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 1

Выполнил студент гр. 121701 Липский Р. В.

Проверил Ивашенко В. П.

Минск 2023

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

Вариант: 1: Два береги реки. На одном берегу есть 3 миссионера и 3 людоеда, требуется с помощью лодки вмещающей не более 2 человек, переправить всех на другой берег. Число присутствующих миссионеров на берегу и в лодке должно быть всегда не меньше числа людоедов.

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 алгоритма

Код программы

```
legal(CL, ML, CR, MR) :-
      ML >= 0, CL >= 0, MR >= 0, CR >= 0, % Количество людей не может быть отрицательным
    (ML >= CL; ML = 0), % На левом берегу должно быть больше миссионеров, чем людоедов, либо не должно быть миссионеров вовсе
  (MR >= CR; MR = 0). % То же, но для правого

% Возможные перемещения:
move([CL,ML,lft,CR,MR],[CL,ML2,rgt,CR,MR2]) :-
        % Два миссионера переплывают реку слева направо
        MR2 is MR+2,
        ML2 is ML-2,
        legal(CL,ML2,CR,MR2).

move([CL,ML,lft,CR,MR],[CL2,ML,rgt,CR2,MR]) :-
        % Два людоеда переплывают реку слева направо
        CR2 is CR+2,
        CL2 is CL-2,
        legal(CL2,ML,CR2,MR).

move([CL,ML,lft,CR,MR],[CL2,ML2,rgt,CR2,MR2]) :-
        %  Один миссионер и один людоед переплывают реку слева направо
        CR2 is CR+1,
        CL2 is CL-1,
        MR2 is MR+1,
        ML2 is ML-1,
        legal(CL2,ML2,CR2,MR2).
```

```
move([CL,ML,lft,CR,MR],[CL,ML2,rgt,CR,MR2]) :-
        % Один миссионер переплывает реку слева направо
        MR2 is MR+1,
        ML2 is ML-1,
        legal(CL,ML2,CR,MR2).

move([CL,ML,lft,CR,MR],[CL2,ML,rgt,CR2,MR]) :-
        % Один людоед перерлывают реку слева направо
        CR2 is CR+1,
        CL2 is CL-1,
        legal(CL2,ML,CR2,MR).

move([CL,ML,rgt,CR,MR],[CL,ML2,lft,CR,MR2]) :-
        % Два миссионера переплывают реку справа налево
        MR2 is MR-2,
        ML2 is ML+2,
        legal(CL,ML2,CR,MR2).

move([CL,ML,rgt,CR,MR],[CL2,ML,lft,CR2,MR]) :-
        % Два людоеда переплывают реку справа налево
        CR2 is CR-2,
        CL2 is CL+2,
        legal(CL2,ML,CR2,MR).

move([CL,ML,rgt,CR,MR],[CL2,ML2,lft,CR2,MR2]) :-
        %  Один миссионер и один людоед переплывают реку справа налево
        CR2 is CR-1,
        CL2 is CL+1,
        MR2 is MR-1,
        ML2 is ML+1,
        legal(CL2,ML2,CR2,MR2).

move([CL,ML,rgt,CR,MR],[CL,ML2,lft,CR,MR2]) :-
        % Один миссионер переплывает реку справа налево
        MR2 is MR-1,
        ML2 is ML+1,
        legal(CL,ML2,CR,MR2).

move([CL,ML,rgt,CR,MR],[CL2,ML,lft,CR2,MR]) :-
        % Один людоед переплывает реку справа налево
        CR2 is CR-1,
        CL2 is CL+1,
        legal(CL2,ML,CR2,MR).

% Перемещение из состяния 1 в 2 существует, если существует такой ход из 1 в 3, что
3 ещё не был достигнут
% и существует переход из состояния 3 в состояние 2 (рекурсия)
path([CL1,ML1,B1,CR1,MR1], [CL2,ML2,B2,CR2,MR2], Explored, MovesList) :-
```

```
    move([CL1, ML1, B1, CR1, MR1], [CL3, ML3, B3, CR3, MR3]),
    not(member([CL3, ML3, B3, CR3, MR3], Explored)),
    path([CL3,ML3,B3,CR3,MR3],
        [CL2,ML2,B2,CR2,MR2],
        [[CL3,ML3,B3,CR3,MR3] | Explored],
        [ [[CL3,ML3,B3,CR3,MR3], [CL1,ML1,B1,CR1,MR1]] | MovesList ]).

path([CL, ML, B, CR, MR], [CL, ML, B, CR, MR], _, MovesList) :- output(MovesList).

output([]) :- nl.
output([[A, B] | MovesList]) :-
    output(MovesList),
    write(B), write(' -> '), write(A), nl.

find(X, Y) :- path(X, Y, [X], _).
find       :- find([3, 3, lft, 0, 0], [0, 0, rgt, 3, 3]).
```

Начальное состояние миссионеров и людоедов:

[3, 3, lft, 0, 0] - 3 людоеда и 3 миссионера находятся на левом берегу.

Целевое состояние программы, необходимое для завершения работы:

[0, 0, rgt, 3, 3] - 3 людоеда и 3 миссионера находятся на правом берегу.

Правила перехода между состояниями (1) должны учитывать, что в лодке помещается только два человека, количество миссионеров на берегу и в лодке должно быть всегда не меньше числа людоедов. . Правила описаны ниже:

```
 legal(CL, ML, CR, MR) :-
      ML >= 0, CL >= 0, MR >= 0, CR >= 0, % Количество людей не может быть отрицательным
     (ML >= CL; ML = 0), % На левом берегу должно быть больше миссионеров, чем людоедов, либо не должно быть миссионеров вовсе
      (MR >= CR; MR = 0). % То же, но для правого
```

Так же есть правила возможных перемещений (2), в которых описываются все варианты перемещения миссионеров и людоедов в лодке (их всего 10):

```
    move([CL,ML,lft,CR,MR],[CL,ML2,rgt,CR,MR2]) :-
            % Два миссионера переплывают реку слева направо
            MR2 is MR+2,
            ML2 is ML-2,
            legal(CL,ML2,CR,MR2).
```

```
move([CL,ML,lft,CR,MR],[CL2,ML,rgt,CR2,MR]) :-
        % Два людоеда переплывают реку слева направо
        CR2 is CR+2,
        CL2 is CL-2,
        legal(CL2,ML,CR2,MR).

move([CL,ML,lft,CR,MR],[CL2,ML2,rgt,CR2,MR2]) :-
        %  Один миссионер и один людоед переплывают реку слева направо
        CR2 is CR+1,
        CL2 is CL-1,
        MR2 is MR+1,
        ML2 is ML-1,
        legal(CL2,ML2,CR2,MR2).

move([CL,ML,lft,CR,MR],[CL,ML2,rgt,CR,MR2]) :-
        % Один миссионер переплывает реку слева направо
        MR2 is MR+1,
        ML2 is ML-1,
        legal(CL,ML2,CR,MR2).

move([CL,ML,lft,CR,MR],[CL2,ML,rgt,CR2,MR]) :-
        % Один людоед перерлывают реку слева направо
        CR2 is CR+1,
        CL2 is CL-1,
        legal(CL2,ML,CR2,MR).

move([CL,ML,rgt,CR,MR],[CL,ML2,lft,CR,MR2]) :-
        % Два миссионера переплывают реку справа налево
        MR2 is MR-2,
        ML2 is ML+2,
        legal(CL,ML2,CR,MR2).

move([CL,ML,rgt,CR,MR],[CL2,ML,lft,CR2,MR]) :-
        % Два людоеда переплывают реку справа налево
        CR2 is CR-2,
        CL2 is CL+2,
        legal(CL2,ML,CR2,MR).

move([CL,ML,rgt,CR,MR],[CL2,ML2,lft,CR2,MR2]) :-
        %  Один миссионер и один людоед переплывают реку справа налево
        CR2 is CR-1,
        CL2 is CL+1,
        MR2 is MR-1,
        ML2 is ML+1,
        legal(CL2,ML2,CR2,MR2).


move([CL,ML,rgt,CR,MR],[CL,ML2,lft,CR,MR2]) :-
```

```
        % Один миссионер переплывает реку справа налево
        MR2 is MR-1,
        ML2 is ML+1,
        legal(CL,ML2,CR,MR2).

move([CL,ML,rgt,CR,MR],[CL2,ML,lft,CR2,MR]) :-
        % Один людоед переплывает реку справа налево
        CR2 is CR-1,
        CL2 is CL+1,
        legal(CL2,ML,CR2,MR).
```

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:

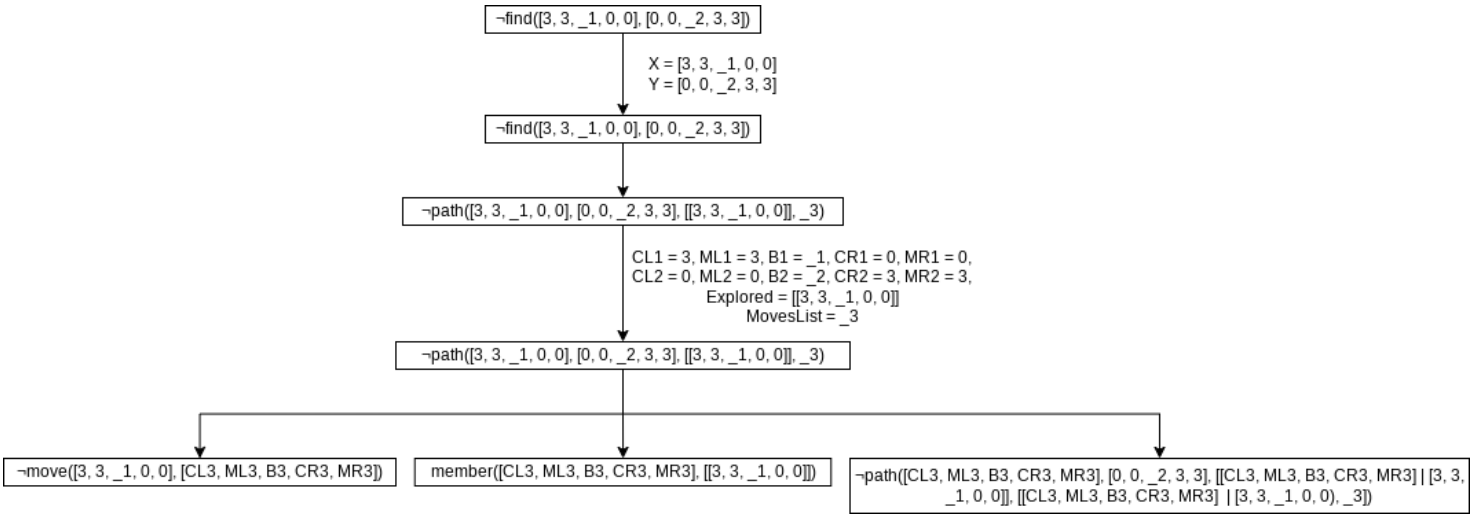
```
path([CL1,ML1,B1,CR1,MR1], [CL2,ML2,B2,CR2,MR2], Explored, MovesList) :-
        move([CL1, ML1, B1, CR1, MR1], [CL3, ML3, B3, CR3, MR3]),
        not(member([CL3, ML3, B3, CR3, MR3], Explored)),
        path([CL3,ML3,B3,CR3,MR3],
            [CL2,ML2,B2,CR2,MR2],
            [[CL3,ML3,B3,CR3,MR3] | Explored],
            [ [[CL3,ML3,B3,CR3,MR3], [CL1,ML1,B1,CR1,MR1]] | MovesList ]).

path([CL, ML, B, CR, MR], [CL, ML, B, CR, MR], _, MovesList) :- output(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 результата

[3,3,lft,0,0] -> [1,3,rgt,2,0]

[1,3,rgt,2,0] -> [2,3,lft,1,0]

[2,3,lft,1,0] -> [0,3,rgt,3,0]

[0,3,rgt,3,0] -> [1,3,lft,2,0]

[1,3,lft,2,0] -> [1,1,rgt,2,2]

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

[2,2,lft,1,1] -> [2,0,rgt,1,3]

[2,0,rgt,1,3] -> [3,0,lft,0,3]

[3,0,lft,0,3] -> [1,0,rgt,2,3]

[1,0,rgt,2,3] -> [1,1,lft,2,2]

[1,1,lft,2,2] -> [0,0,rgt,3,3]

В данном выводе программы описываются действия задачи. Например:

[3,3,lft,0,0] -> [1,3,rgt,2,0] - это означает переход из одного состояния в другое.

В фигурных скобках указываются следующие атрибуты: количество людоедов, количество миссионеров, на каком берегу реки(правый/левый) находится лодка, количество людоедов на противоположном берегу, количество миссионеров на противоположном берегу.

В приведенном примере указано, что в изначальном состоянии на левом берегу реки находится лодка, 3 людоеда и 3 миссионера, на втором берегу 0 людоедов и 0 миссионеров. Во втором же состоянии лодка находится на правом берегу, на левом берегу 1 людоед и 3 миссионера, а на правом - 2 людоеда и 0 миссионеров.

Теперь же разберем вывод подробнее:

1. Изначально на левом берегу находятся три миссионера, три людоеда и лодка. ([3, 3, lft, 0, 0])
2. Два людоеда переправляются на правый берег. ([1, 3, rgt, 2, 0])
3. Один людоед переправляется на левый берег. ([2, 3, lft, 1, 0])
4. Два людоеда переправляются на правый берег. ([0, 3, rgt, 3, 0])
5. Один людоед переправляются на левый берег. ([1, 3, lft, 2, 0])
6. Два миссионера переправляются на правый берег. ([1, 1, rgt, 2, 2])
7. Один миссионер и один людоед переправляются на левый берег. ([2, 2, lft, 1, 1])
8. Два миссионера переправляются на правый берег. ([2, 0, rgt, 1, 3])
9. Один людоед переправляется на левый берег. ([3, 0, lft, 0, 3])
10. Два людоеда переправляются на правый берег. ([1, 0, rgt, 2, 3])
11. Один миссионер переправляется на левый берег. ([1, 1, lft, 2, 2])
12. Один миссионер и один людоед переправляются на правый берег. ([0, 0, rgt, 3, 3])

Таким образом, после всех перемещений все миссионеры и людоеды оказываются на правом берегу реки.

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.