|  |  |
| --- | --- |
| 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 №18**

**по курсу “Функциональное и логическое программирование”**

1. **по теме** “ **Рекурсия на Prolog**”

|  |  |
| --- | --- |
| Студент: | Уласик Е.А. |
| Группа: | ИУ7-61 |
| Преподаватель: | Толпинская Н.Б. |

*2020 г.*

**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**.**

**Для одного** из вариантов **ВОПРОСА** и каждого **задания составить таблицу**, отражающую конкретный порядок работы системы:

Т.к. резольвента хранится в виде стека, то состояние резольвенты требуется отображать в столбик: вершина – сверху! Новый шаг надо начинать с нового состояния резольвенты!

**Вопрос:…..**

|  |  |  |  |
| --- | --- | --- | --- |
| № 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?) |
| 1… | … | … | Комментарий, вывод… |
| … | … | … | … |

Вопросы

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

Рекурсия – это ссылка на определяемый объект во время его определения. Для организации хвостовой рекурсии в Prolog рекурсивный вызов определяемого предиката должен быть последней подцелью в теле рекурсивного правила. Выход из рекурсии в Prolog организован с помощью использования отсечения.

1. **Какое первое состояние резольвенты?**

Первое состояние резольвенты – это вопрос.

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

Алгоритм унификации запускается автоматически, если есть что доказывать. Система распознаёт это, если резольвента не пуста и есть неотмеченные знания в базе знаний. Назначение алгоритма унификации в сопоставлении двух термов. Это сопоставление может завершиться получением одного из двух результатов – успехом или тупиковой ситуацией, то есть неудачей.

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

Именованные переменные уникальны в рамках одного предложения, анонимные уникальны все.

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

Подстановка применяется к подцели резольвенты, конкретизируя значения переменных.

1. **Как изменяется резольвента?**

Резольвента изменяется с использованием редукции, то есть замены подцели телом правила, с которым унифицируется данная подцель, и механизма отката. В ходе отката резольвента возвращается в предыдущее состояние.

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

Механизм отката запускается в случае, если в ходе поиска доказательства подцели программа зашла в тупиковую ситуацию, и, если резольвента не пуста и решение найдено, но в базе знаний остались не отмеченные предложения.

Листинг программы

domains

num = integer.

predicates

**factorial**(num, num).

**factorial\_inner**(num, num, num).

**fibonacci**(num, num).

**fibonacci\_inner**(num, num, num, num).

clauses

**factorial**(NUM, RES) :- **factorial\_inner**(NUM, RES, **1**).

**factorial\_inner**(NUM, RES, M) :-

NUM = **0**, RES = M, !;

N1 = NUM - **1**,

M1 = M \* NUM,

**factorial\_inner**(N1, RES, M1).

**fibonacci**(NUM, RES) :- **fibonacci\_inner**(NUM, RES, **0**, **1**).

**fibonacci\_inner**(NUM, RES, S\_last, S\_next) :-

NUM = **0**, RES = S\_last, !;

N1 = NUM - **1**,

S\_l = S\_next,

S\_n = S\_next + S\_last,

**fibonacci\_inner**(N1, RES, S\_l, S\_n).

goal

% factorial(4, RES).

**fibonacci**(**4**, RES).

Обозначения

* factorial – правило-обёртка, которое получает 2 числа:
  + число, факториал которого нужно найти;
  + число, в которое будет записан результат.
* factorial\_inner – правило, в котором осуществляется хвостовая рекурсия и которое получает 3 аргумента:
  + число, факториал которого нужно найти;
  + число, в которое будет записан результат.
  + число, промежуточное произведение–результат;
* fibonacci – правило-обёртка, которое получает 2 числа:
  + число Фибоначчи, которое нужно найти;
  + число, в которое будет записан результат.
* fibonacci\_inner – правило, в котором осуществляется хвостовая рекурсия и которое получает 4 аргумента:
  + число Фибоначчи, которое нужно найти;
  + число, в которое будет записан результат;
  + число, текущие число Фибоначчи.
  + число, следующее число Фибоначчи после текущего;

Результат работы

1. Вопрос: factorial(0, RES)

Ответ: RES = 1

1. Вопрос: factorial(1, RES)

Ответ: RES = 1

1. Вопрос: factorial(5, RES)

Ответ: RES = 120

1. Вопрос: fibonacci(0, RES)

Ответ: RES = 0

1. Вопрос: fibonacci(1, RES)

Ответ: RES = 1

1. Вопрос: fibonacci(2, RES)

Ответ: RES = 1

1. Вопрос: fibonacci(6, RES)

Ответ: RES = 8

1. Вопрос: fibonacci(10, RES)

Ответ: RES = 55

Таблица

Вопрос: factorial(4, RES).

Сравнение в таблице осуществляется между подцелью из резольвенты и каждого знания из базы знаний.

|  |  |  |  |
| --- | --- | --- | --- |
| № 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?) |
| 1 | Резольвента:  factorial(4, RES).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение factorial(4, RES). и factorial(NUM, RES).  Результат: термы унифицируемые.  Подстановка:  {NUM = 4,  RES = RES}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 2 | Резольвента: factorial\_inner(4, RES, 1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение factorial\_inner(4, RES,1).и factorial\_inner(NUM, RES, M)  Результат: термы унифицируемые.  Подстановка:  {NUM = 4,  RES = RES,  M = 1}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 3 | Резольвента:  4 = 0,  RES = 1,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 4 и 0 на =(4, 0).  Результат: “нет” | Откат к предыдущему состоянию. |
| 4 | Резольвента:  N1 = 4 - 1,  M1 = 1 \* 4,  factorial\_inner(N1, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 3.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 3} | Прямой ход. Производится редукция. |
| 5 | Резольвента:  M1 = 1 \* 4,  factorial\_inner(3, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение M1 и 4.  Результат:  Термы унифицируемы.  Подстановка:  {M1 = 4} | Прямой ход. Производится редукция. |
| 6 | Резольвента: factorial\_inner(3, RES, 4).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение factorial\_inner(3, RES,4).и factorial\_inner(NUM, RES, M)  Результат: термы унифицируемые.  Подстановка:  {Num = 3,  RES = RES,  M = 4}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 7 | Резольвента:  3 = 0,  RES = 4,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 3 и 0 на =(3, 0).  Результат: “нет” | Откат к предыдущему состоянию. |
| 8 | Резольвента:  N1 = 3 - 1,  M1 = 4 \* 3,  factorial\_inner(N1, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 2.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 2} | Прямой ход. Производится редукция. |
| 9 | Резольвента:  M1 = 4 \* 3,  factorial\_inner(2, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение M1 и 12.  Результат:  Термы унифицируемы.  Подстановка:  {M1 = 12} | Прямой ход. Производится редукция. |
| 10 | Резольвента: factorial\_inner(2, RES, 12).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение factorial\_inner(2, RES,12). и factorial\_inner(NUM, RES, M)  Результат: термы унифицируемые.  Подстановка:  {Num = 2,  RES = RES,  M = 12}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 11 | Резольвента:  2 = 0,  RES = 12,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 2 и 0 на =(2, 0).  Результат: “нет” | Откат к предыдущему состоянию. |
| 12 | Резольвента:  N1 = 2 - 1,  M1 = 12 \* 2,  factorial\_inner(N1, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 1.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 1} | Прямой ход. Производится редукция. |
| 13 | Резольвента:  M1 = 12 \* 2,  factorial\_inner(1, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение M1 и 24.  Результат:  Термы унифицируемы.  Подстановка:  {M1 = 24} | Прямой ход. Производится редукция. |
| 14 | Резольвента: factorial\_inner(1, RES, 24).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение factorial\_inner(1, RES,24). и factorial\_inner(NUM, RES, M)  Результат: термы унифицируемые.  Подстановка:  {Num = 1,  RES = RES,  M = 24}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 15 | Резольвента:  1 = 0,  RES = 24,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 1 и 0 на =(1, 0).  Результат: “нет” | Нет успешных унификаций. Откат к предыдущему состоянию. |
| 16 | Резольвента:  N1 = 1 - 1,  M1 = 24 \* 1,  factorial\_inner(N1, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 0.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 0} | Прямой ход. Производится редукция. |
| 17 | Резольвента:  M1 = 24 \* 1,  factorial\_inner(0, RES, M1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение M1 и 24.  Результат:  Термы унифицируемы.  Подстановка:  {M1 = 24} | Прямой ход. Производится редукция. |
| 18 | Резольвента: factorial\_inner(0, RES, 24).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение factorial\_inner(0, RES,24). и factorial\_inner(NUM, RES, M)  Результат: термы унифицируемые.  Подстановка:  {Num = 0,  RES = RES,  M = 24}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 19 | Резольвента:  0 = 0,  RES = 24,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 0 и 0 на =(0, 0).  Результат: “да” | Прямой ход. Производится редукция. |
| 20 | Резольвента:  RES = 24,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение RES и 24.  Результат:  Термы унифицируемы.  Подстановка:  {RES = 24} | Прямой ход. Производится редукция. |
| 21 | Резольвента:  !;  Дальнейшие действия:  Отсечение |  | Вывод результата. Конец. |

Вопрос: fibonacci(4, 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?) |
| 1 | Резольвента: fibonacci(4, RES)  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение fibonacci(4, RES) и fibonacci(NUM, RES)  Результат: термы унифицируемые.  Подстановка:  {NUM = 4,  RES = RES}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 2 | Резольвента: fibonacci\_inner(4, RES, 0, 1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение fibonacci\_inner(4, RES, 0, 1). и fibonacci\_inner(NUM, RES, S\_last, S\_next).  Результат: термы унифицируемые.  Подстановка:  {NUM = 4,  RES = RES,  S\_last = 0,  S\_next = 1}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 3 | Резольвента:  4 = 0,  RES = 0,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 4 и 0 на =(4, 0).  Результат: “нет” | Откат к предыдущему состоянию. |
| 4 | Резольвента:  N1 = 4 - 1,  S\_l = 1,  S\_n = 1 + 0,  fibonacci\_inner(N1, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 3.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 3} | Прямой ход. Производится редукция. |
| 5 | Резольвента:  S\_l = 1,  S\_n = 1 + 0,  fibonacci\_inner(3, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_l и 1.  Результат:  Термы унифицируемы.  Подстановка:  {S\_l = 1} | Прямой ход. Производится редукция. |
| 6 | Резольвента:  S\_n = 1 + 0,  fibonacci\_inner(3, RES, 1, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_n и 1.  Результат:  Термы унифицируемы.  Подстановка: {S\_n = 1} | Прямой ход. Производится редукция. |
| 7 | Резольвента:  fibonacci\_inner(3, RES, 1, 1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение fibonacci\_inner(3, RES, 1, 1). и fibonacci\_inner(NUM, RES, S\_last, S\_next).  Результат: термы унифицируемые.  Подстановка:  {NUM = 3,  RES = RES,  S\_last = 1,  S\_next = 1}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 8 | Резольвента:  3 = 0,  RES = 1,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 3 и 0 на =(3, 0).  Результат: “нет” | Откат к предыдущему состоянию. |
| 9 | Резольвента:  N1 = 3 - 1,  S\_l = 1,  S\_n = 1 + 1,  fibonacci\_inner(N1, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 2.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 2} | Прямой ход. Производится редукция. |
| 10 | Резольвента:  S\_l = 1,  S\_n = 1 + 1,  fibonacci\_inner(2, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_l и 1.  Результат:  Термы унифицируемы.  Подстановка:  {S\_l = 1} | Прямой ход. Производится редукция. |
| 11 | Резольвента:  S\_n = 1 + 1,  fibonacci\_inner(2, RES, 1, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_n и 2.  Результат:  Термы унифицируемы.  Подстановка:  {S\_n = 2} | Прямой ход. Производится редукция. |
| 12 | Резольвента:  fibonacci\_inner(2, RES, 1, 1).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение fibonacci\_inner(2, RES, 1, 2). и fibonacci\_inner(NUM, RES, S\_last, S\_next).  Результат: термы унифицируемые.  Подстановка:  {NUM = 2, RES = RES, S\_last = 1, S\_next = 2}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 13 | Резольвента:  2 = 0,  RES = 1,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 2 и 0 на =(2, 0).  Результат: “нет” | Откат к предыдущему состоянию. |
| 14 | Резольвента:  N1 = 2 - 1,  S\_l = 2,  S\_n = 2 + 1,  fibonacci\_inner(N1, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 1.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 1} | Прямой ход. Производится редукция. |
| 15 | Резольвента:  S\_l = 2,  S\_n = 2 + 1,  fibonacci\_inner(1, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_l и 2.  Результат:  Термы унифицируемы.  Подстановка:  {S\_l = 2} | Прямой ход. Производится редукция. |
| 16 | Резольвента:  S\_n = 2 + 1,  fibonacci\_inner(1, RES, 2, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_n и 3.  Результат:  Термы унифицируемы.  Подстановка:  {S\_n = 3} | Прямой ход. Производится редукция. |
| 17 | Резольвента:  fibonacci\_inner(1, RES, 2, 3).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение fibonacci\_inner(1, RES, 2, 3). и fibonacci\_inner(NUM, RES, S\_last, S\_next).  Результат: термы унифицируемые.  Подстановка:  {NUM = 1,  RES = RES,  S\_last = 2,  S\_next = 3}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 18 | Резольвента:  1 = 0,  RES = 2,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 1 и 0 на =(1, 0).  Результат: “нет” | Откат к предыдущему состоянию. |
| 19 | Резольвента:  N1 = 1 - 1,  S\_l = 3,  S\_n = 3 + 2,  fibonacci\_inner(N1, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение N1 и 0.  Результат:  Термы унифицируемы.  Подстановка:  {N1 = 0} | Прямой ход. Производится редукция. |
| 20 | Резольвента:  S\_l = 3,  S\_n = 3 + 2,  fibonacci\_inner(0, RES, S\_l, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_l и 3.  Результат:  Термы унифицируемы.  Подстановка: {S\_l = 3} | Прямой ход. Производится редукция. |
| 21 | Резольвента:  S\_n = 3 + 2,  fibonacci\_inner(0, RES, 3, S\_n).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение S\_n и 5.  Результат:  Термы унифицируемы.  Подстановка:  {S\_n = 5} | Прямой ход. Производится редукция. |
| 22 | Резольвента:  fibonacci\_inner(0, RES, 3, 5).  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение fibonacci\_inner(0, RES, 3, 5). и fibonacci\_inner(NUM, RES, S\_last, S\_next).  Результат: термы унифицируемые.  Подстановка:  {NUM = 0,  RES = RES,  S\_last = 3,  S\_next = 5}. | Прямой ход. Термы унифицируемы. Производится редукция. К резольвенте применяется подстановка. |
| 23 | Резольвента:  0 = 0,  RES = 3,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение 0 и 0 на =(0, 0).  Результат: “да” | Прямой ход. Производится редукция. |
| 24 | Резольвента:  RES = 3,  !;  Дальнейшие действия:  унификация, так как резольвента не пуста. | Сравнение RES и 3.  Результат:  Термы унифицируемы.  Подстановка:  {RES = 3} | Прямой ход. Производится редукция. |
| 25 | Резольвента:  !;  Дальнейшие действия:  Отсечение |  | Вывод результата. Конец. |

Вывод

Изучены рекурсивные способы организации программ на Prolog, методы формирования эффективных рекурсивных программ и порядок реализации таких программ. Были реализованы две программы с хвостовой рекурсией. Отсечение в хвостовой рекурсии используется для предотвращения отката после нахождения результата. Таким образом, удаётся сократить количество сравнений.