PROG
TP#8

2019-11-25

Résolution de SUDOKU

Le but de cet exercice est d'implémenter un *solveur* de SUDOKU qui permette de trouver une solution à une instance du problème (parmi éventuellement plusieurs) ou de montrer qu'aucune solution n'existe.

La description d'une instance est donnée *via* l'entrée standard sous la forme de 9 lignes de 9 chiffres chacun séparés d'une espace, où tout chiffre non nul indique une contrainte de valeur devant être prise dans la solution, et un zéro indique une absence de contrainte. Une solution doit être affichée sur la sortie standard de la même façon.

On rappelle qu'une solution valide est un carré de 81 chiffres tel que chaque ligne, colonne, et chacun des 9 sous-carrés 3×3 définis de façon évidente forme une permutation des chiffres de 1 à 9.

Par exemple, une solution de l'instance « wikipedia » décrite par :

```
5 3 0 0 7 0 0 0 0
6 0 0 1 9 5 0 0 0
0 9 8 0 0 0 0 6 0
8 0 0 0 6 0 0 0 3
4 0 0 8 0 3 0 0 1
7 0 0 0 2 0 0 0 6
0 6 0 0 0 0 2 8 0
0 0 0 4 1 9 0 0 5
0 0 0 0 8 0 0 7 9
```

est :

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

On conseille de procéder de la façon suivante :

1. Choisissez avec précaution la façon de représenter une instance ; celle-ci doit notamment permettre de représenter aisément toutes les sous-grilles devant être testées.

2. Écrivez une fonction calculant pour chaque position libre l'ensemble des contraintes s'y appliquant. Comment pouvez-vous détecter qu'un remplissage partiel de la grille ne mène à aucune solution complète ?
3. Étant donnée la fonction précédente, quelle heuristique simple pouvez vous utiliser pour résoudre « la plupart » des instances (par ex. l'exemple ci-dessus) ?
4. Implémentez une recherche par *backtracking* vous permettant de résoudre n'importe quelle instance.