Performance

COE 501

Computer Architecture

Prof. Muhamed Mudawar

Computer Engineering Department

King Fahd University of Petroleum and Minerals

What is Performance?

- How do we measure the performance of a computer?
- A user of a computer system may say:
  - A computer is faster when a program runs in **less time**
- A server administrator may say:
  - A computer is faster if it completes **more transactions per minute**
- What factors are hardware related?
- What factors are software related?

Time as a Measure of Performance

- **Response Time**
  - Time between start and completion of a task
  - As observed and measured by the end user
  - Called also **Wall-Clock Time** or **Elapsed Time**
  - Response Time = CPU Time + Waiting Time (I/O, scheduling, etc.)
- **CPU Execution Time**
  - Time spent executing the program instructions
  - **CPU time = User CPU time + Kernel CPU time**
  - Can be measured in seconds. msec, µsec, etc.
  - Can be related to the **number of CPU clock cycles**

Throughput as a Performance Metric

- **Throughput = Total work done per unit of time**
  - Tasks per hour
  - Transactions per minute
- Decreasing the execution time improves throughput
  - Example: using a faster version of a processor
  - Less time to run a task $\Rightarrow$ more tasks can be executed per unit of time
- Parallel hardware improves throughput and response time
  - By increasing the number of processors in a multiprocessor
  - More tasks can be executed in parallel
  - Execution time of individual sequential tasks is not changed
  - Less waiting time in queues reduces (improves) response time

Relative Performance

- For some program running on computer *X*

$$\boxed{\text{Performance}_X = \frac{1}{\text{Execution time}_X}}$$

- Computer *X* is *n* times faster than *Y*

$$\boxed{\frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution time}_Y}{\text{Execution time}_X} = n}$$

- Example: A program takes 10 sec on A and 15 sec on B

  Which computer is faster and by what factor?

  Computer A is faster than B by a factor = 15/10 = 1.5

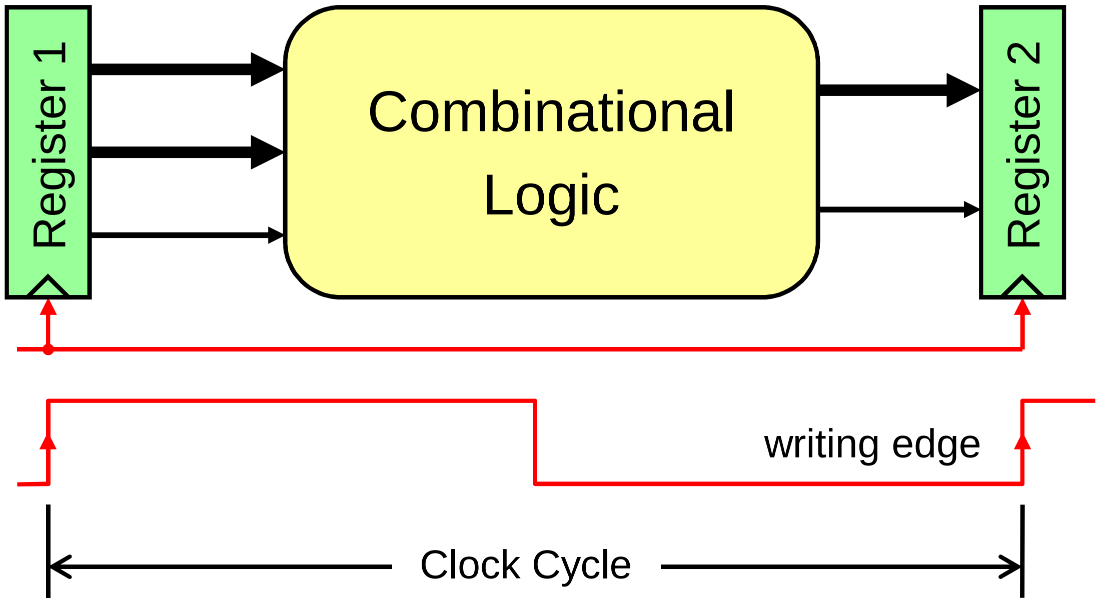
Processor Performance Equation

$$\text{CPU Time (sec)} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

	Instruction Count	CPI	Clock Cycle
Program	X		
Compiler	X	X	
ISA	X	X	
Organization		X	X
Technology			X

Determining the Clock Cycle

❖ Sum of circuit element delays
  - ✧ Register clock-to-output delay + register setup time
  - ✧ Combinational Logic delay along longest (critical) path
  - ✧ Wire delays
  - ✧ Clock skew



Clock cycle should be long enough to ensure correct timing of writes

CPI: Clock cycles Per Instruction

$$\text{Average CPI for a given program} = \frac{\text{Total CPU clock cycles}}{\text{Total Instruction Count}}$$

- Different instructions have different CPI

  Let CPI_i = clocks per instruction for class *i* of instructions

  Let C_i = instruction count for class *i* of instructions

$$\text{CPU cycles} = \sum_{i=1}^{n} (CPI_i \times C_i)$$

$$CPI = \frac{\sum_{i=1}^{n} (CPI_i \times C_i)}{\sum_{i=1}^{n} C_i}$$

- Obtain CPI by detailed simulation
- Use hardware counters in a CPU

Example 1 on CPI

Given the following instruction mix of a program on a processor

What is the average CPI?

What is the percent of time used by each instruction class?

Class_i	**Freq_i**	**CPI_i**	**$\text{Freq}_i \times \text{CPI}_i$**	**Percent of Time**
ALU	20%	1	$0.2 \times 1 = 0.2$	$0.2 / 3.1 = 6.4\%$
FPU	30%	4	$0.3 \times 4 = 1.2$	$1.2 / 3.1 = 38.7\%$
Load	20%	5	$0.2 \times 5 = 1.0$	$1.0 / 3.1 = 32.3\%$
Store	10%	3	$0.1 \times 3 = 0.3$	$0.3 / 3.1 = 9.7\%$
Branch	20%	2	$0.2 \times 2 = 0.4$	$0.4 / 3.1 = 12.9\%$

Average CPI = $0.2 + 1.2 + 1.0 + 0.3 + 0.4 = 3.1$

Example 2 on CPI

- Suppose we make the following measurements:
  - Frequency of **ALL** FP ops (including SQRT) = 30%, with average CPI = 4.0
  - Frequency of SQRT only = 2%, with CPI of SQRT = 10
- What is the average CPI of FP operations excluding SQRT?
- **Answer:** Frequency of FP ops – SQRT = 30% – 2% = 28%

$$\text{CPI of ALL FP} = \frac{0.02 \times \text{CPI SQRT} \ + 0.28 \times \text{CPI FP excluding SQRT}}{0.3}$$

$$4.0 = \frac{0.02 \times 10 + 0.28 \times \text{CPI of FP ops excluding SQRT}}{0.3}$$

$$\text{CPI of FP ops excluding SQRT} = \frac{4.0 \times 0.3 - 0.02 \times 10}{0.28} = 3.57$$

MIPS as a Performance Measure

- MIPS: Millions Instructions Per Second (execution rate)
- Used as a performance metric
  - Faster machine $\Rightarrow$ larger MIPS
- MIPS specifies instruction execution rate

$$\boxed{\text{MIPS} = \frac{\text{Instruction Count}}{\text{Execution Time} \times 10^6} = \frac{\text{Clock Rate}}{\text{CPI} \times 10^6}}$$

- We can also relate execution time to MIPS

$$\boxed{\text{Execution Time} = \frac{\text{Inst Count}}{\text{MIPS} \times 10^6} = \frac{\text{Inst Count} \times \text{CPI}}{\text{Clock Rate}}}$$

Drawbacks of MIPS

Three problems using MIPS as a performance metric

1. Does not take into account the capability of instructions
   - ✧ Cannot use MIPS to compare computers with different instruction sets because the instruction count will differ
2. MIPS varies between programs on the same computer
   - ✧ A computer cannot have a single MIPS rating for all programs
3. MIPS can vary inversely with performance
   - ✧ A higher MIPS rating does not always mean better performance
   - ✧ Example in next slide shows this anomalous behavior

MIPS Example

- Consider the following performance measurements

Measurement	Computer A	Computer B
Instruction Count for a given program	10 Billion	8 Billion
Clock Rate	4.2 GHz	4.0 GHz
CPI	1.0	1.1

  - Which computer has the higher MIPS rating? Which is faster?

- Answer
  - MIPS (Computer A) = $(4.2 \times 10^9)/(1.0 \times 10^6)$ = 4200 MIPS
  - MIPS (Computer B) = $(4.0 \times 10^9)/(1.1 \times 10^6)$ = 3636 MIPS
  - CPU Time (A) = $(10 \times 10^9 \times 1.0)/(4.2 \times 10^9)$ = 2.38 sec
  - CPU Time (B) = $(8 \times 10^9 \times 1.1)/(4 \times 10^9)$ = 2.20 sec
  - Computer A has a higher MIPS rating, but B has less execution time

Amdahl's Law

- Amdahl's Law is a measure of Speedup
  - How a program performs after improving some portion of a computer

$$Speedup = \frac{Execution\ time\ of\ the\ program\ without\ using\ the\ enhancement}{Execution\ time\ of\ the\ program\ using\ the\ enhancement}$$

- Let ***f*** = Fraction of the computation time that is enhanced
- Let **s** = Speedup factor of the enhancement only

Execution Time $_{old}$	Fraction ***f*** of old time to be enhanced	**1 – *f***
Execution Time $_{new}$	***f* / *s*** of old time	**1 – *f***

$$\boxed{\text{Speedup} = \frac{\text{Execution Time}_{\text{old}}}{\text{Execution Time}_{\text{new}}} = \frac{1}{((1 - f) + f / s)}}$$

Example on Amdahl's Law

- Suppose that floating-point square root is responsible for 20% of the execution time of a graphics benchmark and ALL FP instructions are responsible for 60%
- One proposal is to speedup FP SQRT by a factor of 10
- Alternative choice: make ALL FP instructions 2X faster, which choice is better?
- Answer:
  - Choice 1: Improve FP SQRT by a factor of 10
  - Speedup (FP SQRT) = 1/(0.8 + 0.2/10) = 1.22
  - Choice 2: Improve ALL FP instructions by a factor of 2
  - Speedup = 1/(0.4 + 0.6/2) = 1.43 ➔ **Better**

Amdahl's Law of Diminishing Return

❖ Overall speedup gained by just improving a portion of the computer diminishes as improvements are added

$$\text{Speedup}_{\text{overall}} = \frac{1}{((1 - f) + f / s)} \rightarrow \frac{1}{(1 - f)} \text{ as } \mathbf{s} \rightarrow \infty$$

❖ Example: A program spends 80% of its time running on a processor and 20% of its time waiting for I/O. What is the overall speedup if we run the computation on a 10X faster processor? or on a 100X faster processor?

❖ Speedup (10X processor) = 1/(0.2+0.8/10) = 3.57 X

❖ Speedup (100X processor) = 1/(0.2+0.8/100) = 4.81 X

❖ Overall speedup **cannot exceed 5X** because of I/O

Latency versus Bandwidth

- Latency = **Elapsed time for a single event**
  - Example: DRAM memory access time in nanoseconds
- Bandwidth = **Number of events per unit time**
  - Example: Mbytes per second from DRAM memory
- Tracking Four Technology Improvements
  - DRAM Memory (1980 – 2016)
  - Processors (1982 – 2017)
  - Disk Storage (1983 – 2016)
  - Local Area Network (Ethernet 1978 – 2017)

DRAM Memory Improvements

Year	1980	1986	1993	1997	2000	2010	2016
Type	DRAM	Fast Page	Fast Page	SDRAM	DDR	DDR3	DDR4
Capacity	64 Kbit	1 Mbit	16 Mbit	64 Mbit	256 Mbit	2048 Mb	4096 Mb
Die (mm^2)	35	70	130	170	204	50	50
Pins / Chip	16	18	20	54	66	96	134
Bandwidth	13 MB/s	160	267	640	1600	16,000	27,000
Latency	225 ns	125 ns	75 ns	62 ns	52 ns	37 ns	30 ns

- Capacity Improvements = 4096 / 0.064 = 64,000 X
- Bandwidth Improvements = 27000 / 13 = 2077 X
- Latency Improvements = 225 / 30 = 7.5 X
- Bandwidth Improvements >> Latency Improvements

Microprocessor Improvements

Year	1982	1985	1989	1993	2001	2010	2017
Product	Intel 286	Intel 386	Intel 486	Pentium	Pentium 4	Core i7	Core i7
Die (mm^2)	47	43	81	90	217	240	122
Transistors	134,000	275,000	1.2 M	3.1 M	42 M	1170 M	1750 M
Pins/chip	68	132	168	273	423	1366	1400
Bus width	16 bits	32	32	64	64	3×64	3×64
Cores/chip	1	1	1	1	1	4	4
Clock MHz	12.5	16	25	66	1500	3333	4000
MIPS	2	6	25	132	4500	50,000	64,000
Latency	320 ns	313 ns	200 ns	76 ns	15 ns	4 ns	4 ns

- Bandwidth Improvements (MIPS) = 64000 / 2 = 32,000 X
- Latency Improvements = 320 / 4 = 80 X

Hard Disk Improvements

Year	1983	1990	1994	1998	2003	2010	2016
RPM	3600	5400	7200	10,000	15,000	15,000	15,000
Capacity	0.03 GB	1.4 GB	4.3 GB	9.1 GB	73.4 GB	600 GB	600 GB
Diameter	5.25"	5.25"	3.5"	3.0"	2.5"	2.5"	2.5"
Bandwidth	0.6 MB/s	4 MB/s	9 MB/s	24 MB/s	86 MB/s	204 MB/s	250 MB/s
Latency	48.3 ms	17.1 ms	12.7 ms	8.8 ms	5.7 ms	3.6 ms	3.6 ms

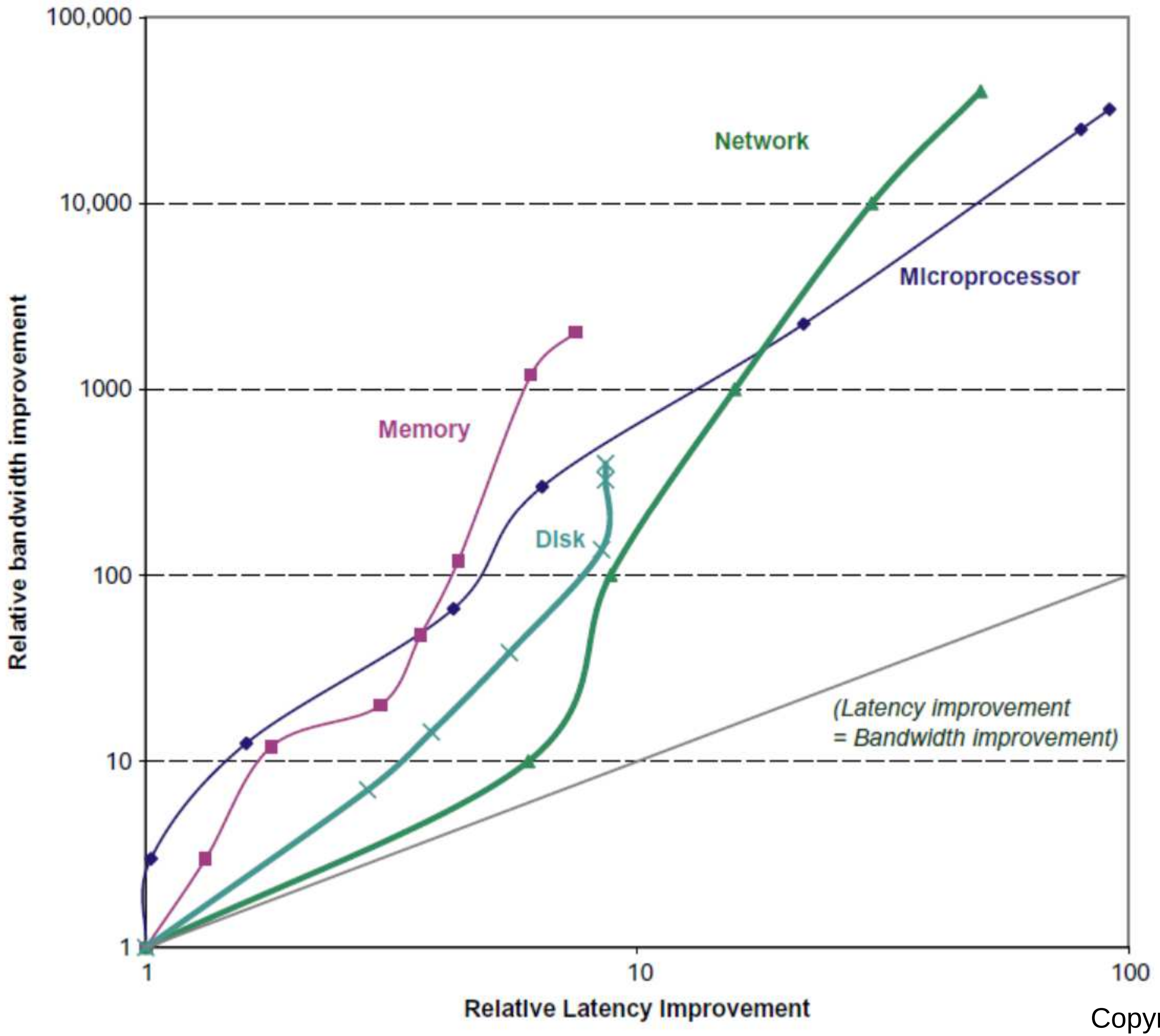
- RPM Improvements = 15000 / 3600 = 4.2 X
- Capacity Improvements = 600 / 0.03 = 20,000 X
- Bandwidth Improvements = 250 / 0.6 = 417 X
- Latency Improvements = 48.3 / 3.6 = 13.4 X
- Bandwidth Improvements >> Latency Improvements

Local Area Network Improvements

Year	1978	1995	1999	2003	2010	2017
LAN	Ethernet	Fast	Gigabit	10 Gigabit	100 Gigabit	400 Gigabit
Standard	802.3	803.3u	802.3ab	802.3ac	802.3ba	802.3bs
Bandwidth	10 Mbit/s	100 Mb/s	1 Gbit/s	10 Gbit/s	100 Gbit/s	400 Gbit/s
Latency	3000 µs	500 µs	340 µs	190 µs	100 µs	60 µs

- ❖ Bandwidth Improvements = 400 / 0.01 = 40,000 X
- ❖ Latency Improvements = 3000 / 60 = 50 X
- ❖ Bandwidth Improvements >> Latency Improvements

Bandwidth versus Latency Improvements



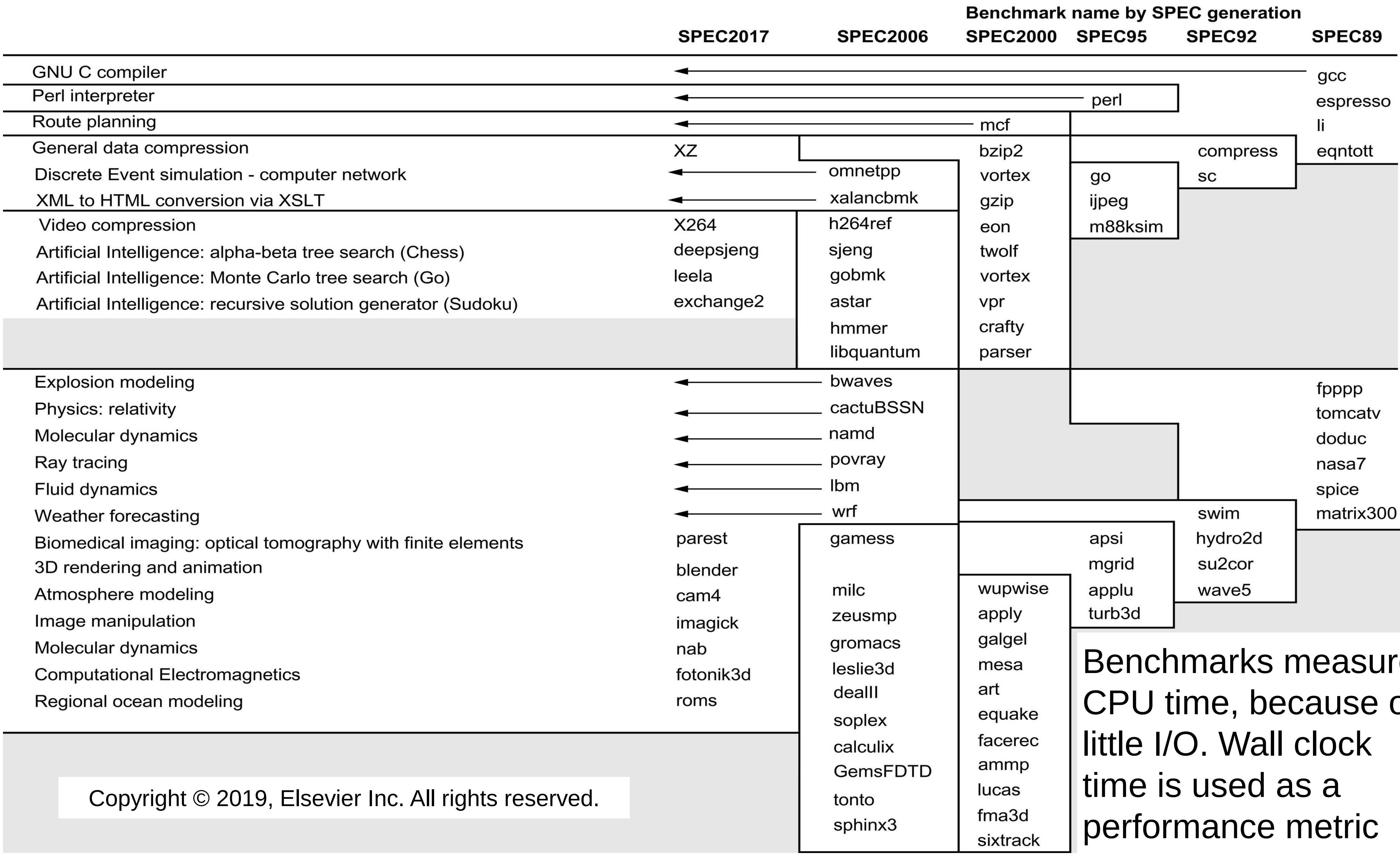
Five Reasons Latency Lags Bandwidth

1. Moore's Law helps Bandwidth more than Latency
   - ✧ Smaller, faster, more transistors, more I/O pins help bandwidth
   - ✧ Wire delay does not improve with smaller feature size
2. Distance increases latency
   - ✧ Relatively long wires in DRAM limit latency
3. Bandwidth easier to sell (Bigger is Better)
   - ✧ 100 Gigabits per second versus 100 µsec latency for Ethernet
4. Latency helps Bandwidth, but not vice versa
   - ✧ Lower DRAM latency ➔ more DRAM access/sec (bandwidth)
5. OS overhead hurts Latency more than Bandwidth
   - ✧ Scheduling queues help bandwidth but hurt latency

Benchmarks

- Performance is measured by running real applications
  - Use programs typical of expected workload
  - Representatives of expected classes of applications
  - Examples: compilers, editors, scientific applications, graphics, ...
- SPEC (System Performance Evaluation Corporation)
  - Website: www.spec.org
  - Various benchmarks for CPU performance, graphics, high-performance computing, Web servers, etc.
  - Specifies rules for running list of programs and reporting results
  - Valuable indicator of performance (and compiler technology)
  - SPEC CPU 2017 (10 integer + 13 FP programs in C, C++, and Fortran)

SPEC CPU Benchmarks over Time



Benchmarks measure CPU time, because of little I/O. Wall clock time is used as a performance metric

Summarizing Performance Results

$$SPEC\ Ratio = \frac{Time\ on\ Reference\ Computer}{Time\ of\ Computer\ being\ Rated}$$

$$\frac{SPEC\ Ratio\ A}{SPEC\ Ratio\ B} = \frac{\dfrac{Reference\ Execution\ Time}{Execution\ Time\ A}}{\dfrac{Reference\ Execution\ Time}{Execution\ Time\ B}} = \frac{Execution\ Time\ B}{Execution\ Time\ A}$$

$$Geometric\ Mean = \sqrt[n]{\prod_{i=1}^{n} SPEC\ Ratio}$$

Choice of Reference Computer is Irrelevant

SPEC CPU INT 2006 Execution Times

Benchmarks	Sun Ultra Enterprise 2 time (seconds)	AMD A10-6800K time (seconds)	SPEC 2006Cint ratio	Intel Xeon E5-2690 time (seconds)	SPEC 2006Cint ratio	AMD/Intel times (seconds)	Intel/AMD SPEC ratios
perlbench	9770	401	24.36	261	37.43	1.54	1.54
bzip2	9650	505	19.11	422	22.87	1.20	1.20
gcc	8050	490	16.43	227	35.46	2.16	2.16
mcf	9120	249	36.63	153	59.61	1.63	1.63
gobmk	10,490	418	25.10	382	27.46	1.09	1.09
hmmer	9330	182	51.26	120	77.75	1.52	1.52
sjeng	12,100	517	23.40	383	31.59	1.35	1.35
libquantum	20,720	84	246.08	3	7295.77	29.65	29.65
h264ref	22,130	611	36.22	425	52.07	1.44	1.44
omnetpp	6250	313	19.97	153	40.85	2.05	2.05
astar	7020	303	23.17	209	33.59	1.45	1.45
xalancbmk	6900	215	32.09	98	70.41	2.19	2.19
Geometric mean			31.91		63.72	2.00	2.00

Geometric mean of ratios = 2.00 = Ratio of Geometric means = 63.72 / 31.91

Things to Remember about Performance

- Two common measures: Response Time and Throughput
- CPU Execution Time = Instruction Count × CPI × Cycle
- MIPS = Millions of Instructions Per Second (is a rate)
  - FLOPS = Floating-point Operations Per Second
- Latency is the time of a single event, Bandwidth is a rate
  - Latency improvements lag Bandwidth over the past 30 years
- Amdahl's Law is a measure of speedup
  - When improving part of a computer (fraction of execution time)
- Benchmarks: real applications are used
  - To compare the performance of computer systems
  - Geometric mean of SPEC ratios (for a set of applications)