Licence Informatique 1^e année
Algorithmique et Programmation

Projet 2009-2010

1 Organisation

Le projet est à réaliser en binôme. En fin de semestre, lors de la dernière séance de TP, les binômes présenteront leur travail aux encadrants de TP. Lors des présentations, chaque binôme rendra un rapport qui présentera le programme réalisé (structures de données utilisées, principe des algorithmes implémentés, ...), les résultats obtenus, les problèmes rencontrés, les solutions apportées et tout élément permettant d'évaluer le travail réalisé. Chaque binôme devra également, au moment des soutenances, envoyer le code des programmes écrits aux responsables de TP (*marisela.hernandez@u-picardie.fr* et *frederic.furst@u-picardie.fr*).

2 Sujet

Le projet consiste à écrire un programme qui résout des grilles de sudoku. Le jeu de sudoku se joue sur une grille de 9 cases sur 9, divisée en carrés de 3×3.

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

Au début, quelques cases contiennent des valeurs entières fixées entre 1 et 9. Le but du jeu est de remplir toutes les cases vides avec des valeurs entières comprises entre 1 et 9, de manière à ce que chaque ligne, chaque colonne et chaque carré 3×3 contiennent une et une seule fois toutes les valeurs de 1 à 9.

L'algorithme le plus simple (mais de loin le moins efficace) pour résoudre un sudoku consiste à parcourir les cases dans l'ordre (par exemple de gauche

à droite et de haut en bas) et à remplir chaque case avec la première valeur qui respecte les contraintes (en testant successivement les valeurs de 1 à 9 par exemple). Si on arrive ainsi à remplir toutes les cases, on tient une solution. Si, à un moment, on se retrouve sur une case pour laquelle aucune valeur ne permet de respecter les contraintes, on remonte à la précédent case remplie (backtracking), et on essaie la valeur suivante (par exemple si on avait mis 5 dans cette case, on essaie 6). L'algorithme s'arrête soit quand on a rempli la dernière case vide, et alors on tient une solution, soit quand on doit remonter mais qu'il n'existe pas de case où remonter, et alors il n'y a pas de solution pour ce sudoku.

Algorithme de réalisation du projet :

- Choisir une structure de données pour stocker les données (il est par exemple possible d'utiliser un tableau d'entiers à 2 dimensions, en utilisant 0 pour coder une case vide)
- Écrire des fonctions permettant de tester si la valeur d'une case respecte les contraintes (la valeur ne doit pas être déjà présente dans une autre case de la ligne, de la colonne ou du carré)
- Écrire une ou plusieurs fonctions implémentant l'algorithme de résolution

Pour afficher le sudoku et suivre la résolution en direct, il est possible d'utiliser la classe *AffichageSudoku* fournie.

3 Pour aller plus loin ...

Les extensions du projet décrites dans cette partie sont optionnelles, mais en traiter au moins une augmentera très sensiblement la note attribuée, SI LA PARTIE OBLIGATOIRE A ÉTÉ TRAITÉE

Une amélioration possible consiste à permettre de lire une grille de sudoku à partir d'un fichier, ou de la sauver dans un fichier.

Le programme peut être étendu, par exemple pour qu'il permette de générer des grilles de sudoku, ou qu'il permette de trouver toutes les solutions possibles d'un sudoku (et non une seule).

Il est également possible d'écrire un programme résolvant des sudokus de dimension quelconque (et pas seulement 9×9), ce qui peut permettre de tester l'algorithme de résolution sur des sudokus de différentes tailles pour tracer la courbe de complexité de l'algorithme.

La complexité théorique de l'algorithme est un peu ardue à calculer, mais il est possible de la déterminer. Les livres de la BU peuvent y aider.