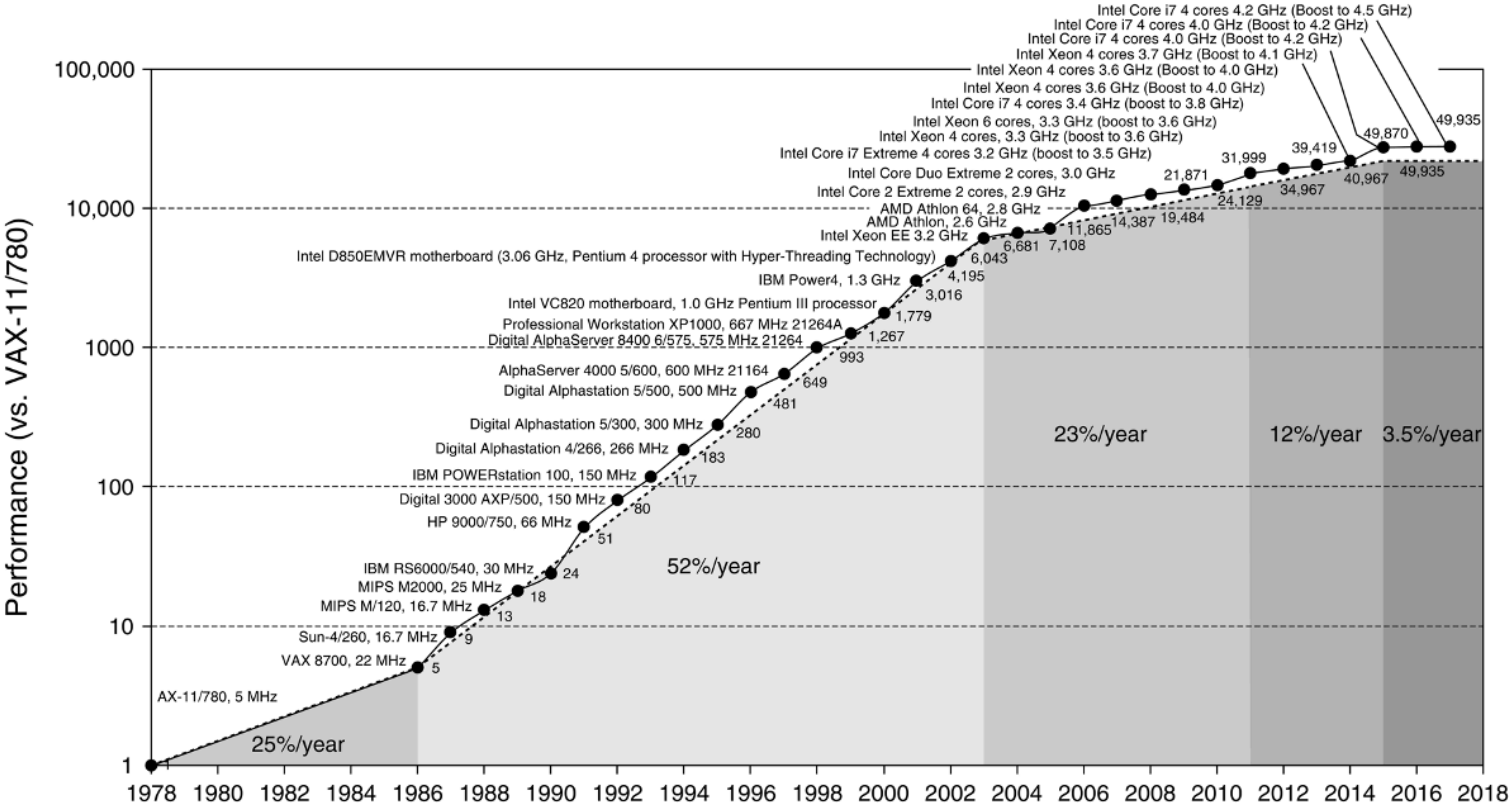
Computer Architecture

Chapter 1


Performance (vs. VAX-11/780)
100,000
10,000
1000
100
10
1
1978 1980 1982 1984 1986 1988 1990 1992 1994 1996 1998 2000 2002 2004 2006 2008 2010 2012 2014 2016 2018
25%/year
52%/year
23%/year
12%/year
3.5%/year
AX-11/780, 5 MHz
VAX 8700, 22 MHz
Sun-4/260, 16.7 MHz
MIPS M/120, 16.7 MHz
MIPS M2000, 25 MHz
IBM RS6000/540, 30 MHz
HP 9000/750, 66 MHz
Digital 3000 AXP/500, 150 MHz
IBM POWERstation 100, 150 MHz
Digital Alphastation 4/266, 266 MHz
Digital Alphastation 5/300, 300 MHz
Digital Alphastation 5/500, 500 MHz
AlphaServer 4000 5/600, 600 MHz 21164
Digital AlphaServer 8400 6/575, 575 MHz 21264
Professional Workstation XP1000, 667 MHz 21264A
Intel VC820 motherboard, 1.0 GHz Pentium III processor
IBM Power4, 1.3 GHz
Intel D850EMVR motherboard (3.06 GHz, Pentium 4 processor with Hyper-Threading Technology)
Intel Xeon EE 3.2 GHz
AMD Athlon, 2.6 GHz
AMD Athlon 64, 2.8 GHz
Intel Core 2 Extreme 2 cores, 2.9 GHz
Intel Core Duo Extreme 2 cores, 3.0 GHz
Intel Core i7 Extreme 4 cores 3.2 GHz (boost to 3.5 GHz)
Intel Xeon 4 cores, 3.3 GHz (boost to 3.6 GHz)
Intel Xeon 6 cores, 3.3 GHz (boost to 3.6 GHz)
Intel Core i7 4 cores 3.4 GHz (boost to 3.8 GHz)
Intel Xeon 4 cores 3.6 GHz (Boost to 4.0 GHz)
Intel Xeon 4 cores 3.6 GHz (Boost to 4.0 GHz)
Intel Xeon 4 cores 3.7 GHz (Boost to 4.1 GHz)
Intel Core i7 4 cores 4.0 GHz (Boost to 4.2 GHz)
Intel Core i7 4 cores 4.0 GHz (Boost to 4.2 GHz)
Intel Core i7 4 cores 4.2 GHz (Boost to 4.5 GHz)
1
5
9
13
18
24
51
80
117
183
280
481
649
993
1,267
1,779
3,016
4,195
6,043
6,681
7,108
11,865
14,387
19,484
21,871
24,129
31,999
34,967
39,419
40,967
49,870
49,935
49,935

Classes of Computers

- Personal Mobile Device (PMD)
  - e.g. smart phones, tablet computers
  - Emphasis on energy efficiency and real-time
- Desktop Computing
  - Emphasis on price-performance
- Servers
  - Emphasis on availability, scalability, throughput
- Clusters / Warehouse Scale Computers
  - Used for “Software as a Service (SaaS)”
  - Emphasis on availability and price-performance
  - Sub-class: Supercomputers, emphasis: floating-point performance and fast internal networks
- Embedded Computers
  - Emphasis: price

Feature	Personal mobile device (PMD)	Desktop	Server	Clusters/warehouse-scale computer	Internet of things/ embedded
Price of system	$100–$1000	$300–$2500	$5000–$10,000,000	$100,000–$200,000,000	$10–$100,000
Price of microprocessor	$10–$100	$50–$500	$200–$2000	$50–$250	$0.01–$100
Critical system design issues	Cost, energy, media performance, responsiveness	Price-performance, energy, graphics performance	Throughput, availability, scalability, energy	Price-performance, throughput, energy proportionality	Price, energy, application-specific performance

Figure 1.2 A summary of the five mainstream computing classes and their system characteristics. Sales in 2015 included about 1.6 billion PMDs (90% cell phones), 275 million desktop PCs, and 15 million servers. The total number of embedded processors sold was nearly 19 billion. In total, 14.8 billion ARM-technology-based chips were shipped in 2015. Note the wide range in system price for servers and embedded systems, which go from USB keys to network routers. For servers, this range arises from the need for very large-scale multiprocessor systems for high-end transaction processing.

Parallelism

- Classes of parallelism in applications:
  - Data-Level Parallelism (DLP)
  - Task-Level Parallelism (TLP)
- Classes of architectural parallelism:
  - Instruction-Level Parallelism (ILP)
  - Vector architectures/Graphic Processor Units (GPUs)
  - Thread-Level Parallelism
  - Request-Level Parallelism

Flynn's Taxonomy (1966)

- Single instruction stream, single data stream (SISD)
- Single instruction stream, multiple data streams (SIMD)
  - Vector architectures
  - Multimedia extensions
  - Graphics processor units
- Multiple instruction streams, single data stream (MISD)
  - No commercial implementation
- Multiple instruction streams, multiple data streams (MIMD)
  - Tightly-coupled MIMD
  - Loosely-coupled MIMD

Instruction Set Architecture

Some examples

- 80x86
- IBM 360
- VAX 11
- MIPS
- SPARC
- ARM
- RISC-V

ISA Components

- ISA Classes (Accumulator, Stack, register-memory, load-store)
- Memory addressing (how processor views memory)
- Addressing modes (ways that memory locations can be referenced)
- Operand types, sizes (bit, byte, word, register, integers, floating pt, etc.)
- Operations (arithmetic, logical, compare, branch, jump, system, etc.)
- Control flow (how are compares done, affect on branches, etc.)
- Bit encoding (how are instructions encoded as bit strings

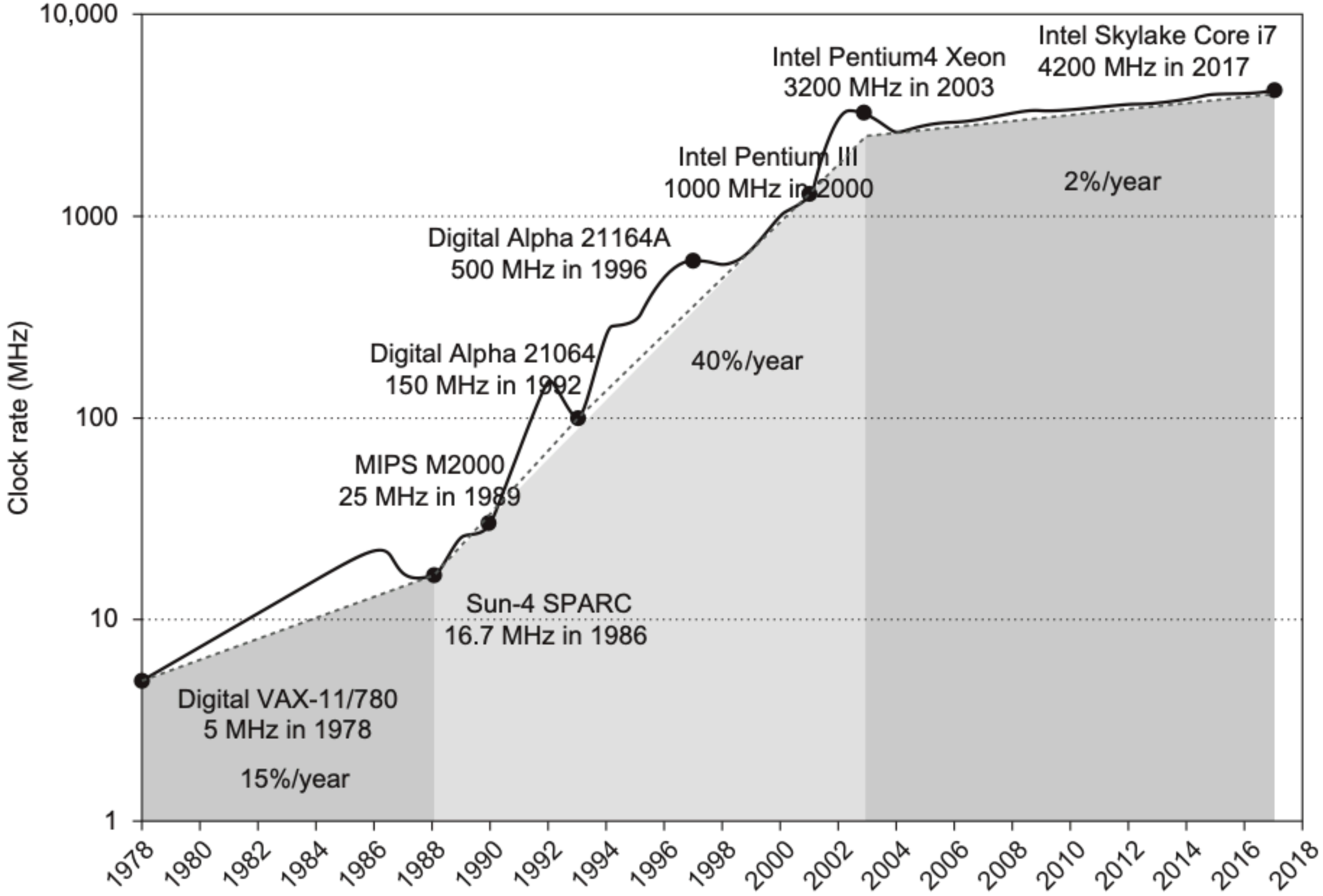


Figure 1.11 Growth in clock rate of microprocessors in Figure 1.1. Between 1978 and 1986, the clock rate improved less than 15% per year while performance improved by 22% per year. During the "renaissance period" of 52% performance improvement per year between 1986 and 2003, clock rates shot up almost 40% per year. Since then, the clock rate has been nearly flat, growing at less than 2% per year, while single processor performance improved recently at just 3.5% per year.

Measuring Performance

"Computer X is n times faster than computer Y"

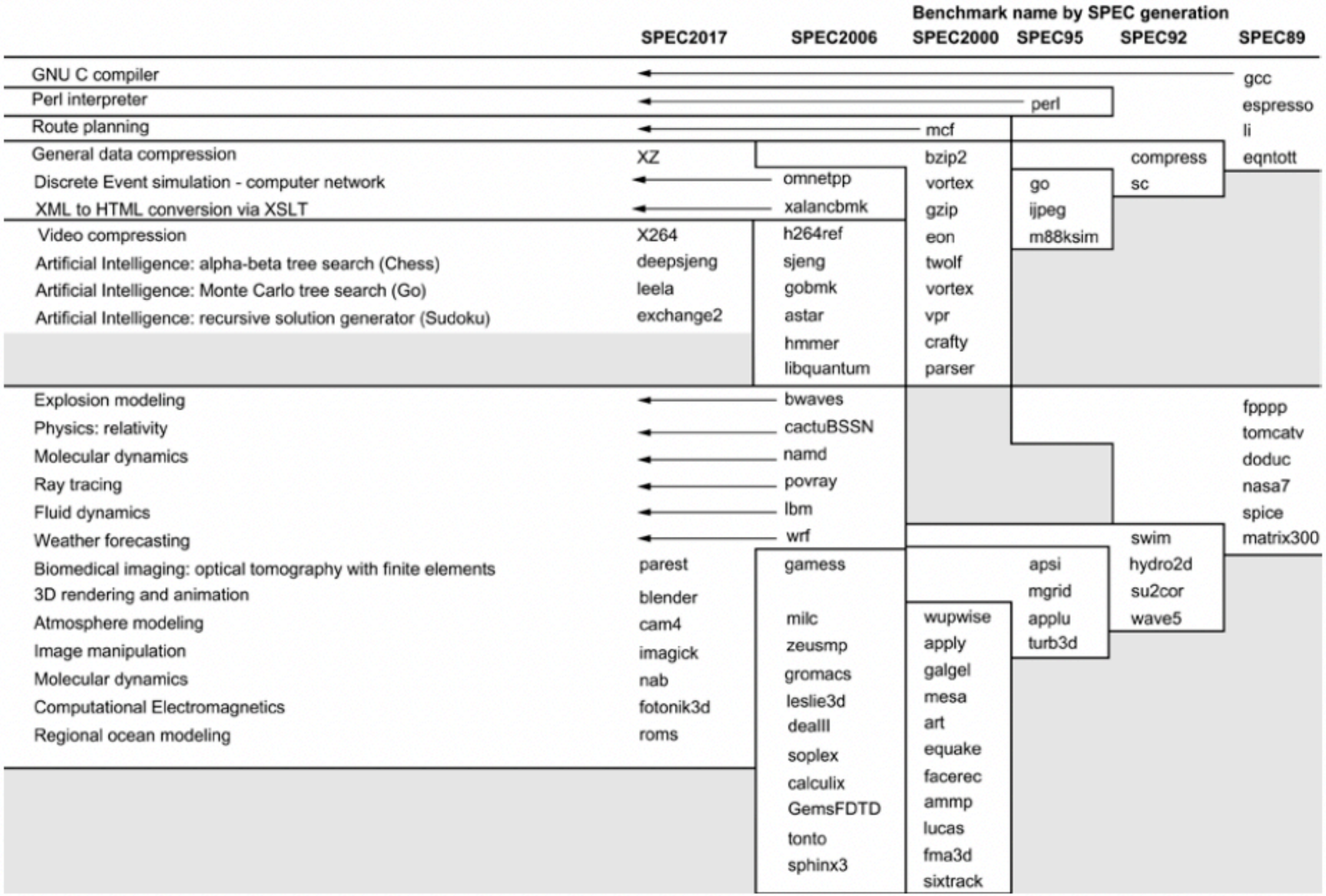
$$\frac{\text{Execution time}_y}{\text{Execution time}_x} = n$$

$$n = \frac{\text{Execution time}_y}{\text{Execution time}_x} = \frac{\text{Performance}_x}{\text{Performance}_y}$$

CHAP10010

Benchmarks

1. *Real Applications*
2. *Modified (or scripted) applications*
3. *Kernels*
4. *Toy benchmarks*
5. *Synthetic benchmarks*


Benchmark name by SPEC generation
SPEC2017
SPEC2006
SPEC2000
SPEC95
SPEC92
SPEC89
GNU C compiler
gcc
Perl interpreter
perl
espresso
Route planning
mcf
li
General data compression
XZ
bzip2
compress
eqntott
Discrete Event simulation - computer network
omnetpp
vortex
go
sc
XML to HTML conversion via XSLT
xalancbmk
gzip
ijpeg
Video compression
X264
h264ref
eon
m88ksim
Artificial Intelligence: alpha-beta tree search (Chess)
deepsjeng
sjeng
twolf
Artificial Intelligence: Monte Carlo tree search (Go)
leela
gobmk
vortex
Artificial Intelligence: recursive solution generator (Sudoku)
exchange2
astar
vpr
hmmer
crafty
libquantum
parser
Explosion modeling
bwaves
fpppp
Physics: relativity
cactuBSSN
tomcatv
Molecular dynamics
namd
doduc
Ray tracing
povray
nasa7
Fluid dynamics
lbm
spice
Weather forecasting
wrf
swim
matrix300
Biomedical imaging: optical tomography with finite elements
parest
gamess
apsi
hydro2d
3D rendering and animation
blender
mgrid
su2cor
Atmosphere modeling
cam4
milc
wupwise
applu
wave5
Image manipulation
imagick
zeusmp
apply
turb3d
Molecular dynamics
nab
gromacs
galgel
Computational Electromagnetics
fotonik3d
leslie3d
mesa
Regional ocean modeling
roms
dealII
art
equake
soplex
facerec
calculix
ammp
GemsFDTD
lucas
tonto
fma3d
sphinx3
sixtrack

Quantitative Principles

1. Make the Common Case Fast

CHAP1030

Quantitative Principles

2. Amdahl's Law

$$\text{Speedup} = \frac{\text{Total Performance using an enhancement}}{\text{Total Performance without the enhancement}}$$

CHAP1040

University of Idaho

Amdahl's Law

$$\text{Speedup} = \frac{\text{Execution time}_{old}}{\text{Execution time}_{new}} = \frac{1}{(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}}}$$

CHAP1065

Example Suppose that we want to enhance the processor used for web serving. The new processor is 10 times faster on computation in the web serving application than the old processor. Assuming that the original processor is busy with computation 40% of the time and is waiting for I/O 60% of the time, what is the overall speedup gained by incorporating the enhancement?

Answer

$$\text{Fraction}_{\text{enhanced}} = 0.4;\ \text{Speedup}_{\text{enhanced}} = 10;\ \text{Speedup}_{\text{overall}} = \frac{1}{0.6 + \frac{0.4}{10}} = \frac{1}{0.64} \approx 1.56$$

Amdahl's Law expresses the law of diminishing returns: The incremental improvement in speedup gained by an improvement of just a portion of the computation diminishes as improvements are added. An important corollary of Amdahl's Law is that if an enhancement is usable only for a fraction of a task, then we can't speed up the task by more than the reciprocal of 1 minus that fraction.

Quantitative Principles

3. CPU Performance Equations

CPU Time = CPU cycles for a program X Clock Cycle Time

or

$$\text{CPU time} = \frac{\text{CPU cycles for a program}}{\text{Clock Rate}}$$

$$\text{CPI} = \frac{\text{CPU cycles for a program}}{\text{Instruction Count}}$$

Quantitative Principles

4. Principle of locality

- *Temporal Locality*
- *Spacial Locality*

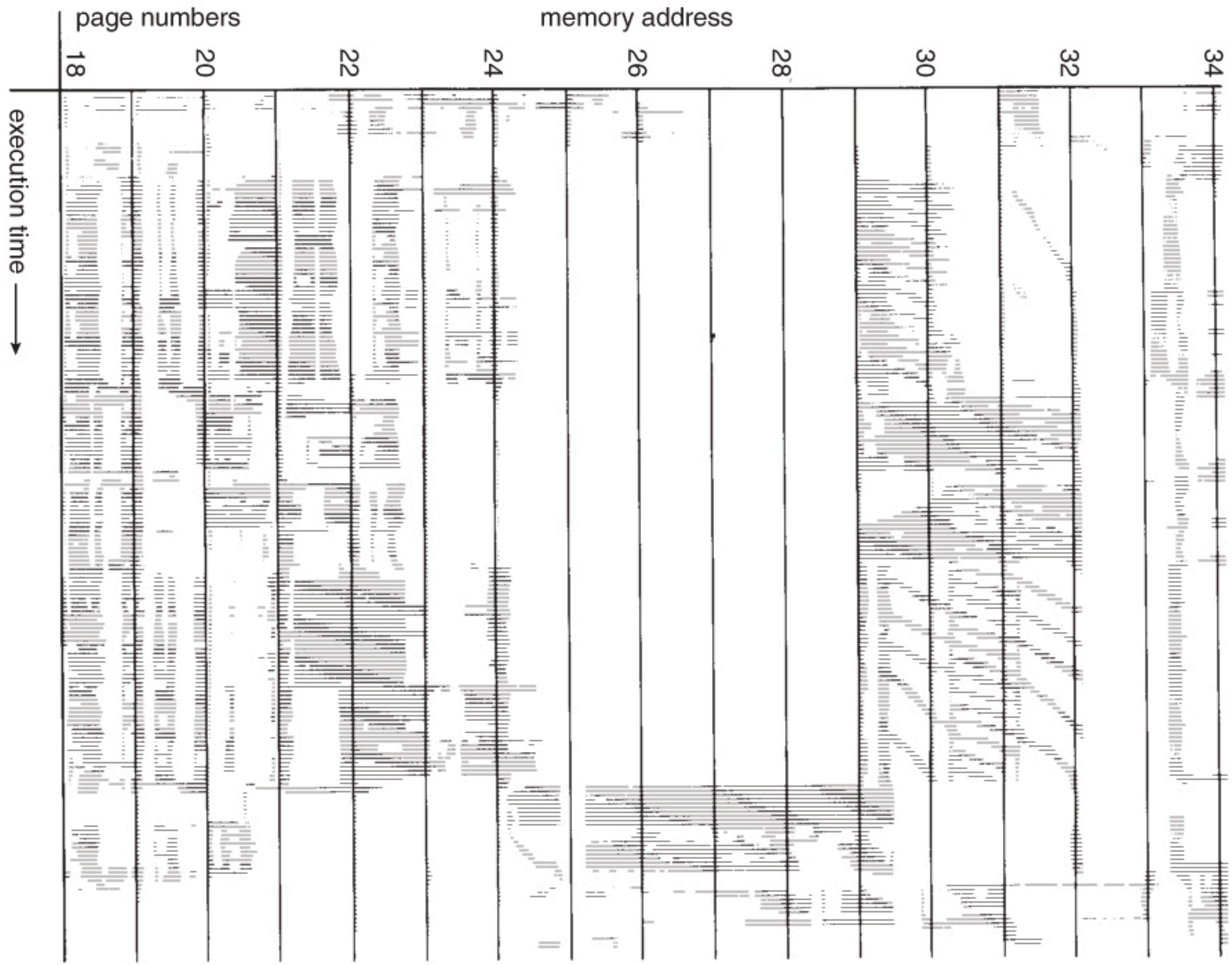


Figure 9.19 - Locality in a memory-reference pattern.

Quantitative Principles

5. Take advantage of parallelism

CHAP1070