Práctica 7: circuitos combinacionales

1.- Puertas lógicas

Puntuación: 8/10.
Calendario y puntuación máxima: 14/04/16 a las 14:50h: 8 puntos. Hasta el 21/04/16 a las 13h: 7 puntos.

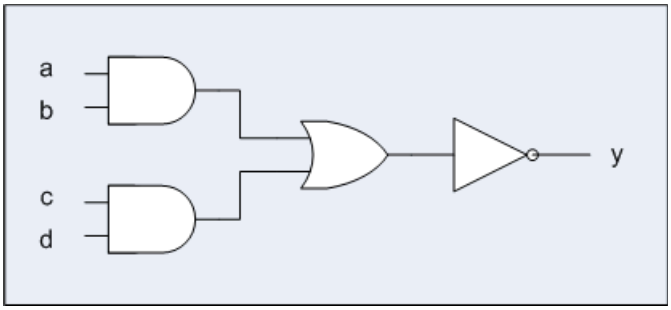
Queremos representar puertas lógicas para construir circuitos combinacionales. Para no mezclar el concepto de entrada con las restricciones sobre booleanos en MiniZinc vamos a suponer que los valores booleanos de los circuitos son enteros en el rango 0..1.

Escribir el código de los siguientes predicados:

Se debe escribir un predicado por cada una de las operaciones. Todos los predicados se encontrarán en un fichero, por ejemplo "puertas.mzn":

```
% r es a AND b
predicate and(var 0..1:a, var 0..1: b, var 0..1: r) =

% r es  a OR b
predicate or(var 0..1:a, var 0..1: b, var 0..1: r) =

% r es la negación de a
predicate neg(var 0..1:a,  var 0..1: r) =

% r es 0
predicate cero(var 0..1: r) =

% r es 1
predicate uno(var 0..1: r) =

% r es a
predicate cable(var 0..1:a,  var 0..1: r) =

% construir nor a partir de las anteriores
predicate nor(var 0..1:a, var 0..1: b, var 0..1: r) =

% construir nand a partir de las anteriores
predicate nand(var 0..1:a, var 0..1: b, var 0..1: r) =


....
```

Escribir un programa en MiniZinc que utilizando estos predicados y las variables de decisión necesarias represente el siguiente circuito combinacional



La llamada minizinc -a programa.mzn debe mostrar la tabla de verdad del circuito

2.- Circuitos combinacionales

Ahora se trata de encontrar circuitos que cumplan una determinada tabla de verdad. Se trata de escribir un predicado *circuito* tal que reciba como entrada un array v de 2^m representando la tabla de verdad de un circuito con m posibles entradas , y un array de la forma:

array [1..n,1..4] of var int:t;

representando un circuito de n es el número de puertas y donde cada t[i, _] representa una puerta del circuito, sabiendo que:

- t[i,1] indica el tipo de puerta, siguiendo la siguiente tabla: 0->cero, 1->uno, 2->neg, , 3->cable, 4->and, 5->or, 6->nand, 7->nor
- t[i,2] es la posición de la puerta cuya salida se toma la primera entrada, o 0 si no tiene sentido (caso de puertas tipo cero, uno)
- t[i,3] es la posición de la puerta cuya salida se toma la segunda entrada, o 0 si no tiene sentido (caso de puertas tipo cero, uno, neg, cable)
- t[i,4] representa la salida de la puerta.

Para que el circuito sea combinacional se debe asegurar que t[i,2] < i, t[i,3]<i.

Se debe cumplir que al hacer t[1,1] = v1, ... t[m,1] = vm , con v1...vm o bien 0 o bien 1, se cumpla que t[n,4] = v[v1,...,vm]

Puntuación: 2/10.
Calendario y puntuación máxima: el 14/04/16 a las 14:50h->2, hasta el 21/04/16 a las 13h ->1
