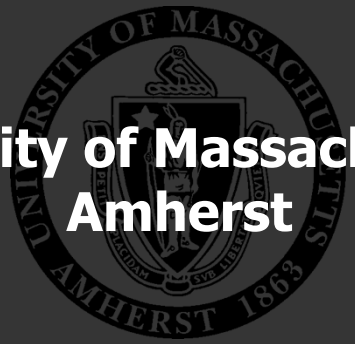
Transparent Contribution of Memory

James Cipar, Mark Corner, Emery Berger



University of Massachusetts, Amherst

Motivation

- Applications depend on contributed resources
  - CPU cycles, disk space, memory, network bandwidth etc.
  - Condor, Folding/SETI@home
- Compete with local applications for memory
  - Up to 50% performance impact on local applications

Transparent Memory Manager

- Operating system mechanism
  - Contributory applications can be run as-is
  - No special programming techniques or recompilation
- Contribute memory without affecting local performance
  - Detects local application's working set size
  - Limits contributory applications accordingly
- Dynamically adjusts to local working set
  - Can protect the performance of any local working set
  - Gives as much memory as possible to contributory applications

Overview: What it Does

- Trace memory accesses of local applications
- Simulate LRU queue of local memory
  - Include out-of-core memory in simulation
- Determine local working set size
- Limit contributory applications to the leftovers

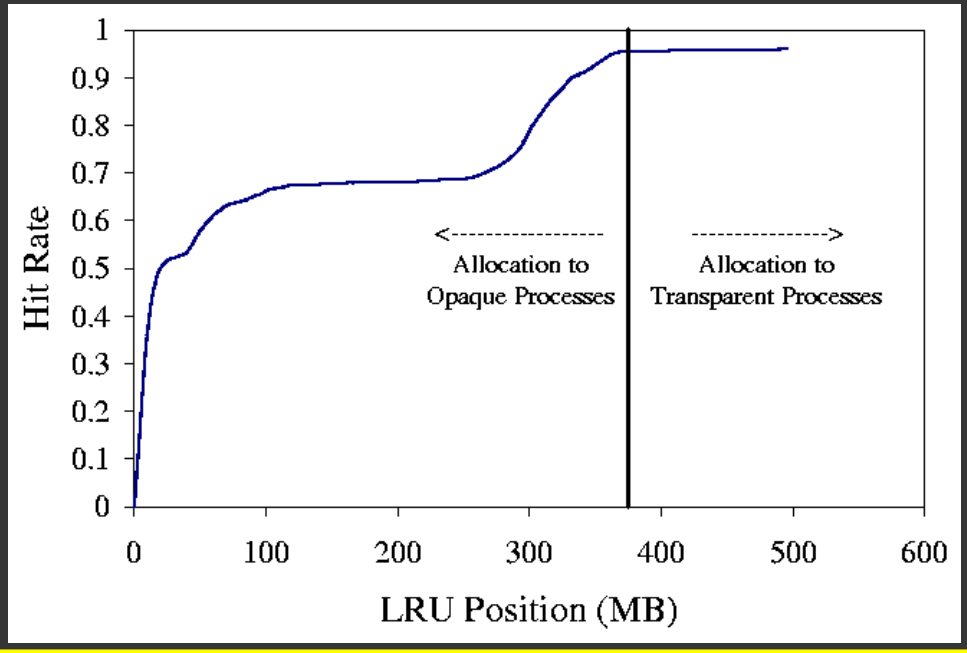
Measuring Working Set Size

Trace memory accesses

Simulate LRU queue

Cumulative histogram of LRU access pattern

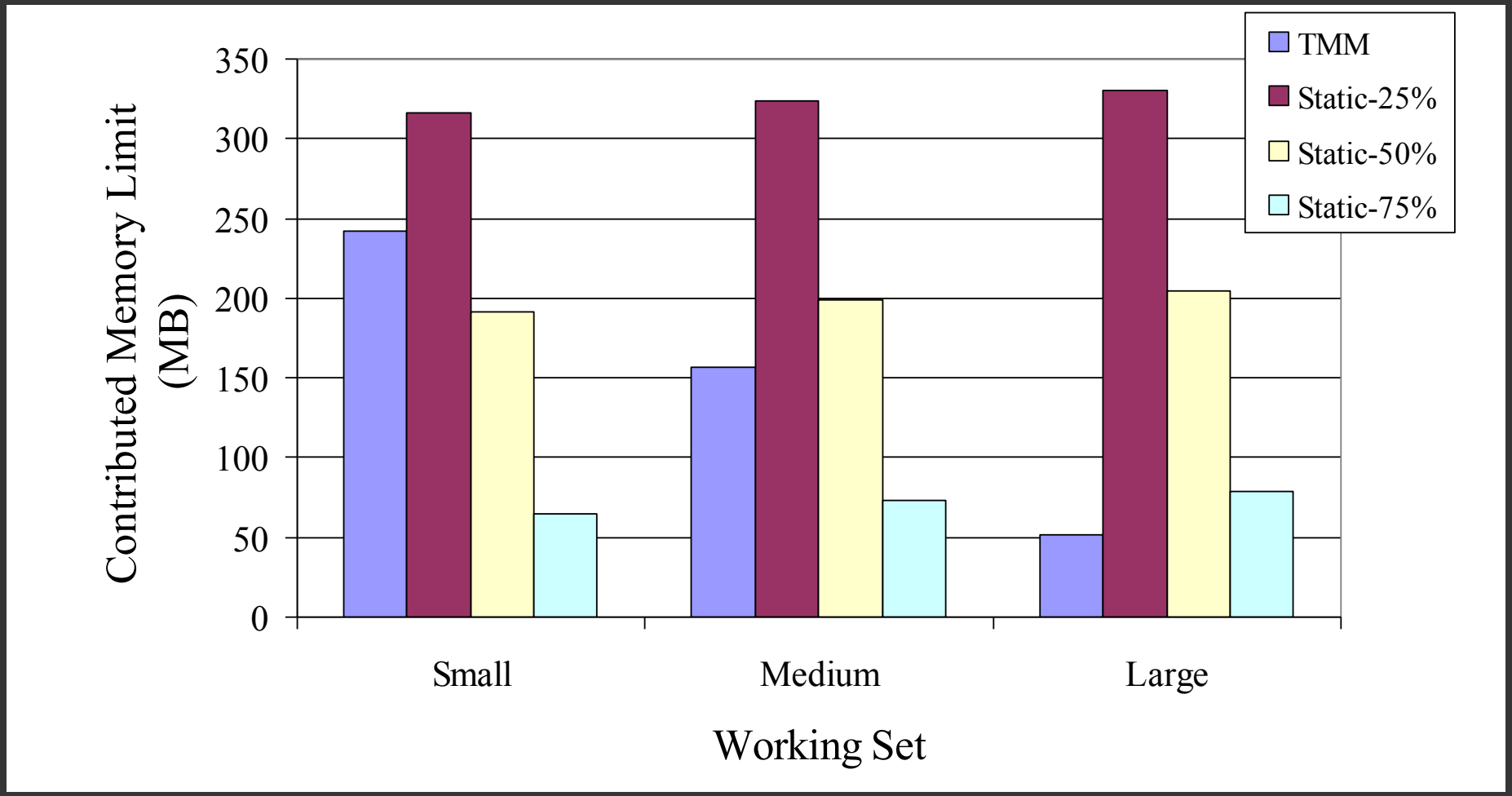
- This shows predicted hit rate for any allocation
- Allow a 5% increase in miss rate



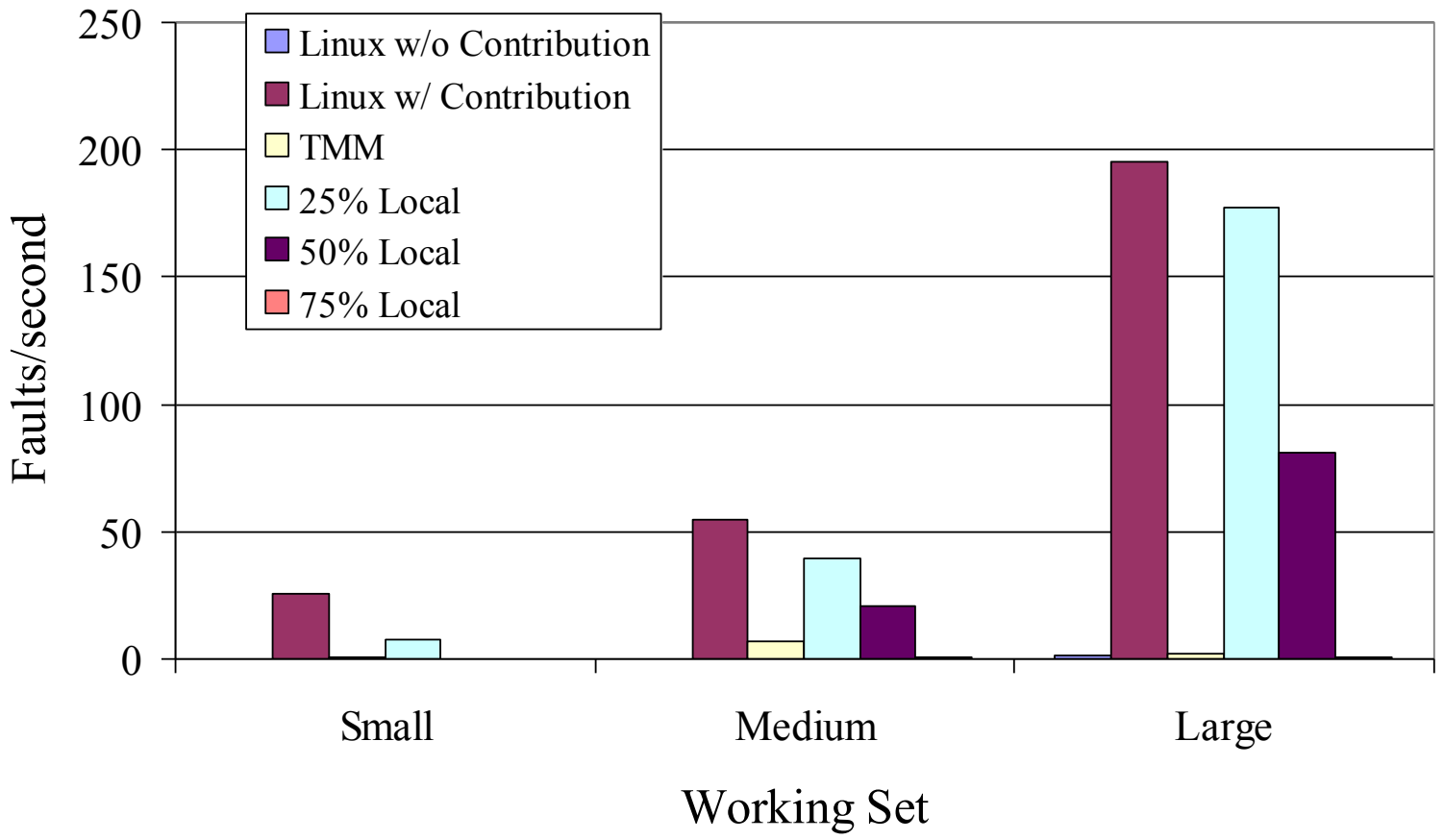
Experimental Setup

- Compare TMM to static allocations
  - How much memory is contributed
  - What is the performance impact
- Three typical working sets of different sizes
  - Web browsing
  - Viewing a large image
  - Plotting a large dataset
- User takes a break, and resumes after a few minutes
  - Contributory application runs during break
- Record page faults after user resumes work

Amount of Memory Contributed



Local Application's Page Faults



Related Work

Conclusions

Dynamically determines appropriate memory limits

Prevents local applications from being swapped out

Allows memory-intensive contributory applications