HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Alberto Dassatti, Bruno
Da Rocha Carvalho

Laboratoire de High Performance Coding

semestre printemps 2024

Laboratoire 7 : Parallélisme des tâches

Temps à disposition : 4 périodes (2 séances de laboratoire).

1 Introduction

Ce laboratoire a pour objectif de vous familiariser avec les techniques et outils de parallélisme des tâches. Vous apprendrez à analyser et optimiser des programmes multithreadés en utilisant un outil spécifique, COZ, et à implémenter des solutions multithreadées pour des problèmes de traitement de données. Le but est de vous donner les compétences nécessaires pour identifier les points critiques de performance dans vos programmes et de les améliorer efficacement.

2 Outil d'analyse de code multithreadé : COZ

La première partie du laboratoire est dédiée à l'introduction d'un outil d'analyse de code multithreadé nommé COZ. Cet outil exécute votre code plusieurs fois pour estimer l'impact d'optimisations potentielles sur certaines parties du code. COZ ralentit délibérément certaines sections du code pour simuler l'effet d'une accélération d'autres et fournit un rapport avec des graphiques qui évaluent l'impact potentiel de ces améliorations sur l'exécution globale.

Cet outil permet ainsi de cibler les zones de code qui pourraient significativement améliorer les performances globales de votre programme.

Pour utiliser COZ, la commande est la suivante :

```
coz run --- {mon programme} {args}
```

L'installation se fait en suivant les instructions disponibles sur le GitHub : `https://github.com/plasma-umass/coz`

Une fois le profilage terminé, vous pouvez observer les résultats avec la commande suivante, qui ouvrira un navigateur pour charger le fichier de résultats .coz :

```
coz plot
```

Attention :

- Si l'exécution est trop courte, COZ pourrait ne pas réussir à profiler correctement.
- Votre programme doit être compilé avec les options `-g` et `-gdwarf-3`.

Afin de tester l'outil et comprendre comment l'utiliser, vous pouvez observer les projets dans `benchmarks/`.

3 Recherche de séquence multithreadée

Pour cette partie, vous allez transformer un code donné en exemple qui permet de trouver une séquence de nombres dans un fichier. La séquence est variable, et la taille du fichier l'est également. L'objectif est d'implémenter une solution multithreadée et la plus rapide possible pour cette recherche de séquence, en utilisant soit OpenMP, soit pthreads.

Pour tester votre code, un petit fichier est fourni, mais nous vous conseillons d'utiliser des fichiers plus grands pour tester l'efficacité, par exemple disponibles sur :

- `https://pi2e.ch/blog/2017/03/10/pi-digits-download/#download`
- `https://calculat.io/en/number/download-pi-digits-files-txt-zip`
- Tout autre fichier de votre conception ou que vous trouverez.

Vous serez amenés donc à modifier le CMakeLists également !

4 Travail à rendre

- Votre code fonctionnel et compilable.
- Un rapport expliquant comment vous avez décidé de paralléliser le travail.
- Une analyse montrant la réelle pertinence de votre implémentation, en prenant en compte que le code fourni a été pensé pour être le moins performant possible.
- Si vous parvenez à analyser votre implémentation avec COZ et à en tirer des conclusions pertinentes dans votre rapport, vous obtiendrez un bonus de 0,5 point sur votre note.