Coloring the Cloud for Predictable Performance

Alberto Scolari, Filippo Sironi, Davide B. Bartolini, Donatella Sciuto, Marco D. Santambrogio
Politecnico di Milano

Motivation and Background

The shared last-level cache (LLC) in commodity multicore can lead to contention among co-located virtual machines (VMs).

VMs contending for the LLC may experience varying slowdown, leading to unpredictable performance, as in the example below [1].

Recent research on LLC partitioning usually requires architectural extensions unavailable on commodity multicore.

Page coloring allows LLC partitioning via software
+ support commodity multicore
− high cost of re-coloring to handle variable memory footprints.

Insights and Contributions

VMs have bounded memory footprint
+ can use page coloring; no need for re-coloring
+ can partition LLC on commodity multicore.

We propose Rainbow: a page allocator leveraging page coloring to manage cache allocation
+ predictable performance
+ higher utilization through safe VMs co-location.

Implementation

We modify the buddy page allocator, splitting the lists of free pages into per-virtual color (i.e., vcolor, a set of 2^n contiguous physical colors - pcolors) lists.

Preliminary Results

We vary cache allocation to 2-threaded instances of canneal and streamcluster, from the PARSEC 2.1 benchmark suite, co-located on a quad-core.

Rainbow partitions the LLC, avoiding contention and achieving almost perfect isolation for canneal.

The gap between solo and co-location with more than 1.25 MB of LLC depends on unpartitionable resources (i.e., on-chip interconnect and memory controller bandwidth; prefetchers are disabled).

Work in Progress / Open Issues

We are finalizing our design and implementation to evaluate Rainbow with VMs running representative cloud workloads.

We will evaluate the performance penalty due to the lack of support for huge pages, which is an inherent limitation of page coloring, with respect to commodity hypervisors.

Support for overlapping vcolor allocations could improve utilization, but requires co-scheduling.

---