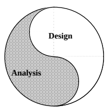
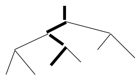
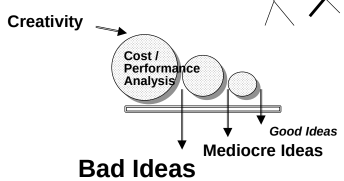
Measurement and Evaluation



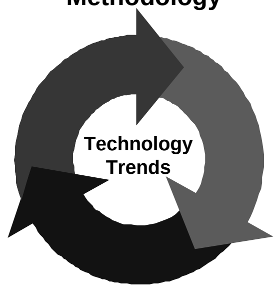
Architecture is an iterative process:
- **Searching the space of possible designs**
- **At all levels of computer systems**





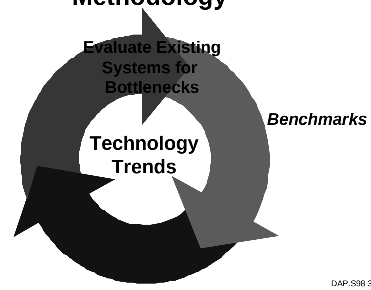
DAP.S98 1

Computer Engineering Methodology



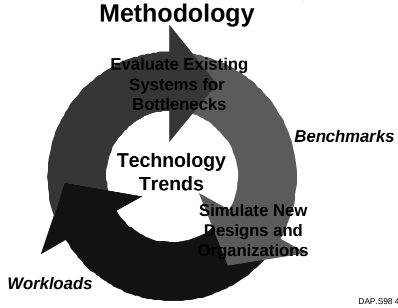
DAP.S98 2

Computer Engineering Methodology



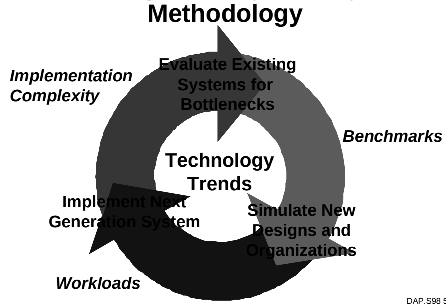
DAP.S98 3

Computer Engineering Methodology



DAP.S98 4

Computer Engineering Methodology



DAP.S98 5

This class: tools for doing this

- Benchmarks, Traces, Mixes
- Hardware: Cost, delay, area, power estimation
- Simulation (many levels)
  - ISA, RT, Gate, Circuit
- Queuing Theory
- Rules of Thumb
- Fundamental “Laws”/Principles

DAP.S98 6

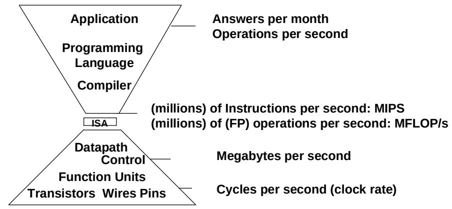
The Bottom Line: Performance (and Cost)

Plane	DC to Paris	Speed	Passengers	Throughput (pmph)
Boeing 747	6.5 hours	610 mph	470	286,700
BAD/Sud Concodre	3 hours	1350 mph	132	178,200

- **Latency: Time to run the task**
  - Execution time, response time, latency
- **Throughput: Tasks per day, hour, week, sec, ns …**
  - Throughput, bandwidth

DAP.S98 7

Metrics of Performance



DAP.S98 8

Performance Evaluation

- **"For better or worse, benchmarks shape a field"**
- **Good products created when have:**
  - Good benchmarks
  - Good ways to summarize performance
- **Given sales is a function in part of performance relative to competition, investment in improving product as reported by performance summary**
- **If benchmarks/summary inadequate, then choose between improving product for real programs vs. improving product to get more sales; Sales almost always wins!**
- **<u>Execution time is the measure of computer performance!</u>**

DAP.S98 9

Benchmarking Problems

- **Bad benchmarks: MIPS, Drystone, MFLOPS, Toys (quicksort, fibonacii, …)**
  - What you care about is how long to run *your* problem
  - Better benchmark = looks more like your problem
- **Benchmarking games (commercial and research)**
  - Different configurations to run same workload on 2 systems
  - Comiler wired to optimize workload
  - Test specification biased towards one machine
  - Arbitrary workload
  - Small benchmark
  - Benchmark manually translated to optimize performance

DAP.S98 10

Benchmarking Problems

- **Common mistakes**
  - Only average behavior in test workload
    - » Average load on machine is about 0!
    - » You care about 98% load
  - Skewing of requests ignored
  - Caching effects ignored
  - Inaccurate sampling
    - » e.g. when timer goes off take sample
    - » timer interrupts lost when machine busy
  - Ignoring monitoring overhead
  - Not validating measurements
  - Not ensuring same initial conditions
  - Not meauring transient cold-start performance
  - **Collecting too much data but doing too little analysis**

DAP.S98 11

How to Summarize Performance

- **"Faster than"**
- **X is n times faster than Y means**
  - performance(X)/performance(Y)
    = throughput(X)/throughput(Y)
    = ExecutionTime(Y)/ExecutionTime(X)
  - Notice: performance is inverse of execution time
- **Never say "slower than"**

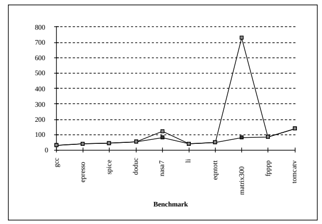
DAP.S98 12

How to Summarize Several Numbers

- Arithmetic mean (weighted arithmetic mean) tracks execution time: $(T_i)/n$ or (W_i*T_i)
- Harmonic mean (weighted harmonic mean) of rates (e.g., MFLOPS) tracks execution time: $n/ (1/R_i)$ or $n/ (W_i/R_i)$
- Normalized execution time is handy for scaling performance (e.g., X times faster than SPARCstation 10)
- But do not take the arithmetic mean of normalized execution time, use the geometric$_i$)^1/n)

SPEC First Round

- One program: 99% of time in single line of code
- New front-end compiler could improve dramatically



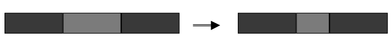
Impact of Means on SPECmark89 for IBM 550

	Ratio to VAX:		Time:		Weighted Time:	
Program	*Before*	*After*	*Before*	*After*	*Before*	*After*
gcc	30	29	49	51	8.91	9.22
espresso	35	34	65	67	7.64	7.86
spice	47	47	510	510	5.69	5.69
doduc	46	49	41	38	5.81	5.45
nasa7	78	144	258	140	3.43	1.86
li	34	34	183	183	7.86	7.86
eqntott	40	40	28	28	6.68	6.68
matrix300	78	730	58	6	3.43	0.37
fpppp	90	87	34	35	2.97	3.07
tomcatv	33	138	20	19	2.01	1.94
Mean	54	72	124	108	54.42	49.99
	Geometric		*Arithmetic*		*Weighted Arith.*	
	Ratio	1.33	Ratio	1.16	Ratio	1.09

Amdahl's Law

Speedup due to enhancement E:

```
                ExTime w/o E        Performance w/  E
Speedup(E) = -------------    =   -------------------
                ExTime w/  E        Performance w/o E
```



Suppose that enhancement E accelerates a fraction F of the task by a factor S, and the remainder of the task is unaffected

Amdahl's Law

$$ExTime_{new} = ExTime_{old} \times \left[(1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}\right]$$

$$Speedup_{overall} = \frac{ExTime_{old}}{ExTime_{new}} = \frac{1}{(1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}}$$

Amdahl's Law

- Floating point instructions improved to run 2X; but only 10% of actual instructions are FP

$ExTime_{new} =$

$Speedup_{overall} =$

Amdahl's Law

- Floating point instructions improved to run 2X; but only 10% of actual instructions are FP

$$ExTime_{new} = ExTime_{old} \times (0.9 + .1/2) = 0.95 \times ExTime_{old}$$

$$Speedup_{overall} = \frac{1}{0.95} = 1.053$$

DAP.S98 19

Aspects of CPU Performance

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

	Inst Count	CPI	Clock Rate
Program	X		
Compiler	X	(X)	
Inst. Set.	X	X	
Organization		X	X
Technology			X

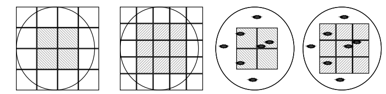
DAP.S98 20

Integrated Circuits Costs

$$\text{IC cost} = \frac{\text{Die cost} + \text{Testing cost} + \text{Packaging cost}}{\text{Final test yield}}$$

$$\text{Die cost} = \frac{\text{Wafer cost}}{\text{Dies per Wafer} * \text{Die yield}}$$

$$\text{Dies per wafer} = \frac{\pi * (\text{Wafer\_diam} / 2)^2}{\text{Die Area}} - \frac{\pi * \text{Wafer\_diam}}{\sqrt{2 * \text{Die Area}}} - \text{Test dies}$$



$$\text{Die Yield} = \text{Wafer yield} * \left\{1 + \frac{\text{Defects\_per\_unit\_area} * \text{Die\_Area}}{\alpha}\right\}^{-\alpha}$$

Die Cost goes roughly with die area^4

DAP.S98 21

Real World Examples

Chip	Metal layers	Line width	Wafer cost	Defect /cm^2	Area mm^2	Dies/ wafer	Yield	Die Cost
386DX	2	0.90	$900	1.0	43	360	71%	$4
486DX2	3	0.80	$1200	1.0	81	181	54%	$12
PowerPC 601	4	0.80	$1700	1.3	121	115	28%	$53
HP PA 7100	3	0.80	$1300	1.0	196	66	27%	$73
DEC Alpha	3	0.70	$1500	1.2	234	53	19%	$149
SuperSPARC	3	0.70	$1700	1.6	256	48	13%	$272
Pentium	3	0.80	$1500	1.5	296	40	9%	$417

– From "Estimating IC Manufacturing Costs," by Linley Gwennap, *Microprocessor Report*, August 2, 1993, p. 15

DAP.S98 22

Summary, #1

- Designing to Last through Trends

	Capacity	Speed
Logic	2x in 3 years	2x in 3 years
DRAM	4x in 3 years	2x in 10 years
Disk	4x in 3 years	2x in 10 years

- 6yrs to graduate => 16X CPU speed, DRAM/Disk size
- Time to run the task
  - Execution time, response time, latency
- Tasks per day, hour, week, sec, ns, …
  - Throughput, bandwidth
- "X is n times faster than Y" means

```
ExTime(Y)          Performance(X)
---------    =     --------------
ExTime(X)          Performance(Y)
```

DAP.S98 23

Summary, #2

- Amdahl's Law:

$$Speedup_{overall} = \frac{ExTime_{old}}{ExTime_{new}} = \frac{1}{(1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}}$$

- CPI Law:

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

- Execution time is the REAL measure of computer performance!
- Good products created when have:
  - Good benchmarks, good ways to summarize performance
- Die Cost goes roughly with die area^4
- Can PC industry support engineering/research investment?

DAP.S98 24