Cache Pirating: Understanding Cache Contention

UPPSALA UNIVERSITET

Researcher:
Ragnar Hägg
raha4459@student.uu.se

Supervisor:
Andreas Sandberg
andreas.sandberg@it.uu.se
UART

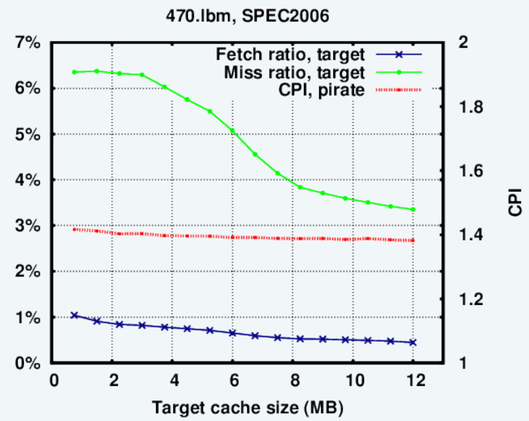
Department of Information Technology
Box 337
SE-751 05 Uppsala
Sweden

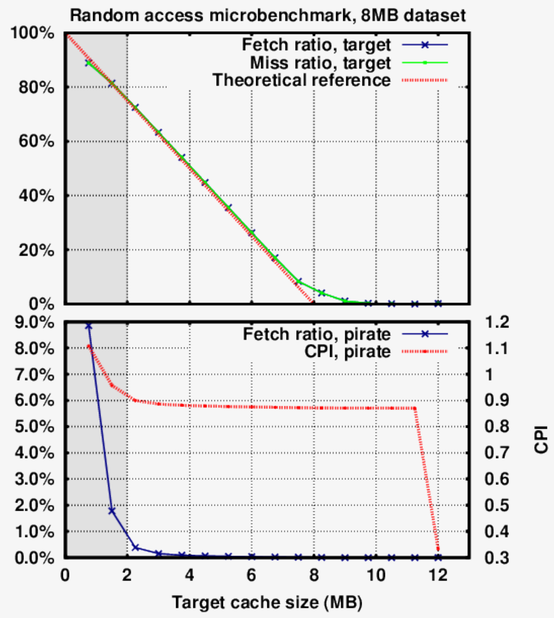
Motivation

Today cache and off-chip bandwidth is typically shared between cores. Since both available cache and memory bandwidth strongly influences performance of an application it is increasingly important to understand how performance, such as memory bandwidth and throughput depends on the amount of shared cache available.

Purpose

To implement a Cache Pirate (Eklöv *et al.)* with on-line size adjustment under an open source license as an easy-to-use command line application.
Hardware performance-event counters should be used for measuring. The pirate should work on as many architectures as possible.





Result

The pirate application

- uses the measured CPI of the pirate to validate that it can keep a sufficient amount of the cache. This removes the need for two off-core per-core counters.
- is started with an easy terminal command.
- automatically adjust microbenchmark parameters such as stride and dataset size using processor information.
- is verified on Westmere, Nehalem and Sandy Bridge architectures.

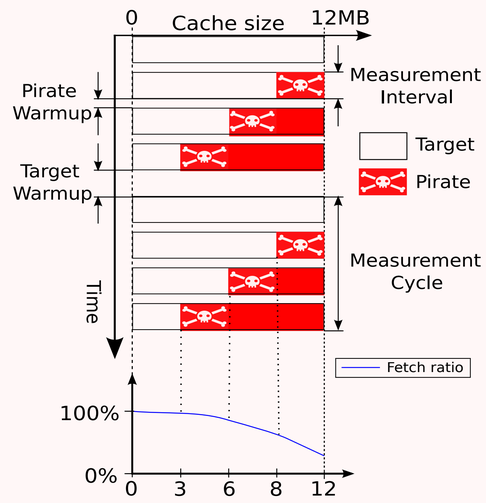
Method

The pirate and the target run simultaneously. The target is stopped after each sample, which is a fix number of instructions executed in the target. The hardware performance counters are then used to measure the performance of the target and monitor the behavior of the pirate, meanwhile the pirate is warmed for the next measurement size.

When all sizes are sampled, the pirate stops and the target is warmed with the full cache size. The sampling cycle is then repeated.

For specifying the the performance events the pfm4 library is used to encode the performance events for the Linux kernel.

The output is given in a Google Protobuf file which can be processed or converted to a csv file with provided Python scripts.



Usage

```
./perfpirate -e PFM4_PERF_EVENT_NAME --sample-period=10000000 -- ./my_target_command target_options
```