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 2025

Laboratoire 5 : Profiling

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

1 Objectifs de ce laboratoire

L'objectif de ce laboratoire est de se familiariser avec des outils de profiling afin d'analyser les performances d'un programme. Pour cela, vous aller utiliser les outls proposé pour profiler des programmes.

2 Perf

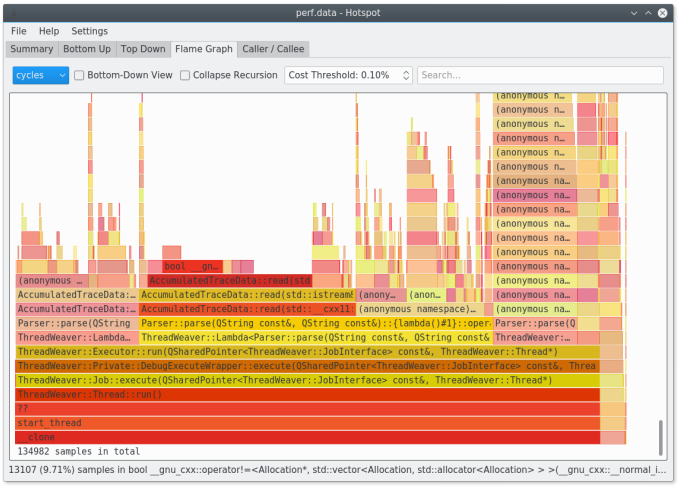
Le kernel Linux implémente des moyens de profiling, accessibles depuis l'outil `perf`. Beaucoup de ressources en ligne décrivent l'architecture et l'utilisation de `perf`. Pour vous familiariser avec cet outil, vous pouvez suivre le tutoriel en trois étapes qui décrit son utilisation, ainsi que consulter cette source d'informations.

Vous pouvez utiliser les commandes `perf stat` et `perf top` pour commencer votre analyse, puis utiliser `perf record` et `perf report` pour approfondir l'analyse.

Pour plus d'informations techniques sur `perf` (facultatif), vous pouvez consulter sa documentation officielle.

3 Hotspot

Vous l'aurez remarqué, `perf` ne permet pas de visualiser facilement les "hotspots" d'un programme qui a été profilé. Ces hotspots peuvent être visualisés sous la forme d'un "flame graph", comme celui proposé par le programme "Hotspot" ci-dessous :



Pour vous familiariser avec Hotspot, vous pouvez lire la page README de son dépôt Git, puis essayer le programme. Vous pouvez effectuer des mesures de cycles mais également de cache misses.

4 Valgrind

Valgrind est un outil sous Linux initialement conçu pour le débogage de l'allocation dynamique de mémoire sur le heap (memcheck). Valgrind a depuis évolué en une suite d'outils (helgrind, cachegrind, callgrind, ...) permettant également de faire du profiling. Pour utiliser les outils cachegrind et callgrind, vous pouvez suivre ce tutoriel : Guide to Callgrind.

Si vous souhaitez utiliser l'outil memcheck, vous pouvez consulter la documentation officielle à cette adresse : Manual for Memcheck (facultatif).

5 Travail à effectuer

5.1 Familiarisation

Afin de vous familiariser avec les outils proposés, nous vous fournissons un code que vous pouvez analyser.

Le code disponible dans le dossier `code` a pour fonction de calculer la température moyenne par ville, ainsi que de déterminer les températures minimale et maximale pour chaque ville.

Chaque ligne du fichier de données est au format `Ville;température`.

Lorsque vous compilez le projet, deux exécutables seront générés :

- `build/create_sample` : cet exécutable permet de générer un fichier de mesures contenant les villes et leurs températures. Il prend un seul argument `N`, qui spécifie le nombre de lignes que contiendra le fichier texte.
- `build/analyse` : cet exécutable permet d'analyser le fichier de mesures et d'en extraire la

température minimale, maximale et moyenne par ville. Il prend comme paramètre le chemin vers le fichier texte généré.

Votre objectif est d'analyser le code fourni à l'aide des outils `perf` et `valgrind`, afin d'identifier les éventuels problèmes et les améliorations possibles. Il ne vous est en revanche pas demandé de les corriger ou de les appliquer.

5.2 Profiling

Maintenant que vous avez pris en main les outils, nous vous demandons de reprendre votre programme du `lab01`. L'objectif est de profiler ce projet à l'aide des outils que vous avez testés, en complément des analyses précédentes que vous avez pu effectuer.

Vous devrez analyser le projet et, en le profilant, identifier des pistes d'amélioration ou de modification.

6 Travail à rendre

Vous devrez remettre un rapport présentant votre analyse de la partie 5.1, ainsi que votre hypothèse d'amélioration. Ce rapport devra également inclure le profilage de votre code du `lab01`, les problèmes que vous y avez identifiés et les améliorations que vous y avez apportées. Pensez à effectuer des mesures avant et après les améliorations afin de comparer les performances de manière objective.