

SAT Solver : Picross

Alexandre Kirchmeyer - Dorian Biichlé - Benjamin Loison -
Léo Mangel - Florent Pollet

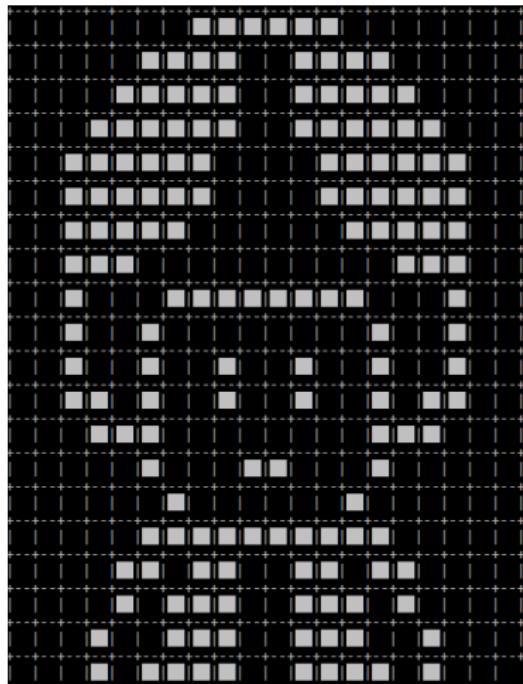
Mathinfo

6 octobre 2020



Introduction

Laissez-vous guider !



- C'est amusant !
- C'est parti !

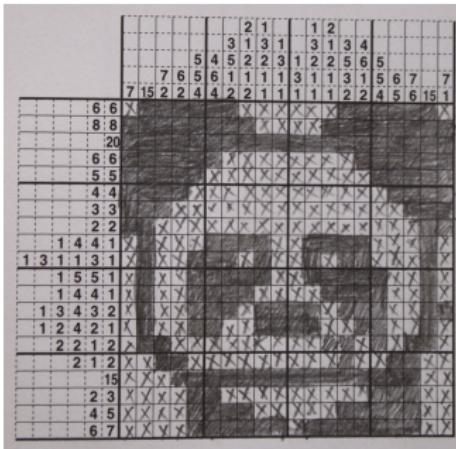
Un joli puzzle

Plan

- 1 Introduction
- 2 Le jeu
- 3 Modélisation
- 4 Complexité
- 5 Performances
- 6 Démonstration du programme
- 7 Conclusion

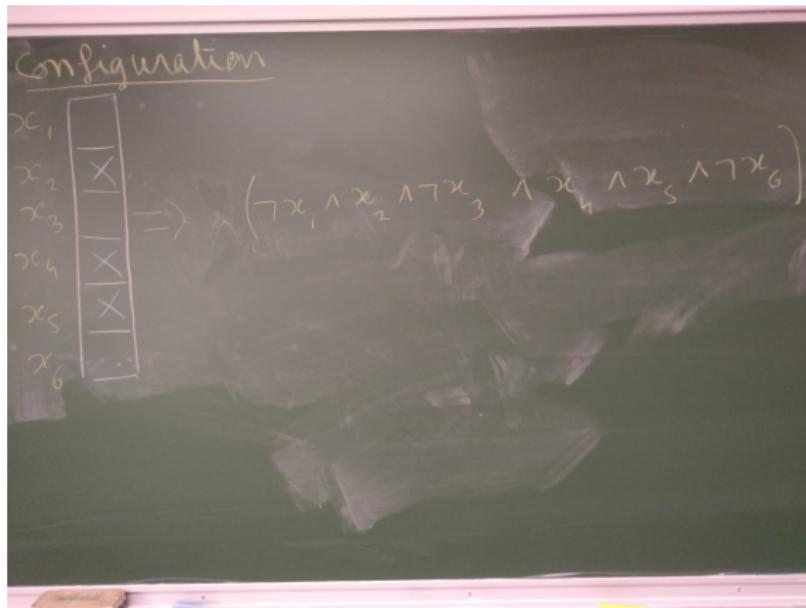
Le jeu

- Un puzzle = un hanjie = un nonogram
- 2 couleurs
- Contraintes sur les lignes et les colonnes : blocs de la longueur indiquée, dans l'ordre, séparé par au moins une case de l'autre couleur
- Une ou plusieurs solutions (selon les contraintes)



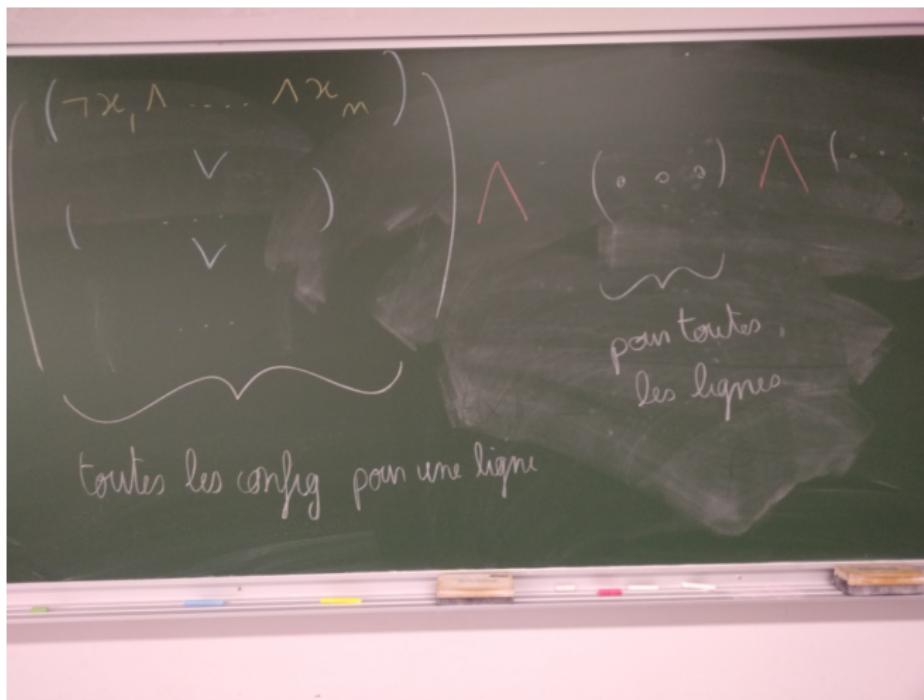
Modélisation - Premier modèle

- Modèle naïf : une variable par case, génération de toutes les possibilités.
- Utilisation de sympy pour le passage en CNF.



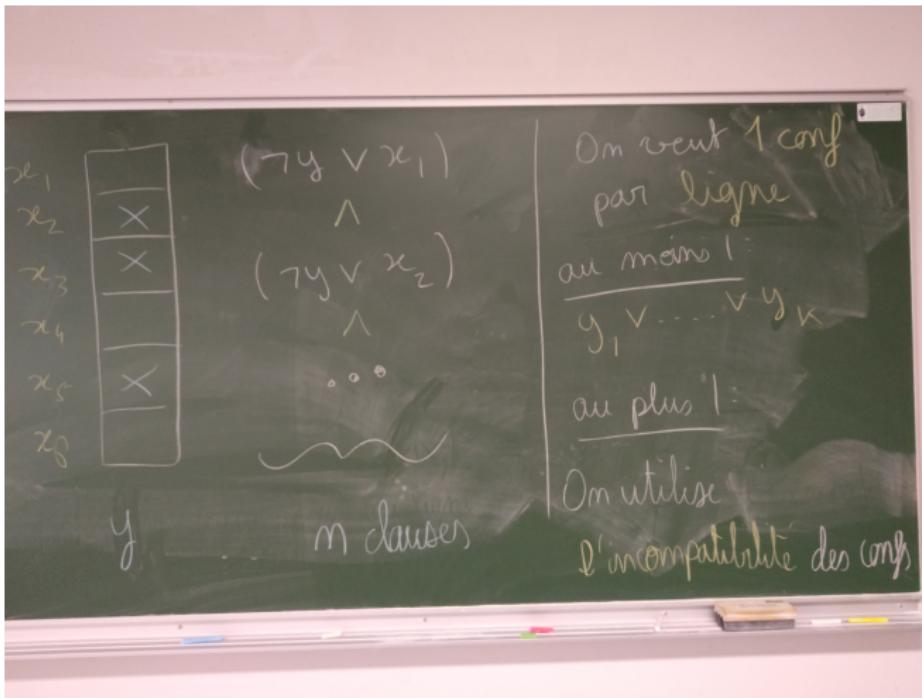
Ecriture des conditions sous forme logique.

Modélisation - Premier modèle



Bruteforce des familles.

Modélisation - Second modèle



Utilisation de variables secondaires

Complexité

• Si $\text{cond} = (k_1, \dots, k_p)$,

$$m(\text{config}) = \binom{m - \sum k_i}{p}$$

• 2^m lignes de colonnes

• Par ligne, au plus $\binom{m+1-p}{p} \leq \binom{m}{\lceil \frac{m}{2} \rceil}$

• D'après la formule de Stirling,

$$\Rightarrow O\left(\frac{2^m}{\sqrt{m}}\right)$$

• Nombre de clauses par config : m

• On veut au moins une config : $+1$

• D'où :

$$m(\text{clauses}) \leq (m+1) \left[O\left(\frac{2^m}{\sqrt{m}}\right) \times m + 1 \right] = O\left(m\sqrt{m}2^m\right)$$

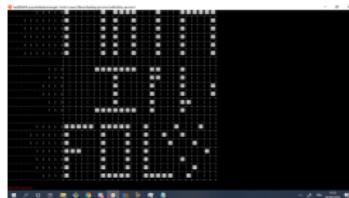
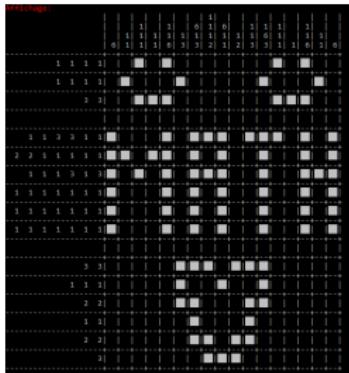
Performances

Utilisation de boardgen

i	Size	Generation time	Glucose time	Density
1	20	1.115s	0.965s	0.5
2	20	1.156s	2.919s	0.475
3	20	10.416s	2.33	0.45
4	20	7.239s	2.7s	0.425
5	20	26.811s	10.103s	0.4
6	30	15.58s	73.094s	0.5
7	30	10.81s	17.215s	0.5
8	33	641.249s	1861.86s	0.5

Démonstration du programme

- Fonctionnement avec pypy utilisation de boardgen.py
- Cf à la fin pour le voir en direct !



Solution non unique (n et y)

Quelque chose de formidable

Démonstration du programme

- Détection des puzzles sans solutions



Un exemple

```
1 $1
2 6
3 1 1
4 1 1 1
5 1 1
6 1 1 6
7 1 3
8 6 1 3
9 1 1 1 2
10 6 1 1
11 1 2
12 1 1 3
13 1 6 3
14 1 1 1
15 1
16 1 1 6
17 1 1
18 6
19 1 1 1 1
20 1 1 1 1
21 3 3
22
23 1 1 3 3 1 1
24 2 2 1 1 1 1 1
25 1 1 1 3 1 3
26 1 1 1 1 1 1 1
27 1 1 1 1 1 1 1
28 1 1 1 1 1 1 1
29
30 3 3
31 1 1 1
32 2 2
33 1 1
34 2 2
35 3
```

Le code

Conclusion

- Recherche très intéressante
- Des progrès sont encore possibles :

B. Nonogram Solver

Rank	Program Name	Solved	Time
1	LalaFrogKK	1000	645.544s
2	Yayagram	766	4989.594s
3	Naughty	509	2325.271s

Nonograms 25x25

- Merci pour votre attention !