

PerfMiner: Cluster-wide Collection, Storage and Presentation of Application Level Hardware Performance Data

Europar 2005: Philip Mucci, Daniel Ahlin, Johan
Danielsson, Per Ekman and Lars Malinowski

Presented by Michael McCracken

Background

- Hardware performance counters
 - Events: Cache hits, branch mispredictions, total instructions, etc.
- PAPI: portable access to performance counters
- PapiEx: utility to run its arguments with PAPI instrumentation

Problems

- Inefficiencies in HPC system administration and use stem from ignorance of workloads
- Suboptimal codes with large allocations waste CPU time
- Inappropriate systems purchased
- Programmers aren't measuring their code
- HW counters let us measure efficiency

Previous work

- /proc based approaches
 - Linux, low granularity
- NCSA PerfSuite
 - Linux, HW counters, less web-focused

Solution

- Using HW counters can be difficult on a per-project basis, so:
- Measure everything with HW counters!
 - every process, every thread, all the time
- Do it without perturbing jobs.
- How do we store and use the info later?

Transparency

- Must be invisible to users - where to intercept jobs?
- Wrap shell on compute nodes, and modify batch submission script
- PapiEx Uses dynamic linker to transparently instrument

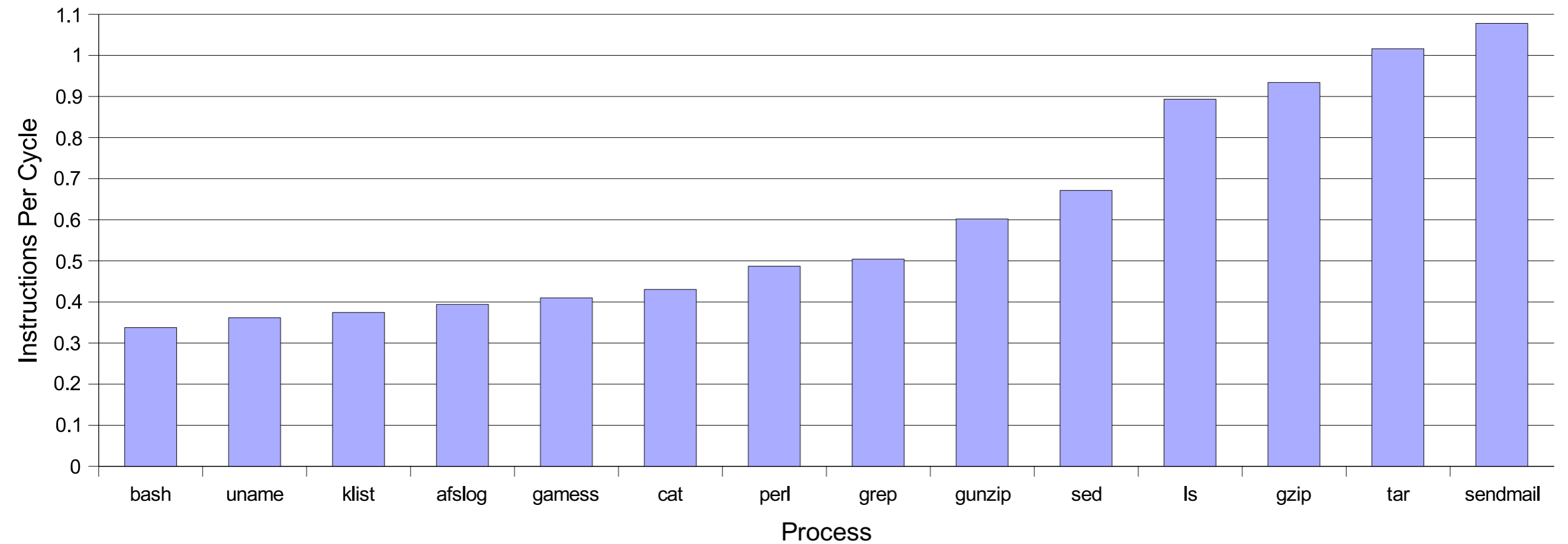
Scalability

- Decoupled data writing and insertion into database
- how much data?
- query efficiency: processing done on server

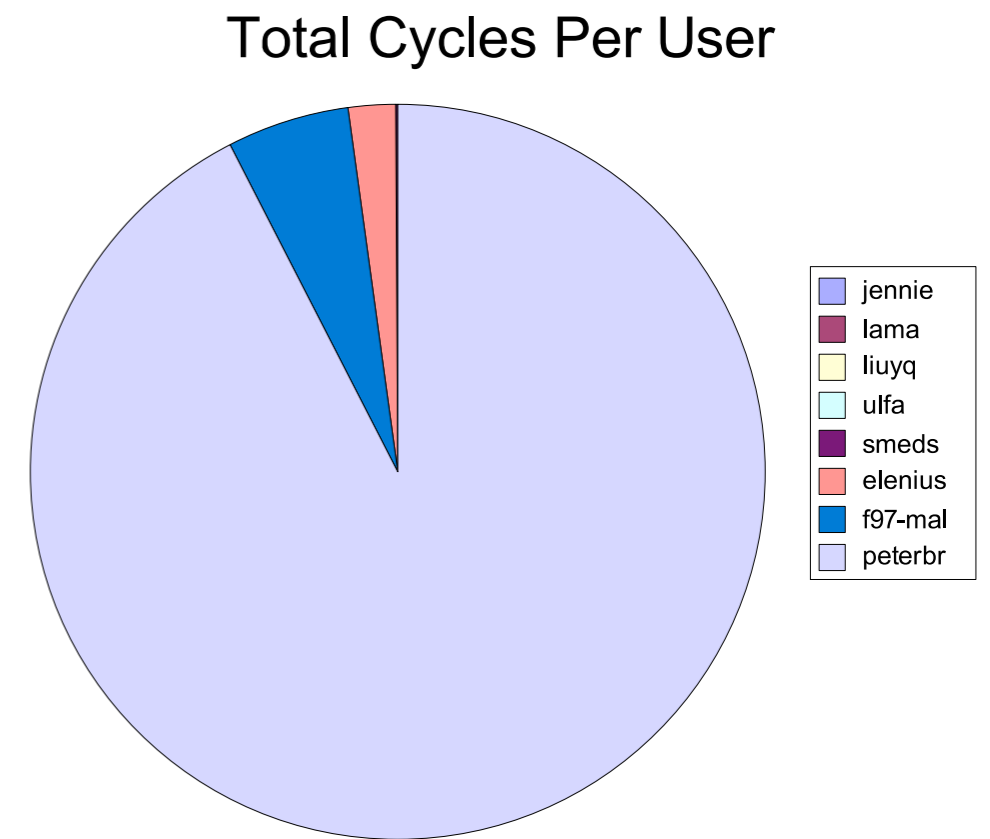
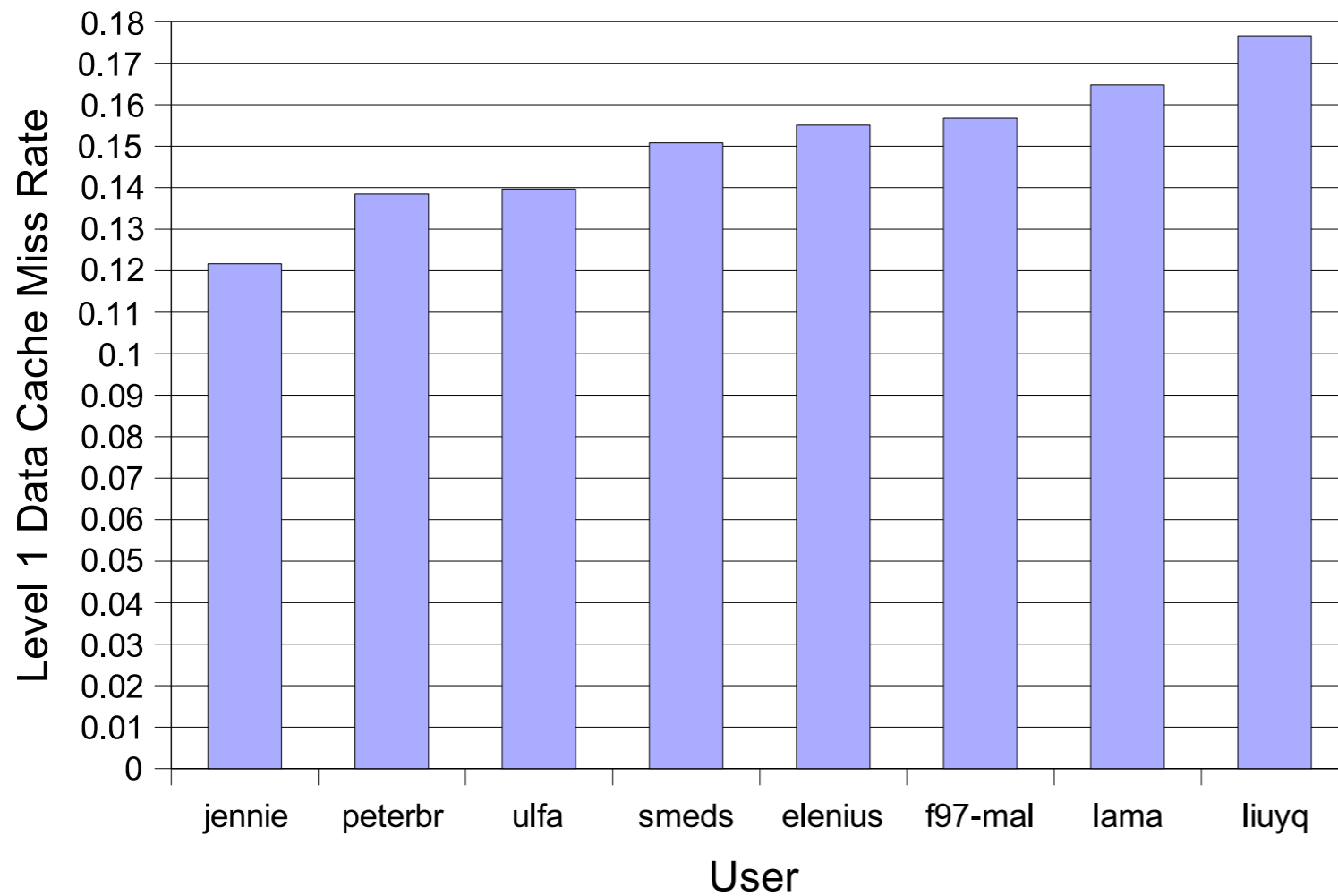
Interface

- Exposed limited amount of data
- Web interface with graphing
- Not an interactive tuning environment
- Most useful for system owners/
administrators?

Single Job Evaluation



User comparisons



Questions

- Real contributions?
 - portability, focus on interface, and?
- Will this work as well for other systems?
- Can HW counters really give us a good measure of efficiency?
- Automatic data gathering “behind our back” - what other data could be useful?