Artificial Intelligence course

6th Semester

Bachelor in Informatics and Computer Engineering

Eight-queens problem

Nuno Leite

17 March 2023

Eight-queens problem

- The problem here is to place eight queens on the empty chessboard in such a way that no queen attacks any other queen
- A solution is shown in Figure 4.9
- We will develop a program to solve this puzzle as an unary predicate `solution(Pos)`
  - which is true if and only if `Pos` represents a position with eight queens that do not attack each other

Eight-queens problem

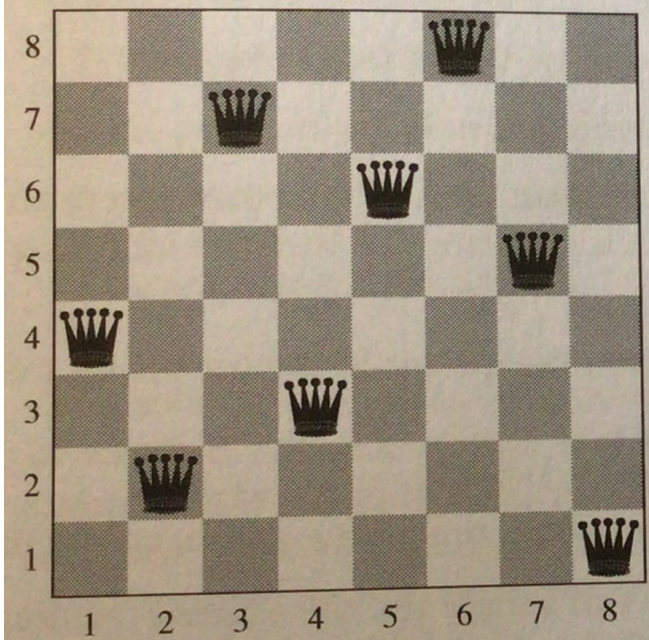

Figure 4.9 A solution to the eight-queens problem. This position can be represented by the list [**1/4, 2/2, 3/7, 4/3, 5/6, 6/8, 7/5, 8/1**]

Program 1

- First, we have to choose a representation of the board position
- One possible representation is by using a list of eight items
  - Each of them corresponding to one queen
  - Hence, each item in the list will specify a square of the board on which the corresponding queen is standing
  - Each square can be specified by a pair of coordinates (X and Y) on the board, where each coordinate is an integer between 1 and 8

Program 1

- We can write such a pair as: `X/Y`
  - Where the `/` operator is not meant to indicate division, but it simply combines both coordinates together into a pair (X, Y) of coordinates, that is, a square
  - The position in Figure 4.9 will thus be represented by the list `[1/4, 2/2, 3/7, 4/3, 5/6, 6/8, 7/5, 8/1]`

Program 1

- The problem is then to find such a list of the form:
  `[X1/Y1, X2/Y2, X3/Y3, ..., X8/Y8]`
  which satisfies the no-attack requirement
- Our procedure `solution` will have to search for a proper instantiation of the variables `X1`, `Y1`, `X2`, `Y2`, ..., `X8`, `Y8`
  - As all the queens will have to be in different columns to prevent vertical attacks, we can fix the X-coordinates so that the solution list will fit the following *template*:
    `[1/Y1, 2/Y2, 3/Y3, ..., 8/Y8]`

Program 1

- The solution predicate can then be formulated by considering two cases:

  Case 1 The list of queens is empty: the empty list is certainly a solution because there is nothing to attack

  Case 2 The list of queens is non-empty: then it looks like this:

  `[X/Y | Others]`

- In case 2, the first queen is at some square `X/Y` and the other queens are at squares specified by the list `Others`

Program 1

- The following conditions must hold:
  1. There must be no attack between the queens in the list `Others`; that is, `Others` itself must also be a solution
  2. `X` and `Y` must be integers between 1 and 8
  3. A queen at square `X/Y` must not attack any of the queens in the list `Others`

Program 1

- We also need a `noattack` relation:

  `noattack(Q, Qlist)`

  Again, this can be broken down into two cases:

  1. If the list `Qlist` is empty then the relation is certainly true because there is no queen to be attacked
  2. If `Qlist` is not empty then it has the form `[Q1 | Qlist1]` and now two conditions must be satisfied:
     - 2.1 The queen at `Q` must not attack the queen at `Q1`, and
     - 2.2 The queen at `Q` must not attack any of the queens in `Qlist1`

Program 1

- To ensure that a queen at some square does not attack another square
  - The two squares must not be in the same row, the same column or the same diagonal
  - Our solution template guarantees that all the queens are in different columns, so it only remains to specify explicitly that:
    - The Y-coordinates of the queens are different, and
    - They are not in the same diagonal, either upward or downward;
    - that is, the absolute distance between the squares in the X-direction must not be equal to the absolute distance in the Y-direction

Eight-queens problem


```
% solution( BoardPosition) if BoardPosition is a list of non-attacking queens

solution( [ ] ).

solution( [X/Y | Others] ) :-
  solution( Others),                    % First queen at X/Y, other queens at Others
  member( Y, [1,2,3,4,5,6,7,8] ),       % Find solution for all other queens
  noattack( X/Y, Others).               % Choose Y coordinate of first queen

noattack( _ , [ ] ).                    % First queen does not attack others

noattack( X/Y, [X1/Y1 | Others] ) :-    % Nothing to attack
  Y =\= Y1,                             % Different Y-coordinates
  Y1 - Y =\= X1 - X,                    % Different upward diagonals
  Y1 - Y =\= X - X1,                    % Different downward diagonals
  noattack( X/Y, Others).

member( Item, [Item | Rest] ).

member( Item, [First | Rest] ) :-
  member( Item, Rest).

% A solution template
template( [1/Y1,2/Y2,3/Y3,4/Y4,5/Y5,6/Y6,7/Y7,8/Y8] ).
```

Figure 4.10 Program 1 for the eight-queens problem.