Performance Summary

The BIG Picture

$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, clock rate

SPEC CPU Benchmark

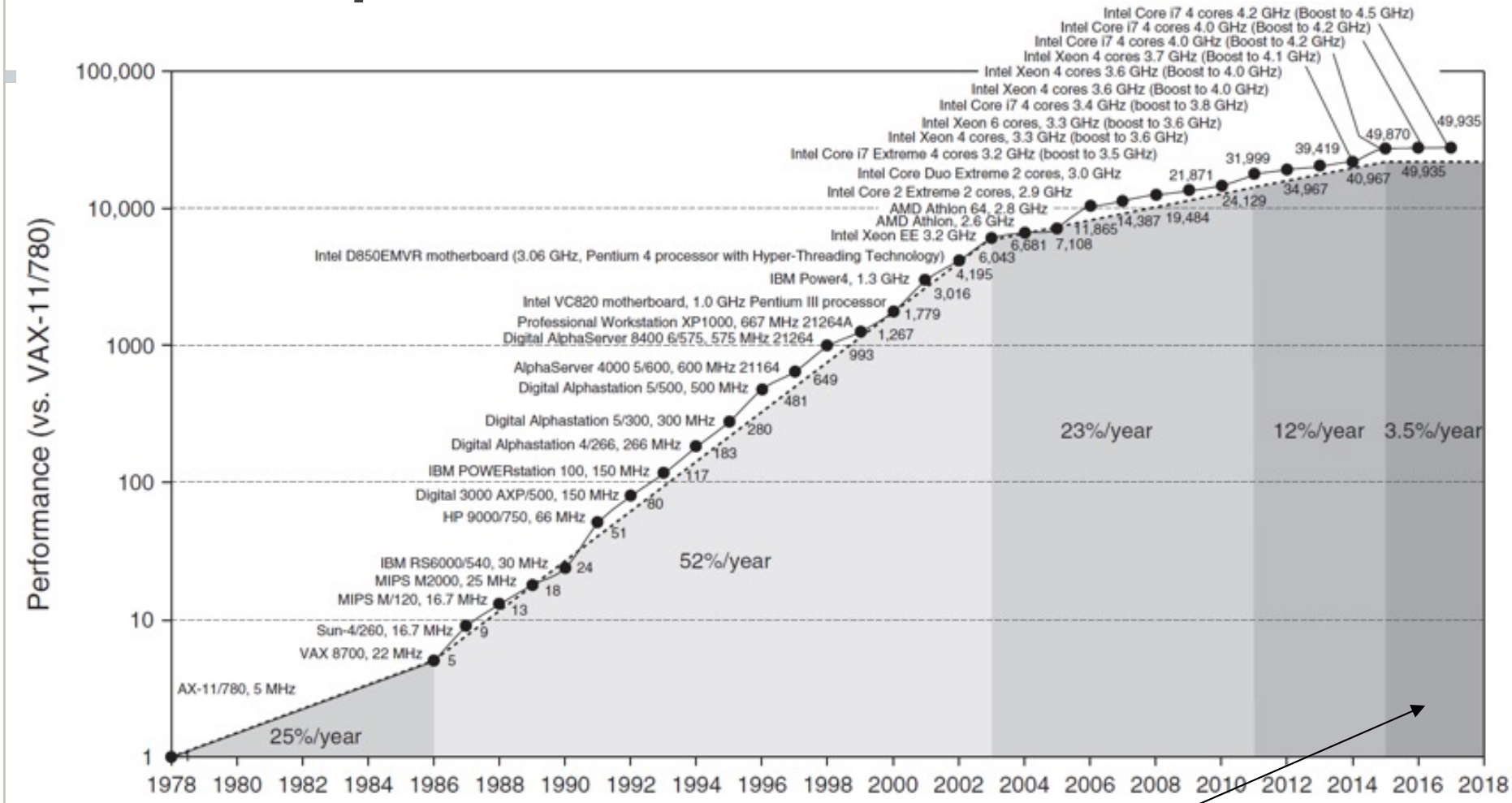
- Standard Performance Evaluation Corp (SPEC)
  - Develops benchmarks for CPU, I/O, Web, …
- SPEC ~~CPU2006~~ CPU2017
  - Elapsed time to execute a selection of programs
    - Negligible I/O, so focuses on CPU performance
  - Normalize relative to reference machine
  - Summarize as geometric mean of performance ratios
    - CINT (integer) and CFP (floating-point)
    - SPEC Speed and SPEC Rate

$$\sqrt[n]{\prod_{i=1}^{n} \text{Execution time ratio}_i}$$

SPECspeed 2017 Integer
Intel Xeon E5-2650L

Description	*Name*	*Instruction Count x 10^9*	*CPI*	*Clock cycle time (seconds x 10^–9)*	*Execution Time (seconds)*	*Reference Time (seconds)*	*SPECratio*
Perl interpreter	perlbench	2684	0.42	0.556	627	1774	2.83
GNU C compiler	gcc	2322	0.67	0.556	863	3976	4.61
Route planning	mcf	1786	1.22	0.556	1215	4721	3.89
Discrete Event simulation - computer network	omnetpp	1107	0.82	0.556	507	1630	3.21
XML to HTML conversion via XSLT	xalancbmk	1314	0.75	0.556	549	1417	2.58
Video compression	x264	4488	0.32	0.556	813	1763	2.17
Artificial Intelligence: alpha-beta tree search (Chess)	deepsjeng	2216	0.57	0.556	698	1432	2.05
Artificial Intelligence: Monte Carlo tree search (Go)	leela	2236	0.79	0.556	987	1703	1.73
Artificial Intelligence: recursive solution generator (Sudoku)	exchange2	6683	0.46	0.556	1718	2939	1.71
General data compression	xz	8533	1.32	0.556	6290	6182	0.98
Geometric mean							2.36

Uniprocessor Performance



Constrained by power, instruction-level parallelism, memory latency

Pitfall: Amdahl's Law

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

- Example: suppose for a program that runs for 100s, multiplication accounts for 80s while addition accounts for 10s. Which one to choose?
  - improve multiplication by 10x; or

    ```
    ET1 = 80/10+20 = 28s
    ```

  - improve addition by 20x

    ```
    ET2 = 10/20+90 = 90.5s
    ```

Pitfall: Amdahl's Law

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

- Example: suppose for a program that runs for 100s, multiplication accounts for 80s while addition accounts for 10s. Which one to choose?
  - improve multiplication by 10x; or
  - improve addition by 20x
- Corollary: make the common case fast

Pitfall: Performance Metric

- Pitfall: using a subset of the performance equation as a performance metric
  - The most important metric is execution time
- One alternative metric is MIPS (Million Instructions Per Second)
  - Is this a good candidate?

Exercise: Performance Comparison

Measurement	Computer A	Computer B
Instruction Count	10 billion	8 billion
Clock Rate	4 Ghz	4 Ghz
CPI	1.0	1.1

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

1) Which computer has a higher MIPS rating ?

2) Which computer is faster ?

Pitfall: MIPS as a Performance Metric

- MIPS: Millions of Instructions Per Second

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

$$= \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

  - Missing instruction count and not measuring exe time
  - Only instruction execution rate for the given instruction mix

Performance Improvement

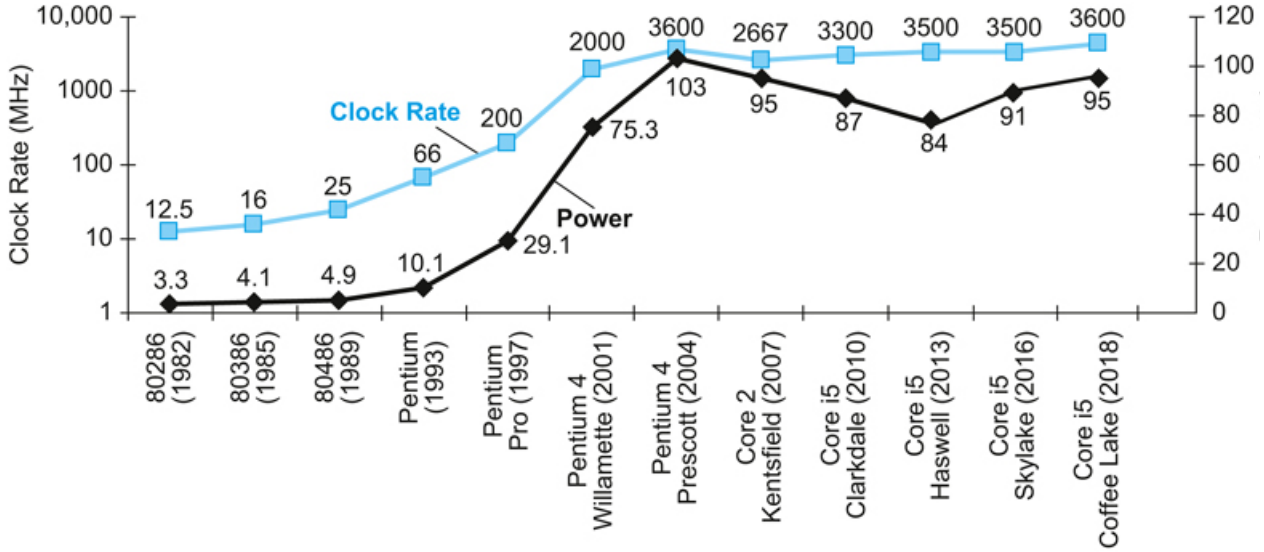
$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- How can we reduce CPU time?
  - Faster machine usually means a shorter cycle time (i.e. higher frequency)
  - How about power?

Power Trends



- In CMOS IC technology

$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

×30 | 5V → 1V | ×300

Reducing Power

- Suppose a new CPU has
  - 85% of capacitive load of old CPU
  - 15% voltage and 15% frequency reduction
  - What will be the ratio of new power to old power

$$\frac{P_{new}}{P_{old}} = \frac{C_{old} \times 0.85 \times (V_{old} \times 0.85)^2 \times F_{old} \times 0.85}{C_{old} \times V_{old}^{\ 2} \times F_{old}} = 0.85^4 = 0.52$$

- The power wall
  - We can't reduce voltage further
  - We can't remove more heat
- How else can we improve performance?

Multiprocessors

- Multi-core microprocessors
  - More than one processor per chip
- What do you think are the challenges of dealing with multiprocessors ?

Challenges of Multicore Multiprocessors

- Requires explicit parallel programming
  - Compared with instruction level parallelism
    - Hardware executes multiple instructions at once
    - Hidden from the programmer
  - Hard to do
    - Programming for performance
    - Load balancing
    - Optimizing communication and synchronization

Concluding Remarks

- Hierarchical layers of abstraction
  - In both hardware and software
- Instruction set architecture
  - The hardware/software interface
- Execution time: the best performance measure
- Cost/performance is improving
  - Due to underlying technology development
- Power is a limiting factor
  - Use parallelism to improve performance