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

«Логическое программирование»

Вариант 9

Выполнил студент гр. 121702 Заломов Р.А.

Проверил Ивашенко В. П.

Минск 2023

Цель: приобрести навыки логического программирования поиска решения задачи.

Задача: расставить на шахматной доске восемь ферзей так, чтобы ни один ферзь не находился под боем другого ферзя.

Теоретические сведения

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

В рамках лабораторной работы на языке Prolog была реализована программа для поиска решения задачи о восьми ферзях. Данная версия программы может работать с более общим случаем, а именно N ферзей на шахматной доске NxN, где N – целое число. Так как для решения задачи в любом случае требуется занять все столбцы доски, то задача сводится к поиску конкретной позиции на каждом столбе для ферзей.

Листинг программы:

```
% Лабораторная работа №2 по дисциплине ЛОИС
% Выполнена студентом группы 121702 БГУИР Заломовым Романом Андреевичем
% 18.05.22
% Файл программы осуществляет решение задачи N ферзей, размещённых на шахматной доске NxN
% таким образом, что ни один ферзь не может бить любого другого

% Учитывая то, что все колонки шахматной доски должны быть заняты,
% суть задачи осуществляется в поиске клетки на соответствующей колонке.


% Главный предикат, отвечающий за решение задачи
queens(N, Solution) :-
    length(Solution, N),
    numlist(1, N, Rows),
    permutation(Rows, Solution),
    safe_all(Solution).

% Факт, указывающий на то, что доска 0x0 всегда имеет решение.
safe_all([]).

% Предикат, определяющий условия безопасности какого-либо ферзя относительно всех остальных
safe_all([Queen|Queens]) :-
    safe(Queens, 1, Queen),
    safe_all(Queens).

% Факт, указывающий на то, что на пустой доске никто никого бить не может
safe([], _, _).

% Предикат, определяющий условия безопасности какого-либо ферзя относительно другого (но не самого себя)
safe([OtherQueen|Queens], Offset, Queen) :-
    Queen =\= OtherQueen,
    Queen + Offset =\= OtherQueen,
    Queen - Offset =\= OtherQueen,
    NewOffset is Offset + 1,
    safe(Queens, NewOffset, Queen).
```

Рис 1. Листинг программы

Дерево вывода для данной программы (для случая, если требуется расставить 4 королев на доске 4×4):

Рис 2. Дерево вывода программы

Пример выполнения программы:

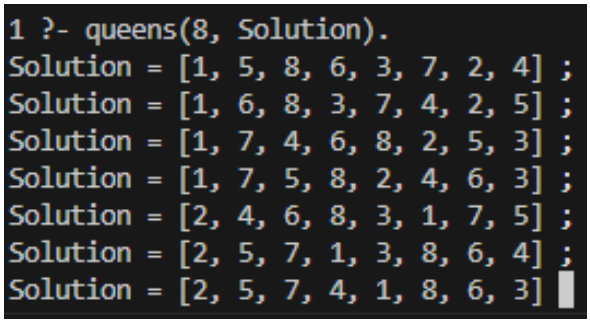

Рис 3. Пример выполнения программы

Встроенные предикаты

length(*?List, ?Length*) - истинно, если Length представляет собой количество элементов в List. Предикат может быть использован для нахождения длины списка или создания списка (содержащего переменные) длины Length.

Предикат является недетерминированным и производит списки возрастающей длины, если List - частичный список, а Length - переменная.

numlist(*+Low, +High, -List*) – заполняет список List целыми числами от Low до High.

permutation(*?Xs, ?Ys*) - истинно, если Xs является перестановкой Ys. Предикат может решить задачу Ys по Xs или Xs по Ys, или даже перечислить Xs и Ys вместе. Предикат предназначен в первую очередь для генерации перестановок.

Формализация на языке логики предикатов первого порядка:

queens	Q
length	L
safe_all	A
safe	S
_=/=_	N

Табл. 1 – Таблица предикатов

numlist	n
permutation	p
[_|_]	h
_+_	f
_-_	m
1	a

Табл. 2 – Таблица термов

Описание предикатов

1. **Правило**: queens(N, Solution) :-
   length(Solution, N),
   numlist(1, N, Rows),
   permutation(Rows, Solution),
   safe_all(Solution).
   Результат: (\-/x(\-/y((L(y)/\A(p(n(x), y)))->Q(x, y))))

2. **Правило**: safe_all([Queen|Queens]) :-
   safe(Queens, 1, Queen),
   safe_all(Queens).
   Результат: (\-/y((S(h(y),a)/\A(h(y)))->A(y)))

3. **Правило**: safe([OtherQueen|Queens], Offset, Queen) :-
   Queen =\= OtherQueen,
   Queen + Offset =\= OtherQueen,
   Queen - Offset =\= OtherQueen,
   NewOffset is Offset + 1,
   safe(Queens, NewOffset, Queen).
   Результат: (\-/x(\-/y(\-/z((((N(z,h(x))/\N(f(z,y),h(x)))/\N(m(z,y),h(x)))/\
   S(x,f(y,a),z))->S(x,y,z)))))

Вывод

В процессе выполнения лабораторной работы были приобретены навыки логического программирования на примере языка Prolog. На данном языке была реализована программа поиска решение задачи восьми ферзей.

Список использованных источников:

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 Documentation [Электронный ресурс]. — Режим доступа: https://www.swi-prolog.org — Дата доступа: 15.05.2023