# Pràctica de PDA, Constraints i SAT (2021-2022)

Data màxima de lliurament Moodle: 16 de Gener, lliurament únic.

Entrevista de lliurament obligatòria: a concretar amb cada grup. Setmana 17-21 de Gener.

Aquesta és la segona part de la pràctica que consiteix a resoldre un problema combinatori amb SAT. Aquesta part contarà un 35% de la nota de pràctiques de constraints (per tant la de MiniZinc conta un 65%). La pràctica es realitzarà per parelles.

## Part SAT (Binary Sudoku), fins a 3.5 punts

En aquesta part es demana que amplieu l'objecte ScalAT amb els mètodes per a codificar cardinality constraints que es descriuran a continuació per a poder donar un model i resoldre el problema *Binary Sudoku* (descrit a més envall).

### Cardinality Constraints (fins a 1 punt)

Implementeu dins de la llibreria ScalAT les següents codificacions de constraints:

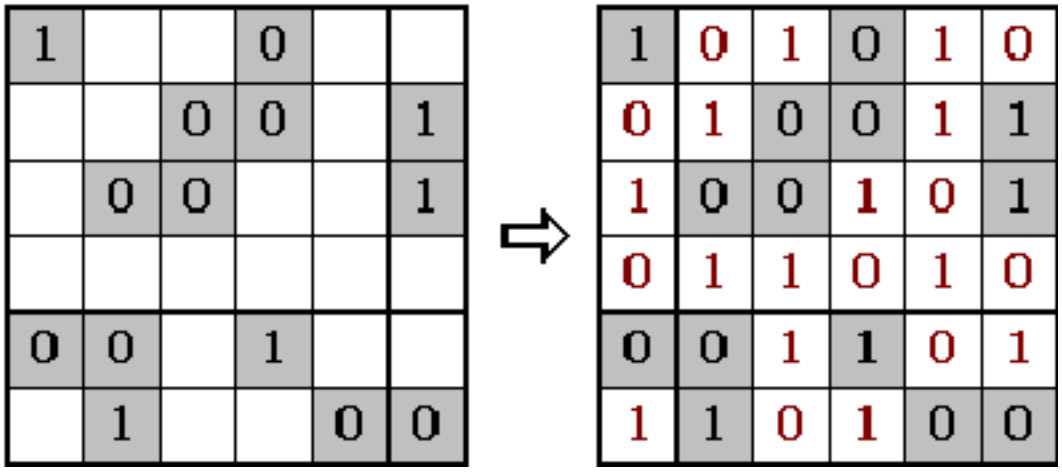
- el mètode `addAMOLog(x)` que codifiqui una restricció de tipus *At-Most-One* a una llista de variables Booleanes `x`, és a dir $(x[0]+x[1]+x[2]+\ldots+x[size(x)-1] \leq 1)$, amb l'encoding logarítmic.
- El mètode `addAMK(x,K)` que codifiqui una restricció de tipus *At-Most-K* a una llista de variables Booleanes `x` amb un enter positiu $K$, és a dir $(x[0]+x[1]+x[2]+\ldots+x[size(x)-1] \leq K)$, fent servir *sorting networks*. La implementació de la sorting network ja ve donada a ScalAT.
- El mètode `addALK(x,K)` que codifiqui una restricció de tipus *At-Least-K* a una llista de variables Booleanes `x` amb un enter positiu $K$, és a dir $(x[0]+x[1]+x[2]+\ldots+x[size(x)-1] \geq K)$, fent servir *sorting networks*.
- El mètode `addEK(x,K)` que codifiqui una restricció de tipus *Exactly-K* a una llista de variables Booleanes `x` amb un enter positiu $K$, és a dir $(x[0]+x[1]+x[2]+\ldots+x[size(x)-1] = K)$, fent servir *sorting networks*.

### Binary Sudoku (fins a 2.5 punts)

Creeu un fitxer `Bsudoku.scala` que llegeixi instàncies del problema i les resolgui. Per fer això us caldrà codificar i resoldre amb ScalAT el problema del *Binary Sudoku* (o *tauler atapeït*). Aquest problema consisteix en acabar d'omplir de 0s i 1s una quadricula parcialment plena d'acord amb les següents regles:

- a cada fila i a cada columna hi ha d'haver el mateix nombre d'uns que de zeros
- no hi pot haver 3 uns o 3 zeros seguits (ni per files ni per columnes)
- no pot haver-hi dues files iguals ni dues columnes iguals

Per llegir les instàncies, llegiu fitxers de text on per casella indeterminada hi hagi un `.`, per a zeros, un `0` i per a uns un `1`. Per exemple, pel següent sudoku binary (amb la solció a la dreta):




el fitxer que haurieu de llegir com a entrada hauria de ser:

```
1..0..
..00.1
.00..1
......
00.1..
.1..00
```

### Què es demana

Caldrà que lliureu un enllaç al vostre Github privat, i m'hi dongueu accés, on hi haurà d'haver l'ScalAT ampliat i un objecte `Bsudoku` (podeu fer servir el `Queens` com a llavor) que codifiqui i resolgui el *sudoku binary* usant ScalAT.
Caldrà també lliurar (el podeu posar al Github) un informe PDF (fet amb LaTeX):

1. Doneu i expliqueu el vostre model: viewpoint, restriccions bàsiques, restriccions implicades que hagueu trobat, trencament de simetries, etc.

2. Òbviament fareu servir cardinality constraints. Cal que proveu les dues configuracions següents del vostre model:

   **Conf1** feu servir només codificacions basades amb sorter.

   **Conf2** Feu servir només codificacions basades en pseudobooleans.

   Considerant la instància del tauler de $6 \times 6$ que hi ha en aquest enunciat a tall d'exemple,

   - Quines diferències observeu entre les dues configuracions pel que fa a la mida?
   - Quina de les dues configuracions és més ràpida? Per què creieu que passa?

3. Mireu de resoldre tantes instàncies com pogueu i quant més grosses millor:

   `https://www.binarypuzzle.com/index.php`

   podeu fer servir un *timeout* d'una hora. Doneu una taula de temps per instància. A la taula s'hi ha de poder veure què us ha suposat de millora de temps les millores que hagueu proposat al model (és a dir, feu columnes amb els temps per la codificació bàsica, per la codificació amb restriccions implicades, etc)

**Extres**

A més a més, es **donarà 0,5 punts extra** al grup que sigui capaç de resoldre més instàncies amb menor temps d'execució.