# Lecture 3: Evaluating Computer Architectures

- Announcements
  - (none)

- Last Time – constraints imposed by technology
  - Computer elements
  - Circuits and timing

- Today
  - Performance analysis
    - Amdahl's Law
    - Performance equation
  - Computer benchmarks



# How to design something:

- List goals
- List constraints
- Generate ideas for possible designs
- Evaluate the different designs
- Pick the best design
- Refine it

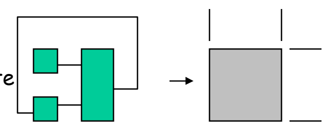
# Evaluation Tools

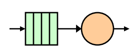
- **Benchmarks, traces, & mixes**
  - macrobenchmarks & suites
    - application execution time
  - microbenchmarks
    - measure one aspect of performance
  - traces
    - replay recorded accesses
      - cache, branch, register

| | |
|---|---|
| MOVE | 39% |
| BR | 20% |
| LOAD | 20% |
| STORE | 10% |
| ALU | 11% |

LD 5EA3
ST 31FF
....
LD 1EA2
....

- Simulation at many levels
  - ISA, cycle accurate, RTL, gate, circuit
    - trade fidelity for simulation rate
- Area and delay estimation
- Analysis
  - e.g., queuing theory







# Evaluation metrics

- Metric = something we measure
- Goal: Evaluate how good/bad a design is
- Examples
  - Execution time for a program
  - Cycles per instruction
  - Clock rate
  - Power consumed by a program

# Different metrics for different purposes

- Chose a metric that's appropriate for design level
- Examples
  - Applications perspective
    - Time to run task (Response Time)
    - Tasks run per second (Throughput)
  - Systems perspective
    - Millions of instructions per second (MIPS)
    - Millions of FP operations per second (MFLOPS)
  - Bus/network bandwidth: megabytes per second
  - Function Units: cycles per instruction (CPI)
  - Fundamental elements (transistors, wires, pins): clock rate



# Each metric has strengths and weaknesses

Pros

Cons

Actual Target Workload

Full Application Benchmarks

Small "Kernel" Benchmarks

Microbenchmarks

## Each metric has strengths and weaknesses

| Pros | | Cons |
|---|---|---|
| • representative | Actual Target Workload | • very specific<br>• non-portable<br>• difficult to run, or measure<br>• hard to identify cause |
| • portable<br>• widely used<br>• improvements useful in reality | Full Application Benchmarks | •less representative |
| • easy to run, early in design cycle | Small "Kernel" Benchmarks | • easy to "fool" |
| • identify peak capability and potential bottlenecks | Microbenchmarks | • "peak" may be a long way from application performance |



## Some Warnings about Benchmarks

- Benchmarks measure the <u>whole</u> system
  - application
  - compiler
  - operating system
  - architecture
  - implementation
- Popular benchmarks typically reflect yesterday's programs
  - what about the programs people are running today?
  - need to design for tomorrow's problems
- Benchmark timings are sensitive
  - alignment in cache
  - location of data on disk
  - values of data
- Danger of *inbreeding* or positive feedback
  - if you make an operation fast (slow) it will be used more (less) often
    - therefore you make it faster (slower)
      - and so on, and so on…
  - the optimized NOP

## Know what you are measuring!

- Compare apples to apples
- Example
  - Wall clock execution time:
    - User CPU time
    - System CPU time
    - Idle time (multitasking, I/O)

```
csh> time latex lecture2.tex
csh> 0.68u 0.05s 0:01.60 45.6%
```

user

system

elapsed

**% CPU time**



## Two notions of "performance"

| Plane | DC to Paris | Speed | Passengers | Throughput (pmph) |
|---|---|---|---|---|
| **Boeing 747** | 6.5 hours | 610 mph | 470 | 286,700 |
| **Concorde** | 3 hours | 1350 mph | 132 | 178,200 |

Which has higher performance?

° **Time to do the task (Execution Time)**
  – execution time, response time, latency

° **Tasks per day, hour, week, sec, ns. .. (Performance)**
  – throughput, bandwidth

## Tradeoff: latency vs. throughput

- Pizza delivery example
  - Do you want your pizza hot?
  - Or do you want your pizza to be inexpensive?
  - Two different delivery strategies for pizza company!

In this course:

We will focus primarily on latency
(execution time for a single task)



## Definitions

- Performance is in units of things-per-second
  - bigger is better
- If we are primarily concerned with response time
  - performance(x) = $\frac{1}{\text{execution\_time}(x)}$

" X is n times faster than Y" means

$$n = \frac{\text{Performance}(X)}{\text{Performance}(Y)}$$

## Brief History of Benchmarking

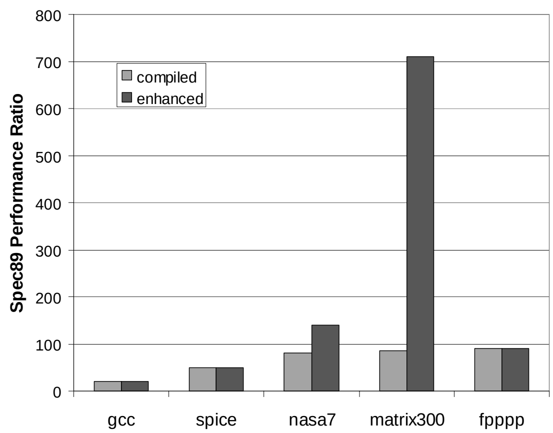
- Early days (1960s)
  - Single instruction execution time
  - Average instruction time [Gibson 1970]
  - Pure MIPS (1/AIT)
- Simple programs(early 70s)
  - Synthetic benchmarks (Whetstone, etc.)
  - Kernels (Livermore Loops)
- Relative Performance (late 70s)
  - VAX 11/780 ≡ 1-MIPS
    - but was it?
  - MFLOPs
- "Real" Applications (late 80s-now)
  - SPEC
    - Scientific
    - Irregular
  - TPC
    - Transaction Processing
  - Winbench
    - Desktop
  - Graphics
    - Quake III, Doom 3
    - MediaBench



## SPEC: Standard Performance Evaluation Corporation (`www.spec.org`)

- System Performance and Evaluation Cooperative
  - HP, DEC, Mips, Sun
  - Portable O/S and high level languages
- Spec89 ⇒ Spec92 ⇒ Spec95 ⇒ Spec2000 ⇒ ....
- Categories
  - CPU (most popular)
  - JVM
  - SpecWeb - web server performance
  - SFS - file server performance
- Benchmarks change with the times and technology
  - Elimination of Matrix 300
  - Compiler restrictions

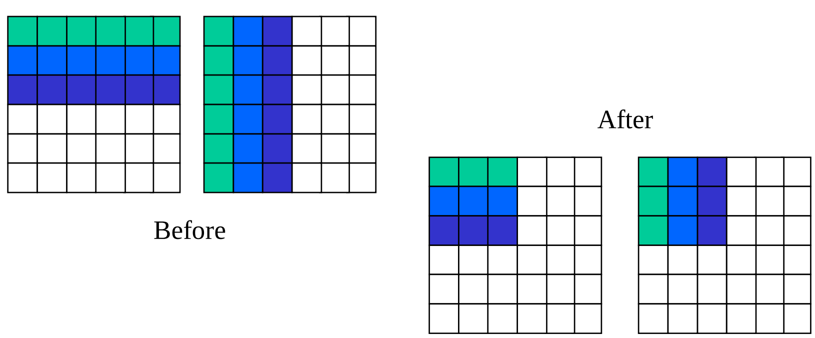
# How to Compromise a Benchmark






# The compiler reorganized the code!

- Change the memory system performance
  - Matrix multiply cache blocking



## Spec2000 Suite

- 12 Integer benchmarks (C/C++)
  - compression
  - C compiler
  - Perl interpreter
  - Database
  - Chess
- 14 FP applications (Fortran/C)
  - Shallow water model
  - 3D graphics
  - Quantum chromodynamics
  - Computer vision
- Characteristics
  - Computationally intensive
  - Little I/O
  - Small code size
  - Variable data set sizes

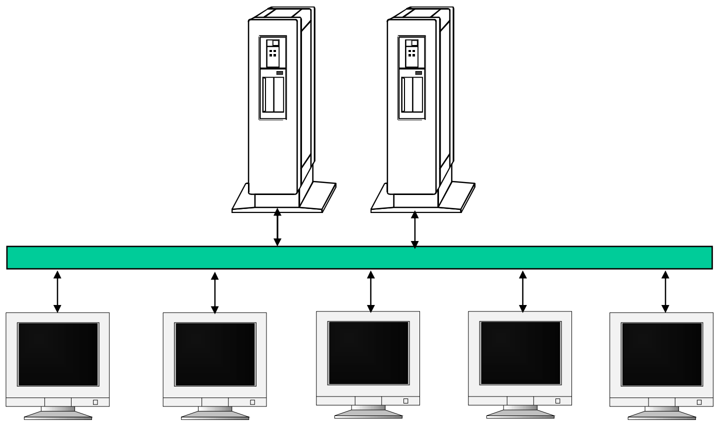


## SPEC Leaders (4/00)

| | Intel Pentium III | AMD Athlon | Compaq Alpha 21264 | Sun Ultra-2 | IBM Power3 | HP PA-8600 |
|---|---|---|---|---|---|---|
| Clock rate | 1000MHz | 1000 MHz | 700MHz | 450MHz | 400MHz | 552MHz |
| Issue rate | 3 x86 | 3 x86 | 4 | 4 | 4 | 4 |
| Cache (I/D) | 16/16/256K | 64K/64K | 64K/64K | 16K/16K | 32K/64K | 512K/1M |
| # transistors | 24 million | 22 million | 15.2 million | 3.8 million | 23 million | 130 million |
| Technology | 0.18µm | 0.18µm | 0.25µm | 0.29µm | 0.22µm | 0.25µm |
| Die Size | $106mm^2$ | $102mm^2$ | $205mm^2$ | $126mm^2$ | $163mm^2$ | $477mm^2$ |
| Estimated mfg. Cost | $40 | $70 | $160 | $70 | $110 | $330 |
| SPECint95 | 46.6 | 42.0 | 34.7 | 16.2 | 23.5 | 38.4 |
| SPECfp95 | 31.9 | 29.4 | 54.5 | 24.6 | 46.0 | 61.0 |

**12/2003**: AMD Opteron 148, 2.0 GHz:
SPECint2000base 16.3
SPECfp2000base 17.5

## Transaction-based Systems

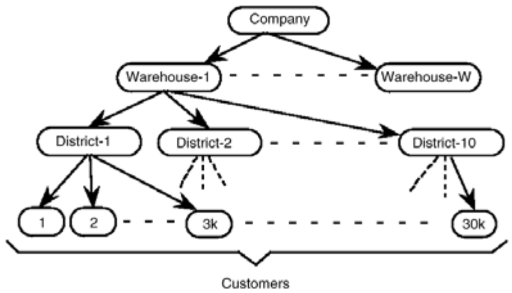




## TPC - Transaction Processing Council

- Established in 1988
- Measure whole system performance and cost
  - Maximum TPS (transactions per second)
  - $/TPS
- Test specifies high level functional requirements
  - Independent of platform
- Workload scales
- Transaction rate constrained by response time requirement
  - 90% of transactions complete in < 2 seconds

## TPC-C: OLTP




- W warehouses
- 10 districts/warehouse
- 3,000 customers/district
- 100,000 items

- 10 items per order
- 1% not in stock at regional warehouse
- Frequent reads and writes



## TPC-C Results (8/00)

- IBM Netfinity 8500R
  - Platform
    - 32 servers, 4 CPUs each
    - 700MHz PentiumIII - Xeon
    - 128 GB memory
    - 4TB disk
    - Windows 2000 server
    - IBM DB2 database server
    - 368,640 users
  - Results
    - Cost: $14.2M
    - Throughput: 440K tpm
    - Price/perf: $32/tpm
- Compaq ProLiant ML570
  - Platform
    - 2 servers, 3 CPUs
    - 700MHz PentiumIII - Xeon
    - 2.5 GB memory
    - 1.5TB disk
    - Windows 2000 server
    - Microsoft SQL
    - 16,200 users
  - Results
    - Cost: $200K
    - Throughput: 20K tpm
    - Price/perf: $10/tpm

## Desktop/Graphics Benchmarks

- WinStone
  - Corel WordPerfect suite, Lotus Smartsuite, Microsoft Office
  - High end: Photoshop
- CPU Mark99
  - Synthetic benchmark - tests CPU, caches, external memory
- 3DMark2003
  - Synthetic benchmark for 3-D rendering



## Performance Measurement Summary

- Best benchmarks are real programs
  - Spec, TPC, Doom3
- Pitfalls still exist
  - Whole system measurement
  - Workload may not match user's
- Key concepts
  - Throughput and Latency
- Next time
  - Performance Equations: CPI, Amdahl's Law, …
  - Instruction set architectures (ISA)
  - Read P&H 2.1 – 2.6