложение заканчивается точкой. Предложения бывают двух видов: факты и правила.

Правило имеет вид:

<заголовок> **:-** <тело правила>**.**

Вопрос состоит только из тела – составного терма (или нескольких составных термов). Вопросы используются для выяснения выполнимости некоторого отношения между описанными в программе объектами. Система рассматривает вопрос как цель, к которой (к истинности которой) надо стремиться. Ответ на вопрос может оказаться логически положительным или отрицательным, в зависимости от того, может ли быть достигнута соответствующая цель.

Поиск содержательного ответа на поставленный вопрос, с помощью имеющейся базы знаний, фактически заключается в поиске нужного знания, но какое знание понадобится – заранее неизвестно. Этот поиск осуществляется формально с помощью механизма унификации, встроенного в систему и не доступного программисту. Упрощенно, процесс унификации можно представить как формальный процесс сравнивания (сопоставления) терма вопроса с очередным термом знания. При этом, знания по умолчанию просматриваются сверху вниз, хотя такой порядок и не очевиден. В процессе сравнивания для переменных «подбираются», исходя из базы знаний, значения (для именованных переменных). И эти подобранные для переменных значения возвращаются в качестве побочного эффекта ответа на поставленный вопрос.

Цель работы программы – определить является ли вопрос логическим следствием программы или нет, что выполняется с применения правил вывода. Правила вывода – это утверждения о взаимосвязи между допущениями и заключениями, которые с позиции исчисления предикатов верны всегда.

Для выполнения логического вывода используется механизм (алгоритм) унификации, встроенный в систему.

Унификация – операция, которая позволяет формализовать процесс логического вывода (наряду с правилом резолюции). С практической точки зрения - это основной вычислительный шаг, с помощью которого происходит:

* Двунаправленная передача параметров процедурам,
* Неразрушающее присваивание,
* Проверка условий (доказательство).

В процессе работы система выполняет большое число унификаций. Процесс унификации запускается автоматически, но пользователь имеет право запустить его принудительно с помощью утверждения (немного нарушает форму записей): T1 = T2 . Унификация – попытка "увидеть одинаковость" – сопоставимость двух термов, может завершаться успехом или тупиковой ситуацией (неудачей). В последнем случае включается механизм отката к предыдущему шагу.

Практическая часть

Составить программу – базу знаний, с помощью которой можно определить, например, множество студентов, обучающихся в одном ВУЗе. Студент может одновременно обучаться в нескольких ВУЗах. Привести примеры возможных вариантов вопросов и варианты ответов (не менее 3-х). Описать порядок формирования вариантов ответа.

Исходную базу знаний сформировать с помощью только фактов.

**Листинг**

1. domains
2. name, university = symbol.
3. predicates
4. student(name, university).
5. oneUnivers(name, university).
6. notUnivers(name, university).
7. clauses
8. student("Ivanov", "MSU").
9. student("Petrov", "BMSTU").
10. student("Smirnov", "MIPT").
11. student("Kuznetsov", "RUDN").
12. student("Mirzoyan","BMSTU").
13. student("Goncharova", "MGOY").
14. student("Mirzoyan", "MSU").
16. oneUnivers(Name, University) :- student(X, University), student(Name, University), X <> Name.
17. notUnivers(Name, Y) :- student(Name, X), X <> Y.
18. goal
19. %oneUnivers(Name, University).
20. notUnivers(Name, "MSU").
21. %student("Mirzoyan", University).

Варианты вопросов:

1. Список студентов, у которых совпадает университет
2. Студенты, которые в любом ВУЗе, помимо указанного.
3. Список университетов, в которых учится указанный студент

Если цель содержит переменные, Prolog выдает те значения переменных, которые приводят к решению (в вопросы переменные входят только с квантором существования, а в факты и правила входят только с квантором всеобщности). Таким образом student("Mirzoyan", University) — дословно: «В каких ВУЗах учится Мирзоян?».

Пролог всегда ищет ответ на запрос, начиная с верха фактов. Он смотрит на каждый терм знания, пока не достигнет низа, где его больше нет. В данной программе, помимо фактов, присутствует правило, которое можно трактовать как: «Студенты, которые учатся в одном ВУЗе».