Problema de las 8 Reinas

> Fuente: Ejercios de Lógica Computacional
> (https://www.cs.us.es/~fsancho/?p=logica-informatica-2020-21)

El Problema de las N reinas es un problema clásico dentro de los Problemas de Satisfacción de Restricciones. El problema fue originalmente propuesto en 1848 por el ajedrecista Max Bezzel para el tablero de ajedrez y considerando 8 Reinas. Durante años, muchos matemáticos, incluyendo a Gauss y Cantor, han trabajado en él y lo han generalizado a N-reinas. El problema se enuncia como:

Hallar, si existe, una distribución de N reinas en un tablero de ajedrez $N \times N$ de manera no haya dos reinas que se ataquen entre sí.

En este ejercicio trabajaremos el problema original considerando un tablero de ajedrez clásico 8×8 y 8 reinas a colocar en el mismo. Se pide formalizar el problema y resolviéndolo haciendo uso de fórmulas proposicionales (parametrizadas).

Solución

Cualquiera que conozca el ajedrez sabe que dos reinas se atacan entre sí si ocupan la misma fila, la misma columna o la misma diagonal. Por tanto para que el problema tenga solución se debe cumplir que para cada pareja de reinas se tiene que ambas ocupan filas, columnas y diagonales (principal y secundaria) distintas. Formalicémos el problema haciendo uso de fórmulas parametrizadas y resolvámoslo haciendo uso de LogicUS.

Bien, consideremos el tablero

p_{81}	p_{82}	p_{83}	p_{84}	p_{85}	p_{86}	p_{87}	p_{88}
p_{71}	p_{72}	p_{73}	p_{74}	p_{75}	p_{76}	p_{77}	p_{78}
p_{61}	p_{62}	p_{63}	p_{64}	p_{65}	p_{66}	p_{67}	p_{68}
p_{51}	p_{52}	p_{53}	p_{54}	p_{55}	p_{56}	p_{57}	p_{58}
p_{41}	p_{42}	p_{43}	p_{44}	p_{45}	p_{46}	p_{47}	p_{48}
p_{31}	p_{32}	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}	p_{38}
p_{21}	p_{22}	p_{23}	p_{24}	p_{25}	p_{26}	p_{27}	p_{28}

p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}	p_{17}	p_{18}

```
import LogicUS.PL.CSP exposing (..)
import LogicUS.PL.SintaxSemantics exposing (..)

bfplRead : String -> BigFPL
bfplRead = bfplReadExtraction << bfplReadFromString
```

Bien, ténganse las variables proposicionales $P_{i,j}$ con $i, j \in \{1, \ldots, 8\}$ tal que $P_{i,j}$ vale 1 si una reina está colocada en la posición $p_{i,j}$.

Entonces las restricciones pueden modelarse como:

- Dos reinas no ocupan una misma fila. Esto es en toda fila se cumple que si una casilla de la fila tiene una reina entonces las demás no pueden tener reina:

```
f1 : BigFPL
f1 = bfplRead "!&[i(1:8), j(1:8)]{T} (P_{i,j} -> !&[k(1:8)]{[k!=j]}(¬P_{i,k})))"
```

$$\bigwedge_{\substack{i \in \{1..8\} \\ j \in \{1..8\}}} \left(P_{i,j} \rightarrow \bigwedge_{\substack{k \in \{1..8\} \\ [k \neq j]}} \neg P_{i,k} \right)$$

- Dos reinas no ocupan una misma columna. Esto es en toda columna se cumple que si una casilla de la columna tiene una reina entonces las demás no pueden tener reina:

```
f2 : BigFPL
f2 = bfplRead "!&[j(1:8)]{T} (!&[i(1:8)]{T} (P_{i,j} -> !&[k(1:8)]{[k!=i]}(¬P_{k,j})))"
```

$$\bigwedge_{j \in \{1..8\}} \bigwedge_{i \in \{1..8\}} \left(P_{i,j} \rightarrow \bigwedge_{\substack{k \in \{1..8\} \\ [k \neq i]}} \neg P_{k,j} \right)$$

- Dos reinas no ocupan una misma diagonal principal. Esto es en toda casilla se cumple que si esa casilla contiene una reina entonces las demás casillas de esa diagonal principal no pueden tener una reina:

```
f3 : BigFPL
f3 = bfplRead "!&[i(1:8)]{T} (!&[j(1:8)]{T} (P_{i,j} -> !&[k(-7:7)]{[k!=0]AND[i+k>0]AND[
```

$$\bigwedge_{i \in \{1..8\}} \bigwedge_{j \in \{1..8\}} \left(P_{i,j} \rightarrow \bigwedge_{\substack{k \in \{-7..7\} \\ s.t.\, \theta_1}} \neg P_{(i+k),(j+k)} \right)$$

$$\theta_1 \equiv \Big(\big[k \neq 0\big] \wedge \Big(\big[(i+k) > 0\big] \wedge \Big(\big[(i+k) \leq 8\big] \wedge \Big(\big[(j+k) > 0\big] \wedge \big[(j+k) \leq 8\big] \Big) \Big) \Big) \Big)$$

- Dos reinas no ocupan una misma diagonal principal. Esto es en toda casilla se cumple que si esa casilla contiene una reina entonces las demás casillas de esa diagonal principal no pueden tener una reina:

```
f4 : BigFPL
f4 = bfplRead "!&[i(1:8)]{T} (!&[j(1:8)]{T} (P_{i,j} -> !&[k(-7:7)]{[k!=0]AND[i+k>0]AND[
```

$$\bigwedge_{i \in \{1..8\}} \bigwedge_{j \in \{1..8\}} \left(P_{i,j} \rightarrow \bigwedge_{\substack{k \in \{-7..7\} \\ s.t.\, \theta_1}} \neg P_{(i+k),(j-k)} \right)$$

$$\theta_1 \equiv \Big(\big[k \neq 0\big] \wedge \Big(\big[(i+k) > 0\big] \wedge \Big(\big[(i+k) \leq 8\big] \wedge \Big(\big[(j-k) > 0\big] \wedge \big[(j-k) \leq 8\big] \Big) \Big) \Big) \Big)$$

- Han de colocarse 8 reinas. Esto es equivalente a decir que en cada fila ha de haber una reina.

```
f5 : BigFPL
f5 = bfplRead "!&[i(1:8)]{T} (![j(1:8)]{T} (P_{i,j}))"
```

$$\bigwedge_{i \in \{1..8\}} \bigvee_{j \in \{1..8\}} P_{i,j}$$

Resolvamos el problema. Esto es comprobemos si el conjunto es satisfactible y si lo es entonces encontremos un modelo. Esto es precisamente lo que posibilita la función `sbfplsolver` .

```
sol : (Bool, Interpretation)
sol = sbfplsolver [f1,f2,f3,f4,f5]
```

(True,[("P",[6,7]),("P",[1,4]),("P",[5,1]),("P",[2,8]),("P",[3,5]),("P",[7,2]),("P",[4,3]),("P",[8,6])])

¿Es satisfactible? True

Un modelo: $\{P_{6,7}, P_{1,4}, P_{5,1}, P_{2,8}, P_{3,5}, P_{7,2}, P_{4,3}, P_{8,6}\}$

Que correspondería en el tablero a:

p_{81}	p_{82}	p_{83}	p_{84}	p_{85}	p_{86}	p_{87}	p_{88}
p_{71}	p_{72}	p_{73}	p_{74}	p_{75}	p_{76}	p_{77}	p_{78}
p_{61}	p_{62}	p_{63}	p_{64}	p_{65}	p_{66}	p_{67}	p_{68}
p_{51}	p_{52}	p_{53}	p_{54}	p_{55}	p_{56}	p_{57}	p_{58}
p_{41}	p_{42}	p_{43}	p_{44}	p_{45}	p_{46}	p_{47}	p_{48}
p_{31}	p_{32}	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}	p_{38}
p_{21}	p_{22}	p_{23}	p_{24}	p_{25}	p_{26}	p_{27}	p_{28}
p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}	p_{17}	p_{18}