# Programación con Restricciones - Práctica 5 – Hitori

En el pasatiempo Hitori se parte de un tablero cuadrado repleto de dígitos

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4 | 5 | 6 | 8 | 3 | 7 | 8 | 1 |
| 1 | 1 | 1 | 4 | 8 | 6 | 7 | 2 |
| 5 | 4 | 7 | 6 | 3 | 1 | 2 | 8 |
| 7 | 4 | 5 | 3 | 2 | 8 | 1 | 6 |
| 2 | 2 | 2 | 7 | 1 | 4 | 4 | 4 |
| 8 | 7 | 4 | 5 | 5 | 2 | 6 | 3 |
| 6 | 7 | 8 | 5 | 5 | 4 | 3 | 2 |
| 3 | 3 | 3 | 2 | 4 | 6 | 1 | 7 |

El objetivo es “tachar” dígitos hasta conseguir:

1.- Que cada fila y columna esté formada por dígitos distintos
2.- Que no haya dos tachones adyacentes
3.- Que cada dígito tenga otro adyacente

Aquí “adyacente” significa, a la izquierda, a la derecha, arriba o abajo. Estar en diagonal no se considera adyacente.

En el ejemplo anterior una solución sería:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4 | 5 | 6 | **8** | 3 | **7** | 8 | 1 |
| **1** | 1 | **1** | 4 | 8 | 6 | **7** | 2 |
| 5 | **4** | 7 | 6 | **3** | 1 | 2 | 8 |
| 7 | 4 | 5 | **3** | 2 | 8 | **1** | 6 |
| **2** | 2 | **2** | 7 | 1 | **4** | 4 | **4** |
| 8 | **7** | 4 | **5** | 5 | 2 | 6 | 3 |
| 6 | 7 | 8 | 5 | **5** | 4 | 3 | **2** |
| **3** | 3 | **3** | 2 | 4 | **6** | 1 | 7 |

Nota: en la realidad existe una condición adicional: cada dígito debe ser accesible desde todos los demás por un camino de dígitos adyacentes, pero no lo tenemos en cuenta aquí.

Vamos a resolver el problema partiendo de los ficheros 5.mzn y 5.dzn del campus.
El array de  variables t utiliza el valor 0 para indicar un tachón.

Para ello se pide:

1) Escribir un predicado con esta cabecera:

predicate inBounds(array[int,int] of var int:t, var int:i, var int: j)

que establece si las coordenadas i,j son válidas en el array t.

**Nota**: para obtener los conjuntos de índices válidos de t for filas ypor columnas usar index_set_1of2(t) /\ j in index_set_2of2(t) (ver http://www.minizinc.org/doc-lib/doc-builtins-array.html para un listado completo de funciones válidas sobre arrays).

2) Escribir un predicado

predicate nohayzeroalrededor(array[int,int] of var int:t, var int:i, var int:j)

que asegure que la posición i,j no tiene ceros adyacentes (ni a la izquierda, ni a la derecha, ni arriba ni abajo).

Nota:

a) Se puede asumir que i,j es una posición válida para t, pero hay que tener cuidado al comprobar las posiciones adyacente y usar inBounds.

b) Aún así nos saldrán warnings al ejecutar el modelo, podeos ignorarlos.

3) Escribir un predicado

predicate haynozeroalrededor(array[int,int] of var int:t,var int:i, var int:j)

que asegure que la posición i,j tiene adyacente al menos un valor distinto de cero.

4) Escribir un predicado

predicate copiadeh(array[int,int] of var int:t)

constraint que asegure que para toda coordenada de t el valor correspondiente verifica que si es distinto de cero coincide con el valor de h en la misma coordenada (t es una copia de h, excepto por los ceros que representan los valroes tachados).

5) Escribir un predicado

predicate cerobien(array[int,int] of var int:t)

que asegure que toda posición de t que contenga un cero no tiene ningún cero alrededor (usar el predicado de 2)

6) Escribir una restricción que asegure que toda posición de que contenga un valor distinto de cero tenga otro valor distinto de cero alrededor (usar el predicado de 3)

predicate nocerobien(array[int,int] of var int:t)

7) Escribir un predicado

predicate diferenteporfilas(array[int,int] of var int:t)

que asegura que para las filas de t tienen valores distintos excluyendo los valores a cero.

8) Escribir un predicado

predicate diferenteporcolumnas(array[int,int] of var int:t)

que asegura que para las columnas de t tienen valores distintos excluyendo los valores a cero.

9) Escribir constraints llamando a los predicados 4,5,6,7,8 para resolver el problema

10) Escribir un output que muestre el array t en "bonito", es decir algo así:

4 5 6 8 0 7 0 1
0 1 0 4 8 6 7 0
5 0 7 6 3 0 2 8
7 4 5 0 2 8 0 6
0 2 0 7 1 0 4 0
8 0 4 5 0 2 6 3
6 7 8 0 5 4 0 2
0 3 0 2 4 0 1 0

Notas finales: Incluir el nombre de los integrantes del grupo con comentarios al principio (%). Subir un solo fichero .mzn al campus.