

Introduction à la programmation par contraintes

Journées d'ouvertures professionnelles HERS 2018

Pierre Talbot
(pierre.talbot@univ-nantes.fr)

Université de Nantes, laboratoire LS2N, équipe TASC

30 Novembre 2018



UNIVERSITÉ DE NANTES

Programmation par contraintes

Paradigme de programmation

- ▶ Vous êtes habitué au paradigme impératif (C, C++, Java, ...),
- ▶ peut-être objet (C++, Java, ...),
- ▶ voir fonctionnel (OCaml, Haskell, Scala, ...) ?

Aujourd'hui, on va voir le paradigme de la programmation par contraintes !

Programmation par contraintes

Paradigme déclaratif, surnommé "holy grail of computing" : on déclare notre problème et laisse l'ordinateur le résoudre pour nous.



Paradigme reconnu

Applications

La PPC a beaucoup d'applications du solveur de Sudoku, planification, packing, composition musicale...



Intuitions

Les problèmes de contraintes sont *combinatoires* et généralement *NP-complet*, très long voir impossible à résoudre sans optimisation.

Comment ça marche ?

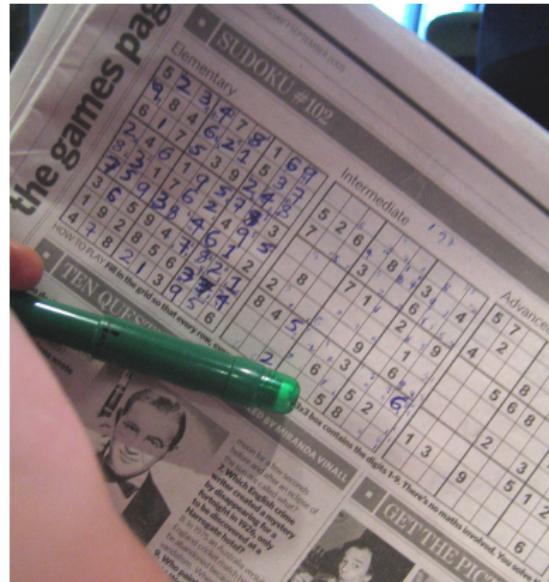
On déclare un ensemble de variables et pose des contraintes/relations sur ces variables de tel sorte qu'on obtient une solution si toutes les contraintes sont vérifiées.

```
var x: 0..10;  
var y: 0..10;  
constraint x > y;
```

Menu

- ▶ Introduction
- ▶ Modélisation par contraintes avec MiniZinc
- ▶ Comment ça marche
- ▶ Autres exemples de problèmes par contraintes
- ▶ Conclusion

Premier exemple : Sudoku



Comment trouver les nombres manquants par un algorithme ? Difficile ?

Représentation mathématique

Soit une matrice M de taille 9×9 où $1 \leq M_{I,c} \leq 9$ et I est le numéro de la ligne et c celui de la colonne, on veut :

- ▶ Lignes différentes : $\forall I, c, c'. M_{I,c} \neq M_{I,c'}$ tel que $c \neq c'$.
- ▶ Colonnes différentes : $\forall I, I', c. M_{I,c} \neq M_{I',c}$ tel que $I \neq I'$.
- ▶ Sous-carré différents :
 $\forall I, c \in \{1, 4, 7\}. \forall sl, sc, sl', sc' \in \mathbb{Z}_3. M_{I+sl, c+sc} \neq M_{I+sl', c+sc'}$ tel que $sl \neq sl' \wedge sc \neq sc'$.

Représentation informatique

Modèle proche de la définition mathématique.

Pour résoudre : solve satisfy;

```
1 include "alldifferent.mzn";
2 int: N = 9;
3 array[1..N,1..N] of var 1..N: sudoku;
4 constraint forall(l in 1..N)
5   (alldifferent([sudoku[l,c] | c in 1..N]));
6 constraint forall(c in 1..N)
7   (alldifferent([sudoku[l,c] | l in 1..N]));
8 constraint forall(l, c in {1,4,7})
9   (alldifferent([sudoku[l + sl, c + sc] | sl,sc in 0..2]));
10
11 solve satisfy;
12 output [show2d(sudoku)];
```

Input et Output

Une grille de Sudoku en entrée où les chiffres inconnus sont _.
Le système nous répond avec la grille pleine.

```
14 sudoku=[ |  
15 _, _, _, _, _, _, _, _, _ |  
16 _, 6, 8, 4, _, 1, _, 7, _ |  
17 _, _, _, _, 8, 5, _, 3, _ |  
18 _, 2, 6, 8, _, 9, _, 4, _ |  
19 _, _, 7, _, _, _, 9, _, _ |  
20 _, 5, _, 1, _, 6, 3, 2, _ |  
21 _, 4, _, 6, 1, _, _, _, _ |  
22 _, 3, _, 2, _, 7, 6, 9, _ |  
23 _, _, _, _, _, _, _, _, _ |  
24 ];
```

Compiling Sudoku.mzn

Running Sudoku.mzn

```
[| 5, 9, 3, 7, 6, 2, 8, 1, 4  
2, 6, 8, 4, 3, 1, 5, 7, 9  
7, 1, 4, 9, 8, 5, 2, 3, 6  
3, 2, 6, 8, 5, 9, 1, 4, 7  
1, 8, 7, 3, 2, 4, 9, 6, 5  
4, 5, 9, 1, 7, 6, 3, 2, 8  
9, 4, 2, 6, 1, 8, 7, 5, 3  
8, 3, 5, 2, 4, 7, 6, 9, 1  
6, 7, 1, 5, 9, 3, 4, 8, 2 |]
```

Finished in 8msec

Démo live.

Quels outils pour résoudre un problème de contraintes ?

- ▶ *Librairies* : GeCode (C++), Choco (Java), ...
- ▶ *Langages* : Prolog, **MiniZinc**, ...

Pourquoi MiniZinc ?

- ▶ Facilité : Syntaxe proche du raisonnement mathématique.
- ▶ Modulaire : Compile le code vers différents solveurs (GeCode, Choco, ...).
- ▶ Encore plus modulaire : Compile vers différents "sous-paradigmes" (CP, ILP, MIP, ...). Avec le même code !

Déclarer des variables

```
int: n = 9; % Parametre  
var 1..9: y; % Variable
```

Deux types de variables

- ▶ *Paramètre* : La variable `n` est un paramètre qui est fixé avant l'exécution et ne peut prendre qu'une seule valeur. Dans le problème du Sudoku, c'est la taille de la matrice.
- ▶ *Variables de décision* : La variable `y` prendra une valeur entre 1 et 9 à la fin de l'exécution. Dans le Sudoku, c'est par exemple une case de la matrice.

Types des variables

`int, bool, float, string, set and array.`

Ajouter des contraintes

Une contrainte est une relation bidirectionnelle sur les variables. Exemple :

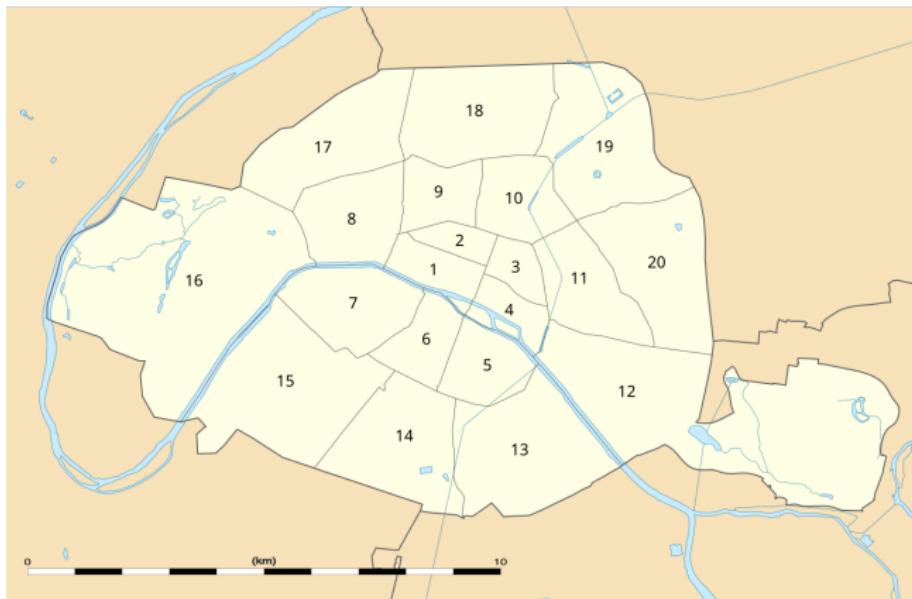
```
var 1..10: x;  
var 1..10: y;  
int: n = 4;  
constraint x = y + n;
```

Si x change, ça impact y qui change pour respecter l'égalité, et inversement.

- ▶ Contraintes arithmétiques : $x < y$ (les classiques : $<, \leq, >, \geq, =$)
- ▶ Expressions arithmétiques : $y - z = x + 3$ ($- , + , * , / , \text{mod} , \text{div}$).
- ▶ Contraintes booléennes : $x \neq y \vee y = 0$ ($\vee , \wedge , \neg , \leftarrow\rightarrow , \text{not}$).

Exemple : Colorier une carte

Selon la carte suivante, trouver un coloriage à 3 couleurs des arrondissements 3 à 5 et 11,12 tel que 2 arrondissements adjacents n'ont pas la même couleur.



Solution : Colorier une carte

```
int: nc = 3;

var 1..nc: arr3;
var 1..nc: arr4;
var 1..nc: arr5;
var 1..nc: arr11;
var 1..nc: arr12;

constraint arr3 != arr4;
constraint arr3 != arr11;
constraint arr4 != arr11;
constraint arr4 != arr12;
constraint arr4 != arr5;
constraint arr11 != arr12;

solve satisfy;
output ["arr3 = \$(arr3)\t arr4 = \$(arr4)\t arr5 = \$(arr5)\n",
       "arr11 = \$(arr11)\t arr12 = \$(arr12)"];
```

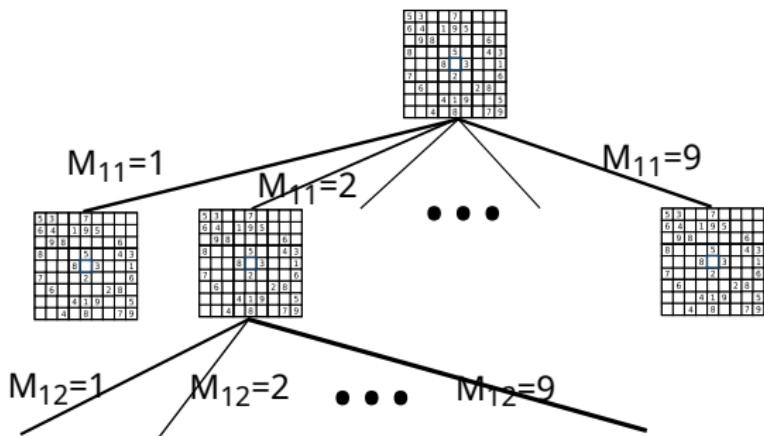
Menu

- ▶ Introduction
- ▶ Modélisation par contraintes avec MiniZinc
- ▶ Comment ça marche
- ▶ Autres exemples de problèmes par contraintes
- ▶ Conclusion

Comment trouver une solution ?

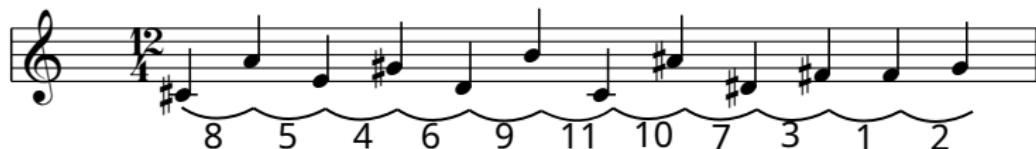
Nature NP-complete

- ▶ Essayer toutes les combinaisons jusqu'à ce qu'on trouve une solution.
- ▶ Algorithme de *backtracking* construisant un arbre de recherche.



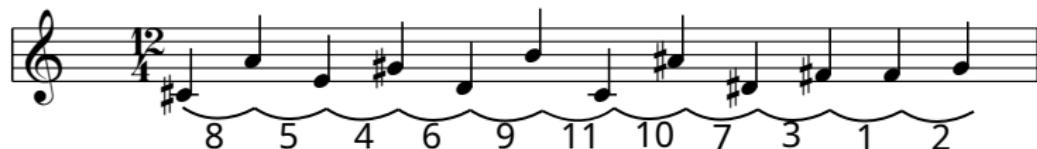
Un exemple de problème de contraintes

Pour une série de 12 notes, chaque note et tout intervalle entre deux notes successives doit être distincts.



Un exemple de problème de contraintes

Pour une série de 12 notes, chaque note et tout intervalle entre deux notes successives doit être distincts.



- ▶ On donne juste les relations entre les données qui nous intéressent.
- ▶ Mais on ne dit pas **comment** on arrive à la solution.

Modélisation des séries tous intervalles

Pour une série de 12 notes, chaque note et tout intervalle entre deux notes successives doit être distincts.

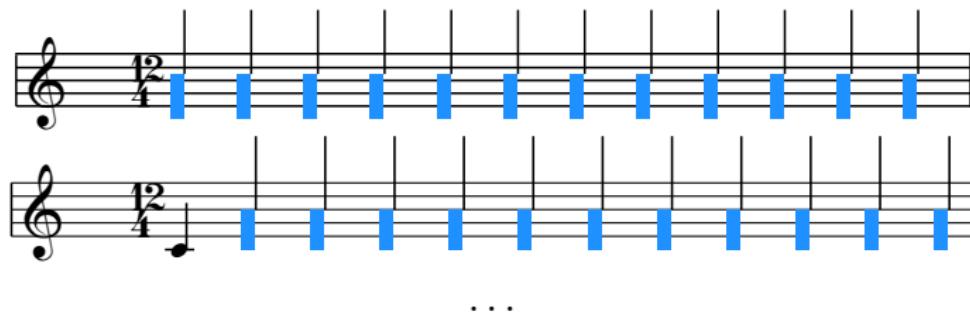


Modèle en MiniZinc :

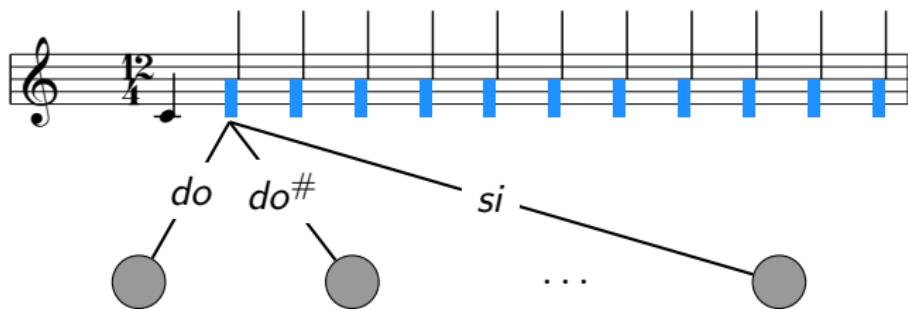
```
int: n = 12;
array[1..n] of var 1..n: pitches;
array[1..n-1] of var 1..n - 1: intervals;
constraint forall(i in 1..n - 1)
    ( intervals [i] = abs(pitches[i+1] - pitches[i]));
constraint alldifferent (pitches);
constraint alldifferent (intervals);

solve satisfy;
```

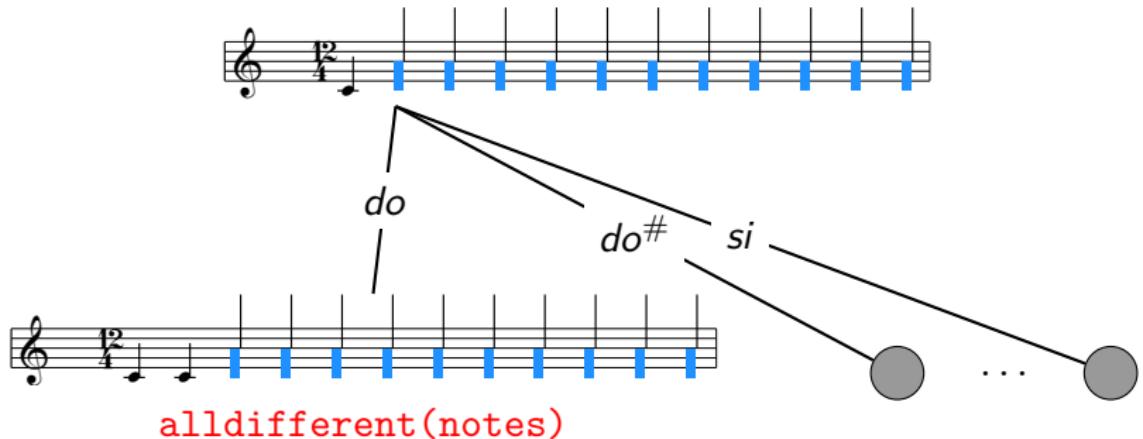
Au début de l'exploration



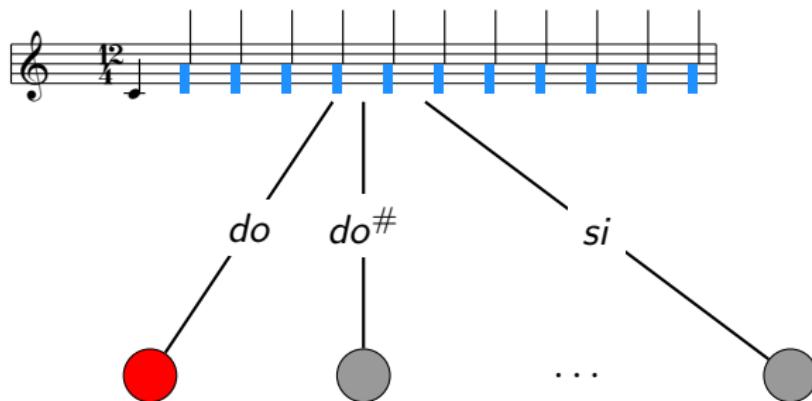
Arbre d'exploration (étape 1)



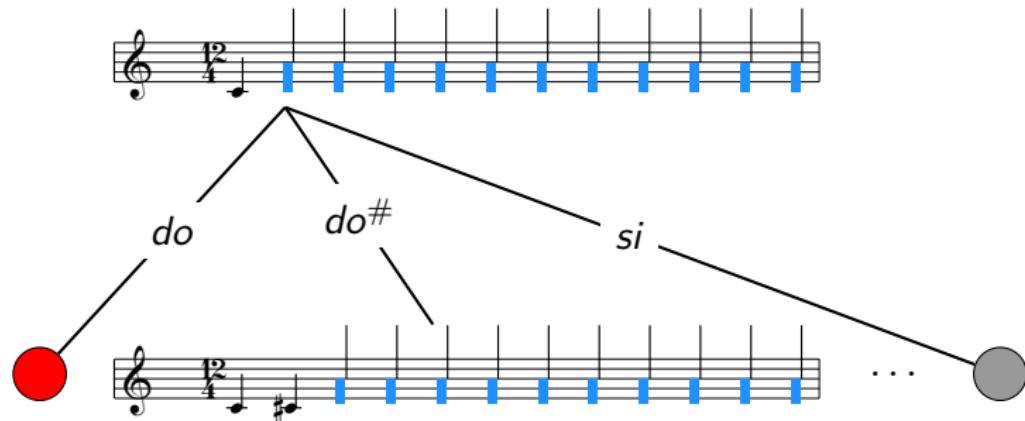
Arbre d'exploration (étape 2)



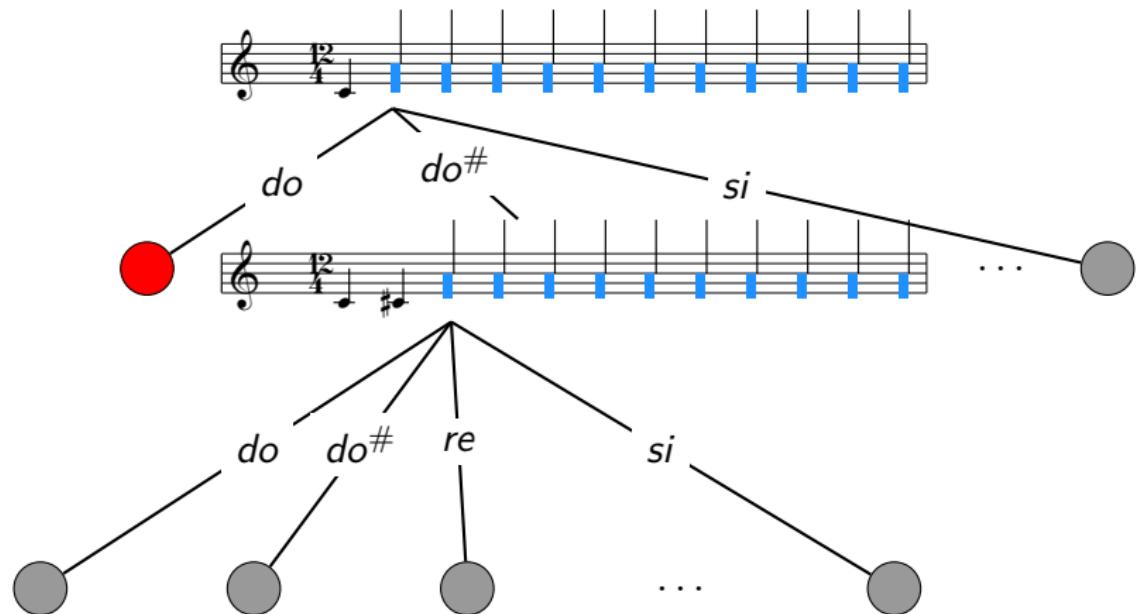
Arbre d'exploration (étape 3)



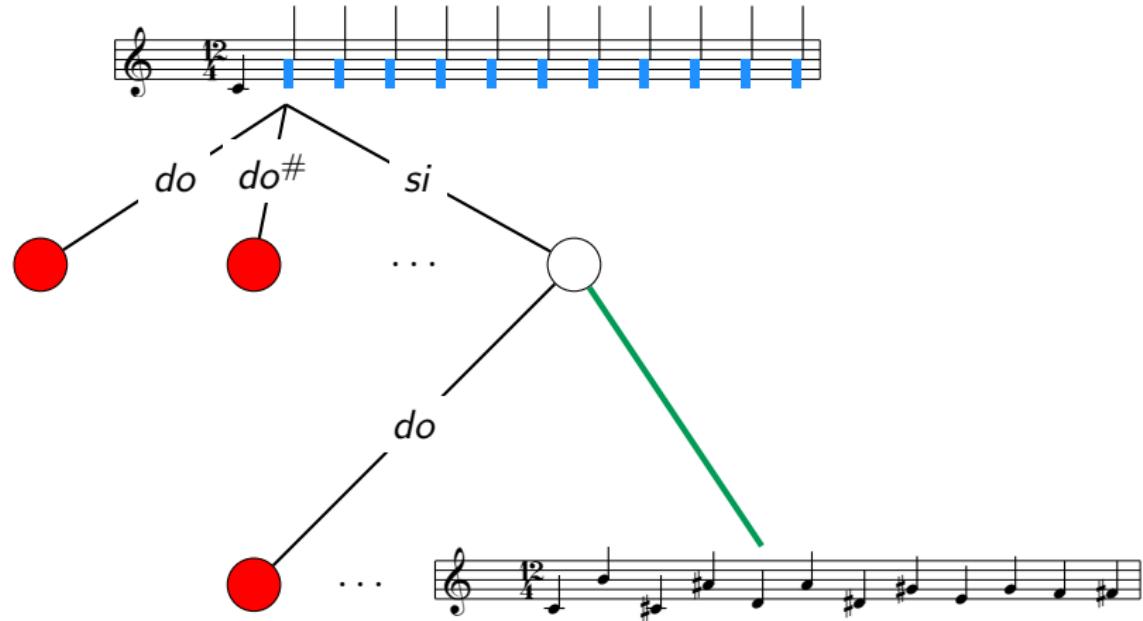
Arbre d'exploration (étape 4)



Arbre d'exploration (étape 5)



Arbre d'exploration (étape 6)



Pouvez-vous penser à des moyens d'améliorer cet algorithme ?

Une optimisation classique : Propagation

- ▶ Jusqu'à présent on se contente d'énumérer toutes les combinaisons possibles mais...
- ▶ Une optimisation consiste à réaliser la propagation.

5	3			7				
6	4		1	9	5			
	9	8				6		
8				5			4	3
			8	3				1
7				2				6
	6				2	8		
		4	1	9				5
	4		8			7	9	

 = { , 4, , 6, }

Menu

- ▶ Introduction
- ▶ Modélisation par contraintes avec MiniZinc
- ▶ Comment ça marche
- ▶ Autres exemples de problèmes par contraintes
- ▶ Conclusion

Nurse Scheduling Problem



N infirmières travaillent M jours en rotation, trouver un planning tel que :

- ▶ Une rotation par jour et infirmière : jour, début nuit, fin nuit.
- ▶ Pas plus de deux nuits d'affilées, un jour pas précédé d'une nuit.
- ▶ ...

Packing Problem

- ▶ Répartition d'un chargement de camion en équilibrant le poids.
- ▶ Plan optimal d'un appartement pour maximiser le nombre de chambres (d'une taille min).
- ▶ Plan de circuit intégré : minimiser le coût en plaçant les blocs suivant certaines contraintes.

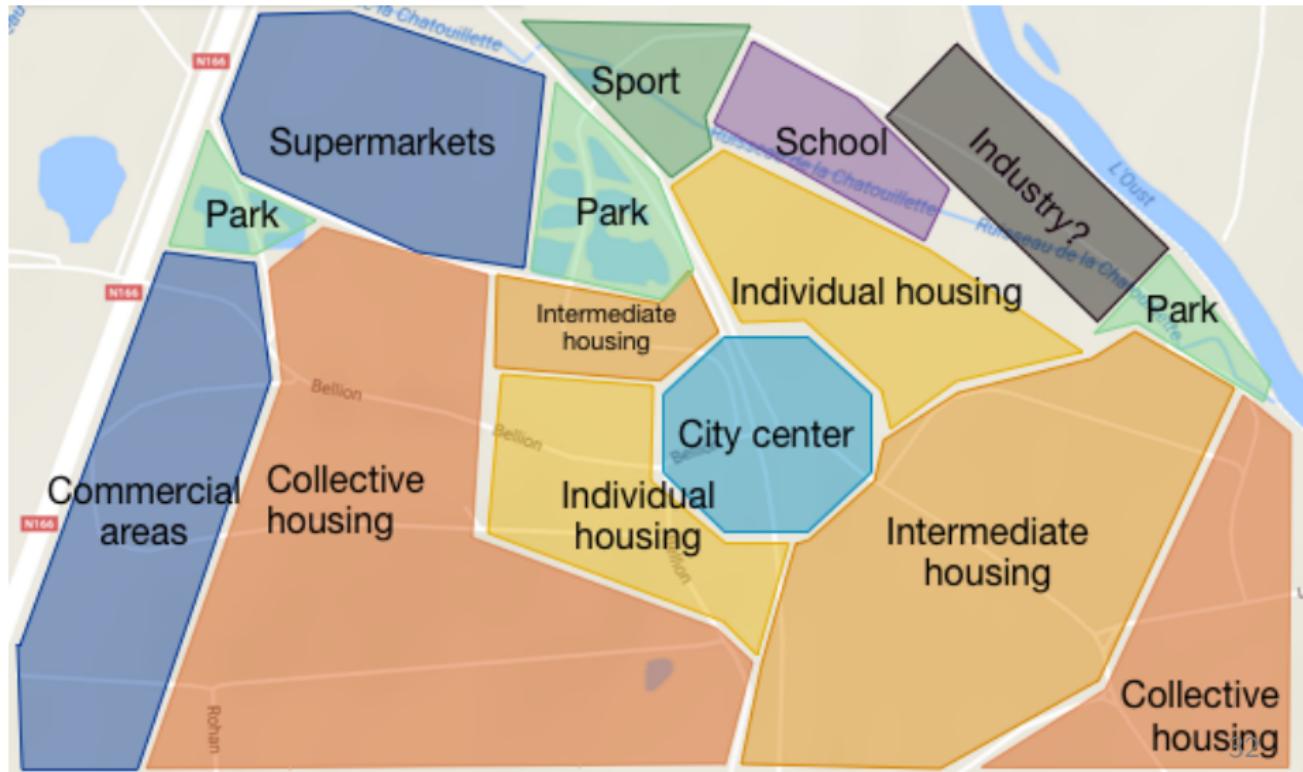


Problème d'optimisation : création d'une nouvelle ville

We need to place :

- Industry
- Intermediate housing
- Park
- Supermarkets & commercial areas
- Sport
- Individual housing
- Collective housing
- School
- City center

Problème d'optimisation : création d'une nouvelle ville



Ordonnancement de tâches sous ressources

- ▶ Établir un planning de cours.
- ▶ Maximiser la productivité de machine dans une usine.
- ▶ ...



Interactive Soccer Queries



Suivant le système de score FIFA (0 défaite / 1 égalité / 3 victoire), répondre à des questions :

- ▶ Est-ce qu'une équipe peut encore devenir championne en théorie ?
- ▶ Est-ce qu'une équipe est sûr / à des chances de se qualifier ?
- ▶ ...

Menu

- ▶ Introduction
- ▶ Modélisation par contraintes avec MiniZinc
- ▶ Comment ça marche
- ▶ Autres exemples de problèmes par contraintes
- ▶ Conclusion

Conclusion

La programmation par contraintes permet principalement de résoudre des problèmes combinatoires et NP-complet.

- ▶ On déclare et le système le résout automatiquement.
- ▶ On se concentre sur le problème plutôt que sa méthode de résolution.

Ceci n'est qu'une introduction...

En savoir plus

- ▶ Coursera – Modeling Discrete Optimization (P. Stuckey, J. Lee) : Focus sur la modélisation par programmation par contraintes.
- ▶ Gérard Assayag et Charlotte Truchet. Constraint Programming in Music.
- ▶ N'hésitez pas à me contacter ;-)

Conclusion

Merci pour votre attention !

