HEC FSIO Workshop, Arlington, VA, August 2-5, 2010

A Top-Down Approach to Dynamically Tune I/O for HPC Virtualization

Xubin (Ben) He

hexb@tntech.edu

Tennessee Tech University

Stephen L. Scott

scottsl@ornl.gov

Oak Ridge National Laboratory

Outcomes of this project

- Student Training
  - Two graduate students and two undergrads
- Related publications
  - "Hint-K: An Efficient Multi-level Cache Using K-step Hints", Proceedings of the 39th International Conference on Parallel Processing, September 13-16, 2010.
  - "A Top-Down Approach to Dynamically Tune I/O for HPC virtualization," Proceedings of the 4th Workshop on System-level Virtualization for High Performance Computing (HPCVirt), in conjunction with the 5th ACM SIGOPS European Conference on Computer Systems (EuroSys), April 2010. http://www.csm.ornl.gov/srt/conferences/hpcvirt2010/
  - "Investigating Locality Reformations for Cluster Virtualization", Poster presentation, the 8th USENIX Conference on File and Storage Technologies (FAST2010), Feb 23-26, 2010.
  - "An Extensible I/O Performance Analysis Framework for Distributed Environments", Euro-Par, Delft, The Netherlands, August 25-28, 2009.
  - "KVM on Clusters: Tackling the Disk I/O Bottleneck for HPC Virtualization", poster presentation at the 7th USENIX conference on File and Storage Technologies (FAST), Feb. 24-27, 2009, San Francisco, CA.

Why HPC & Virtualization

Virtualization in HPC provides exciting possibilities:

- **Build the system according to application.**
  - Right / Light weight kernels
- **The VM *is* the OS…**
- **Resilience possibilities.**
  - VM system migration
  - Migrate application
- **Security & Fault isolation**
- **Dynamic job consolidation.**
  - Interleave applications according to resources
  - Capability computing versus capacity computing
- **Legacy & Future system support & development**
  - Run apps on new hardware without code changes
  - Run apps on future hardware via VM environments

Mission

Provide a runtime framework for dynamically optimizing I/O on virtualized clusters via user-level tools.

Outline

- **Motivation:** Poor locality for virtual I/O and wealth of applicable user-level tools for tackling the problem.
- **Our solution:** ExPerT (**Ex**tensible **Per**formance **T**oolkit)
- Experimental results with pinning
- Conclusions & Future Work

The Current state of the Art

- New technologies have decreased the overhead of virtualization.
  - According to recent studies, virtualization only provides roughly 2-4% overhead in compute-bound scenarios.
- Intel and AMD have also provided hardware support to help boost performance at the CPU.
- Virtualization "platforms" have been rapidly maturing and are gaining widespread acceptance in other areas.

Motivation

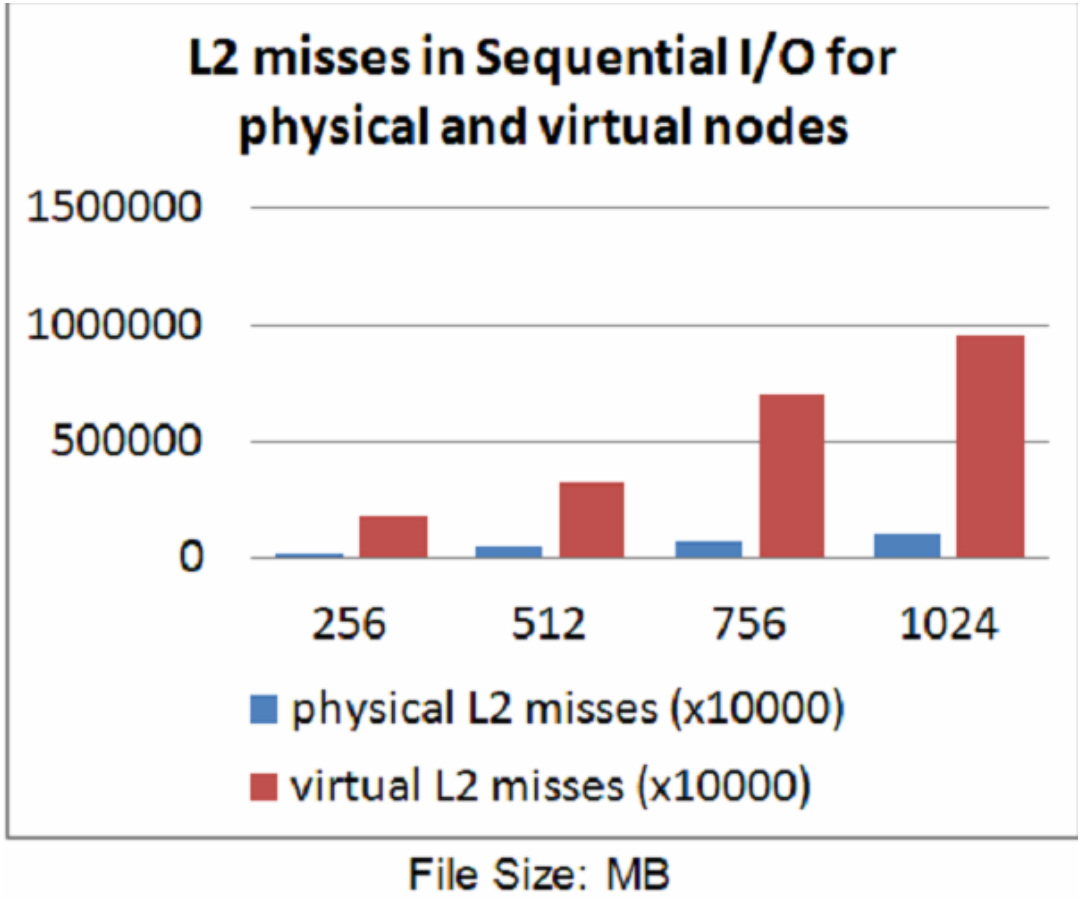
- More work needs to be done that focuses on improving I/O performance within Virtual Machines.
  - Additionally, most work has focused on network I/O and not disk I/O.
- This presents a problem in I/O bound applications in a High Performance Computing (HPC) environment where thousands of virtual machines (VMs) could be running on a limited number of compute nodes creating an I/O bottleneck.

Motivation (cont.)

- Specifically, we work with KVM, which uses virtio
- As I/O requests come in from more and more VMs on the system, virtio will become overloaded with requests and take up a high percentage of CPU usage.
  - Decreasing I/O throughput by decreasing I/O operations per second (IOPS).
  - An increased number of context switches and cache misses

Motivation (cont.)

- Virtualization causes large increases in cache misses
- Order of magnitudes difference
- Throughput suffers… roughly inverse linear correlation between throughput and L2 misses on i/o requests
- **Great opportunity for improvement here…**



Sequential write-intensive workload under VM versus non-VM (physical)

Motivation (cont.)

- Virtualization puts us in a unique position to perform in-depth system monitoring without instrumentation of hardware techniques
- The large performance gap in I/O motivates us to look at how we can leverage **the virtualization platform itself** to optimize the system

Our Solution

- To investigate the I/O bottleneck, we propose a testing and tuning framework with a combination of commonly found user-level tools in order to achieve greater performance.
  - The Extensible Performance Toolkit (ExPerT) is used in this work as it supports such a framework.
- The methods under study are primarily the use of pinning and prioritization. We focus on pinning in this talk.

Our Solution (cont.)

- We use pinning in order to lower cache misses when using virtio, as it is CPU intensive.
  - Pinning refers to the assigning core affinities to processes
  - This should increase the possible IOPS (input/output operations per second) and thus increase performance.
- We use prioritization in order to effect how each VM is scheduled.
  - We prioritize processes by changing their "niceness"
  - Scheduling an I/O intensive VM more frequently should increase I/O throughput vs. a fair scheduling approach.
  - Set process CPU affinity with "taskset"

Our Solution (cont.)

- W… in order to lower cache misses when using v…
  …affinities to processes
  …put
  schedu…
  - We prioritize pro…
  - Scheduling an I/O intensive v… increase I/O throughput vs. a fair schedu…
  - Set process CPU affinity with "taskset"

What is novel here?
- Design of the runtime toolkit
- Methods of auto-tuning via user level tools versus others that require kernel level mods

Research Methodology

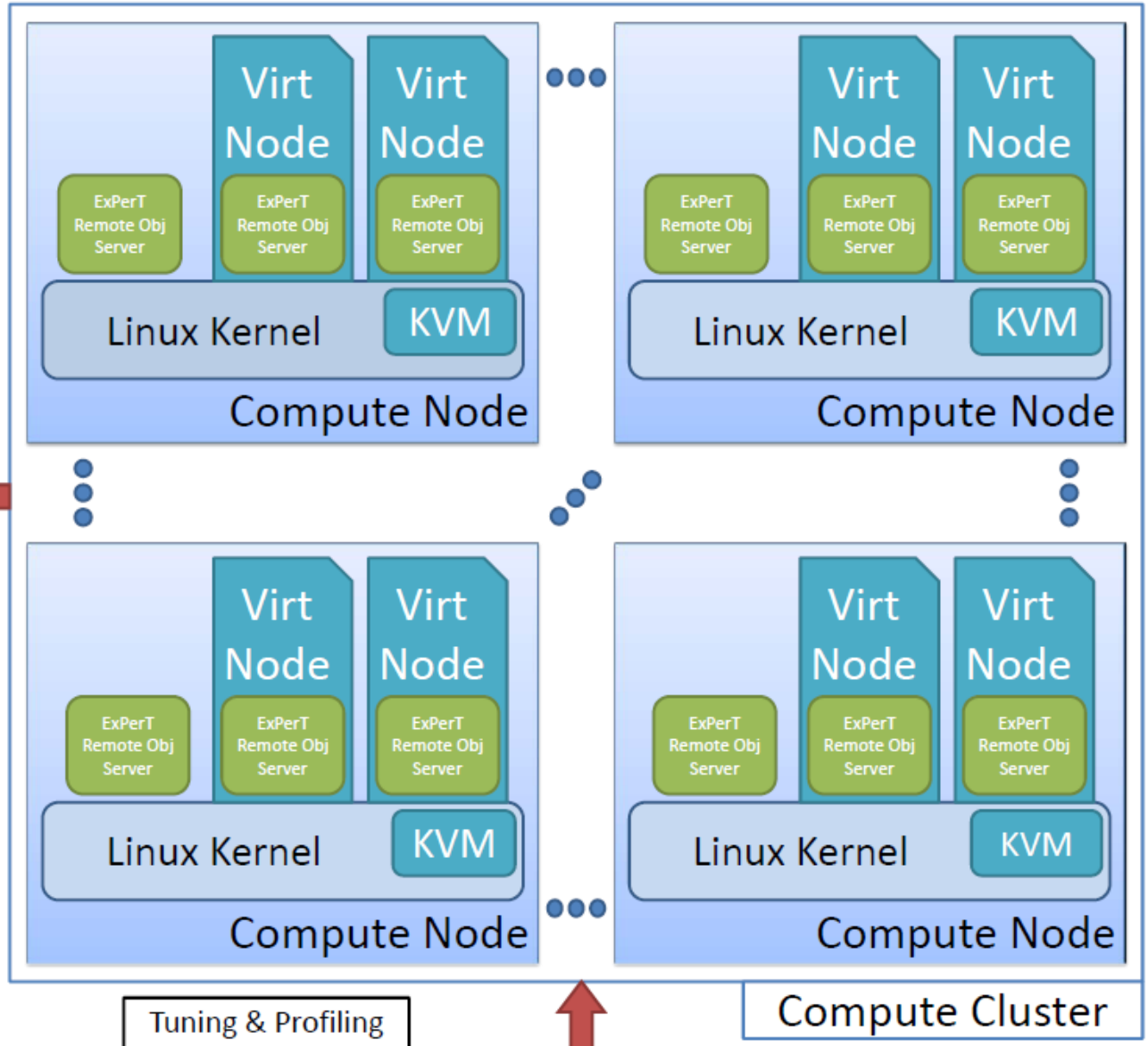
- We wish to look at the Kernel-based Virtual Machine (KVM) as it is more readily available to researchers since it is integrated in the main-line Linux kernel.
  - Simply loading a module loads the hypervisor.
  - VMs are deployed as processes
- User-level tools are used to both speedup development of this approach and to allow for the ease of reproducibility by other researchers.

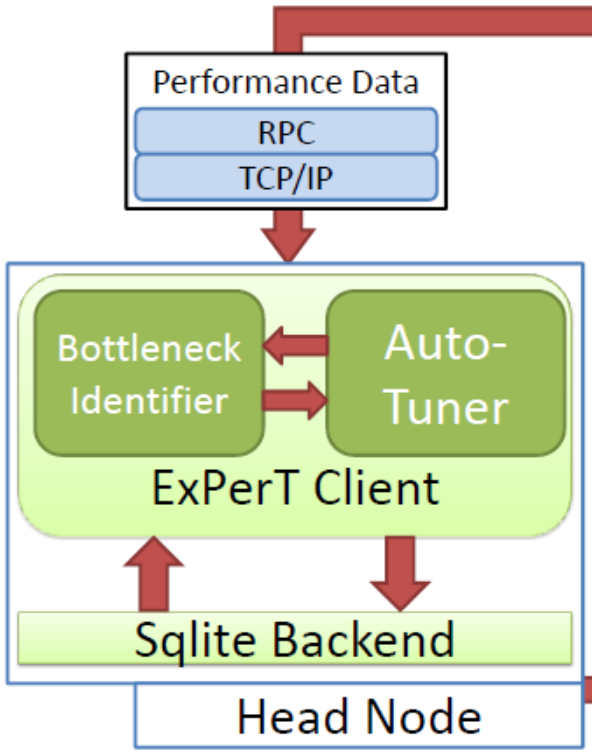
ExPerT

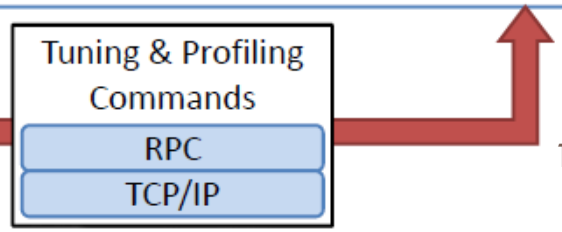
- Distributed testing framework with a database backend, visualization, and test suite creation tools for virtual systems.
- Updates its database in real-time.
- Closely integrates with Oprofile, vmstat, and the sysstat suite of tools.
- Uses a distributed object model.
- Support for automatic tuning and optimization.

The Framework (architecture organization)

- Client on head node registers compute node list & starts remote servers
- RPC achieved through remote objects using remote servers and TCP/IP
- Profiling occurs on both physical compute node and each virtual node
- Bottleneck Identifier and Auto-tuner work in concert to analyze and send tuning commands. Bottleneck Identifier analyzes performance data and auto-tuner sends tuning commands

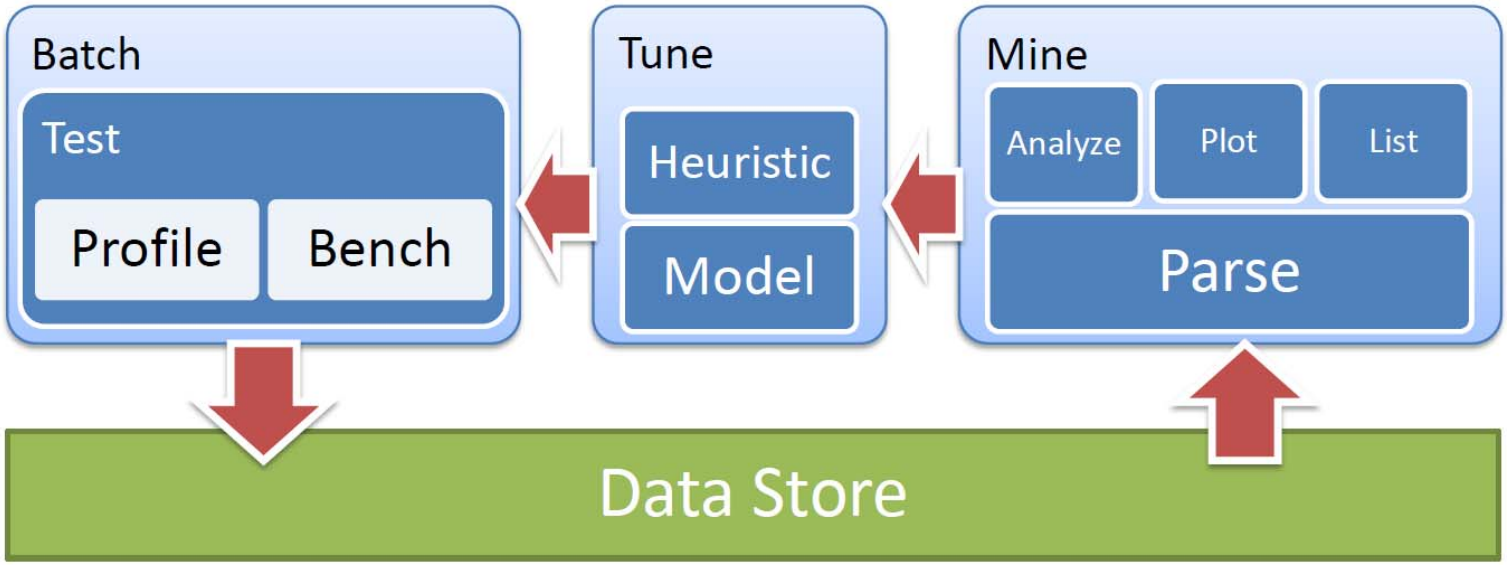






The Framework (logical organization)

- Consists primarily of three parts:
  1. **Batch:** a test creation tool.
  2. **Tune:** a tuning tool.
  3. **Mine:** a data discovery tool.



Batch

- Object-Oriented design
- Uses remote objects
  - **RemoteServer:** a remote process server which maintains a list of processes and defines the methods through which they can be controlled.
  - **RemoteProgram:** contains the basic functionality for communication over the network including the ability to control remote processes.
    - E.g. starting, killing, waiting, gathering output and sending input.

Mine

- Utilizes the results collected from the batch phase.
  - All results during the batch phase are not parsed and instead mine accomplishes this task.
- Allows for the visualization of the results.
  - Through an interactive wizard
  - Or through a declarative syntax similar to the configuration syntax

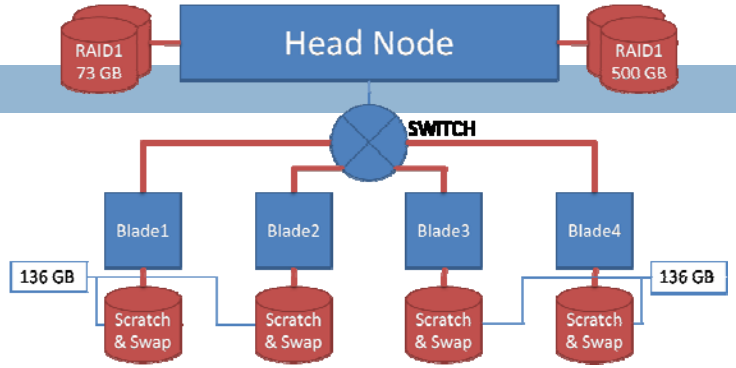
Mine (cont'd)

- Why does mine do the parsing and not batch?
  - **Flexibility:** our parser may change, losing or gaining attributes. Lazy parsing does not lock in past tests.
  - **Efficiency during:** since we delay parsing, we save computation during the data collection process.
  - **Efficiency after:** we can selectively parse out data as we need it (parse on demand).
  - **Lossless accounting:** we can always look at raw output if we need it since parsing for attributes will necessarily remove data.

The Data Store

- A wrapper for sqlite and is essential for making the data coming into the database a standard format.
- The general schema of the database consists of three tables:
  - A high-level batch table that lists saved batch results and short descriptions.
  - A table that lists individual processes and their unique id within a batch.
  - A table that lists raw process output, per line, for a uniquely identified process.

Experimental Testbed

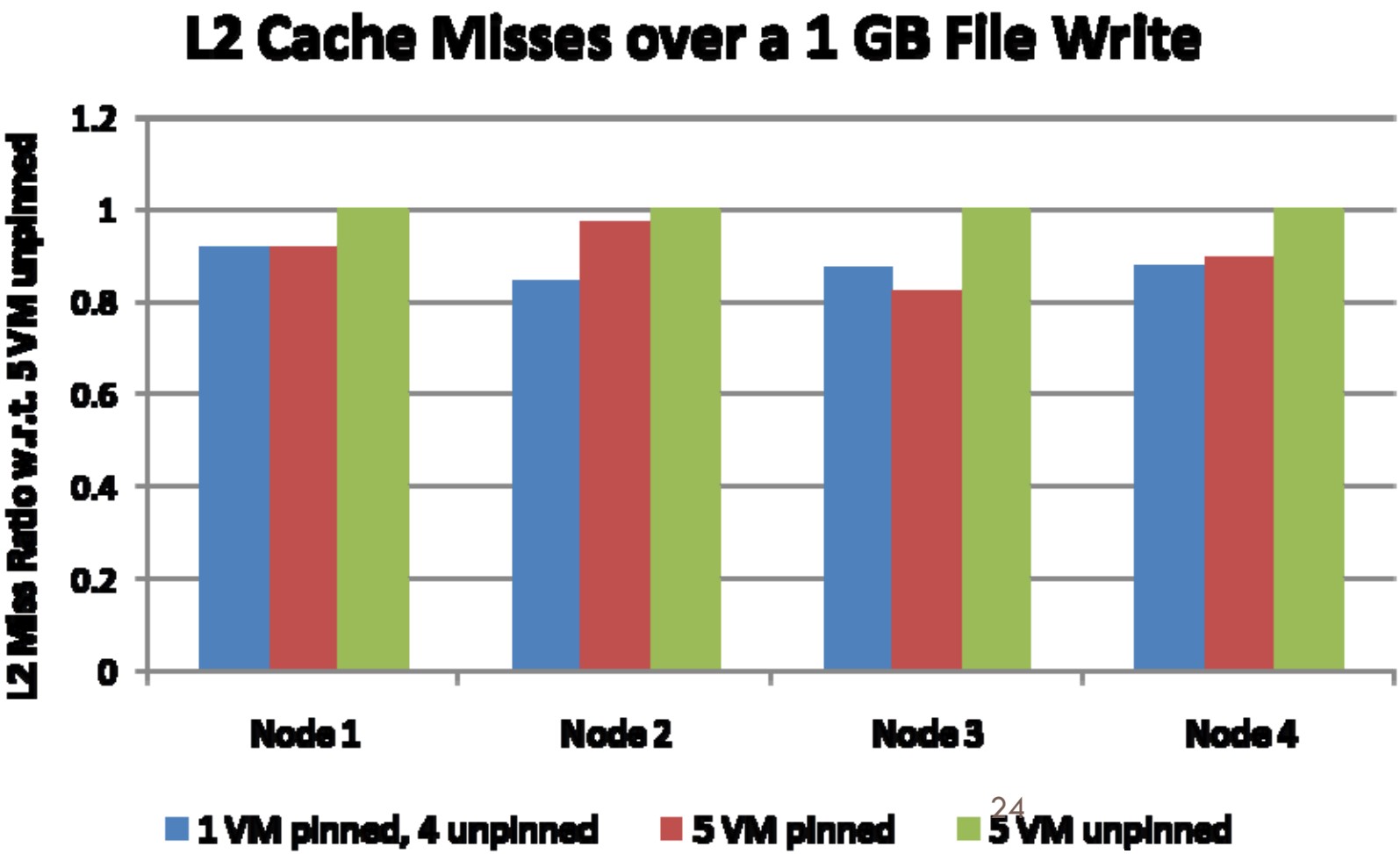


- Mach4
  - 4 node cluster
  - Each node contains two quad-core Xeon 5520 CPUs and 6 GBs of ram
- Used ExPerT and KVM to examine two policies with 5 VMs per node:
  1. Pinning only one VM to a core while performing iozone write benchmarks.
  2. Pinning 5 VMs to a core while performing iozone write benchmarks.

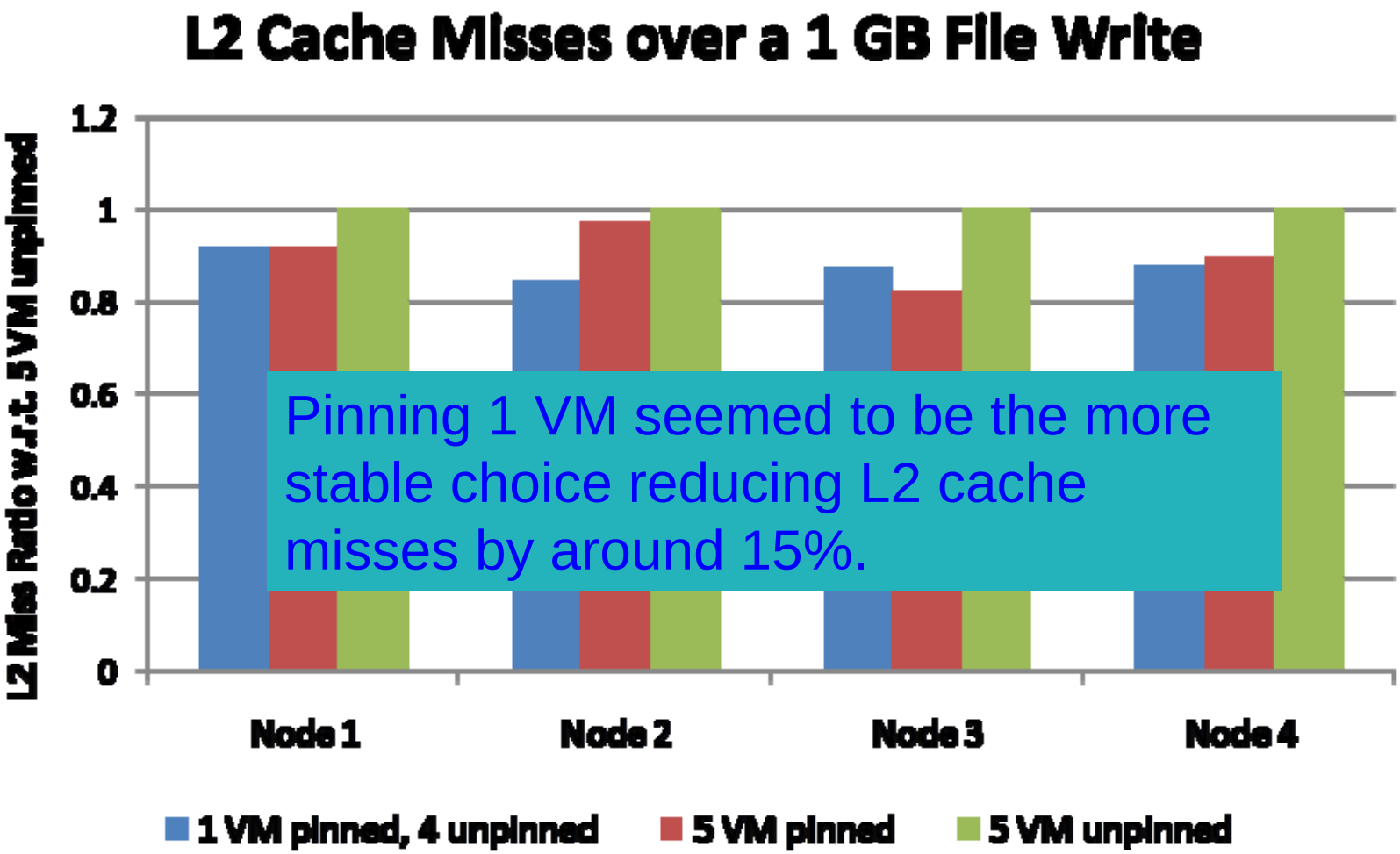
Workflow

- A general workflow consists of the following steps:
  - Start virtual machines, and start the RemoteProcess server on every node, physical and virtual (this may be a startup script).
  - Create a configuration file specifying the batch test(s) to be run, the identification and tuning policies, and the machine map.
  - Run Batch from the head node with the configuration file specified.
  - (Optional) Run any of the post-mortem tools (Mine) for further analysis

Experimental Results

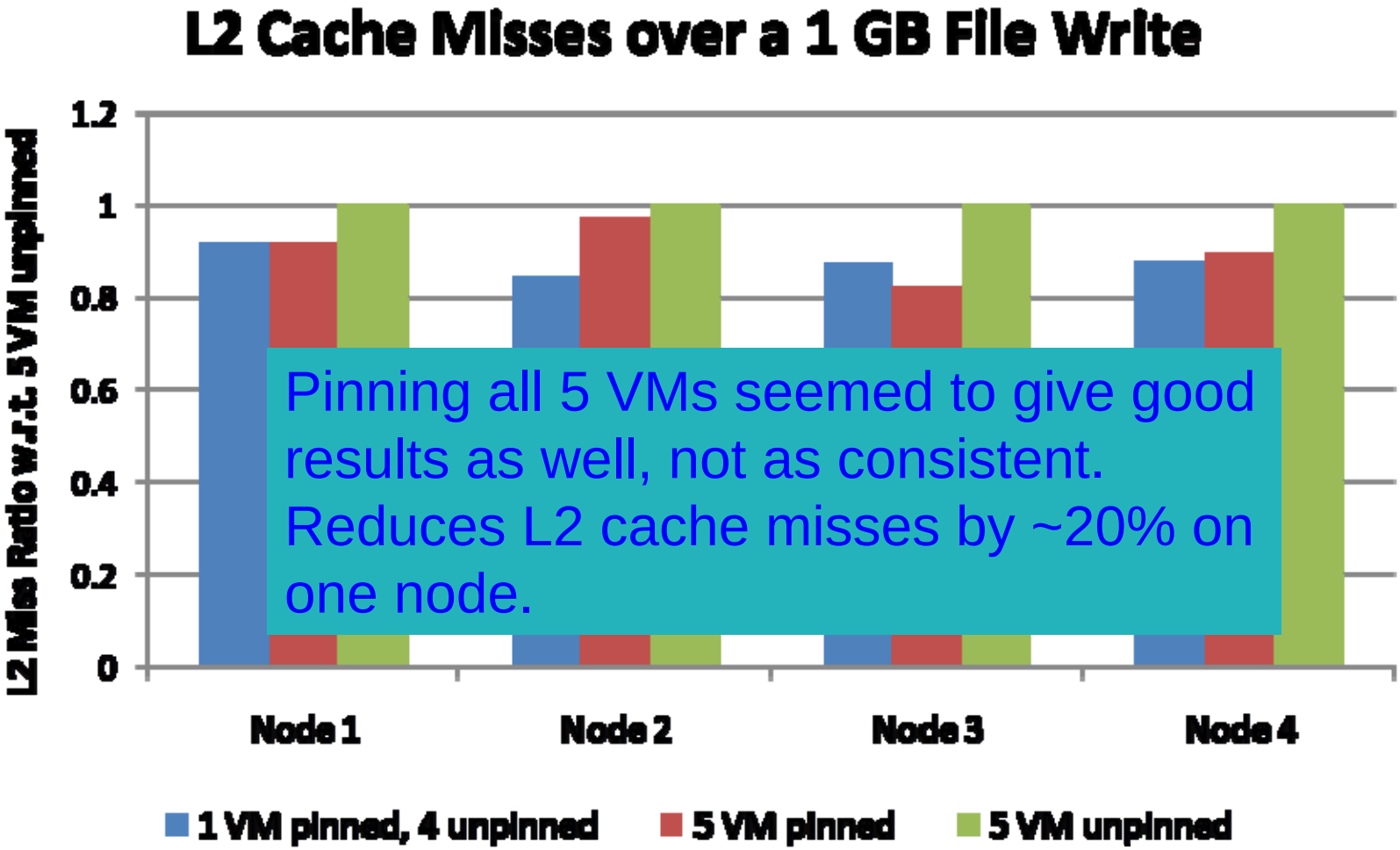


Experimental Results



Pinning 1 VM seemed to be the more stable choice reducing L2 cache misses by around 15%.

Experimental Results



Pinning all 5 VMs seemed to give good results as well, not as consistent.
Reduces L2 cache misses by ~20% on one node.

Conclusions

- There are ways to alleviate the I/O bottleneck by using simple user-level tools.
- In comparison to related work, we consider the use of such a toolset as "performance for free" since we do not compromise portability by modifying the kernel, locking one into a particular platform, etc.
- Through the pinning of VMs it is possible to decrease L2 cache misses by up to 20%.

Future Work

- We wish to move to a more automated approach of self-optimization (using user-defined policies)
- We would like to look towards using more lightweight protocols than TCP/IP for our remote objects usage for increased scalability.
- We would like to investigate other methods of dynamically changing the properties of virtual machines to modify their performance.
- Investigate scalability issues of our methods.

Acknowledgments

- Graduate students: Ben Eckart, Ferrol Aderholdt, and Juho Yoo

- This work is sponsored by U.S. NSF under Grant No. CCF-0937850

HEC FSIO Workshop, Arlington, VA, August 2-5, 2010

A Top-Down Approach to Dynamically Tune I/O for HPC Virtualization

Xubin (Ben) He

hexb@tntech.edu

Tennessee Tech University

Stephen L. Scott

scottsl@ornl.gov

Oak Ridge National Laboratory