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 2023

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 allez utiliser les outils décrits pour profiler deux des programmes proposés :

- PPM non interactif
- FFT
- LZW compression

2 Perf

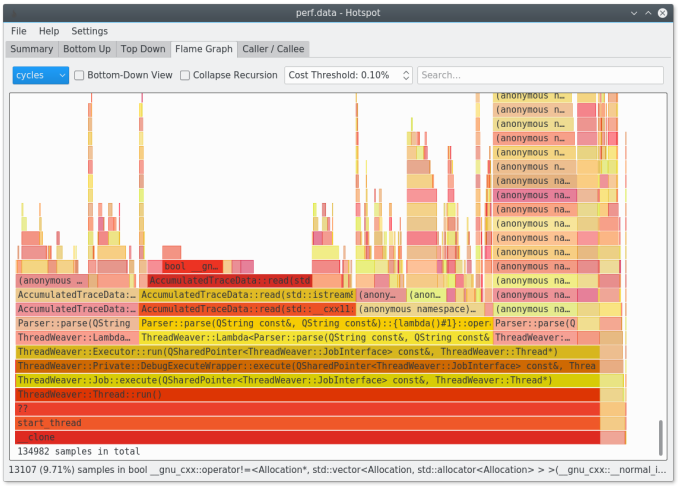
Le kernel Linux implémente des moyens de profiling, accessibles depuis l'outil `perf`. Beaucoup de ressources en ligne décrivent l'architecture et l'utilisation de `perf`. Pour vous familiariser avec cet outil, vous pouvez suivre le tutoriel en trois étapes qui décrit son utilisation, ainsi que consulter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

Enfin, vous pouvez utiliser perf et cachegrind pour profiler le code que vous avez écrit depuis le début des laboratoires. Identifiez les bottlenecks (points de blocage), puis établissez une liste des améliorations possibles que vous pourriez apporter à votre code. Il est évident que vous avez déjà effectué des optimisations dans les laboratoires précédents, mais n'hésitez pas à revenir en arrière et à analyser votre code avec ces outils pour vérifier si vos améliorations ont été pertinentes. Il vous est également demandé d'implémenter au moins une des améliorations que vous aurez identifiées et de nous remettre le code correspondant.

Pour simplifier le mécanisme de mesure, je vous invite à consulter ce lien qui vous montre comment ne mesurer que les parties "intéressantes" de votre code.