|  |  |
| --- | --- |
| 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 **ПРОГРАМНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.03 ПРОГРАММНАЯ ИНЖЕНЕРИЯ**

**Отчет**

|  |  |
| --- | --- |
| **По лабораторной работе №** | 1 18 |

**Дисциплина:** Функциональное и логическое программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-62Б |  |  | Н.А. Гарасев |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Н.Б.Толпинская |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2021

**Цель работы** –изучить рекурсивные способы организации программ на Prolog, методы формирования эффективных рекурсивных программ и порядок реализации таких программ.

**Задачи работы**: приобрести навыки использования рекурсии на Prolog, эффективного способа ее организации и прядка работы соответствующей программы. Изучить возможность и необходимость использования системных предикатов в рекурсивной программе на Prolog, принципы и особенности порядка работы такой программы. Способ формирования и изменения резольвенты в этом случае и порядок формирования ответа.

**Задание**

**Ответить на вопросы:**

1. Что такое рекурсия? Как организуется хвостовая рекурсия в Prolog? Как организовать выход из рекурсии в Prolog?
2. Какое первое состояние резольвенты?
3. В каком случае система запускает алгоритм унификации? Каково назначение использования алгоритма унификации? Каков результат работы алгоритма унификации?
4. В каких пределах программы переменные уникальны?
5. Как применяется подстановка, полученная с помощью алгоритма унификации?
6. Как изменяется резольвента?
7. В каких случаях запускается механизм отката?

**Используя хвостовую рекурсию, разработать программу, позволяющую найти**

1. **n!**,
2. **n-е** число Фибоначчи.

Убедиться в правильности результатов**.**

**Ответы на вопросы**

* **Что такое рекурсия? Как организуется хвостовая рекурсия в Prolog? Как организовать выход из рекурсии в Prolog?**

Рекурсия — это способ организации повторных вычислений. Рекурсия — это ссылка на самого себя. В логическом программировании это способ заставить систему использовать многократно одну и ту же процедуру. Хвостовая же рекурсия достигается путем использования дополнительных переменных и предполагает произведения всех вычислений до вызова знания самим собой.

В Prolog рекурсия организуется с помощью нескольких правил, часть из которых не являются рекурсивными и служат для выхода из рекурсии, они используют отсечения для выхода из рекурсии. Хвостовая рекурсия в Prolog организуется за счет расположения повторного вызова функции последней подцелью в конъюнктивном правиле.

* **Какое первое состояние резольвенты?**

Первое состояние резольвенты - заданный вопрос.

* **В каком случае система запускает алгоритм унификации? Каково назначение использования алгоритма унификации? Каков результат работы алгоритма унификации?**

Процесс унификации запускается в случае, когда необходимо доказать какое-то утверждение. Если резольвента не пуста, есть, что доказывать.Также пользователь может запустить этот процесс вручную с помощью утверждения Т1=Т2.

Унификация позволяет формализовать процесс логического вывода. Назначение - поиск знания, которое является ответом на конкретный вопрос.

Результат работы алгоритма унификации — ответ «да» или «нет», а также конкретизация переменных.

* **В каких пределах программы переменные уникальны?**

Именованные переменные уникальны в рамках одного предложения, т. е. в разных предложениях одно и то же имя переменной может использоваться для обозначения разных объектов. Анонимные переменные уникальны везде.

* **Как применяется подстановка, полученная с помощью алгоритма унификации?**

Если алгоритм унификации завершился успешно и найдена подстановка, соответствующие переменные конкретизируются полученными значениями.

* **Как изменяется резольвента?**

В процессе доказательства утверждений в резольвента меняется. При изменении строится новая резольвента. По стековому принципу берется верхняя подцель и заменяется на тело подходящего правила. Затем применяется найденная на текущем этапе подстановка. Успешное завершение работы программы достигается, когда резольвента пуста.

* **В каких случаях запускается механизм отката?**

Механизм отката к предыдущему шагу выполняется в случае, когда унификация завершается тупиковой ситуацией(неудачей). Кроме того, механизм используется для того, чтобы получить все возможные ответы.

**Текст программы**

predicates

factorial(integer, integer)

f(integer, integer, integer, integer)

fib(integer, integer)

fib1(integer, integer, integer, integer)

clauses

factorial(X, Res):- f(X, Res, 1, 1).

f(X, Res, X, Res):- !.

f(X, Res, X1, Res1):- Tmp = X1+1, Res\_tmp = Res1 \* Tmp, f(X, Res, Tmp, Res\_tmp).

fib(X, R):- fib1(X, R, 1, 0). %obertka

fib1(1, R, R, \_):- !. % x=1 -> exit

fib1(X, R, X1, X2):- R1 = X1 + X2, P = X - 1, fib1(P, R, R1, X1).

goal

%factorial(3, Res).

fib(6, R).

**Вопрос: factorial(3, Res)**

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Состояние резольвенты, и вывод: дальнейшие действия (почему?) | Для каких термов запускается алгоритм унификации: Т1=Т2 и каков результат (и подстановка) | Дальнейшие действия: прямой ход или откат (почему и к чему приводит?) |
| 11 | factorial(3, Res)  В качестве первого состояния в резольвенту помещается вопрос | T1 = factorial(3, Res)  T2 = factorial(X, Res)  Попытка унификации. Унификация успешна.  Подстановка:  {Х=3, Res=Res} | Прямой ход |
| 2 | f(3, Res, 1, 1) | T1 = f(3, Res, 1, 1)  T2 = factorial(X, Res)  Попытка унификации. Унификация не успешна. Разные функторы | Переход к следующему предложению |
|  | f(3, Res, 1, 1) | T1 = f(3, Res, 1, 1)  T2 = f(X, Res, X, Res)  Попытка унификации. Унификация не успешна. 3!= 1 | Переход к следующему предложению |
|  | f(3, Res, 1, 1) | T1 = f(3, Res, 1, 1)  T2 =f(X, Res, X1, Res1)  Попытка унификации. Унификация успешна.  Подстановка:  {Х=3, Res=Res, X1 = 1, Res1 = 1} | Прямой ход |
| 3 | Tmp = 1+1  Res\_tmp = 1 \* Tmp  f(3, Res, Tmp, Res\_tmp) | Tmp = 2 | Прямой ход |
| 4 | Res\_tmp = 1 \* 2  f(3, Res, 2, Res\_tmp) | Res\_tmp = 2 | Прямой ход |
| 5 | f(3, Res,2, 2) | T1 = f(3, Res,2, 2)  T2 = factorial(X, Res)  Попытка унификации. Унификация не успешна. Разные функторы | Переход к следующему предложению |
|  | f(3, Res,2, 2) | T1 = f(3, Res,2, 2)  T2 = f(X, Res, X, Res)  Попытка унификации. Унификация не успешна. 3!= 2 | Переход к следующему предложению |
|  | f(3, Res, 2, 2) | T1 = f(3, Res, 2, 2)  T2 =f(X, Res, X1, Res1)  Попытка унификации. Унификация успешна.  Подстановка:  {Х=3, Res=Res, X1 = 2, Res1 = 2} | Прямой ход |
| 6 | Tmp = 2+1  Res\_tmp = 2 \* Tmp  f(3, Res, Tmp, Res\_tmp) | Tmp = 3 | Прямой ход |
| 7 | Res\_tmp = 2 \* 3  f(3, Res, 3, 6) | Res\_tmp = 6 | Прямой ход |
| 8 | f(3, Res, 3, 6) | T1 = f(3, Res,3, 6)  T2 = factorial(X, Res)  Попытка унификации. Унификация не успешна. Разные функторы | Переход к следующему предложению |
| 9 | f(3, Res,3, 6) | T1 = f(3, Res,3, 6)  T2 = f(X, Res, X, Res)  Попытка унификации. Унификация успешна. Подстановка:  {X=3, Res=Res, X=3, Res=6} | Прямой ход |
| 10 | ! | ! - указывает прологу отменить поиск альтернатив для целей до него | Альтернатив не искать. Завершение работы. Вывод результата. |

**Вопрос:** fib(3, R)

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Состояние резольвенты, и вывод: дальнейшие действия (почему?) | Для каких термов запускается алгоритм унификации: Т1=Т2 и каков результат (и подстановка) | Дальнейшие действия: прямой ход или откат (почему и к чему приводит?) |
| 11 | fib(3, R)  В качестве первого состояния в резольвенту помещается вопрос | T1 = fib(3, R)  T2 = fib(X, R)  Попытка унификации. Унификация успешна.  Подстановка:  {Х=3, R=R} | Прямой ход |
| 22 | fib1(3, R, 1, 0). | T1 = fib1(3, R, 1, 0).  T2 = fib(X, R)  Попытка унификации. Унификация не успешна. Разные функторы | Переход к следующему предложению |
| 3 | fib1(3, R, 1, 0). | T1 = fib1(3, R, 1, 0).  T2 =fib1(1, R, R, \_)  Попытка унификации. Унификация не успешна. 3!=1 | Переход к следующему предложению |
| 4 | fib1(3, R, 1, 0). | T1 = fib1(3, R, 1, 0).  T2 =fib1(X, R, X1, X2)  Попытка унификации. Унификация успешна.  Подстановка  {X=3, R=R, X1=1, X2=0} | Прямой ход |
| 53 | R1 = 1 + 0  P = 3 — 1  fib1(P, R, R1, 1) | R1=1 | Прямой ход |
| 64 | P = 3 — 1  fib1(P, R, 1, 1) | P=2 | Прямой ход |
| 75 | fib1(2, R, 1, 1) | T1 = fib1(2, R, 1, 1).  T2 = fib(X, R)  Попытка унификации. Унификация не успешна. Разные функторы | Переход к следующему предложению |
| 8 |  | T1 = fib1(2, R, 1, 1).  T2 =fib1(1, R, R, \_)  Попытка унификации. Унификация не успешна. 2!=1 | Переход к следующему предложению |
| 9 |  | T1 = fib1(2, R, 1, 1).  T2 =fib1(X, R, X1, X2)  Попытка унификации. Унификация успешна.  Подстановка  {X=2, R=R, X1=1, X2=1} | Прямой ход |
| 16 | R1 = 1 + 1  P = 2 — 1  fib1(P, R, R1, 1) | R1= 2 | Прямой ход |
| 17 | P = 2 — 1  fib1(P, R, 2, 1) | P= 1 | Прямой ход |
| 1 | fib1(1, R, 2, 1) | T1 = fib1(1, R, 2, 1).  T2 = fib(X, R)  Попытка унификации. Унификация не успешна. Разные функторы | Переход к следующему предложению |
| 1 | fib1(1, R, 2, 1) | T1 = fib1(1, R, 2, 1).  T2 =fib1(1, R, R, \_)  Попытка унификации. Унификация успешна.  Подстановка: {  1=1, R=R, R=2, \_=1} | Прямой ход |
| 18 | ! | ! - указывает прологу отменить поиск альтернатив для целей до него | Альтернатив не искать. Завершение работы. Вывод результата. |

**Выводы:**

Эффективность системы достигнута за счет использования хвостовой рекурсии и отсечений. Необходимо обеспечить, чтобы после выхода из рекурсии программа не пыталась искать альтернативные ответы, для этого используется отсечение. Это позволяет сократить количество шагов, необходимых для получения результата.