TP 4 – Informatique Embarquée

Victor Oudin

Makefile.

- **make :** pour compiler et créer l'exécutable.
- **make test :** pour compiler, créer un exécutable de test et le lancer.
- **make clean :** pour effacer les fichiers « .o » et « .a ».
- **make mrproper :** pour effacer les fichiers « .o », « .a » et les exécutables.
- **make pkg :** fais « make mrproper » et créé une archive « .tgz » des sources.

Implémentation.

J'ai choisi de remplir le tableau de façon récursive :

- Si le tableau fais une case : je la remplis.
- Si le tableau fais une ligne : je remplis la première case et appel une récursion sur le sous-tableau.
- Si le tableau fais une colonne : je remplis la première case et appel une récursion sur le sous-tableau.
- Sinon je remplis la couronne extérieur et :
  - si le tableau est plus grand que 2 lignes ou 2 colonnes : j'appelle une récursion sur le sous-tableau.

→	→	→	→	→	→	↓
→	→	→	→	→	↓	↓
↑	→	→	→	↓	↓	↓
↑	↑	Ω	←	←	↓	↓
↑	↑	←	←	←	←	↓
↑	←	←	←	←	←	←

Dans cet exemple, l'implémentation rempli la couronne bleu et fais un appel récursif pour remplir le sous-tableau. Le sous-tableau rempli la couronne verte et fais un appel récursif pour remplir le sous-sous-tableau. Le sous-sous-tableau rempli la couronne jaune et fini.

Temps de travail.

Temps estimé : 6h

Temps travaillé : 10h

Problèmes rencontrés.

Les problèmes rencontrés sont la récursion et le parcours du tableau.

Temps d'exécution.

Temps réel d'exécution pour colimaçon 10 10 : 0sec ; Occupation mémoire 400 bytes

Temps réel d'exécution pour colimaçon 100 100 : 0sec ; Occupation mémoire 40.000 bytes

Temps réel d'exécution pour colimaçon 1000 1000 : 0sec ; Occupation mémoire 4.000.000 bytes

Temps réel d'exécution pour colimaçon 10000 10000 : 2sec ; Occupation mémoire 400.000.000 bytes

mesuré ainsi :

```
time_t start, stop;
start=time(NULL);
if(colimacon(&array, rows, columns) == 0){
        perror("");
        return 0;
}
stop=time(NULL);
```

l'occupation mémoire à été mesurer à l'aide de « valgrind ».

Remarques d'autres étudiants.

Aucune remarques particulières.

Parallélisation

L'algorithme est parallélisable car on peu faire les appels récursif dans des threads.