Programmation C

TP9 – Backtracking

Emilie Marti

Ce TP m'a appris à résoudre un jeu du type sudoku avec de la brute force (on teste les valeurs jusqu'à bloquer, puis on revient à l'arrière pour tester de nouvelles valeurs ; ansi de suite jusqu'à résolution). Ce n'est pas optimisé mais ça marche très bien pour un simple sudoku de 9x9 cases.

Avec la première partie du TP j'ai découvert la technique du backtracking avec un tableau à une dimension et j'ai du l'appliquer sur un tableau à deux dimensions au sudoku (deuxième partie du TP). La partie la plus difficile à été déjà de comprendre ce que c'était le backtracking, puis l'appliquer. Ensuite, il a fallu faire gaffe au tableau à deux dimensions (ne pas inverser lignes et colonnes surtout, ce qui peut tout faire foirer bêtement).

J'ai pas su trouver toutes les possibilités de résolution d'un sudoku à plusieurs solutions … La fonction retourne lorsqu'elle a trouvé la première solution, donc je devrait peut-être changer cela facilement (peut-être pour le deuxième semestre).

- I/O

J'ai affiché le tableau du sudoku sur la sortie standard et ça rend plutot bien ; ça a été plutôt facile surtout parce que les nombres sont de 1 à 9 (ou rien), du coup ils occupaient le même espace.

- Type

Les types utilisés sont principalement des int.

- Programmation

- Module

Le programme est divisé en modules ; le main est tout seul dans son fichier, et les fonctions classées selon ce qu'elles gèrent.

- Compilation

Le makefile va compiler le programe avec un simple make et va donner le nom de l'exécutable sur la sortie standard.

- Récursivité

La technique du backtracking (brute force) est codée en récursive.

- Tableaux

On travaille sur des tableaux 1D pour la première partie, puis 2D pour la deuxième partie.

- Pointeurs

Les tableaux sont sous la forme d'un pointeur qui pointe sur la première case d'un tableau.

- Fichiers

Le programme va récupérer les modèles de sudoku non-résolus dans des .txt. À partir de là un tableau à 2 dimension est rempli.

Pour retrouver l'emplacement de la case, j'utilise une seule variable position et je retrouve les coordonnées, par exemple pour un tableau de 9x9, comme suit :

row = position/9 ;
column = position%9 ;

Ceci me permet de trouver une condition d'arrêt simplement ; en effet, lorsque la position = 9x9 = 81, je viens de sortir du tableau (qui va de 0 à 80) et cela veut dire que le sudoku a été complêtement rempli et je peux terminer ma fonction.