Министерство образования Республики Беларусь

Учреждение образования

“Белорусский государственный университет

информатики и радиоэлектроники”

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

**ЛАБОРАТОРНАЯ РАБОТА №2**

по дисциплине «Логические основы интеллектуальных систем»

на тему

«Решение логических задач на языке Prolog»

**Вариант 4**

Выполнил студент гр. 121701 Олешкевич А.С.

Проверил Ивашенко В. П.

Минск 2023

**Цель:** В соответствии с вариантом необходимо реализовать программу на языке Prolog, решающую поставленную задачу.

**Вариант: 4:** Два береги реки. На одном из них полицейский с заключенным, мама с дочерьми и отец с сыновьями. Необходимо с помощью плота, вмещающего не более двух человек, переправить всех персонажей на другой берег реки. Управлять плотом могут только полицейский и родители. Заключённого нельзя оставлять ни с одним из членов семьи. Папе не разрешается находиться с дочерьми без присутствия матери. Маме не разрешается находиться с сыновьями без присутствия отца.

**Описание лабораторной работы:**

Решение поставленной задачи сводится к следующим подзадачам:

1. Анализ начального и целевого состояния миссионеров и людоедов.
2. Анализ допустимых состояний.
3. Анализ возможных переходов между состояниями.
4. Анализ правил обхода состояний.

Реализация алгоритма осуществляется на языке логического программирования Prolog.

**Дополнительные теоретические сведения:**

Prolog – средство написания выполнимых на ЭВМ программ. Язык логического

Грамматика языка PROLOG.

<ПРОЛОГ-предложение> ::= <правило> | <факт> | <запрос>

<правило> ::= <заголовок> ‘:-’<тело>

<факт> ::= <заголовок> ‘.’

<запрос> ::= <тело>‘.’

<тело> ::= <цель> /’,’<цель>/’.’

<заголовок>::= <предикат>

<цель>::= <предикат> |<выражение>

<предикат>::= <имя>/ ‘(‘<терм> /’,’<терм>/ ‘)’/

<терм>::= <атом> |<предикат>|<список>

<атом>::= <переменная> |<число> |<строка>|<имя>

<список>::= <список с заголовком>| <простой список>

<список с заголовком >::= ‘[‘ <терм >/’,’<терм>/’|’ < терм>’]’

< простой список>::= ‘[‘ <терм >/’,’<терм>/’]’|‘['’]’

<выражение>::= <терм> /<оператор><терм>/

<оператор>::= ‘is’ | '=' | ‘==' | ’\=' | ’>=' | ’=<’ | ‘=\=' |

# 

# **Описание программы и алгоритм**

Начальное состояние миссионеров и людоедов:

[1, 1, 1, 1, 2, 2, lft, 0, 0, 0, 0, 0, 0] - все персонажи находятся на левом берегу.

Целевое состояние программы, необходимое для завершения работы:

[0, 0, 0, 0, 0, 0, rgt,1, 1, 1, 1, 2, 2] - все персонажи находятся на правом берегу.

Правила перехода между состояниями должны учитывать, что в плоте помещается только два человека, управлять плотом могут только полицейский и родители. Заключённого нельзя оставлять ни с одним из членов семьи. Папе не разрешается находиться с дочерьми без присутствия матери. Маме не разрешается находиться с сыновьями без присутствия отца. Правило (1) описано ниже:

legal(CL, PL, DL, ML, SL, DtL, CR, PR, DR, MR, SR, DtR) :-

CL >=0, PL >=0, DL >=0, ML >=0, SL >=0, DtL >=0, CR >=0, PR >=0, DR >=0, MR >=0, SR >=0, DtR >=0,

CL + CR < 2,

PL + PR < 2,

DL + DR < 2,

ML + MR < 2,

check\_all\_parents(DL, ML, SL, DtL, DR, MR, SR, DtR),

check\_prisoner(CL, PL, DL, ML, SL, DtL, CR, PR, DR, MR, SR, DtR).

Так же есть правила возможных перемещений, в которых описываются все варианты перемещения персонажей на плоту:

1. один полицейский слева направо и в обратную сторону
2. один отец слева направо и в обратную сторону
3. одна мать слева направо и в обратную сторону
4. коп и дочь слева направо и в обратную сторону
5. коп и сын слева направо и в обратную сторону
6. коп и заключенный слева направо и в обратную сторону
7. коп и мать слева направо и в обратную сторону
8. коп и отец слева направо и в обратную сторону
9. отец и сын слева направо и в обратную сторону
10. отец и мать слева направо и в обратную сторону
11. мать и дочь слева направо и в обратную сторону

Правила меняют количество людей на берегах и проверяют возможность выполнения такого перехода благодаря правилу (1).

Далее было реализовано правило обхода состояний (3), которое имеет следующий вид:

path([CL1, PL1, DL1, ML1, SL1, DtL1, Boat1, CR1, PR1, DR1, MR1, SR1, DtR1],

[CL2, PL2, DL2, ML2, SL2, DtL2, Boat2, CR2, PR2, DR2, MR2, SR2, DtR2], Explored, MovesList) :-

move([CL1, PL1, DL1, ML1, SL1, DtL1, Boat1, CR1, PR1, DR1, MR1, SR1, DtR1],

[CL3, PL3, DL3, ML3, SL3, DtL3, Boat3, CR3, PR3, DR3, MR3, SR3, DtR3]),

not(member([CL3, PL3, DL3, ML3, SL3, DtL3, Boat3, CR3, PR3, DR3, MR3, SR3, DtR3], Explored)),

path([CL3, PL3, DL3, ML3, SL3, DtL3, Boat3, CR3, PR3, DR3, MR3, SR3, DtR3],

[CL2, PL2, DL2, ML2, SL2, DtL2, Boat2, CR2, PR2, DR2, MR2, SR2, DtR2],

[[CL3, PL3, DL3, ML3, SL3, DtL3, Boat3, CR3, PR3, DR3, MR3, SR3, DtR3] | Explored],

[[[CL3, PL3, DL3, ML3, SL3, DtL3, Boat3, CR3, PR3, DR3, MR3, SR3, DtR3],

[CL1, PL1, DL1, ML1, SL1, DtL1, Boat1, CR1, PR1, DR1, MR1, SR1, DtR1]] | MovesList ]).

Здесь также были реализованы правила вывода списка состояний. Правило имеет следующий вид:

output([]) :- nl.

output([[A, B] | MovesList]) :-

output(MovesList),

write(B), write(' -> '), write(A), nl.

**Дерево вывода**

**Пример вывода**

**[1,1,1,1,2,2,lft,0,0,0,0,0,0] -> [0,0,1,1,2,2,rgt,1,1,0,0,0,0]**

**[0,0,1,1,2,2,rgt,1,1,0,0,0,0] -> [1,0,1,1,2,2,lft,0,1,0,0,0,0]**

**[1,0,1,1,2,2,lft,0,1,0,0,0,0] -> [0,0,1,1,1,2,rgt,1,1,0,0,1,0]**

**[0,0,1,1,1,2,rgt,1,1,0,0,1,0] -> [1,1,1,1,1,2,lft,0,0,0,0,1,0]**

**[1,1,1,1,1,2,lft,0,0,0,0,1,0] -> [1,1,0,1,0,2,rgt,0,0,1,0,2,0]**

**[1,1,0,1,0,2,rgt,0,0,1,0,2,0] -> [1,1,1,1,0,2,lft,0,0,0,0,2,0]**

**[1,1,1,1,0,2,lft,0,0,0,0,2,0] -> [1,1,0,0,0,2,rgt,0,0,1,1,2,0]**

**[1,1,0,0,0,2,rgt,0,0,1,1,2,0] -> [1,1,0,1,0,2,lft,0,0,1,0,2,0]**

**[1,1,0,1,0,2,lft,0,0,1,0,2,0] -> [0,0,0,1,0,2,rgt,1,1,1,0,2,0]**

**[0,0,0,1,0,2,rgt,1,1,1,0,2,0] -> [0,0,1,1,0,2,lft,1,1,0,0,2,0]**

**[0,0,1,1,0,2,lft,1,1,0,0,2,0] -> [0,0,0,0,0,2,rgt,1,1,1,1,2,0]**

**[0,0,0,0,0,2,rgt,1,1,1,1,2,0] -> [0,0,0,1,0,2,lft,1,1,1,0,2,0]**

**[0,0,0,1,0,2,lft,1,1,1,0,2,0] -> [0,0,0,0,0,1,rgt,1,1,1,1,2,1]**

**[0,0,0,0,0,1,rgt,1,1,1,1,2,1] -> [1,1,0,0,0,1,lft,0,0,1,1,2,1]**

**[1,1,0,0,0,1,lft,0,0,1,1,2,1] -> [0,1,0,0,0,0,rgt,1,0,1,1,2,2]**

**[0,1,0,0,0,0,rgt,1,0,1,1,2,2] -> [1,1,0,0,0,0,lft,0,0,1,1,2,2]**

**[1,1,0,0,0,0,lft,0,0,1,1,2,2] -> [0,0,0,0,0,0,rgt,1,1,1,1,2,2]**

**true.**

В данном выводе программы описываются действия задачи. Например:

[1,1,1,1,2,2,lft,0,0,0,0,0,0] -> [0,0,1,1,2,2,rgt,1,1,0,0,0,0] - это означает переход из одного состояния в другое.

В квадратных скобках указываются следующие атрибуты: количество полицейских, количество миссионеров, количество отцов, количество матерей, количество сыновей, количество дочерей, на каком берегу реки(правый/левый) находится лодка, количество полицейских, количество миссионеров, количество отцов, количество матерей, количество сыновей, количество дочерей на противоположном берегу.

***Вывод:*** В рамках лабораторной работы была реализована программа решения логической задачи на языке Prolog. Также был описан алгоритм решения задачи. Ознакомились с логическим языком программирования Prolog.