6. ОПИСАНИЕ НА СРЕДАТА И РЕШЕНИЕ

Въведение. Към решаването на конкретна задача може да се подходи по различен начин. Често всеки подход изисква и подходящо представяне на данните, или казано по друг начин – изисква подходящо описание на средата.

При описание на средата водещи могат да бъдат различни критерии, например:

-описанието да бъде по-икономично;

-описанието да бъде по-прегледно и разбираемо.

Прегледността води често до излишък в описанието, но икономисването на някои данни от своя страна може да усложни програмата, като наложи те да се изчисляват от наличните.

Тези зависимости могат да се илюстрират, като се разгледат няколко варианта на класическата задача за разполагане на N царици върху шахматна дъска NxN. Единият от вариантите ще бъде разгледан подробно, а другите ще се развият като задачи за самостоятелна работа.

Постановка. Задачата е да се разположат върху шахматна дъска с размер NxN N царици по такъв начин, че нито една да не е под заплахата на нито една от другите царици. Това означава, че не трябва да има две царици на един и същи ред, колона или диагонал.

При решаването на задачата на преден план излиза начинът за описване на редовете, колоните и диагоналите така, че всеки да има уникално име. Възможен е следният вариант на описанието. Редовете(R) и колоните(C) на шахматната дъска се номерират от 1 до N. Уникалните номера на диагоналите се получават от следните формули:

Диагонал_Тип_/ = R + C – 1 Диагонал_Тип_\ = N+ C – R

За дъска 4x4 това води до следната номерация.

	1	*2*	*3*	*4*
1	**1**	**2**	3	**4**
2	**2**	3	**4**	5
3	3	**4**	5	**6**
4	**4**	5	**6**	**7**

	1	*2*	*3*	*4*
1	**4**	5	**6**	**7**
2	3	**4**	5	**6**
3	**2**	3	**4**	5
4	**1**	**2**	3	**4**

Упътване. Необходим е предикат за позиция на царицата

queen = q(integer, integer)

и списък от царици

queens = queen *

Свободните (без царици) редове, колони и диагонали от първи и втори тип се оформят в списъци от вида

freelist = integer*

Това става с предикат makelist (брой елементи, списък). При формиране на списъците с редове и колони броят елементи е равен на размерността на дъската (N), а броят на диагоналите се изчислява по формулата (2*N)-1.

Състоянието на шахматната дъска ще се задава с петместен предикат – списък от местоположения на царици, свободни редове, свободни колони, свободни Диагонал_Тип_/ , свободни Диагонал_Тип_\ ) – и ще изглежда по следния начин:

board = board(queens, freelist, freelist, freelist, freelist).

Например шахматна дъска без царици (4 x 4) ще изглежда така:

board = ([], [1,2,3,4], [1,2,3,4], [1,2,3,4,5,6,7], [1,2,3,4,5,6,7]).

Шахматна дъска с една царица на първи ред и първа колона се описва така:

board = (q[1,1], [2,3,4], [2,3,4], [2,3,4,5,6,7], [1,2,3,5,6,7]).

При така въведените означения задачата може да се доуточни по следния начин: да се съставят N четворки (ред, колона, Диагонал_Тип_/ , Диагонал_Тип_\), така че да няма едноименна позиция с еднакви елементи. Изборът на ред и колона предопределя другите два елемента.

Стратегията за търсене на решение би могла да изглежда по следния начин: генерират се четирите области за възможните стойности на елементите на четворките; избира се местоположение на първата царица; отстраняват се заетите от тази царица стойности от съответните им области; от останалите елементи в областите се избира място за следващата царица и така до изчерпване на поне една от областите с допустими стойности.

Ключов за програмата наред с предиката **board** за текущото състояние на дъската е и триместният предикат place_a_queen(**N -** размер на дъската, **board** - начално състояние на дъската, **board -** ново състояние на дъската). Програмата започва от празна дъска и с предикат place_a_queen се поставя една царица на последния ред и последната колона. Изпълнението продължава до изчерпване на списъка от свободни редове или колони.

При поставяне на царица върху свободни ред и колона те трябва да се изтрият от списъка на свободните редове и колони (предикат findandremove (избран ред, списък от свободни редове, списък с останалите след избора свободни редове)). Изчисляват се и новозаетите диагонали от двата типа и също се изтриват от списъка на свободните.

Предикатът nextrow(избран ред, списък от свободни редове, списък с останалите след избора свободни редове) търси свободен ред за разполагане на поредната царица.

Със задаването на броят на цариците, които ще се разполагат на шахматната дъска, се определя броя на редовете и колоните, с които ще се работи.

Програма.

```
nqueens(N):-
        makelist(N,L),              Diagonal is N*2-1,
        makelist(Diagonal,LL),
        placeN(N,board([],L,L,LL,LL),Final),
```

```
        write(Final).

placeN(_, board(D,[],[],D1,D2),board(D,[],[],D1,D2)):-!.
placeN(N,Board1, Result):-
        place_a_queen(N,Board1,Board2),
        placeN(N,Board2, Result).

place_a_queen(N,
board(Queens, Rows, Columns, Diag1, Diag2),
board([q(R,C)|Queens], NewR, NewC, NewD1, NewD2)):-
nextrow(R, Rows, NewR),
findandremove(C, Columns,NewC),
D1 is N+C-R, findandremove(D1, Diag1, NewD1),
D2 is R+C-1, findandremove(D2, Diag2, NewD2).

findandremove(X, [X|Rest], Rest).
findandremove(X, [Y|Rest], [Y|Tail]):-
        findandremove(X, Rest, Tail).

makelist(1,[1]).
makelist(N,[N|Rest]):-
        N1 is N-1, makelist(N1, Rest).

nextrow(Row, [Row|Rest], Rest).
```

Решенията за две цели имат следния вид:

```
nqueens(4)
board([q(1,2),q(2,4),q(3,1),q(4,3)],[],[],[7,4,1],[7,4,1])True
1 Solution

nqueens(8)
board([q(1,5),q(2,7),q(3,2),q(4,6),q(5,3),q(6,1),q(7,4),q(8,8)],[],[],[15,14,11,9,4,
2,1, 14,13,12,11,3,2,1])True
1 Solution
```

Работа на Пролог.

След като се формират списъците [4,3,2,1], [4,3,2,1], [7,6,5,4,3,2,1] и [7,6,5,4,3,2,1] за N=4, интересните детайли на програмата могат да се проследят от трасировката.

```
Trace:>>CALL:
placen(4,board([],[4,3,2,1],[4,3,2,1],[7,6,5,4,3,2,1],[7,6,5,4,3,2,1]),_)
Trace:>>CALL:
place_a_queen(4,board([],[4,3,2,1],[4,3,2,1],[7,6,5,4,3,2,1],[7,6,5,4,3,2,1]),_)
Trace: >> CALL:   nextrow(_,[4,3,2,1],_)
```

Trace: >> RETURN: nextrow(4,[4,3,2,1],[3,2,1])
Trace: >> CALL: findandremove(_,[4,3,2,1],_)
Trace: >> RETURN: findandremove(4,[4,3,2,1],[3,2,1])
…….
Trace:>>RETURN:
place_a_queen(4,board([],[4,3,2,1],[4,3,2,1],[7,6,5,4,3,2,1],[7,6,5,4,3,2,1]),board([q(4,4)],[3,2,1],[3,2,1],[7,6,5,3,2,1],[6,5,4,3,2,1]))
Trace: >> CALL:
placen(4,board**[q(4,4)],[**3,2,1],[3,2,1],[7,6,5,3,2,1],[6,5,4,3,2,1]),_)
Trace:>>CALL:
place_a_queen(4,board([q(4,4)],[3,2,1],[3,2,1],[7,6,5,3,2,1],[6,5,4,3,2,1]),_)
Trace: >> CALL: nextrow(_,[3,2,1],_)
….
Trace: >> CALL:
placen(4,board**[q(3,2),q(4,4)],[**2,1],[3,1],[7,6,5,2,1],[6,5,3,2,1]),_)
Trace: >> CALL:
place_a_queen(4,board([q(3,2),q(4,4)],[2,1],[3,1],[7,6,5,2,1],[6,5,3,2,1]),_)
Trace: >> CALL: nextrow(_,[2,1],_)
Trace: >> RETURN: nextrow(2,[2,1],[1])
…
Trace: >> CALL:
placen(4,board**[q(3,1),q(4,4)],[**2,1],[3,2],[7,6,5,3,1],[6,5,4,2,1]),_)
Trace: >> CALL:
place_a_queen(4,board([q(3,1),q(4,4)],[2,1],[3,2],[7,6,5,3,1],[6,5,4,2,1]),_)
Trace: >> CALL: nextrow(_,[2,1],_)
Trace: >> RETURN: nextrow(2,[2,1],[1])
….
Trace: >> CALL: placen(4,board**[q(2,3),q(3,1),q(4,4)],[**1],[2],[7,6,3,1],[6,5,2,1]),_)
Trace: >> CALL:
place_a_queen(4,board([q(2,3),q(3,1),q(4,4)],[1],[2],[7,6,3,1],[6,5,2,1]),_)
Trace: >> CALL: nextrow(_,[1],_)
…
Trace: >> CALL:
placen(4,board**[q(4,3)],[**3,2,1],[4,2,1],[7,6,5,4,2,1],[7,5,4,3,2,1]),_)
Trace:>>CALL:
place_a_queen(4,board([q(4,3)],[3,2,1],[4,2,1],[7,6,5,4,2,1],[7,5,4,3,2,1]),_)
Trace: >> CALL: nextrow(_,[3,2,1],_)
….
Trace: >> CALL:
placen(4,board**[q(3,1),q(4,3)],[**2,1],[4,2],[7,6,5,4,1],[7,5,4,2,1]),_)
Trace: >> CALL:

За препоръчване е да се следи всеки номер на свободен ред, колонка и диагонали от първи и втори тип едновременно.

Задание

1. Анализирайте изпълнението на програмата за N=5.

Какво ще се случи, ако се отстрани операторът (!) в седмия ред на програмата?

2. Променете програмата така, че за разполагане на царица да се избира ред (а не колона, както е в текста на програмата).

3. Разгледайте следната програма:

Забележка. Търсеното решение е списък от позиции на цариците от следния вид: [Xl/Yl, X2/Y2, X3/Y3, X4/Y4, X5/Y5, X6/Y6, X7/Y7, X8/Y8]

Ако се фиксират първите координати, за да се намали броят на вариантите, този списък добива следния вид:[1/Y1, 2/Y2, 3/Y3, 4/Y4, 5/Y5, 6/Y6, 7/Y7, 8/Y8]

Решението се търси, като се намерят неизвестните координати в този списък.

Редовете, които започват с %, са коментар.

Програма.

```
Solution( [ ] ).
Solution( [“X/Y”| Rest] ) :-
% Първата царица е на позиция X/Y, където X е номер на ред, а Y – номер на колона
%  Останалите царици ще бъдат на позиции от списъка Rest
Solution(Rest),
belong (Y, [1, 2, 3, 4, 5, 6, 7, 8]),
free([“X/Y”| Rest ).          % Първата царица не застрашава останалите
free( _, [ ]).                % Никой никого не застрашава
free( “X/Y”, [“X1/Y1” | Rest] ) :-
Y =\= Y1,                     % Различни колони
Y1-Y =\= X1-X                 % Различни  диагонали
Y1-Y =\= X-X1,
free(“X/Y”, [“X1/Y1” | Rest).
belong ( X, [X | L] ).
belong ( X, [Y | L] ) :- belong ( X, L).
```

Упътване. Решението представлява списък от позиции на царици, които не се застрашават една друга. Координатите са число/число, където наклонената черта е само разделител. Знакът “=\=” означава “отлично от”.

4. Проведете експерименти с програмата от точка 3.

Изяснете идеите, които лежат в основата на тази програма

Упътване. В началото списъкът от разположени царици е празен. В общия случай предикатът изглежда така: Solution([“X/Y”| Rest]). За да бъде наистина решение така представеният списък, необходимо е списъкът Rest да бъде решение, X и Y трябва да са цели числа от 1 до 8 и царицата на поле “X / Y” не трябва да застрашава позициите от списъка Rest.

Третото изискване се гарантира, като се зададе правило за предиката free(head| Rest). Без съмнение free(_, [ ]) е вярно, тъй като в празния списък никой никого не застрашава. За да е вярно free("X/Y", ["X1/Y1" | Rest]), необходимо е Y и Y1 да са различни (това означава цариците да са в различни колони) и на различни диагонали. Второто условие се гарантира чрез изискването разликата между номерата на колоните да е различна от разликата между номерата на редовете.

5. Сравнете двете програми по ефективност и прегледност. Къде в програмата се задава редът ?

6. Анализирайте следната програма (Bratko, Ivan, Prolog Programming for Artificial Intelligence, Addison-Wesley publishing company):

```
Solution(queens) :-
permutation ( [1, 2, 3, 4, 5, 6, 7, 8], queens ),
safe (queens).
permutation ( [ ], [ ]).
permutation ( [Head | Tail], Tailperm ) :-
permutation(Tail , Tailperm ),
delete( Head, listperm, Tailperm).
        % Поставяне на главата в пермутираната опашка
delete ( A, [A | list).
delete ( A, [B | list], [B, list1] ) :-
delete ( A, list, list1).
safe ( [ ]).
safe ( [queen | Rest]) :-
safe (Rest),
noattack (queen, Rest, 1).
noattack( _, [ ], _ ).
noattack ( Y, [Y1 | listY], distanceX) :-
Y1-Y =\= distanceX,
Y-Y1 =\= distanceX,
distance1 is distanceX + 1,
noattack ( Y, listY, distance1).
```

7. Преобразувайте програмата така, че да работи със задаван с целта брой царици.

8. Сравнете ефективността на тази програма с ефективността на предишните програми.