BASTIEN ROBERT – THEO GENIESSE – NICOLAS VAUVILLIER – QUENTIN BONNEFOND

ALGORITHME X & DANCING LINKS

EPF
ÉCOLE D'INGÉNIEUR·E·S

SUJET

- Résolution d'un SUDOKU par les **Dancing Links** et **l'algorithme X**
- Intégration du code au projet commun dans le but de tester la performance

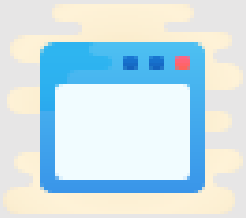
DANCING LINKS & ALGORITHME X

Sommaire

Phases du projet

Documentation et acquisitions techniques

Analyse de l'algorithme



Adaptation et implémentation de l'algorithme

Test

Teamwork

Bastien

- Technique
- Aide théorique

Théo

- Théorie
- Aide technique

Quentin

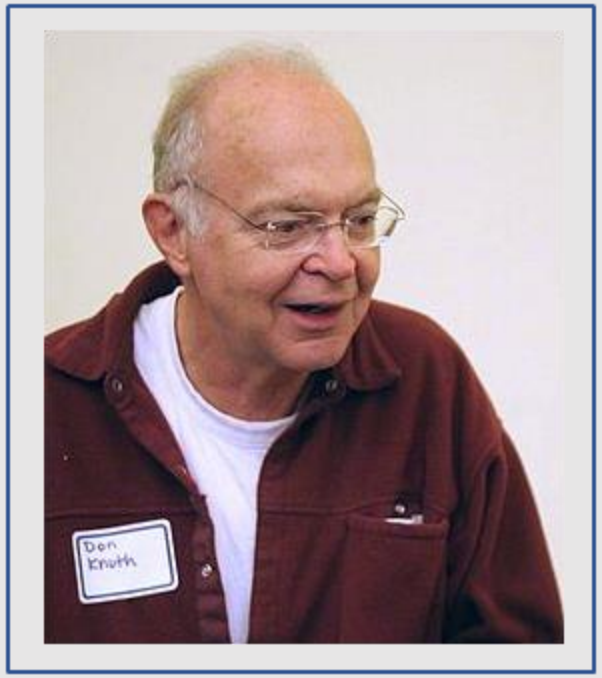
- Théorie
- Aide technique

Nicolas

- Technique
- Aide théorique

ALGORITHME X & DANCING LINKS

Un peu d'histoire

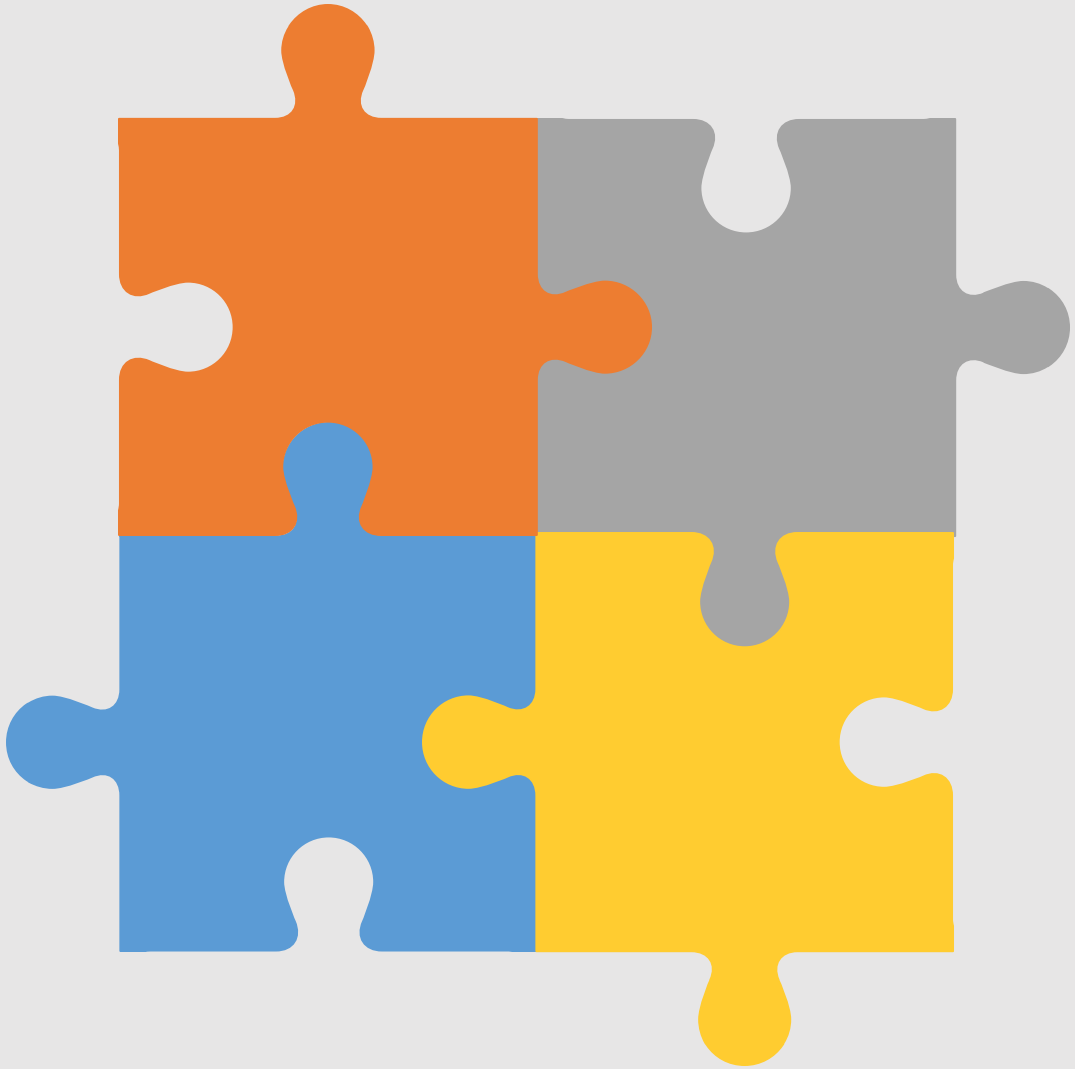


Dancing Links est la technique suggérée par Donald Knuth pour mettre en œuvre efficacement son **Algorithme X**.

Exact Cover (ou "Problème de couverture exacte")

Soit

- Un ensemble U
- Une collection S de sous-ensemble



Définition

Une couverture exacte de U est une sous-collection S* de S telle que tout élément de U est élément d'exactement un des ensembles de S*

Question

Quelle collection S* est une couverture exacte ?

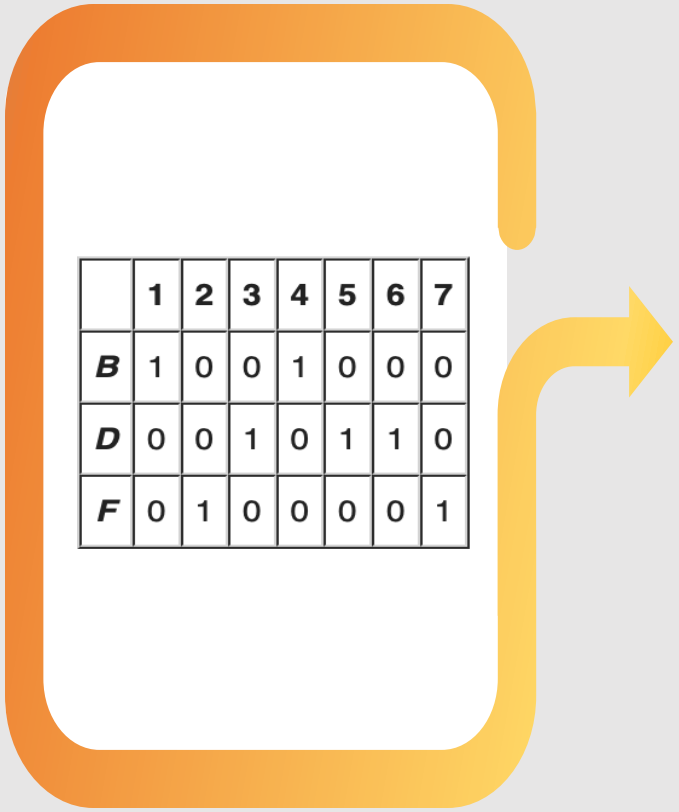
Exemple simple

Soit U = {0, 1, 2, 3, 4} et soit S= {E, I, P} une collection de trois ensembles :
E = {0, 2, 4} ; I = {1, 3} ; P = {2, 3}

Représentation matricielle

Un exemple simple pour expliquer :
6 ensembles dans S = {A, B, C, D, E, F} sous forme de lignes et les 7 éléments dans U = {1, 2, 3, 4, 5, 6, 7} sous forme de colonnes alors le problème de la couverture exacte est représenté par la matrice 6×7 :

	1	2	3	4	5	6	7
A	1	0	0	1	0	0	1
B	1	0	0	1	0	0	0
C	0	0	0	1	1	0	1
D	0	0	1	0	1	1	0
E	0	1	1	0	0	1	1
F	0	1	0	0	0	0	1



	1	2	3	4	5	6	7
B	1	0	0	1	0	0	0
D	0	0	1	0	1	1	0
F	0	1	0	0	0	0	1

Comme dans l'exemple 2, la sélection S∗= {B, D, F} de lignes est une solution de ce problème de couverture exacte :

Déroulé de l'algorithme X

	1	2	3	4	5	6	7
A	1	0	0	1	0	0	1
B	1	0	0	1	0	0	0
C	0	0	0	1	1	0	1
D	0	0	1	0	1	1	0
E	0	1	1	0	0	1	1
F	0	1	0	0	0	0	1

Sélectionner la colonne avec le moins de 1

Les lignes A et B ont des 1 dans leur première colonne

	1	2	3	4	5	6	7
A	1	0	0	1	0	0	1
B	1	0	0	1	0	0	0
C	0	0	0	1	1	0	1
D	0	0	1	0	1	1	0
E	0	1	1	0	0	1	1
F	0	1	0	0	0	0	1

Sous algo sur ligne A

La ligne A possède des 1 sur les colonnes 1,4,7

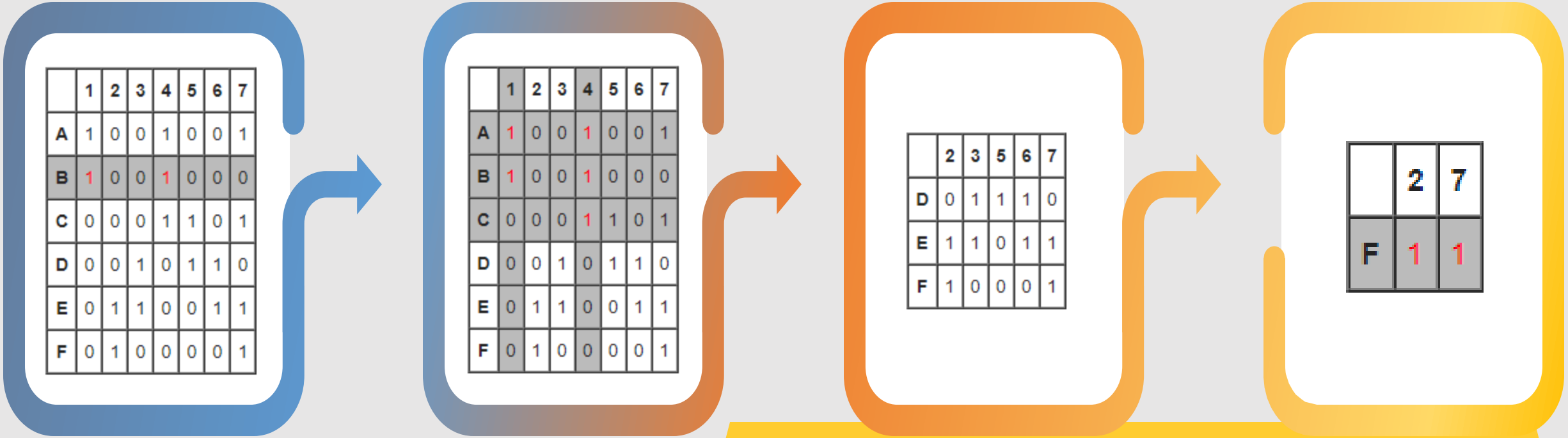
	1	2	3	4	5	6	7
A	1	0	0	1	0	0	1
B	1	0	0	1	0	0	0
C	0	0	0	1	1	0	1
D	0	0	1	0	1	1	0
E	0	1	1	0	0	1	1
F	0	1	0	0	0	0	1

La colonne *1* contient des 1 aux lignes *A* et *B* ; la colonne *4*, aux lignes *A*, *B*, *C* ; la colonne *7*, aux lignes *A*, *C*, *E* et *F*.

	2	3	5	6
D	0	1	1	1

La matrice n'est pas vide, on a donc une solution

On ne peut plus réduire



	1	2	3	4	5	6	7
A	1	0	0	1	0	0	1
B	1	0	0	1	0	0	0
C	0	0	0	1	1	0	1
D	0	0	1	0	1	1	0
E	0	1	1	0	0	1	1
F	0	1	0	0	0	0	1

	1	2	3	4	5	6	7
A	1	0	0	1	0	0	1
B	1	0	0	1	0	0	0
C	0	0	0	1	1	0	1
D	0	0	1	0	1	1	0
E	0	1	1	0	0	1	1
F	0	1	0	0	0	0	1

	2	3	5	6	7
D	0	1	1	1	0
E	1	1	0	1	1
F	1	0	0	0	1

	2	7
F	1	1

Solution S={B,D,F}

La première colonne avec un minimum de 1 est la colonne *2*, qui n'en a qu'un. On la sélectionne.

La colonne *2* contient un 1 à la ligne *F*; la colonne *7*, à la ligne *F* aussi.

On élimine la ligne et les colonnes ci-dessus.

Résolution de sudoku avec les dancing links

Le but sudoku s'est d'assigner une certaine quantité de nombres et des cellules d'une grille tout en satisfaisant les **4 contraintes** suivantes (pour un sudoku 9x9) :

- Lignes/colonnes (triviale mais essentielle)
- Lignes/nombres
- Colonnes/nombres
- Boite/nombres

Le sudoku se ramène à un **problème d'ensembles intersectant** et devant respecter les 4 contraintes précédentes avec exactement une seule possibilité. C'est ainsi que l'on peut ramener le sudoku à un problème de couverture exacte.

Pour chacune de ces contraintes il existe 81 solutions et donc 729 possibilités et 324 ensembles de contraintes.

On peut donc représenter ce problèmes sous la forme d'une **matrice 729x324**, que nous traiterons comme ci-dessus.

DlxLib

Entrée : Matrice de 0 et 1

Cette classe librairie prend en entrée une matrice de 0 et 1.

Sortie

Sort une matrice avec que des colonnes avec un seul 1.

C#

DlxLib est une classe librairie en C#

Dancing Links

La classe va effectuer l'algorithme X et les dacing links

PARTIE TECHNIQUE

→ Gestion des dépendances

→ Résolution avec DlxLib

→ Sudoku → Grille

PARTIE TECHNIQUE

Facile (x50 sudokus) :

Relativement très performant

```
Solver NorvigSolver a tout résolu en 00:00:01.3360307
Solver SolverDlx a tout résolu en 00:00:08.8113776
Solver SolverSMT a tout résolu en 00:00:12.8785386
Solver CSP a tout résolu en 00:00:28.8581932
```

Moyen (x95 sudokus) :

```
Solver NorvigSolver a tout résolu en 00:00:05.2800076
Solver SolverDlx a tout résolu en 00:00:14.7065695
Solver SolverSMT a tout résolu en 00:00:47.7664814
Solver CSP a tout résolu en 00:07:25.3920180
```

Difficile (x10 sudokus) :

```
Solver NorvigSolver a tout résolu en 00:00:00.3934492
Solver SolverDlx a tout résolu en 00:00:01.8288250
Solver SolverSMT a tout résolu en 00:00:04.2463292
Solver CSP a tout résolu en 00:00:11.1661411
```

Retour sur le projet

Comprendre et assimiler un concept complexe

Target 1

Implémenter un algorithme complexe

Target 2

Tester la validité et la performance de l'algorithme

Target 3

Travail au sein de l'équipe et intergroupe

Target 4

MERCI DE VOTRE ATTENTION