Projet – Programmation parallèle : Tetravex

Tetravex

Tetravex est un jeu de réflexion de type puzzle. Au début du jeu, la partie commence sur un plateau vierge de taille 3x3 (taille par défaut) et neuf carrés, ayant chacun un chiffre sur chaque bord (donc quatre chiffres par carré). Ces chiffres varient de 0 à 9 inclus. Le but est de placer ces carrés sur la grille, en faisant en sorte que le chiffre d'un bord soit le même que celui du carré adjacent. Chaque face de chaque carré est donc posée à côté d'une face d'un même chiffre. Le jeu est terminé quand la grille est remplie avec tous les carrés, correctement placés.

Quelques sites internet permettant de jouer, sur le web, au jeu Tetravex :
https://gamegix.com/tetravex/game
http://smart-games.org/en/tetravex/game/

Objectif

Vous devez écrire un programme en C++ qui permette de résoudre (remplir correctement le plateau) une partie de jeu Tetravex via un algorithme de type *backtracking* (retour sur trace). Le *backtracking* est un algorithme qui permet de trouver toutes les solutions à un problème donné. L'algorithme construit progressivement des candidats aux solutions et abandonne un candidat dès lors qu'il ne satisfait pas la solution.

Votre programme doit prendre en entrée un fichier texte décrivant une partie (plateau et carré) et devra afficher la solution de cette partie.

Exemple de fichier texte décrivant une partie de jeu :

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

La première ligne indique la taille du plateau (nombre de colonne et nombre de ligne) et les autres lignes décrivent les 9 pièces du jeu.

Quelques sites internet expliquant l'algorithme de *backtracking* :
https://openclassrooms.com/fr/courses/1256706-le-backtracking-par-lexemple-resoudre-un-sudoku
http://www-igm.univ-mlv.fr/~dr/XPOSE2013/sudoku/backtracking.html

Travail à réaliser

1. Modéliser et implémenter le jeu Tetravex
2. Implémenter un algorithme de *backtracking* séquentielle
3. Implémenter un algorithme de *backtracking* parallèles en utilisant la librairie thread de C++11. L'algorithme peut être paralléliser à différents niveaux, vous devez implémenter au minimum deux variantes dont une variante utilisant un thread pool.

Rendu du travail

Vous devrez fournir les code sources en C++ du projet ainsi qu'un rapport. Ce rapport devra contenir :

1. Un manuel d'utilisation de votre programme
2. Les explications concernant vos différentes implémentations des algorithmes de *backtracking*
3. Les résultats et/ou courbes de comparaisons entre les différentes variantes de *backtracking*
4. Pour chaque comparaison, vous donnerez une interprétation des résultats obtenus.

Le rapport ne devra pas excéder 3 pages.

Rendu du travail

Il est fortement conseillé de prendre le temps de la réflexion avant d'arrêter un choix pour vos structures de données. En particulier, les structures de données choisies devront permettre une mise en œuvre aisée et efficace de résolution de *backtracking*.

Durant la phase de débogage, je vous invite à compiler en utilisation les options –Wall et –g3 du compilateur g++ et d'utiliser alternativement un débogueur comme gdb et un programme vérifiant les accès à la mémoire comme valgrind.

Une fois votre programme au point, il est conseillé de le compiler avec l'option d'optimisation –O3 du compilateur g++ afin d'obtenir un exécutable le plus performant possible. Dans le cas de figure où vos expérimentations sont effectuées sur un ordinateur portable, il est important de s'assurer que la politique de gestion de l'énergie est réglée sur la puissance maximum afin de ne pas biaiser les comparaisons.

Attention : de nombreuses ressources et programmes (de qualité variable) sont disponibles très facilement, notamment sur Internet. Vous pouvez bien évidemment les consulter, mais gardez bien à l'idée que c'est un travail personnel : l'enseignant responsable sera particulièrement vigilant sur ce point et prendra les sanctions adéquates en cas de détection.