Performance Analysis

CSE 410, Spring 2004
Computer Systems

http://www.cs.washington.edu/education/courses/410/04sp/

Readings and References

• Reading
  » Chapter 2, Computer Organization & Design, Patterson and Hennessy

• Other References

Performance Analysis - When?

• Evaluation prior to system purchase
  » deciding which system is right for the need
  » very speculative and uncertain
    • unknown future workload, future capability
• Tuning prior to product release
  » product must meet expectations or it won’t sell
  » very speculative and uncertain
    • sensitive to configuration, workload

Understand your Environment

• The question is “what system to buy?”
  » or design to select, or vendor to hire, etc
• Performance analysis is fun
  » you get to look at all sorts of machines and their inner workings
• It’s very easy to analyze the wrong things ...
  ... and then your analysis will lead you to a wrong answer
Time to complete task

- The best metric is time to complete the task, as perceived by the user
- Total performance is the sum of many individual factors and perceptions
- Understand the task and the expectations
  » write down the tasks to be accomplished
  » write down the user expectations for performance

Evaluation tools

- Simulation
  » necessary if there is no system in place yet
  » accuracy of the simulation is critical
    - how is accuracy defined?
- Prototypes
  » accuracy is critical - it looks real, but you don’t have the real product in hand yet
    - where are the simplifications, are they important?

Benchmarks

- High potential for misleading results
- May not reflect actual operation at all
  » reading from disk controller cache, not disk
  » product vendor tweaked system for benchmark
- Simplified representation of the workload may completely miss critical factors
  » eg, operating system performance under heavy user load

Better benchmarks

- Benchmarks derived from real applications
  » Standard Performance Evaluation Corp (SPEC)
- Test hardware
  » simulated users at simulated keyboards
- Prototype deployment
  » actually build an initial version of the application and deploy it on a limited scale
Validate!

- Your users do not think of the system the same way you do - guaranteed!
- Document your assumptions
- Use several different types of benchmarks
- Keep an open mind
  » pay attention to business issues too, or you will be surprised when your choice is not selected

SPEC Benchmarks

- “Establish, maintain, and endorse standardized set of relevant benchmarks and metrics for performance evaluation of computer systems”
  » numeric computing
  » web servers
  » graphic subsystems
- Test the CPU, memory hierarchy, compilers
  » Fortran, C, C++
  » Integer and Floating Point sections

CPU 2000

- Compression
- FPGA circuit placement
- C compiler
- Combinatorial Optimizer
- Chess
- Word Processing
- Visualization
- Perl
- Group Theory
- OO Database
- Place and route simulator
- Quantum physics
- Shallow water model
- Multi-grid field
- Partial Dif. Equations
- 3D Graphics
- Fluid Dynamics
- Image recognition
- Seismic wave simulation
- Image processing
- Chemistry, Meteorology
- Number theory

More SPEC Benchmarks

- Java Virtual Machines
- server-side Java
- shared-memory parallel programming
- SMTP and POP3 mail servers
- NFS (network file server) computers
- World Wide Web Servers
- System MultiTasking performance
- high-end industrial-style applications
- graphics performance
**Tuning - Design for speed**

- Think about performance early and often
- Good algorithms first
  - throw faster hardware at it only when you can’t think of anything else
- Product is almost ready, but ...
  - particular data models cause problems
  - many concurrent users cause problems
  - etc

**Tuning - The product is slow**

- Let the numbers tell you what to work on
- Don’t assume anything!
  - I/O bottlenecks, memory thrashing, unusual data sets, unexpected usage patterns, ...
  - the problems can be anywhere
  - the solutions can be anywhere
- Make sure you are solving real performance issues as perceived by the real users

**Tuning Tools**

- Clock on the wall
  - Remember your customer - web user with modem, corporate end user, sysadmin
- High level statistics from the program
  - functions performed and time elapsed
- Profiling
  - detailed information about instruction history
  - logic analyzer tracing for embedded systems

**Profiling**

- Continuously sample CPU state
  - built-in performance counters (eg I-cache miss)
  - sample program counter and context
- Procedure level
  - Where are we spending the most time?
- Instruction level
  - What are we doing there?
  - Why is it taking so long?