Trabajo Practico 1: Problema de las 8-Reinas

Daniel Mugica, *P. 87.967*
fiubadaniel@gmail.com

Sergio Matias Piano, *P. 85.191*
smpiano@gmail.com

1er. Cuatrimestre de 2013
75.23 Inteligencia Artificial
Facultad de Ingenieria, Universidad de Buenos Aires

Resumen

Este informe sumariza el desarrollo del trabajo practico 1 de la materia Inteligencia Artificial (75.23) dictada en el primer cuatrimestre de 2013 en la Facultad de Ingenieria de la Universidad de Buenos Aires. El mismo consiste en la construccion de un sistema minimalista que resuelva el conocido problema de las ocho reinas mediante el lenguaje de programación *PROLOG*

Índice

Parte I

Desarrollo

1. Introduccion

El objectivo principal es resolver el problema de las ocho reinas, dicho problema es un pasatiempo en el que se colocan ocho reinas en un tablero de ajedréz sin que se amenacen. En el juego del ajedrez la reina amenaza a aquellas piezas que se encuentren en su misma fila, columna o diagonal.

Principalmente el juego consiste en colocar sobre un tablero de ajedréz ocho reinas sin que estas se amenacen entre ellas. Éste mismo fue propuesto por el ajedrecista alemán Max Bezzel en 1848.

2. Implementación

La implementación coniste en utilizar el lenguaje *Prolog*, pero antes analicemos que necesitamos para una posible solución.

De acuerdo a los movimientos admitidos en el juego del ajedréz, cada reina puede amenazar a todas las reinas que estén en la misma fila, cada una puede situarse en una fila diferente. Tratando el caso a un ejemplo podemos representar las 8 reinas mediante un vector[1-8], teniendo en cuenta que cada índice del vector representa una fila y el valor una columna. Así cada reina estaría en la posición (i, v[i]) para i = 1-8.

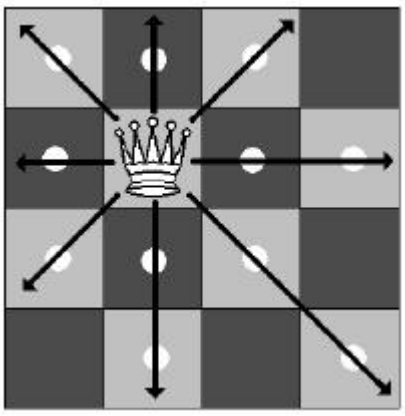

Figura 1: Movimientos posibles de una reina en un juego de Ajedréz

Una solución posible puede escribirse de la siguiente manera con un vector solución (1,5,8,6,3,7,2,4). Éste mismo significa que se pueden ubicar las reinas

en las siguientes posiciones definidas por el binomio (fila, columna)

```
(1, 1) : fila 1, columna 1
(2, 5) : fila 2, columna 5
(3, 8) : fila 3, columna 8
(4, 6) : fila 4, columna 6
(5, 3) : fila 5, columna 3
(6, 7) : fila 6, columna 7
(7, 2) : fila 7, columna 2
(8, 4) : fila 8, columna 4
```

Las reinas no son amenazadas entre ellas mismas con sus movimientos (ver Figura 2).

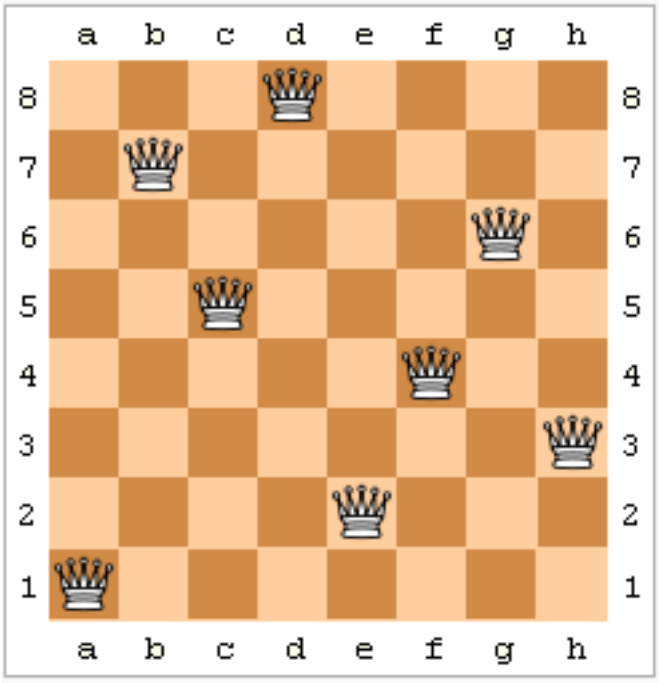

Figura 2: Representación de la solución (1,5,8,6,3,7,2,4)

3. Conclusiones

El problema de las ocho reinas tiene 92 soluciones, de las cuales 12 son esencialmente distintas, es decir las 92 soluciones existentes se pueden obtener a partir de simetrías, rotaciones y traslaciones de las 12 soluciones únicas.

Parte II

Apendice

A. Enunciado original

Trabajo Práctico Número 1

Se debe realizar un sistema de producción, que resuelva el problema de las 8 reinas.

El problema consiste en ubicar en un tablero de ajedrez 8 reinas, de modo que ninguna de ellas ataque a las demás.

El TP se debe realizar en PROLOG, usando un sistema de producción.

Se debe entregar código fuente y la salida.

El programa se deberá ejecutar en las máquinas del laboratorio de la Facultad o en notebook propia.

B. Codigo fuente

source/8reinas.pro

```
1  % Defino el dominio
2  domains
3
4      % Represento a la reina por medio de su posicion en la
           columna
5      reina=integer
6
7      % El tablero es una lista de n elementos con los naturales
           de 1 a n
8      % donde cada elemento especifica la columna que ocupa la
           reina en el
9      % tablero, y la fila se especifica por el orden del elemento
            en la lista.
10     tablero=reina*
11
12
13 % Defino los predicados
14 predicates
15
16     % Devuelve un tablero solucion del problema de la amenaza
           entre las reinas de
17     % un tablero de ajedrez de tamanio dimension [reina].
18     nondeterm nreinas(reina, tablero).
19
20     % Verifica si el tablero solucion es una lista de elementos
           que contiene los naturales
21     % comprendidos entre 1 y dimension del mismo [reina], ambos
           inclusive.
22     nondeterm generarTablero(reina, tablero).
23
24     % Verifica si la primer tabla es una permutacion de los
           elementos de la segunda.
25     nondeterm permutar(tablero, tablero).
26
27     % Indica si existe alguna reina que amenace a otra.
28     nondeterm buenTablero(tablero).
29
30     % Verifica si la reina esta en la primer tabla y
31     % si la segunda tabla es la primer lista sin la reina.
32     nondeterm seleccionar(tablero, reina, tablero).
33
34     % Indica si una reina ubicada en la columna [reina] y fila n
            del tablero amenaza
35     % a las demas reinas del tablero.
36     nondeterm amenaza(reina, tablero).
37     nondeterm amenaza(reina, reina, tablero).
38
39 clauses
40
41     % Genera un tablero de tamanio dimensio
42     n, permuta y obtiene una solucion al problema de las 8
           reinas.
```

```
43      nreinas(Dimension,Solucion):- generarTablero(Dimension,
            Tablero), permutar(Tablero,Solucion), buenTablero(
            Solucion).
44
45      %Caso en que el tablero tenga dimension 0, el tablero es
            vacio.
46      generarTablero(0,[]).
47      %Caso en que el tablero tenga una dimension > 0, genera
            filas y columnas.
48      generarTablero(Dimension,[Dimension|Columnas]):-
            DimensionAnterior = Dimension - 1, DimensionAnterior >=
            0, generarTablero(DimensionAnterior,Columnas).
49
50      %Permutacion de un tablero vacio, es un tablero vacio.
51      permutar([],[]).
52      %Se selecciona el primer valor de la lista y se permuta con
             la cola.
53      permutar(Tablero,[SolCabecera|SolCola]):- seleccionar(
            Tablero,SolCabecera,R), permutar(R,SolCola).
54
55      %Verifica si X es el primer elemento de la primer lista y R
             siendo
56      %la cola de la primer lista es igual a la segunda lista.
57      seleccionar([X|R],X,R).
58      %Verifica si los primeros elementos de las listas son
            iguales y verifica si
59      %las colas de las listas seleccionan a X.
60      seleccionar([C|R],X,[C|Y]):- seleccionar(R,X,Y).
61
62      %Tablero vacio es considerado buen tablero porque no se
            amenazan reinas.
63      buenTablero([]).
64      %Un buen tablero se considera si no hay reina que amenace a
             otras.
65      buenTablero([C|R]):- not(amenaza(C,R)), buenTablero(R).
66
67      %Verifica si X es la cabecera + Profundidad o X es la
            cabecera - Profundidad o X es la cabecera.
68      amenaza(X,Profundidad,[C|_]):- X = C+Profundidad; X = C-
            Profundidad; X = C.
69      %Incrementa la Profundidad y verifica la amenaza sobre la
            reina.
70      amenaza(X,Profundidad,[_|R]):- ProfundidadSiguiente =
            Profundidad + 1, amenaza(X,ProfundidadSiguiente,R).
71
72      %En caso que la lista sea vacia, no hay amenaza.
73      amenaza(_,[]):- fail.
74      %Verifico la amenaza entre las listas a partir de la
            posicion 1.
75      amenaza(X,Y):- amenaza(X,1,Y).
76
77
78  goal
79
```

```
80 % Ejecucion del problema de las 8 reinas.
81 nreinas(8, Solucion).
```

C. Soluciones

Mostrando las 92 Soluciones:

docs/soluciones.txt

```
Solucion =[8 ,4 ,1 ,3 ,6 ,2 ,7 ,5]
Solucion =[8 ,3 ,1 ,6 ,2 ,5 ,7 ,4]
Solucion =[8 ,2 ,5 ,3 ,1 ,7 ,4 ,6]
Solucion =[8 ,2 ,4 ,1 ,7 ,5 ,3 ,6]
Solucion =[7 ,5 ,3 ,1 ,6 ,8 ,2 ,4]
Solucion =[7 ,4 ,2 ,8 ,6 ,1 ,3 ,5]
Solucion =[7 ,4 ,2 ,5 ,8 ,1 ,3 ,6]
Solucion =[7 ,3 ,8 ,2 ,5 ,1 ,6 ,4]
Solucion =[7 ,3 ,1 ,6 ,8 ,5 ,2 ,4]
Solucion =[7 ,2 ,6 ,3 ,1 ,4 ,8 ,5]
Solucion =[7 ,2 ,4 ,1 ,8 ,5 ,3 ,6]
Solucion =[7 ,1 ,3 ,8 ,6 ,4 ,2 ,5]
Solucion =[6 ,8 ,2 ,4 ,1 ,7 ,5 ,3]
Solucion =[6 ,4 ,7 ,1 ,8 ,2 ,5 ,3]
Solucion =[6 ,4 ,7 ,1 ,3 ,5 ,2 ,8]
Solucion =[6 ,4 ,2 ,8 ,5 ,7 ,1 ,3]
Solucion =[6 ,4 ,1 ,5 ,8 ,2 ,7 ,3]
Solucion =[6 ,3 ,7 ,4 ,1 ,8 ,2 ,5]
Solucion =[6 ,3 ,7 ,2 ,8 ,5 ,1 ,4]
Solucion =[6 ,3 ,7 ,2 ,4 ,8 ,1 ,5]
Solucion =[6 ,3 ,5 ,8 ,1 ,4 ,2 ,7]
Solucion =[6 ,3 ,5 ,7 ,1 ,4 ,2 ,8]
Solucion =[6 ,3 ,1 ,8 ,5 ,2 ,4 ,7]
Solucion =[6 ,3 ,1 ,8 ,4 ,2 ,7 ,5]
Solucion =[6 ,3 ,1 ,7 ,5 ,8 ,2 ,4]
Solucion =[6 ,2 ,7 ,1 ,4 ,8 ,5 ,3]
Solucion =[6 ,2 ,7 ,1 ,3 ,5 ,8 ,4]
Solucion =[6 ,1 ,5 ,2 ,8 ,3 ,7 ,4]
Solucion =[5 ,8 ,4 ,1 ,7 ,2 ,6 ,3]
Solucion =[5 ,8 ,4 ,1 ,3 ,6 ,2 ,7]
Solucion =[5 ,7 ,4 ,1 ,3 ,8 ,6 ,2]
Solucion =[5 ,7 ,2 ,6 ,3 ,1 ,8 ,4]
Solucion =[5 ,7 ,2 ,6 ,3 ,1 ,4 ,8]
Solucion =[5 ,7 ,2 ,4 ,8 ,1 ,3 ,6]
Solucion =[5 ,7 ,1 ,4 ,2 ,8 ,6 ,3]
Solucion =[5 ,7 ,1 ,3 ,8 ,6 ,4 ,2]
Solucion =[5 ,3 ,8 ,4 ,7 ,1 ,6 ,2]
Solucion =[5 ,3 ,1 ,7 ,2 ,8 ,6 ,4]
Solucion =[5 ,3 ,1 ,6 ,8 ,2 ,4 ,7]
Solucion =[5 ,2 ,8 ,1 ,4 ,7 ,3 ,6]
Solucion =[5 ,2 ,6 ,1 ,7 ,4 ,8 ,3]
Solucion =[5 ,2 ,4 ,7 ,3 ,8 ,6 ,1]
Solucion =[5 ,2 ,4 ,6 ,8 ,3 ,1 ,7]
Solucion =[5 ,1 ,8 ,6 ,3 ,7 ,2 ,4]
Solucion =[5 ,1 ,8 ,4 ,2 ,7 ,3 ,6]
Solucion =[5 ,1 ,4 ,6 ,8 ,2 ,7 ,3]
Solucion =[4 ,8 ,5 ,3 ,1 ,7 ,2 ,6]
Solucion =[4 ,8 ,1 ,5 ,7 ,2 ,6 ,3]
```

```
Solucion =[4 ,8 ,1 ,3 ,6 ,2 ,7 ,5]
Solucion =[4 ,7 ,5 ,3 ,1 ,6 ,8 ,2]
Solucion =[4 ,7 ,5 ,2 ,6 ,1 ,3 ,8]
Solucion =[4 ,7 ,3 ,8 ,2 ,5 ,1 ,6]
Solucion =[4 ,7 ,1 ,8 ,5 ,2 ,6 ,3]
Solucion =[4 ,6 ,8 ,3 ,1 ,7 ,5 ,2]
Solucion =[4 ,6 ,8 ,2 ,7 ,1 ,3 ,5]
Solucion =[4 ,6 ,1 ,5 ,2 ,8 ,3 ,7]
Solucion =[4 ,2 ,8 ,6 ,1 ,3 ,5 ,7]
Solucion =[4 ,2 ,8 ,5 ,7 ,1 ,3 ,6]
Solucion =[4 ,2 ,7 ,5 ,1 ,8 ,6 ,3]
Solucion =[4 ,2 ,7 ,3 ,6 ,8 ,5 ,1]
Solucion =[4 ,2 ,7 ,3 ,6 ,8 ,1 ,5]
Solucion =[4 ,2 ,5 ,8 ,6 ,1 ,3 ,7]
Solucion =[4 ,1 ,5 ,8 ,6 ,3 ,7 ,2]
Solucion =[4 ,1 ,5 ,8 ,2 ,7 ,3 ,6]
Solucion =[3 ,8 ,4 ,7 ,1 ,6 ,2 ,5]
Solucion =[3 ,7 ,2 ,8 ,6 ,4 ,1 ,5]
Solucion =[3 ,7 ,2 ,8 ,5 ,1 ,4 ,6]
Solucion =[3 ,6 ,8 ,2 ,4 ,1 ,7 ,5]
Solucion =[3 ,6 ,8 ,1 ,5 ,7 ,2 ,4]
Solucion =[3 ,6 ,8 ,1 ,4 ,7 ,5 ,2]
Solucion =[3 ,6 ,4 ,2 ,8 ,5 ,7 ,1]
Solucion =[3 ,6 ,4 ,1 ,8 ,5 ,7 ,2]
Solucion =[3 ,6 ,2 ,7 ,5 ,1 ,8 ,4]
Solucion =[3 ,6 ,2 ,7 ,1 ,4 ,8 ,5]
Solucion =[3 ,6 ,2 ,5 ,8 ,1 ,7 ,4]
Solucion =[3 ,5 ,8 ,4 ,1 ,7 ,2 ,6]
Solucion =[3 ,5 ,7 ,1 ,4 ,2 ,8 ,6]
Solucion =[3 ,5 ,2 ,8 ,6 ,4 ,7 ,1]
Solucion =[3 ,5 ,2 ,8 ,1 ,7 ,4 ,6]
Solucion =[3 ,1 ,7 ,5 ,8 ,2 ,4 ,6]
Solucion =[2 ,8 ,6 ,1 ,3 ,5 ,7 ,4]
Solucion =[2 ,7 ,5 ,8 ,1 ,4 ,6 ,3]
Solucion =[2 ,7 ,3 ,6 ,8 ,5 ,1 ,4]
Solucion =[2 ,6 ,8 ,3 ,1 ,4 ,7 ,5]
Solucion =[2 ,6 ,1 ,7 ,4 ,8 ,3 ,5]
Solucion =[2 ,5 ,7 ,4 ,1 ,8 ,6 ,3]
Solucion =[2 ,5 ,7 ,1 ,3 ,8 ,6 ,4]
Solucion =[2 ,4 ,6 ,8 ,3 ,1 ,7 ,5]
Solucion =[1 ,7 ,5 ,8 ,2 ,4 ,6 ,3]
Solucion =[1 ,7 ,4 ,6 ,8 ,2 ,5 ,3]
Solucion =[1 ,6 ,8 ,3 ,7 ,4 ,2 ,5]
Solucion =[1 ,5 ,8 ,6 ,3 ,7 ,2 ,4]
```