Principles of Computer Architecture

CSE 240A

Fall 2024

Hadi Esmaeilzadeh

hadi@ucsd.edu

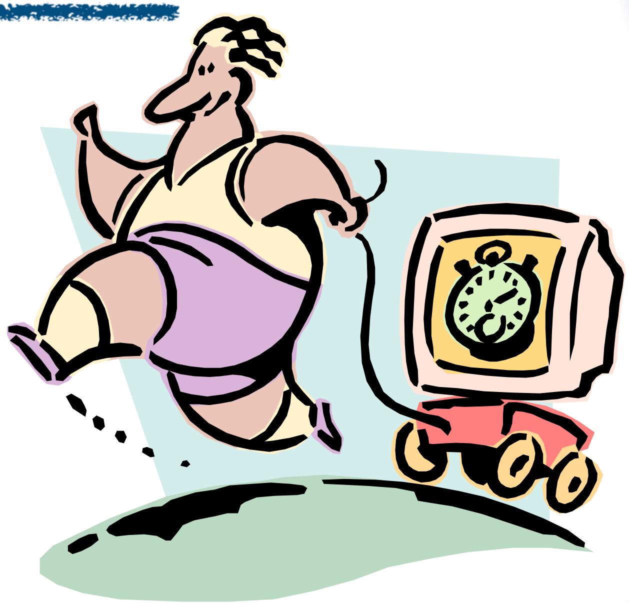
University of California, San Diego

Quantitative Computer Architecture

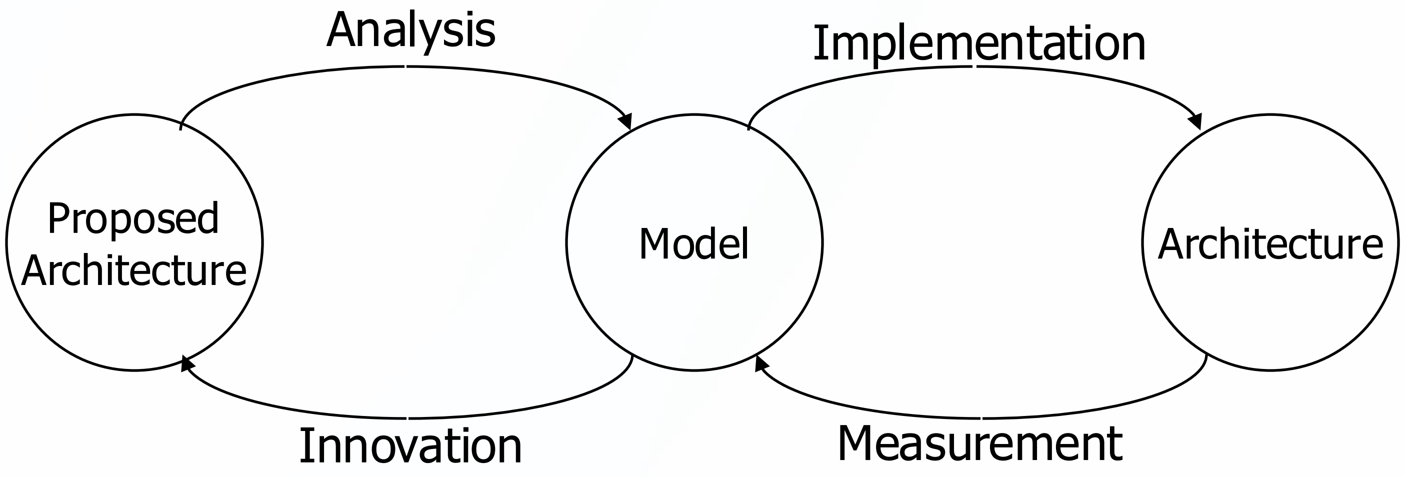
How to measure, analyze, and specify computer system performance

or

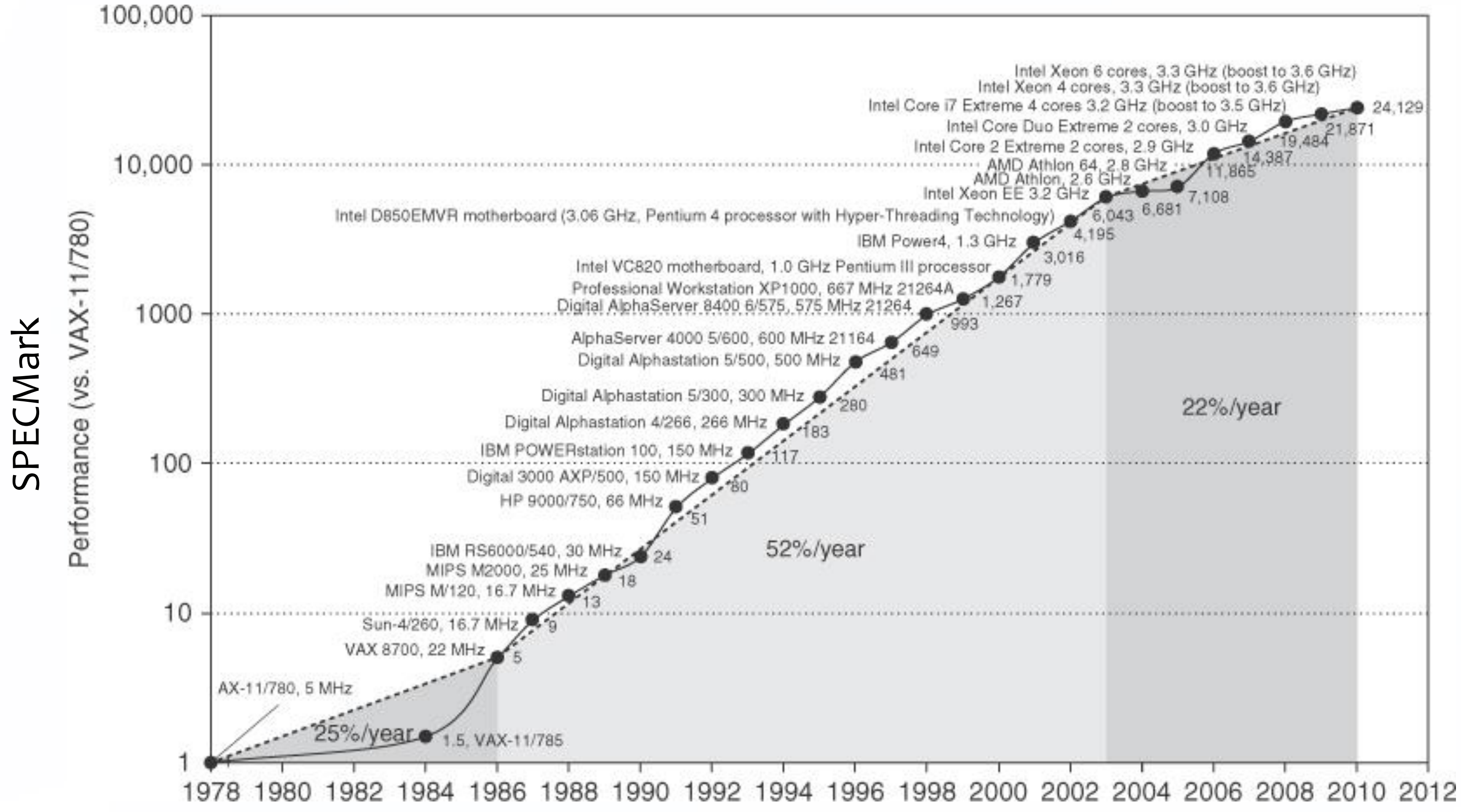
“My computer is faster than your computer!”



Performance Measurement and Analysis in Computer Architecture



Processor Performance



What is the best car you can buy?

Measures of “Performance”

- Execution Time
- Frame Rate
- Throughput (operations/time)
- Responsiveness
- Performance / Cost
- Performance / Power
- Performance / Power^2 (better for arch)
- Fairness (users)

Different Metrics for different folks
ET is standard responsiveness
Frame Rate – FPS/Media App
Throughput (Servers)
Responsiveness – similar to ET but can vary, think ipod/smartphone
Perf/Cost – Server farm / everyone
Perf/Power – power is expensive and causes thermal issues(cooling)
Perf/Power^2 – Embedded devices / server farms

How to measure Execution Time?

```
% time program
... program results ...
160.7u 19.9s 4:15 71%
%
```

- Wall-clock time?

How to measure Execution Time?

```
% time program
... program results ...
160.7u 19.9s 4:15 71%
%
```

- Wall-clock time?
- user CPU time?

How to measure Execution Time?

```
% time program
... program results ...
160.7u 19.9s 4:15 71%
%
```

- Wall-clock time?
- user CPU time?
- user + kernel CPU time?

Attendance

Selection	Answer
Present	A
Present	B
Present	C
Present	B
Present	D

Relative Performance

- can be confusing
  - A runs in 12 seconds
  - B runs in 20 seconds

How much faster is A than B?

Selection	Answer
A	40%
B	67%
C	Neither of the above

Relative Performance

- can be confusing
  - A runs in 12 seconds
  - B runs in 20 seconds
  - ~~A/B = .6 , so A is 40% faster, or 1.4X faster, or B is 40% slower~~
  - B/A = 1.67, so A is 67% faster, or 1.67X faster, or B is 67% slower

- needs a precise definition

Relative Performance, the Definition

$$\text{Relative Performance} = \frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution Time}_Y}{\text{Execution Time}_X} = n$$

Relative Performance, the Definition

Speedup (of x over y)

$$\text{Relative Performance} = \frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution Time}_Y}{\text{Execution Time}_X} = n$$

Relative Performance, the Definition

Speedup (of x over y)

$$\text{Relative Performance} = \frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution Time}_Y}{\text{Execution Time}_X} = n$$

- We can remove all ambiguity by always constraining n to be > 1
- Saying: machine x is *n* times faster than y.
- Saying: machine x is (n – 1)% faster than y

Examples

- Your program runs in 5 minutes on an Intel Xeon, but 2 minutes on a Core i7 processor. How much faster is the i7 processor?

Examples

- Your program runs in 5 minutes on an Intel Xeon, but 2 minutes on a Core i7 processor. How much faster is the i7 processor?
- Another program runs in 10 minutes with the standard compiler, but when recompiled with a new compiler, the program runs in 9 minutes. How much faster is the new compiled program (what is the speedup)?

How to Specify Performance of a machine

- Can we talk just about the performance of a machine in general?

  MIPS – Millions of Instructions Per Second

  MFLOPS – Millions of Floating Point Ops Per Second

How to Specify Performance of a machine

- Can we talk just about the performance of a machine in general?

MIPS – Millions of Instructions Per Second

MFLOPS – Millions of Floating Point Ops Per Second

Are these good metrics?

Selection	Answer
A	Yes
B	No

Summary: How to Specify Performance

- Performance only has meaning in the context of a program or workload
- When talking about the performance of a single machine, we talk about "response time" or "throughput."
- When talking about relative performance, we will say "machine y has a speedup of n over machine x" based on the ratio of their execution times for a workload.
  - "speedup of 1.6"
  - "1.6 times as fast"
  - "60% speedup" [correct but more often misinterpreted]

But What Workload?

- Synthetic workloads
    - whetstone, dhrystone, ...
- Toy benchmarks
    - puzzle, quicksort, sieve, ...
- Kernels
    - livermore loops, linpack
- Real programs

But What Workload?

- Synthetic workloads
  - whetstone, dhrystone, ...
- Toy benchmarks
  - puzzle, quicksort, sieve, ...
- Kernels
  - livermore loops, linpack
- Real programs

But What Workload?

- Synthetic workloads
  - whetstone, dhrystone, …
- Toy benchmarks
  - puzzle, quicksort, sieve, …
- Kernels
  - livermore loops, linpack
- Real programs

To maximize their efforts, architects will attempt to mirror the decision process of the market.

When the market uses poor measurement methodology, we can get poor architectures!

SPEC: System Performance Evaluation Cooperative

- First Round 1989 (SPEC89)
  - 10 programs yielding a single number
- Second Round 1992
  SpecInt92 (6 integer programs) and SpecFP92 (14 floating point programs)
  - Compiler Flags unlimited.
- Third Round 1995
  - Single flag setting for all programs; new set of programs
- Fourth Round, 2000
  - More complex programs, larger data sets
- Fifth Round, 2006
  - Longer running time, some larger data sets, more application areas

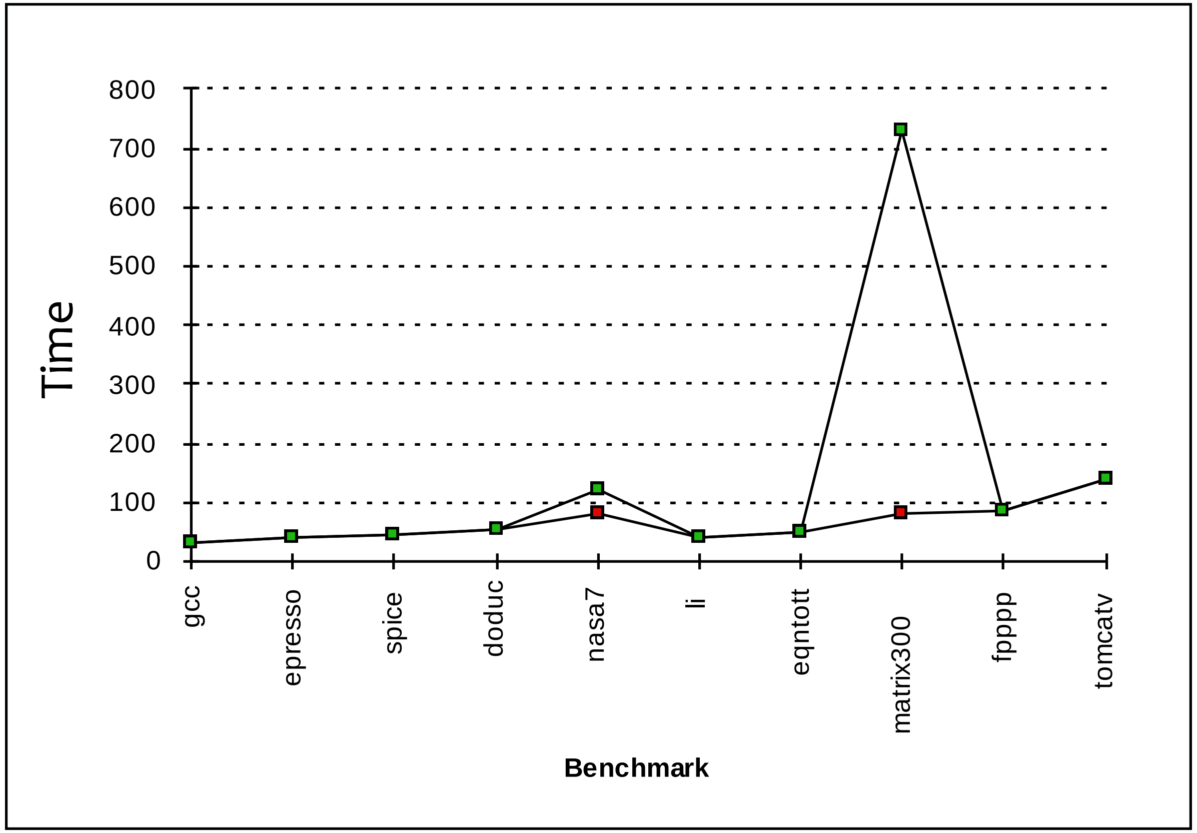
SPEC combines real programs with enforced measurement standards.

SPEC: Standard Performance Evaluation Corporation

- First Round 1989 (SPEC89)
  - 10 programs yielding a single number
- Second Round 1992 (SPEC92)
  SpecInt92 (6 integer programs) and SpecFP92 (14 floating point programs)
  - Compiler Flags unlimited.
- Third Round 1995 (SPEC95)
  - Single flag setting for all programs; new set of programs
- Fourth Round, 2000 (SPEC00)
  - More complex programs, larger data sets
- Fifth Round, 2006 (SPEC06)
  - Longer running time, some larger data sets, more application areas

SPEC combines real programs with enforced measurement standards.

SPEC89/92 Do not Prescribe Compiler Flags



A change in the compiler can have a significant impact!

A Benchmark is not Just Code

- The working data set is also a main part

- A benchmark is
  - The Code
  - The Data
  - The Compiler
    - The Compiler Options

How to Summarize Performance

- Real workloads typically involve multiple programs, and thus, multiple results.
- Popular benchmarks (e.g., SPEC, Parsec, Cloud Suite, ...) involve multiple programs.
- Everyone wants to summarize results with a single number.
- But the summarized result can be dramatically skewed by the method used to combine them.

How to Summarize Performance

	Computer A	Computer B	Computer C
Program 1	1	10	20
Program 2	1000	100	20
Total time	1001	110	40

Which machine is fastest?

How to Summarize Performance

- Arithmetic Mean $\frac{1}{n}\sum_{i=1}^{n} Time_i$
- Weighted Arithmetic Mean $\sum_{i=1}^{n} Time_i * Weight_i$

  where the sum of the weights is 1.
- Geometric Mean $\sqrt[n]{\prod_{i=1}^{n} ExecutionTimeRatio_i} = \frac{\sqrt[n]{\prod_{i=1}^{n} ExecutionTime_i}}{ExecutionTime_{base}}$

Summarizing Performance

Machines:	A	B
Program 1	1	10
Program 2	1000	100

	Set(1)	Set(2)	Set(3)
W_1	.5	.909	.999
W_2	.5	.091	.001

Arith M/Set(1)	500.5	55
Arith M/Set(2)	91.82	18.18
Arith M/Set(3)	2	10.09
Geo M	31.6	31.6

Arith M: Speedup (A/B) = (10 / 1 + 100 / 1000) / 2 = 5.05
Arith M: Speedup (A/B) = (10 + 100)/(1+1000) = 0.10989

Geo M: Speedup (A/B) = sqrt(sqrt(10 / 1) * sqrt(100 /1000)) = 1
Geo M: Speedup (A/B) = sqrt(10 * 100)/sqrt(1*1000) = 1

Summarizing Performance

- Even the unweighted arithmetic mean implies a weighting

- Ratios of geometric means never change (regardless of which machine is used as the base), and always give equal weight to all benchmarks

- To give unequal weight requires weighted arithmetic mean