Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет информатики и

Радиоэлектроники

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №2

по курсу “Логические основы интеллектуальных систем”

Вариант 1

Выполнил:

Студент гр. 221701 Робилко Т.М.

Проверил: Ивашенко В. П.

Минск

2024

**Тема:**

Логическое программирование поиска решения задачи.

**Цель:**

Приобрести навыки логического программирования поиска решения задачи.

**Задание:**

В соответствии с вариантом реализовать программу, решающую поставленную задачу. Предусмотреть возможность задания произвольного начального и целевого состояний задачи, а также – возможность задания произвольных значений количественных характеристик этих состояний. В результате выполнения программы выводить протокол работы, описывающий решение задачи.

Два берега реки. На одном из берегов есть три миссионера и три людоеда, требуется с помощью лодки, вмещающий не более двух человек, переправить всех на другой берег. Число присутствующих миссионеров на берегу и в лодке должно быть всегда не меньше числа людоедов.

**Дополнительные теоретические сведения:**

Prolog – средство написания выполнимых на ЭВМ программ. Язык логического программирования, основанный на логике предикатов первого порядка.

**Грамматика языка PROLOG**:

<Prolog -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>

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

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

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

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

**Реализация:**

Для решения поставленной задачи использовался язык SWI-Prolog. Для представления состояния в решении используется функциональный терм state(MisLeft, CanLeft, MisRight, CanRight, BoatSide), в котором:

MisLeft — количество миссионеров на левом берегу.

CanLeft — количество людоедов на левом берегу.

MisRight — количество миссионеров на правом берегу.

CanRight — количество людоедов на правом берегу.

BoatSide — местоположение лодки, которое может принимать значения left или right, что означает нахождение лодки у левого или правого берега соответственно.

Таким образом, начальное состояние представляется как state(3, 3, 0, 0, left), а целевое состояние как state(0, 0, 3, 3, right).

Решение задачи использует поиск в ширину (BFS) для нахождения кратчайшего способа переправки всех миссионеров и людоедов на другой берег. При каждом перемещении проверяется условие безопасности, чтобы на любом берегу количество миссионеров не было меньше количества людоедов.

Таким образом, алгоритм находит последовательность перемещений, которая позволяет безопасно переправить всех миссионеров и людоедов на другой берег, соблюдая все условия задачи.

**В программе используются:**

1. **Встроенные предикаты языка Prolog:**
2. member: Проверяет, является ли элемент членом списка.
3. \+: Логическое отрицание. Возвращает истину, если предикат внутри него ложен.
4. is: Выполняет арифметическое вычисление и присваивает результат переменной.
5. write: Выводит аргумент на экран.
6. nl: Переводит курсор на новую строку.
7. **Факты:**
8. initial\_state(state(3, 3, 0, 0, left)). - Описывает начальное состояние: 3 миссионера и 3 людоеда на левом берегу, 0 миссионеров и 0 людоедов на правом берегу, лодка на левом берегу.
9. goal\_state(state(0, 0, 3, 3, right)). - Описывает целевое состояние: 0 миссионеров и 0 людоедов на левом берегу, 3 миссионера и 3 людоеда на правом берегу, лодка на правом берегу.
10. **Правила:**
11. move(state(ML, CL, MR, CR, left), state(ML2, CL2, MR2, CR2, right)) :-

move\_action(ML, CL, MR, CR, ML2, CL2, MR2, CR2),

safe(ML2, CL2, MR2, CR2).

Описывает возможные перемещения лодки с миссионерами и людоедами с левого на правый берег.

1. move(state(ML, CL, MR, CR, right), state(ML2, CL2, MR2, CR2, left)) :-

move\_action(MR, CR, ML, CL, MR2, CR2, ML2, CL2),

safe(ML2, CL2, MR2, CR2).

Описывает возможные перемещения лодки с миссионерами и людоедами с правого на левый берег.

1. move\_action(XL, YL, XR, YR, XL2, YL2, XR2, YR2) :-

member([DX, DY], [[2, 0], [0, 2], [1, 1], [1, 0], [0, 1]]),

XL >= DX, YL >= DY,

XL2 is XL - DX, YL2 is YL - DY,

XR2 is XR + DX, YR2 is YR + DY.

Определяет возможные перемещения, учитывая ограничения на количество миссионеров и людоедов. Генерирует новые состояния после перемещения.

1. safe(ML, CL, MR, CR) :-

(ML >= CL ; ML = 0),

(MR >= CR ; MR = 0).

Проверяет, что на каждом берегу количество миссионеров не меньше количества людоедов (или миссионеров нет вообще), чтобы обеспечить их безопасность.

1. print\_solution: Печатает путь от начального состояния к целевому.
2. path(State, State, Path, Path).

path(State, Goal, Visited, Path) :-

move(State, NextState),

\+ member(NextState, Visited),

path(NextState, Goal, [NextState|Visited], Path).

Осуществляет поиск пути от начального состояния State к целевому состоянию Goal. Первый предикат используется, когда текущее состояние совпадает с целевым. Второй предикат описывает рекурсивный процесс поиска пути.

1. solve(State, Goal) :-

path(State, Goal, [State], Path),

print\_solution(Path).

Обеспечивает поиск решения с данными начальным и целевыми состояниями.

**Код программы:**

*% Лабораторная работа №2 по дисциплине "Логические основы интеллектуальных систем"*

*% Выполнил студент группы 221701*

*% Робилко Тимур Маркович*

*% Вариант: 1*

*% Дата изменения: 05.06.2024*

*% В файле содержится исходный код для решения задачи:*

*% Два берега реки. На одном из берегов есть три миссионера и три людоеда, требуется с помощью лодки,*

*% вмещающий не более двух человек, переправить всех на другой берег. Число присутствующих миссионеров*

*% на берегу и в лодке должно быть всегда не меньше числа людоедов.*

*%*

*% Источники:*

*% 1. Логические основы интеллектуальных систем. Практикум : учеб.- метод. пособие / В. В. Голенков [и др.].*

*% – Минск : БГУИР, 2011. – 70 с. : ил. ISBN 978-985-488-487-5.*

*% 2. SWI Prolog [Электронный ресурс]. -- Режим доступа https://www.swi-prolog.org/*

*% Предикаты перемещений*

*move(state(ML, CL, MR, CR, left), state(ML2, CL2, MR2, CR2, right)) :-*

*move\_action(ML, CL, MR, CR, ML2, CL2, MR2, CR2),*

*safe(ML2, CL2, MR2, CR2).*

*move(state(ML, CL, MR, CR, right), state(ML2, CL2, MR2, CR2, left)) :-*

*move\_action(MR, CR, ML, CL, MR2, CR2, ML2, CL2),*

*safe(ML2, CL2, MR2, CR2).*

*move\_action(XL, YL, XR, YR, XL2, YL2, XR2, YR2) :-*

*member([DX, DY], [[2, 0], [0, 2], [1, 1], [1, 0], [0, 1]]),*

*XL >= DX, YL >= DY,*

*XL2 is XL - DX, YL2 is YL - DY,*

*XR2 is XR + DX, YR2 is YR + DY.*

*% Предикат "безопасного" состояния*

*safe(ML, CL, MR, CR) :-*

*(ML >= CL ; ML = 0),*

*(MR >= CR ; MR = 0).*

*% Вывод результата (списка состояний)*

*print\_solution([]).*

*print\_solution([State|Path]) :-*

*print\_solution(Path),*

*write(State), nl.*

*% bfs*

*path(State, State, Path, Path).*

*path(State, Goal, Visited, Path) :-*

*move(State, NextState),*

*\+ member(NextState, Visited),*

*path(NextState, Goal, [NextState|Visited], Path).*

*initial\_state(state(3, 3, 0, 0, left)).*

*goal\_state(state(0, 0, 3, 3, right)).*

*% Вызов решения*

*solve(State, Goal) :-*

*path(State, Goal, [State], Path),*

*print\_solution(Path).*

*solve(State) :-*

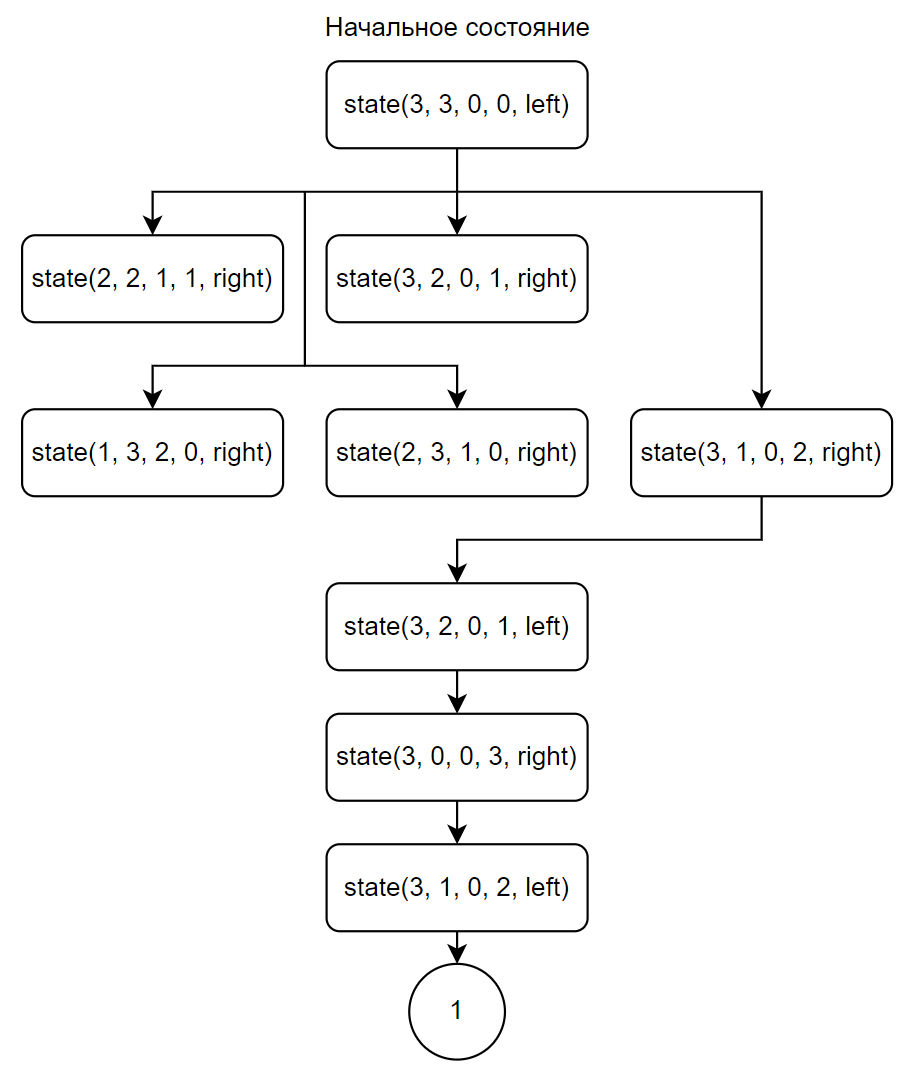
*goal\_state(Goal),*

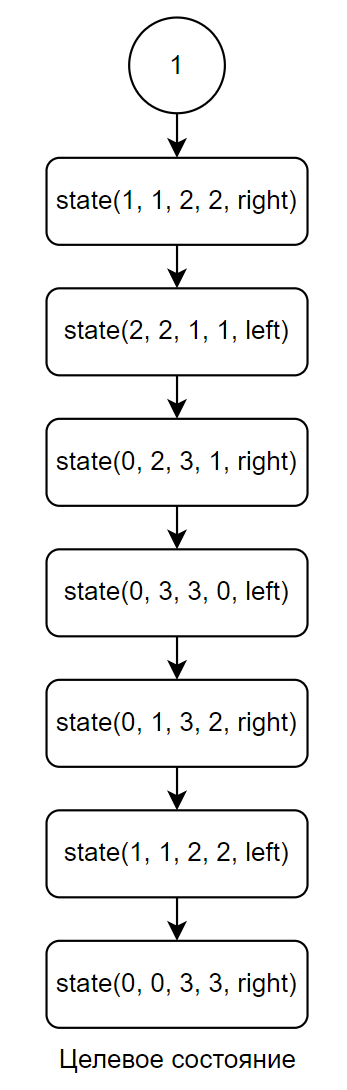
*solve(State, Goal).*

*solve :-*

*initial\_state(State),*

*solve(State).*

**Дерево логического вывода:**

****

**Вывод:**

В рамках лабораторной работы была реализована программа решения логической задачи на языке Prolog. Было построено дерево логического вывода, соответствующее результату работы данной программы.

**Список использованных источников:**

1. Логические основы интеллектуальных систем. Практикум: учебно-методическое пособие / В. В. Голенков, В. П. Ивашенко, Д. Г. Колб, К. А. Уваров. – Минск: БГУИР, 2011.
2. SWI-Prolog Documentation [Электронный ресурс]. –https://www.swi-prolog.org/pldoc/man?section=libpl.