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 12**

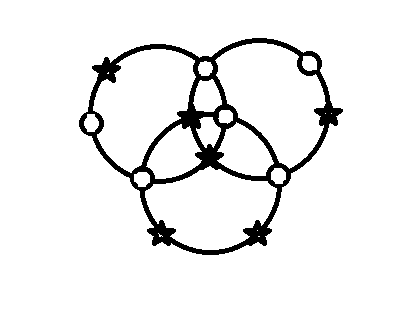
Выполнил студент гр. 221703 Вечорко Д. Н.

Проверил Ивашенко В. П.

Минск 2024

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

**Вариант: 12:** *Задача с тремя окружностями*. Вращением окружностей вокруг своего центра добиться того, чтобы все сферы оказались на пересечениях. Окружности за один шаг поворачиваются на одну шестую круга, т.е. сменяются точки, центральная точка круга не смещается при повороте. Начальное состояние представлено на рисунке:



**Описание лабораторной работы:**

Решение поставленной задачи сводится к следующим подзадачам:

1. Анализ структуры начального состояния окружностей и объектов на них.
2. Анализ целевого состояния окружностей и объектов на них.
3. Анализ правил перехода между состояниями (вращения окружностей).
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***

run(A, B, C):-

iter(A, B, C, 0, 0, 0, [1, 2, 3]);

iter(A, B, C, 0, 0, 0, [1, 3, 2]);

iter(A, B, C, 0, 0, 0, [2, 1, 3]);

iter(A, B, C, 0, 0, 0, [2, 3, 1]);

iter(A, B, C, 0, 0, 0, [3, 1, 2]);

iter(A, B, C, 0, 0, 0, [3, 2, 1]).

log(A, B, C):-

write(A),

write(" "),

write(B),

write(" "),

write(C),

nl.

% Целевое состояние

iter([A1, b, b, b, b, A6], [b, b, b, B4, B5, b], [b, C2, C3, b, b, b], \_, \_, \_, \_):-

log([A1, b, b, b, b, A6], [b, b, b, B4, B5, b], [b, C2, C3, b, b, b]),

nl,

!.

iter(A, B, C, A\_ITER, B\_ITER, C\_ITER, [Q, W, E]):-

log(A, B, C), % <---

(C\_ITER < 7 ->

shift\_with\_order(A, B, C, A\_ITER, B\_ITER, C\_ITER + 1, [Q, W, E], Q), !

;

(B\_ITER < 6 ->

shift\_with\_order(A, B, C, A\_ITER, B\_ITER + 1, 0, [Q, W, E], W), !

;

(A\_ITER < 6 ->

shift\_with\_order(A, B, C, A\_ITER + 1, 0, 0, [Q, W, E], E), !

;

fail

)

)

).

shift\_with\_order(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER, NUM):-

(NUM = 1 -> shift\_first(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER);

(NUM = 2 -> shift\_second(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER);

(NUM = 3 -> shift\_third(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER)))).

shift\_first([A1, A2, A3, A4, A5, A6], [B1, \_, B3, B4, B5, \_], [\_, C2, C3, C4, \_, C6], A\_ITER, B\_ITER, C\_ITER, OREDER):-

iter([A6, A1, A2, A3, A4, A5], [B1, A2, B3, B4, B5, A4], [A1, C2, C3, C4, A3, C6], A\_ITER, B\_ITER, C\_ITER, OREDER).

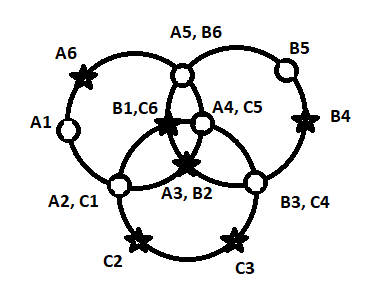
shift\_second([A1, A2, \_, A4, \_, A6], [B1, B2, B3, B4, B5, B6], [C1, C2, C3, \_, C5, \_], A\_ITER, B\_ITER, C\_ITER, OREDER):-

iter([A1, A2, B1, A4, B5, A6], [B6, B1, B2, B3, B4, B5], [C1, C2, C3, B2, C5, B6], A\_ITER, B\_ITER, C\_ITER, OREDER).

shift\_third([A1, \_, A3, \_, A5, A6], [\_, B2, \_, B4, B5, B6], [C1, C2, C3, C4, C5, C6], A\_ITER, B\_ITER, C\_ITER, OREDER):-

iter([A1, C6, A3, C4, A5, A6], [C5, B2, C3, B4, B5, B6], [C6, C1, C2, C3, C4, C5], A\_ITER, B\_ITER, C\_ITER, OREDER).

Для решения поставленной задачи было необходимо описать структуру окружностей и точек, для этого были использованы три списка атомов, каждый из которых представляет собой одну окружность с точками, где b - это синяя точка, а r - красная точка. Например для начального состояния из условия задачи можно использовать следующие списки: [b, b, r, b, b, r], [r, r, b, r, b, b], [b, r, r, b, b, r]. Для наглядности каждой точке поставим в соответствие переменную: [A1, A2, A3, A4, A5, A6], [B1, B2, B3, B4, B5, B6], [C1, C2, C3, C4, C5, C6]. Обозначим позиции точек на рисунке (нумерация начинается с одинаковой позиции для каждой окружности – крайняя левая точка):



Целевое состояние программы, необходимое для завершения работы:

iter([A1, b, b, b, b, A6], [b, b, b, B4, B5, b], [b, C2, C3, b, b, b], \_, \_, \_, \_):-

log([A1, b, b, b, b, A6], [b, b, b, B4, B5, b], [b, C2, C3, b, b, b]),

nl,

!.

Правила перехода между состояниями должны учитывать, что окружности пересекаются, т.е. имеют общие точки, что влечет возможности перемещения элемента несколькими окружностями. Правила описаны ниже:

shift\_first([A1, A2, A3, A4, A5, A6], [B1, \_, B3, B4, B5, \_], [\_, C2, C3, C4, \_, C6], A\_ITER, B\_ITER, C\_ITER, OREDER):-

iter([A6, A1, A2, A3, A4, A5], [B1, A2, B3, B4, B5, A4], [A1, C2, C3, C4, A3, C6], A\_ITER, B\_ITER, C\_ITER, OREDER).

shift\_second([A1, A2, \_, A4, \_, A6], [B1, B2, B3, B4, B5, B6], [C1, C2, C3, \_, C5, \_], A\_ITER, B\_ITER, C\_ITER, OREDER):-

iter([A1, A2, B1, A4, B5, A6], [B6, B1, B2, B3, B4, B5], [C1, C2, C3, B2, C5, B6], A\_ITER, B\_ITER, C\_ITER, OREDER).

shift\_third([A1, \_, A3, \_, A5, A6], [\_, B2, \_, B4, B5, B6], [C1, C2, C3, C4, C5, C6], A\_ITER, B\_ITER, C\_ITER, OREDER):-

iter([A1, C6, A3, C4, A5, A6], [C5, B2, C3, B4, B5, B6], [C6, C1, C2, C3, C4, C5], A\_ITER, B\_ITER, C\_ITER, OREDER).

Правило смещает точки первой окружности против часовой стрелки, что в результате даёт следующий список (для окружности А из начального состояния): [A6, A1, A2, A3, A4, A5]. В остальных списках заменяются те точки, которые пересекаются с первой окружностью. Это точки B2, B6, C1, C5, на их место станут точки списка А, в результате получаем следующие списки для окружностей B и С: [B1, A2, B3, B4, B5, A4], [A1, C2, C3, C4, A3, C6].

Далее было реализовано правило обхода состояний которое имеет следующий вид:

iter(A, B, C, A\_ITER, B\_ITER, C\_ITER, [Q, W, E]):-

log(A, B, C), % <---

(C\_ITER < 7 ->

shift\_with\_order(A, B, C, A\_ITER, B\_ITER, C\_ITER + 1, [Q, W, E], Q), !

;

(B\_ITER < 6 ->

shift\_with\_order(A, B, C, A\_ITER, B\_ITER + 1, 0, [Q, W, E], W), !

;

(A\_ITER < 6 ->

shift\_with\_order(A, B, C, A\_ITER + 1, 0, 0, [Q, W, E], E), !

;

fail

)

)

).

Здесь также используются переменные A\_ITER, B\_ITER и C\_ITER, которые нужны для учёта итераций вращения каждой окружности. После полного оборота окружность возвращается в исходное состояние, затем сдвигается следующая окружность и повторяется полный оборот предыдущей. После полного оборота второй окружности сдвигается последняя и повторяются обороты двух предыдущих. Для обхода всех состояний окружностей необходимо также учесть порядок вращения, для этого используются переменные Q, W, E и правило shift\_with\_order, определяющее какую окружность сдвигать на данной итерации. Правило имеет следующий вид:

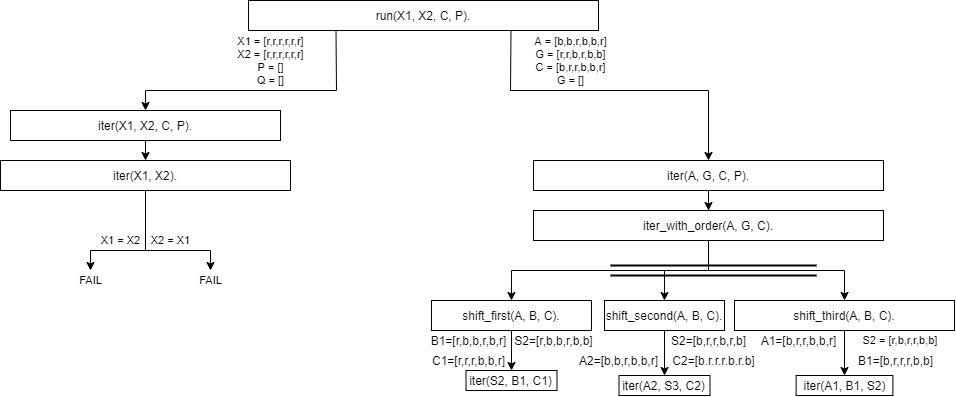
shift\_with\_order(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER, NUM):-

(NUM = 1 -> shift\_first(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER);

(NUM = 2 -> shift\_second(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER);

(NUM = 3 -> shift\_third(A, B, C, A\_ITER, B\_ITER, C\_ITER, OREDER)))).

**Дерево вывода**



В качестве входной точки программы было определено правило run, запускающее обход окружностей с разным порядком вращения. Примеры запросов для запуска программы:

1. run([b, b, r, b, b, r], [r, r, b, r, b, b], [b, r, r, b, b, r]). - начальное состояние из условия задачи
2. run([b, r, b, r, r, r], [r, b, b, r, b, r], [r, b, b, b, r, r]).
3. run([b, b, b, b, b, b], [r, b, r, r, r, b], [b, r, r, r, b, r]). - все элементы первой окружности - сферы, остальные - звездочки.
4. run([b, b, b, b, b, b], [b, b, b, b, b, b], [b, b, b, b, b, b]). - все сферы.
5. run([r, r, r, r, r, r], [r, r, r, r, r, r], [r, r, r, r, r, r]). - все звездочки, достигнуть конечного состояния невозможно и программа вернёт false.
6. run([r, b, r, b, b, r], [r, r, b, r, b, b], [b, r, r, b, b, r]). - начальное состояние из условия задачи, но точка А1 - звездочка, тогда всего синих точек 5, достигнуть конечного состояния невозможно и программа вернёт false.

Пример вывода программы (часть шагов сокращена):

[b,b,r,b,b,r] [r,r,b,r,b,b] [b,r,r,b,b,r]

[r,b,b,r,b,b] [r,b,b,r,b,b] [b,r,r,b,r,r]

[b,r,b,b,r,b] [r,b,b,r,b,r] [r,r,r,b,b,r]

[b,b,r,b,b,r] [r,r,b,r,b,b] [b,r,r,b,b,r]

[r,b,b,r,b,b] [r,b,b,r,b,b] [b,r,r,b,r,r]

[b,r,b,b,r,b] [r,b,b,r,b,r] [r,r,r,b,b,r]

[b,b,r,b,b,r] [r,r,b,r,b,b] [b,r,r,b,b,r]

[r,b,b,r,b,b] [r,b,b,r,b,b] [b,r,r,b,r,r]

[r,b,r,r,b,b] [b,r,b,b,r,b] [b,r,r,b,r,b]

[b,r,b,r,r,b] [b,b,b,b,r,r] [r,r,r,b,r,b]

...

[b,b,r,r,b,r] [b,r,r,b,b,b] [b,r,r,r,r,b]

[b,b,b,r,b,r] [b,b,r,r,b,b] [b,r,r,r,r,b]

[b,b,b,r,b,r] [b,b,b,r,r,b] [b,r,r,b,r,b]

[b,b,b,r,r,r] [b,b,b,b,r,r] [b,r,r,b,r,b]

[r,b,b,b,r,r] [b,b,b,b,r,r] [b,r,r,b,b,b]

[r,b,b,b,r,r] [r,b,b,b,b,r] [b,r,r,b,b,r]

[r,b,r,b,b,r] [r,r,b,b,b,b] [b,r,r,b,b,r]

[r,b,r,b,b,r] [b,r,r,b,b,b] [b,r,r,r,b,b]

[r,b,b,b,b,r] [b,b,r,r,b,b] [b,r,r,r,b,b]

[r,b,b,b,b,r] [b,b,b,r,r,b] [b,r,r,b,b,b]

При начальном состоянии из условия задачи конечное состояние достигается за 870 шагов.

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