Analysis of **performance monitoring** on production batch servers

being done by:
Andreas Hirstius
Paola Cecuk
Gyorgy Balazs

October 2008

Gyorgy Balazs

Performance monitoring

•Access low level information concerning running jobs

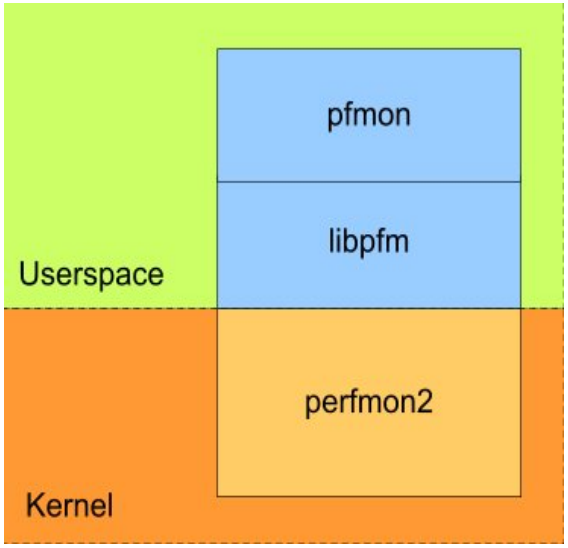
•Modern CPUs provide real time statistics on executed instructions via a Performance Monitoring Unit (PMU)

•A robust tool to access PMU counters:

Perfmon2 –pfmon

•System wide monitoring

•Only minimal overhead on the monitored system



Production batch servers

- CERN Batch Farm of 3000 machines
- Main users: Experiments
- Used for:
  - Data analysis
  - Simulation
  - Reconstruction
  - Etc. Etc. Etc.
- Accessible for everyone from lxplus and the grid (!)
- Queues for jobs of different length: 8nm,1nh,1nw…
- Jobs are scheduled by LSF

Goals

- To get some data:

  - Get low level, system wide information about the running nodes
  - Detailed, instruction level information about jobs
  - Reveal possible performance bottlenecks

- To answer some questions:

  - Who (which experiments) are using the nodes?
  - What are they running?
  - Does it fit their claims?

Monitoring

- 60 std. Batch nodes
- 49 days
- 761 users
- ~200.000 jobs

What to do with the raw data?

- Raw data comes on an hourly basis from each machine

- Generate different reports:
  - oHourly report per node
  - oDaily report per node
  - oAggregated results per node
  - oAggregated results all nodes

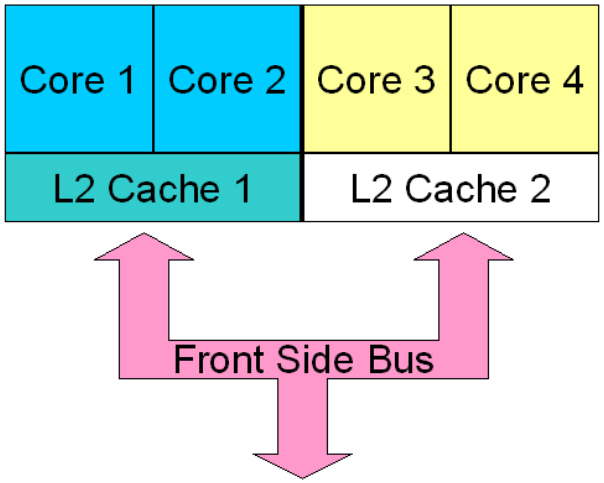
- Merging reports with LSF logs:
  - oGet user and queue data
  - oPin jobs to experiments
    - ccid->groupid->experiment

- Generate files:
  - oAll data/node
  - oSimplified view/node
  - oExperiments

Gained information

Performance information:

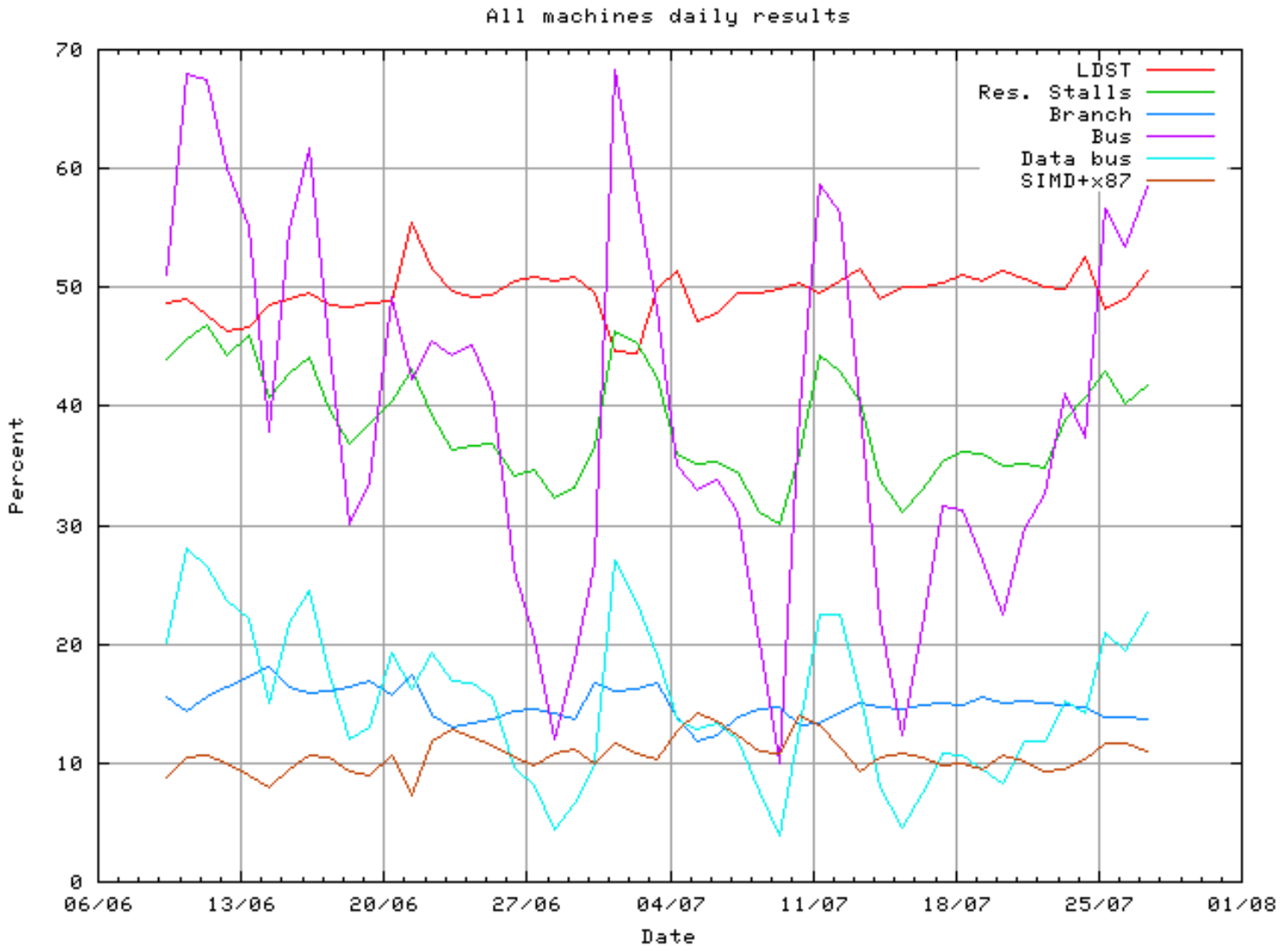
- Unhalted Core Cycles
- Instructions retired
- Cycles Per Instruction (CPI)
- Branch instructions %
- Bus utilization %
- Data load/store %
- L2 cache misses %
- SIMD/x87 %



Running jobs:

- Username
- Group
- Experiment
- Job queue

Aggregated results for all nodes



Analysis of 32bit vs. 64bit applications

- Indicator: x87 (32bit) and SIMD (64bit) instructions

- What are we looking for?
  - Percentage of all instructions
  - Ratio at each experiment
  - Does it fit the claims?

- Problem: one experiment/node is very rare
  Normally 8-9 jobs from 2-4 experiments running on each node

- Solution: find timeframes when one experiment is dominant for a longer time

Analyzing dominant users:

Selecting timeframes, when at least 80% of the running jobs belong to one experiment

Experiment	Hours when dominant	All SIMD+x87 instr.
NA58	4406	~12%
ALICE	565	~7%
CMS	64	~10%
ATLAS	48	~10%

32/64 bit usage at the experiments

	Experiment	Principal instr.	Utilization rate
Dominant users	ATLAS	x87	~90%
	ALICE	SIMD	~90%
	CMS	x87	~90%
	NA58	x87	~90%
Minor users	AT	x87	>90%
	IT-GEANT4	SIMD	>90%
	IT-GEAR	SIMD	>90%
	IT-DTEAM	x87	low amount of data
	LHCB	x87	>90%
	NA45/2	x87	>90%
	PH-TH	SIMD	low amount of data
	PS212	x87	low amount of data

The next steps…

- Analyzing timeframes when the same jobs are running, but the performance is changing
- Castor's impact on performance
- How much data can be extracted from performance monitoring?
- How can we automate the process?

Q & A