Low Cost Working Set Size Tracking

Weiming Zhao, Xinxin Jin, Zhenlin Wang, Xiaolin Wang, Yingwei Luo, Xiaoming Li
Motivation

- Page Level Miss Ratio Curve (MRC)
  - Wide Applications:
    - Working Set Size (WSS) Estimation
    - Memory Resource Balancing
  - Expensive
    - Mean Runtime Overhead of SPEC CPU 2006: 16%

- Goal of This Research
  - Low Cost MRC Construction With Enough Accuracy
Background

- Overhead & Existing Optimizations
  - Memory Intercept # x (time to find LRU distance)
    - Dynamic hot set sizing
    - Less interception if overhead is too high
    - Undermines accuracy
  - Bounded by WSS
  - AVL-Tree LRU list → Linked List
  - O(log(WSS)) → O(WSS)

- Program Phases
  - Most programs show phasing behaviors
    - IPC, WSS, branch prediction, etc.
    - Stable within a phase, disruptive transitions between phases
Our Idea

- Intermittent Memory Tracking (IMT)
  - When WSS is stable, disable memory tracking
  - Re-enable when a phase change occurs
  - How to detect when memory tracking is off?
    - A key observation:
      - Monitor HW events (PMCs) => detect PMC phase changes
      - predict WSS phase changes
  - Challenge
    - Quick and accurate online phase detection
Examples: Correlation Between WSS And Hardware Events

- **429.mcf**
- **473.astar**
- **450.soplex**

**HW Events: degree of fluctuation varies among programs**

A challenge to PMC phase detector

All data are normalized
Phase Detection
- Moving average filter for de-noising, $f(i)$
- Stable phase: $f(i) / (\text{historic mean}) \in [1 \pm T]$
- $T$: detection threshold

$T$ For WSS Phase Detection
- A fixed, empirical value of $T_{wss} = 0.05$
  - Works well because of relatively small fluctuations

$T$ For PMC Phase Detection
- A fixed value of $T_{PMC}$
  - Average performance, not the best fit for all programs
Framework of IMT

- Adaptive Threshold for PMC Phase Detection
  - Compare detection results, if inconsistent:
    - WSS is stable but PMC phase detected: $\uparrow T_{PMC}$
    - WSS phase detected but PMC is stable: $\downarrow T_{PMC}$
  - "Checkpointing": periodically wake up WSS tracker
Experimental Results

- Implementation
  - WSS Tracker in Xen 3.2
  - IMT in Dom–0

- IMT Configuration
  - Use Data TLB misses for PMC phase detection
Evaluation of IMT

- **Metrics**
  - Mean Relative Error (MRE): \( \frac{\sum_{i=1}^{n} |M_i - m_i|}{n} \)
  - Up Ratio (UR): memory tracking time / total time

**IMT Performance of SPEC CPU 2006**

- MRE: 3.9%
- Up Ratio: 12%
Overhead Of WSS Tracking

SPEC CPU 2006

- **Reg Opt**
- **Reg Opt + IMT (adaptive)**

![Graph showing overhead percentages for various SPEC CPU 2006 benchmarks](attachment:image.png)

- **No IMT:** 16%
- **IMT:** 2%

**Regular optimizations:**
- Dynamic hot set sizing
- AVL-tree based LRU list
Application to Memory Balancing For Virtual Machines

Two VMs on one host:
VM1: 470.lbm
VM2: 433.milc

Baseline: 700 MB Memory / VM

Speed-Ups With Memory Balancing

Bal. w/ Reg Opt
Bal. w/ Reg Opt + IMT (adapt.)

470.lbm   433.milc   Overall
2.96   3.56   1.63  1.85
Conclusion

- Our Novel Design Is Capable Of Tracking WSS
  - With very low cost
  - With little accuracy loss
  - Orthogonal to existing optimizations

- More Details Are In Our Technical Report
THANKS!