Cours "Informatique Embarquée"

François Armand

M2 (IMPAIRS, LP, MIC, EIDD...)

Exercice N° 2, 13 Octobre 2017

A rendre avant le Jeudi 19 Octobre Minuit sur Moodle

armand@informatique.univ-paris-diderot.fr

Rappels

- Voir la note sur le contrôle d'Informatique Embarquée
- Voir la note sur les TPS

Sujet du TP N°2

Avant de vous lancer dans la programmation, vous tenterez d'estimer le temps nécessaire (en heures) à la réalisation (réflexion, codage, tests, écriture du rapport) de ce TP, ces temps seront consignés sur le rapport associé au TP.

Le travail demandé est de reprendre le travail commencé la semaine dernière : remplissage d'un tableau à 2 dimensions en spirale / hélice colimaçon… pour le compléter, l'améliorer, discuter, mesurer, réfléchir…

0) Vous finirez de mettre au point votre algorithme si nécessaire. Il doit fonctionner sur des tableaux d'une seule case, d'une seule ligne, d'une seule colonne, carrés de tailles paires, carrés de tailles impaires, rectangles pairs / pairs, rectangles pairs/impairs, rectangles impairs/impairs, rectangles impairs /pairs, que ces rectangles soient verticaux ou horizontaux. Votre programme doit se comporter correctement (pas de plantage) quelles que soient les dimensions demandées.

1) Vous ferez évoluer votre programme de telle sorte qu'il définisse

1.1) une fonction `colimacon` dont vous définirez le prototype, mais qui doit impérativement allouer la mémoire du tableau et initialiser ce tableau.

1.2) un programme prenant les 2 dimensions du tableau en paramètres et invoquant cette fonction, lignes d'abord, colonnes ensuite.

2) Vous fournirez un Makefile, avec toutes les propriétés attendues d'un Makefile. Ce Makefile devra notamment construire une librairie statique fournissant que la fonction `colimacon`, ainsi qu'une librairie dynamique fournissant la même fonction. Il faudra donc produire un fichier `colimacon.a` et un fichier `colimacon.so`

3) Vous fournirez aussi un programme de test automatique validant votre fonction `colimacon` ou votre programme construit sur cette fonction sur un certain nombre d'exemples.

Dans votre compte-rendu :

- Vous consignerez l'estimation initiale que vous avez faite du temps de travail nécessaire. Pour ce second TP, sans prendre en compte le temps estimé / passé pour le premier TP.
- Vous consignerez aussi le temps que vous avez réellement passé pour le codage et la mise au point de ce programme.
- Vous expliquerez brièvement vos choix, les problèmes de mise au point que vous avez rencontrés,

- Argumentez le choix du prototype de votre fonction. Vous comparerez votre prototype avec le prototype suivant : quels sont leurs avantages inconvénients respectifs ?

```
int colimacon (int ** tab, int line, int col)
```

- Votre programme pourrait-il être parallélisé, et fonctionner dans un programme multi-threads ? Sinon, serait-il possible de concevoir un algorithme permettant facilement une parallélisation, argumentez. Pourrait-on paralléliser au point ou chaque cellule serait remplie indépendamment des autres cellules ?

- Quel(s) est (sont) les temps nécessaire(s) à l'exécution de votre programme sur des tableaux de 10x10, 100x100, 1000x1000 et 10 000 x 10 000. Indiquez avec quel(s) moyen(s) vous avez mesurez ce temps, et comment vous avez procéder pour mesurer ce temps.

- Combien d'accès mémoire en lecture et en écriture votre programme fait-il pour remplir un tableau de N cases au total. On ne comptera que les accès au tableau.

- Quel impact votre algorithme a-t-il vis à vis des accès mémoire. On pourra utiliser la commande Linux `perf`(1).

- Quelle est l'occupation mémoire de votre programme dans les cas ci-dessus ?

- Vous rapporterez les commentaires faits par le groupe / les étudiants à qui vous aurez soumis votre travail pour relecture. Vous direz quelles remarques vous avez prises en compte. Vous justifierez votre rejet des remarques non prises en compte.