Performance Tools – Thoughts and Experiences

Seetharami R. Seelam
The University of Texas - El Paso
Computer Science
seelam@cs.utep.edu

Outline

- Issues facing performance analysis research today and in the future
- What should the community do?
  - Standard APIs
  - Smart data collection and analysis
  - Dynamic filtering and instrumentation
- Steps to address some of the issues that performance analysis is facing

Outline

- Issues facing performance analysis research today and in the future
- What should the community do?
  - Standard APIs
  - Smart data collection and analysis
  - Dynamic filtering and instrumentation
- Steps to address some of the issues that performance analysis is facing

Performance Analysis: Current and Future Issues - 1

- Performance problems are vaguely defined
- Fine-grained instrumentation generates GBs of data; coarse-grained instrumentation looses important information
- Appropriate sampling frequency must be determined; use hints from application programmers
- Correlation of GBs of data to program and system components is difficult

Performance Analysis: Current and Future Issues - 2

- Providing checks and balances seems to be a difficult problem
- Scalability of tools is a big concern and it is growing
- Adaptivity in application, operating system or architecture increases complexity of related issues
- Poor communication with application developers exacerbates problems

Outline

- Issues facing performance analysis research today and in the future
- What should the community do?
  - Standard APIs
  - Smart data collection and analysis
  - Dynamic filtering and instrumentation
- Steps to address some of the issues that performance analysis is facing

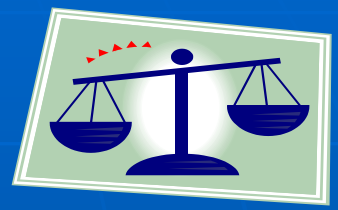
What should the community do?

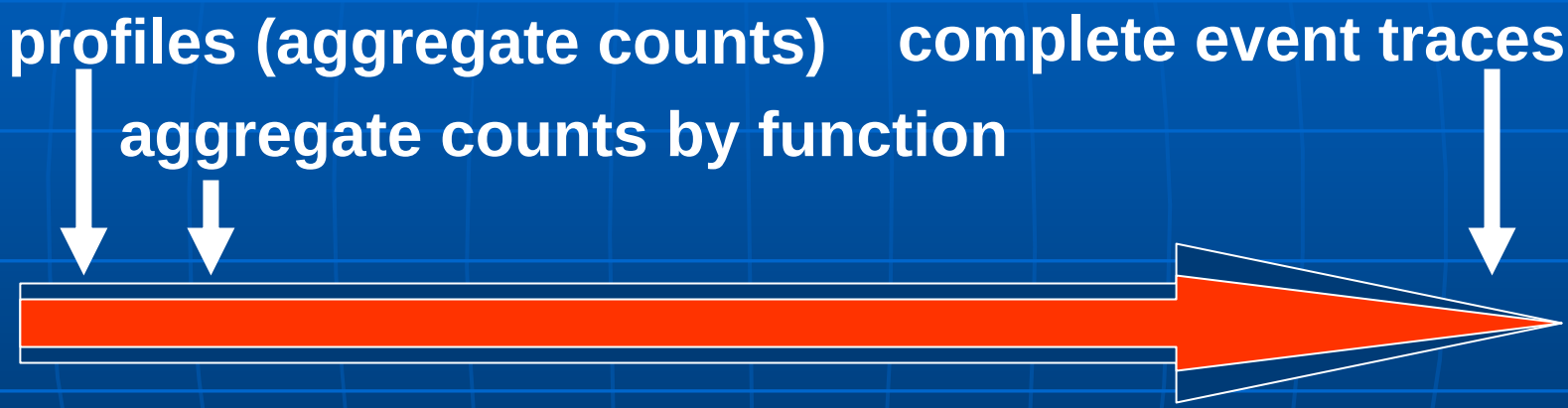
- Provide standard APIs
- Collect relevant performance data
- Collect data via data-centric instrumentation
- Provide means for analysis
- Automatically identify relevant performance data and bottlenecks

Provide Standard APIs

- Well-defined interfaces (e.g., POMP)
- Ability to tightly control fine-grained instrumentation
- Ability to invoke instrumentation per function, per thread, per process, per system
- Instrumentation based on flow of data, as well as flow of program control
- Provision of profile and trace libraries

Collect Relevant Performance Data – Scalability Tradeoffs





less storage
less overhead
lower level of detail

more storage
more overhead
higher level of detail

Data Collection via Data-centric Instrumentation

- Control-centric performance information
- Data-centric performance information
  - Understanding precise memory references is crucial to efficient memory hierarchy utilization
  - Example: c2c identifies performance problems not with control flow but with respect to flow of information through data structures

Example: Analysis of CG code

```
1  !$omp do
2    do j=1,naa+1
...
3      p(j) = r(j)
4    enddo
...
5  !$omp do
6   do j=1,lastrow-firstrow+1
7     sum = 0.d0
8     do k=rowstr(j),rowstr(j+1)-1
9       sum = sum + a(k)*p(colidx(k))
10    enddo
11    q(j) = sum
12 enddo
```

CG code segment with a large number of cache line transfers

```
on load    for a1 do call myHandler
on store   for a1 do call myHandler
on load    for a2 do call myHandler
on store   for a2 do call myHandler
on load    for a3 do call myHandler
on store   for a3 do call myHandler
nCaches 2
TLBEntries 256
TLBAssoc 2-way
TLBRepl LRU
L1Size 64Kb
L1Linesize 128b
L1Assoc 128-way
L1Repl LRU
L2Size 4Mb
L2LineSize 128b
L2Assoc 2-way
L2Repl LRU
```

Figure 3. Event/Architecture Specification

80% total program cache transfers

Array	Function	Memory References	Stores	Cold Misses	c2c transfers
p	conj_grad@OL@B@OL@F	741241600	0	2624	1048576
a	conj_grad@OL@B@OL@F	741241600	0	16588	70057
z	conj_grad@OL@B@OL@13	29649664	0	2621	39427
z	con_grad@OL@B@OL@11	11200000	5600000	0	2395
colidx	conj_grad@OL@B@OL@F	741241600	0	0	403

cache-to-cache transfers in NAS CG

Outline

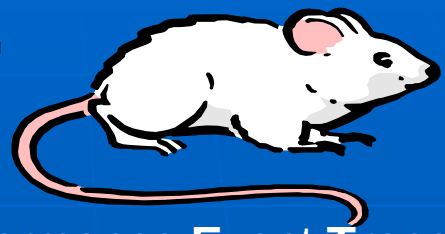
- Issues facing performance analysis research today and in the future
- What should the community do?
  - Standard APIs
  - Smart data collection and analysis
  - Dynamic filtering and instrumentation
- Steps to address some of the issues that performance analysis is facing

Smart Data Collection & Analysis

- Leverage efficient storage techniques from databases
- Use compression as well as sampled event traces to reduce the amount of data
- Use multivariate statistical analysis to eliminate redundant data
- Utilize autonomous agents to dynamically capture and analyze relevant performance data, and “throttle” the amount and type of information collected

Smart Data Collection & Analysis - 2

PETrAT



Performance **E**vent **T**race **A**nalysis **T**ool

Diana Villa's Master's Thesis – Work with IBM-Austin

Data Collection Environment

TPC-C

p690

Sampled Event Traces

PID TID Timestamp Instr.Addr. DataAddr.
PID TID Timestamp Instr.Addr. DataAddr.
PID TID Timestamp Instr.Addr. DataAddr.

Load DB Java Tool

Database

Queries

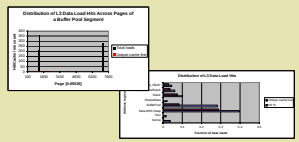
Report Generation Java Tool

Reports

5 BufferPool 56893 29384
6 Data,BSS,Heap 8799 4855
1 Kernel 23485 9840

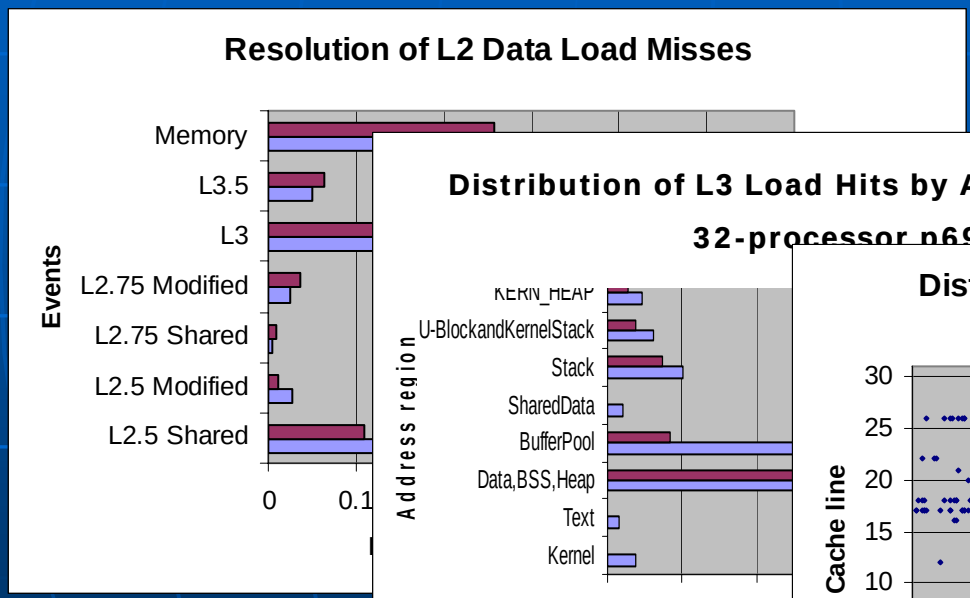
Microsoft® Excel

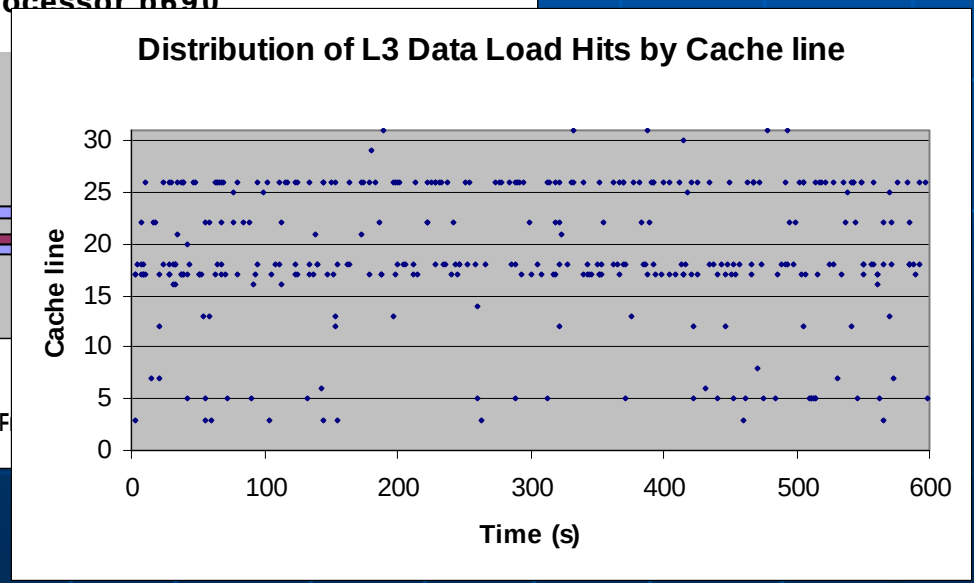
Graphs



Smart Data Collection & Analysis - 3

Sampled Event Traces – Analysis & Results





Smart Data Collection & Analysis - 4

Multivariate Statistical Analysis

Scatterplot /correlation matrix to identify redundant counters

High correlation – just measure one, predict others

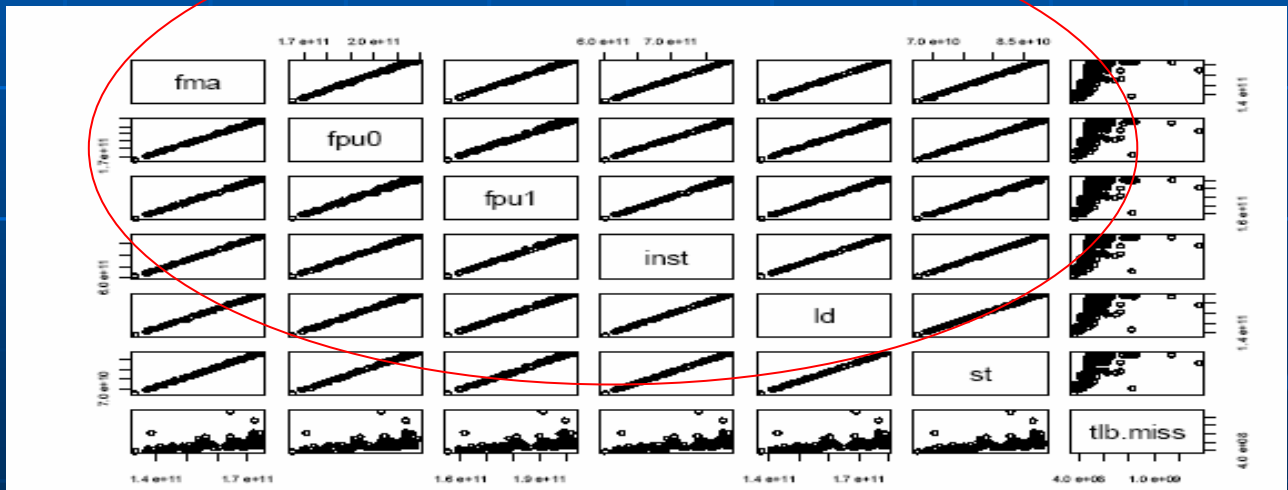


Table 3: Scatterplot matrix for UMT on 288 tasks with raw data from 7 counters.

Smart Data Collection & Analysis - 5
Automatic Identification of Relevant Performance Data

- Autonomous agents that can activate/deactivate or throttle instrumentation based on possible performance problems
- 9.9% of the data contains 89.1% of the problems
- Agents that can map between data flow and control flow of information
- Leverage dynamic adaptation techniques from OS or application layers

More Observations

- We need thermometers and stethoscopes
- X-rays and MRIs are what we have
  - Programmers need to know why the bone is broken
  - We are only showing that the bone is broken
- We need automatic tools to analyze the information
- Finding methods that discard useless information is critical

Addressing the Scalability Challenges of Performance Data Collection & Analysis

- Standardization of APIs
- Scalable data collection
- Throttled data collection
  - Dynamically adapt w.r.t. data being collected
    - Type and amount
- Adaptive instrumentation and data collection
- Use of off- and on-line analysis to adapt heuristics that trigger adaptations
- Autonomous agents
- Scalable analysis of trace data (database queries, EARL modules)
- Scalable visualization of analytic results

Acknowledgements

- David Klepacki and the IBM team that made this workshop a reality
- My mentors and collaborators at IBM: Simone Sbaraglia, Eknath Kattamuri, and Luiz Derose (now at Cray)
- Collaborators from IBM-Austin
- My Ph.D. advisor, Patricia Teller

References

- S. Sbaraglia, K. Ekanadham, S. Crea, and S. Seelam, “pSigma: An Infrastructure for Parallel Application Performance Analysis using Symbolic Specifications,” *Proceedings of the Sixth European Workshop on OpenMP - EWOMP'04*, October 2004.
- L. DeRose, B. Mohr, and S. Seelam, “An Implementation of the POMP Performance Monitoring Interface for OpenMP Based on Dynamic Probes,” *Proceedings of the Fifth European Workshop on OpenMP - EWOMP'03*, September 2003.
- D. Villa, J. Acosta, P. J. Teller, B. Olszewski, and T. Morgan, “A Framework for Profiling Multiprocessor Memory Performance,” *Proceedings of the 10th International Conference on Parallel and Distributed Systems*, July 2004.
- R. Portillo, D. Villa, P. J. Teller, and B. Olszewski, “Mining Performance Data from Sampled Event Traces,” *Proceedings of the 12th Annual Meeting of the IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2004)*, October 2004.
- A. Dong and J. S. Vetter, “Scalable Analysis Techniques for Microprocessor Performance Counter Metrics, “ *SC2002 High Performance Networking and Computing Conference,* November 2002.