CSE 613: Parallel Programming

Department of Computer Science
SUNY Stony Brook
Spring 2019

"We used to joke that
"parallel computing is the future, and always will be,"
but the pessimists have been proven wrong."

— *Tony Hey*

Course Information

- **Lecture Time:** MF 1:00 pm - 2:20 pm
- **Location:** Room 2120, Old CS Building, West Campus
- **Instructor:** Rezaul A. Chowdhury
- **Office Hours:** MF 4:00 pm - 5:30 pm, 239 New CS Building
- **Email:** rezaul@cs.stonybrook.edu
- **TA:** Unlikely
- **Class Webpage:**

  http://www3.cs.stonybrook.edu/~rezaul/CSE613-S19.html

Prerequisites

- **Required:** Background in algorithms analysis
  (e.g., CSE 373 or CSE 548)
- **Required:** Background in programming languages (C / C++)
- **Helpful but Not Required:** Background in computer architecture
- **Please Note:** This is not a course on
  - Programming languages
  - Computer architecture
- **Main Emphasis:** Parallel algorithms

Topics to be Covered

The following topics will be covered

- Analytical modeling of parallel programs
- Scheduling
- Programming using the message-passing paradigm and for shared address-space platforms
- Parallel algorithms for dense matrix operations, sorting, searching, graphs, computational geometry, and dynamic programming
- Concurrent data structures
- Transactional memory, etc.

Grading Policy

- Homeworks (three: lowest score 8%, highest score 20%, and the remaining one 12%): 40%
- Group project (one): 45%
  - Proposal: Feb 22
  - Progress report: Apr 1
  - Final demo / report: May 6 - May 10
- Scribe note (one lecture): 10%
- Class participation & attendance: 5%

Programming Environment

This course is supported by an educational grant from

- Extreme Science and Engineering Discovery Environment (XSEDE): https://www.xsede.org

We have access to the following supercomputing resources

- **Stampede 2 (** **Texas Advanced Computing Center** **):**
  4,200 KNL nodes with 68 cores (Intel Xeon Phi 7250 / "Knights Landing") each; 1,736 SKX nodes each with 48 cores (Intel Xeon Platinum / "Skylake") on two sockets.

- **Comet (** **San Diego Supercomputer Center** **):**
  1,984 nodes with 24 cores (2 Intel Haswell) per node. The Comet GPU resource features 36 K80 GPU nodes (with 2 Intel Haswell processors each), and 36 P100 nodes (with 2 Intel Broadwell processors each).

Programming Environment

World's Most Powerful Supercomputers in November, 2018

(www.top500.org)

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , IBM DOE/SC/Oak Ridge National Laboratory United States	2,397,824	143,500.0	200,794.9	9,783
2	**Sierra** - IBM Power System S922LC, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	1,572,480	94,640.0	125,712.0	7,438.3
3	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway , NRCPC National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
4	**Tianhe-2A** - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000 , NUDT National Super Computer Center in Guangzhou China	4,981,760	61,444.5	100,678.7	18,482
5	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 , Cray Inc. Swiss National Supercomputing Centre (CSCS) Switzerland	387,872	21,230.0	27,154.3	2,384.2
6	**Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , Cray Inc. DOE/NNSA/LANL/SNL United States	979,072	20,158.7	41,461.2	7,578.1
7	**AI Bridging Cloud Infrastructure (ABCI)** - PRIMERGY CX2570 M4, Xeon Gold 6148 20C 2.4GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR , Fujitsu National Institute of Advanced Industrial Science and Technology (AIST) Japan	391,680	19,880.0	32,576.6	1,649.3
8	**SuperMUC-NG** - ThinkSystem SD530, Xeon Platinum 8174 24C 3.1GHz, Intel Omni-Path , Lenovo Leibniz Rechenzentrum Germany	305,856	19,476.6	26,873.9	
9	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x , Cray Inc. DOE/SC/Oak Ridge National Laboratory United States	560,640	17,590.0	27,112.5	8,209
10	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom , IBM DOE/NNSA/LLNL United States	1,572,864	17,173.2	20,132.7	7,890

Programming Environment

World's Most Powerful Supercomputers in November, 2018 (www.top500.org)

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
11	**Lassen** - IBM Power System S922LC, IBM POWER9 22C 3.1GHz, Dual-rail Mellanox EDR Infiniband, NVIDIA Tesla V100 , IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	248,976	15,430.0	19,904.4	
12	**Cori** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , Cray Inc. DOE/SC/LBNL/NERSC United States	622,336	14,014.7	27,880.7	3,939
13	**Nurion** - Cray CS500, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path , Cray Inc. Korea Institute of Science and Technology Information Korea, South	570,020	13,929.3	25,705.9	
14	**Oakforest-PACS** - PRIMERGY CX1640 M1, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path , Fujitsu Joint Center for Advanced High Performance Computing Japan	556,104	13,554.6	24,913.5	2,718.7
15	**HPC4** - Proliant DL380 Gen10, Xeon Platinum 8160 24C 2.1GHz, Mellanox InfiniBand EDR, NVIDIA Tesla P100 , HPE Eni S.p.A. Italy	253,600	12,210.0	18,621.1	1,320
16	**Tera-1000-2** - Bull Sequana X1000, Intel Xeon Phi 7250 68C 1.4GHz, Bull BXI 1.2 , Bull, Atos Group Commissariat a l'Energie Atomique (CEA) France	561,408	11,965.5	23,396.4	3,178
17	**Stampede2** - PowerEdge C6320P/C6420, Intel Xeon Phi 7250 68C 1.4GHz/Platinum 8160, Intel Omni-Path , Dell EMC Texas Advanced Computing Center/Univ. of Texas United States	367,024	10,680.7	18,309.2	
18	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect , Fujitsu RIKEN Advanced Institute for Computational Science (AICS) Japan	705,024	10,510.0	11,280.4	12,659.9
19	**Marconi Intel Xeon Phi** - CINECA Cluster, Lenovo SD530/S720AP, Intel Xeon Phi 7250 68C 1.4GHz/Platinum 8160, Intel Omni-Path , Lenovo CINECA Italy	348,000	10,384.9	18,816.0	
20	**Taiwania 2** - QCT QuantaGrid D52G-4U/LC, Xeon Gold 6154 18C 3GHz, Mellanox InfiniBand EDR, NVIDIA Tesla V100 SXM2 , Quanta Computer / Taiwan Fixed Network / ASUS Cloud National Center for High Performance Computing Taiwan	170,352	9,000.0	15,208.2	797.5

Programming Environment

World's Most Powerful Supercomputers in November, 2017

(www.top500.org)

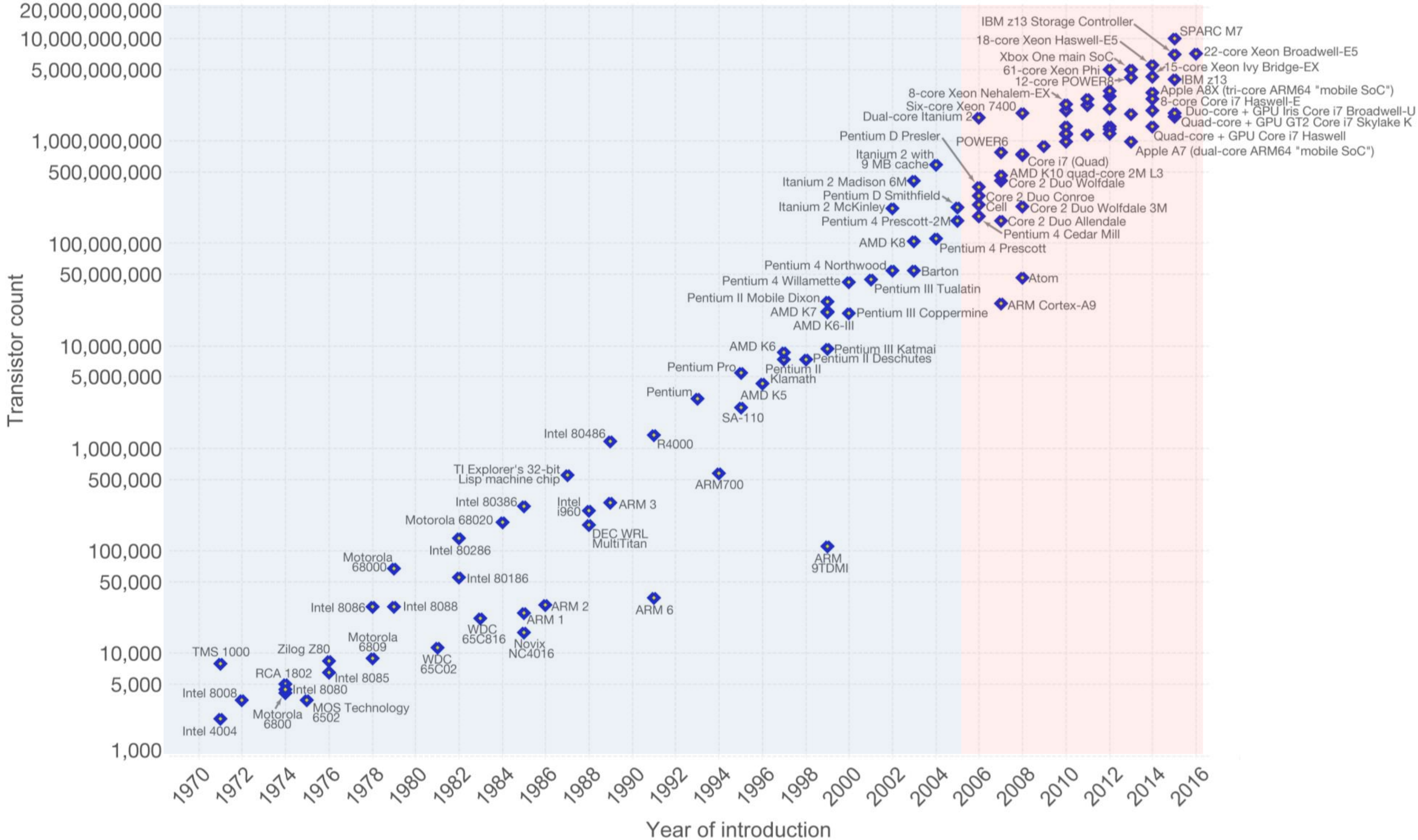
Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway , NRCPC National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
2	**Tianhe-2A** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P , NUDT National Super Computer Center in Guangzhou China	3,120,000	33,862.7	54,902.4	17,808
3	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 , Cray Inc. Swiss National Supercomputing Centre (CSCS) Switzerland	361,760	19,590.0	25,326.3	2,272.0
4	**Gyoukou** - ZettaScaler-2.2 HPC system, Xeon D-1571 16C 1.3GHz, Infiniband EDR, PEZY-SC2 700Mhz , ExaScaler Japan Agency for Marine-Earth Science and Technology Japan	19,860,000	19,135.8	28,192.0	1,350.2
5	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x , Cray Inc. DOE/SC/Oak Ridge National Laboratory United States	560,640	17,590.0	27,112.5	8,209
6	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom , IBM DOE/NNSA/LLNL United States	1,572,864	17,173.2	20,132.7	7,890
7	**Trinity** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , Cray Inc. DOE/NNSA/LANL/SNL United States	979,968	14,137.3	43,902.6	3,843.6
8	**Cori** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , Cray Inc. DOE/SC/LBNL/NERSC United States	622,336	14,014.7	27,880.7	3,939
9	**Oakforest-PACS** - PRIMERGY CX1640 M1, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path , Fujitsu Joint Center for Advanced High Performance Computing Japan	556,104	13,554.6	24,913.5	2,718.7
10	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect , Fujitsu RIKEN Advanced Institute for Computational Science (AICS) Japan	705,024	10,510.0	11,280.4	12,659.9
11	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom , IBM DOE/SC/Argonne National Laboratory United States	786,432	8,586.6	10,066.3	3,945
12	**Stampede2** - PowerEdge C6320P/C6420, Intel Xeon Phi 7250 68C 1.4GHz/Platinum 8160, Intel Omni-Path , Dell EMC Texas Advanced Computing Center/Univ. of Texas United States	368,928	8,317.7	18,215.8	

Recommended Textbooks

– A. Grama, G. Karypis, V. Kumar, and A. Gupta. ***Introduction to Parallel Computing*** (2nd Edition), Addison Wesley, 2003.
– J. JáJá. ***An Introduction to Parallel Algorithms*** (1st Edition), Addison Wesley, 1992.
– T. Cormen, C. Leiserson, R. Rivest, and C. Stein. ***Introduction to Algorithms*** (3rd Edition), MIT Press, 2009.
– M. Herlihy and N. Shavit. ***The Art of Multiprocessor Programming*** (1st Edition), Morgan Kaufmann, 2008.
– P. Pacheco. ***Parallel Programming with MPI*** (1st Edition), Morgan Kaufmann, 1996.

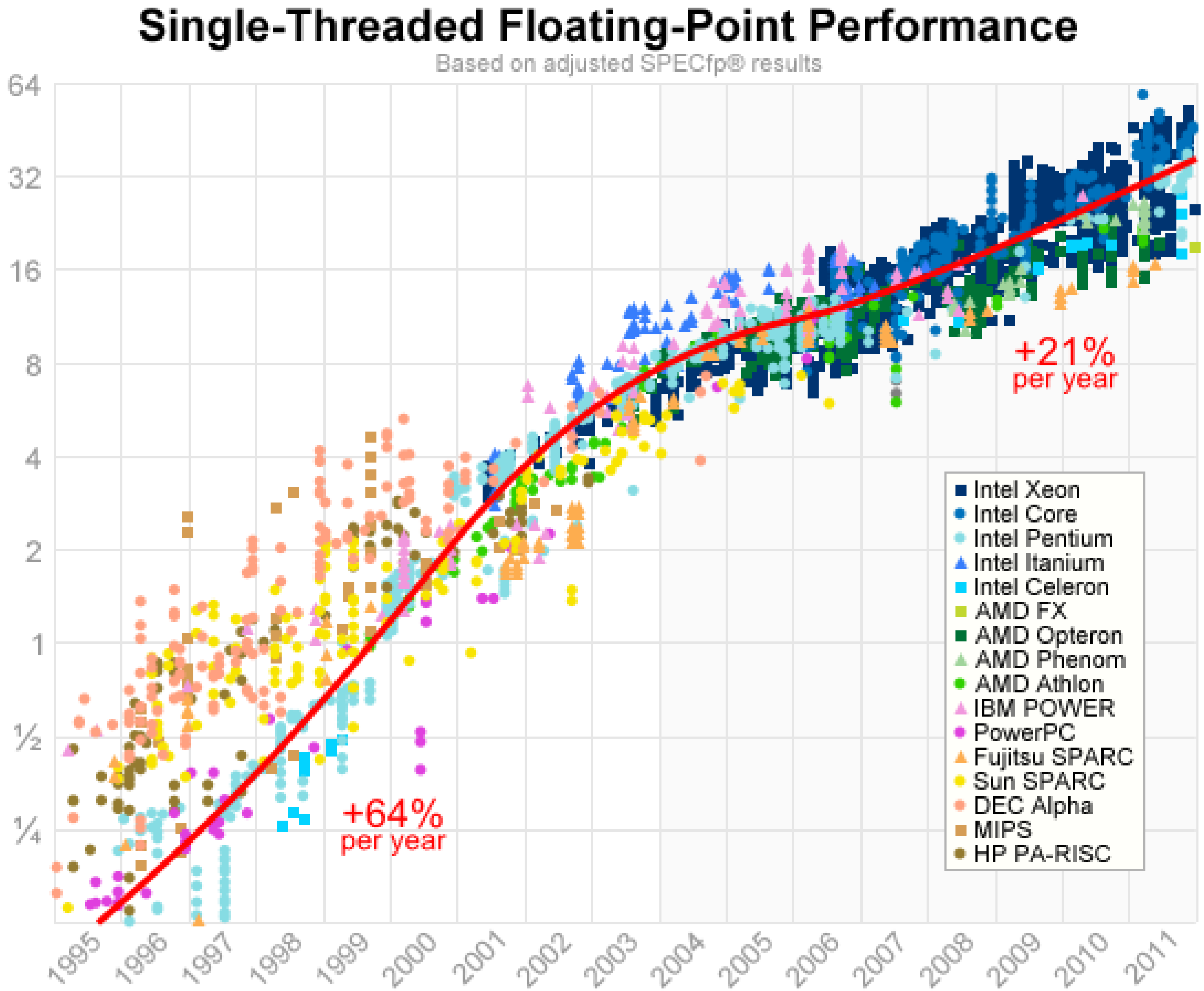
Why Parallelism?

Moore's Law



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Unicore Performance



Source: Jeff Preshing, 2012, http://preshing.com/20120208/a-look-back-at-single-threaded-cpu-performance/

Unicore Performance Has Hit a Wall!

Some Reasons

- Lack of additional ILP
  (Instruction Level Hidden Parallelism)
- High power density
- Manufacturing issues
- Physical limits
- Memory speed

Unicore Performance: No Additional ILP

"Everything that can be invented has been invented."

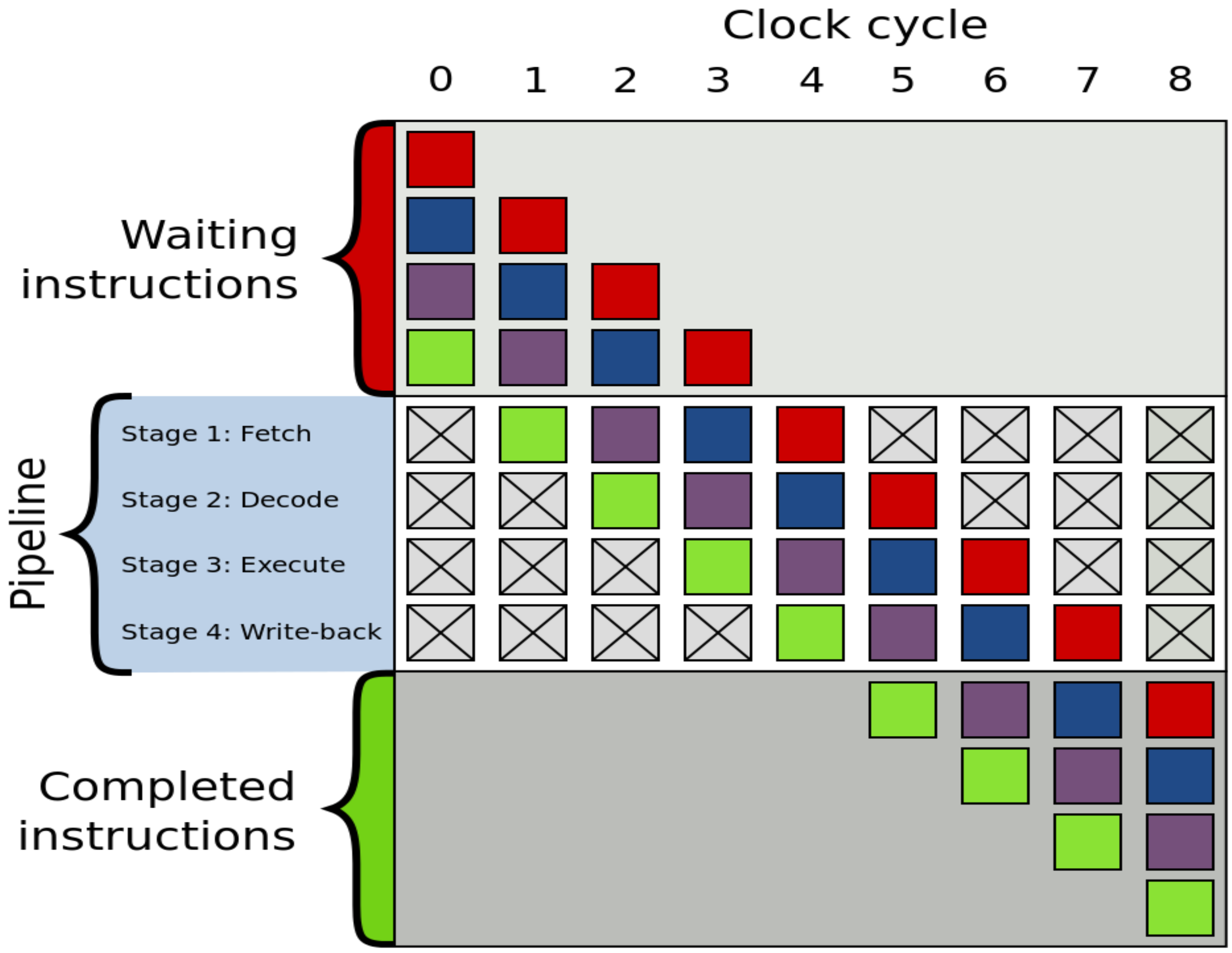
— *Charles H. Duell*

Commissioner, U.S. patent office, 1899

Exhausted all ideas to exploit hidden parallelism?

- Multiple simultaneous instructions
- Instruction Pipelining
- Out-of-order instructions
- Speculative execution
- Branch prediction
- Register renaming, etc.

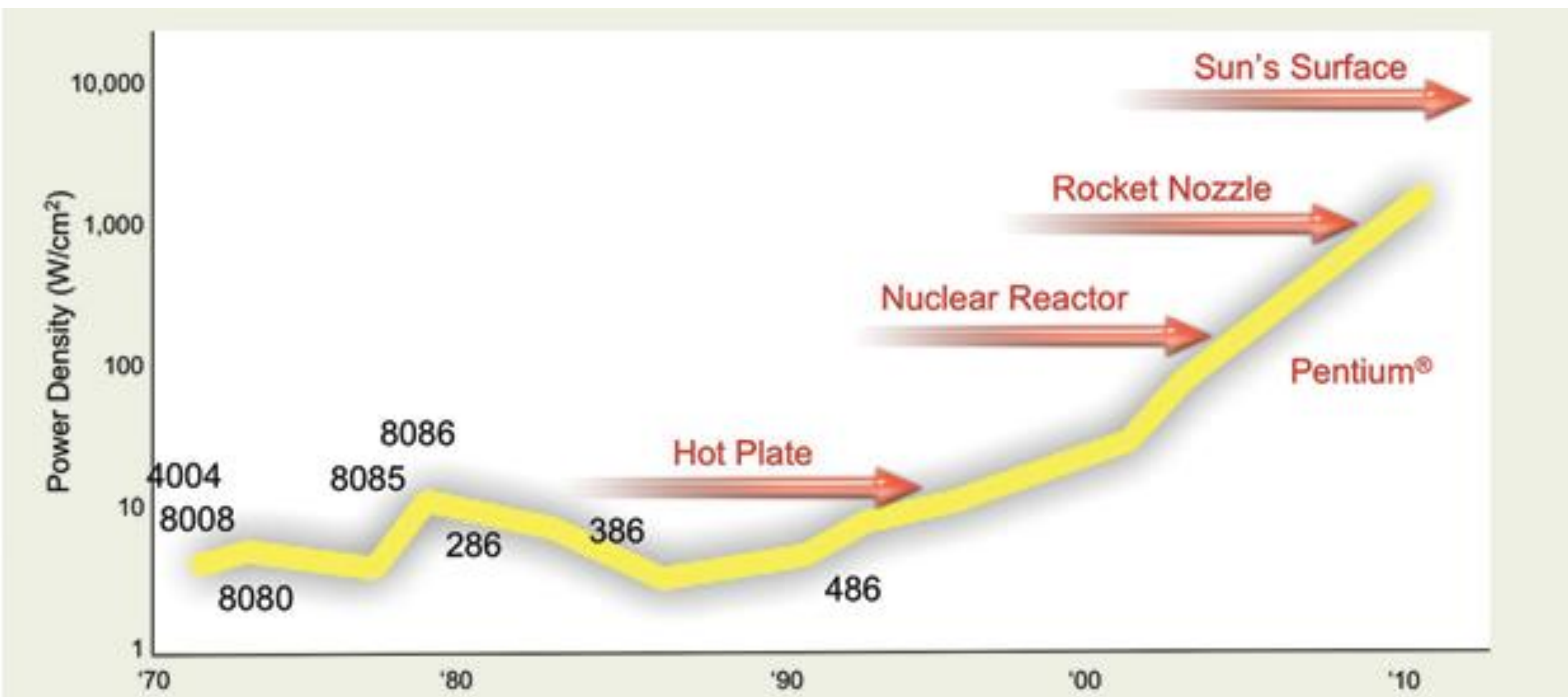
ILP: Instruction Pipelining



Source: Wikipedia

Unicore Performance: High Power Density

- Dynamic power, $P_d \propto V^2 f C$
  - *V = supply voltage*
  - *f = clock frequency*
  - *C = capacitance*
- But $V \propto f$
- Thus $P_d \propto f^3$



Source: Patrick Gelsinger, Intel Developer Forum, Spring 2004 (Simon Floyd)

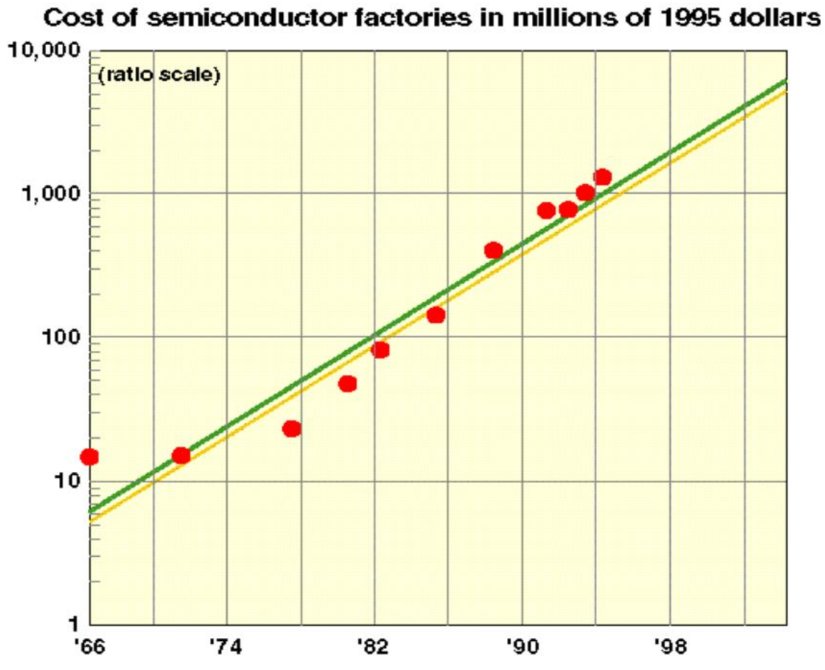
Unicore Performance: Manufacturing Issues

- Frequency, $f \propto 1 / s$
    - *s = feature size (transistor dimension)*
- Transistors / unit area $\propto 1 / s^2$
- Typically, die size $\propto 1 / s$
- So, what happens if feature size goes down by a factor of x?
    - Raw computing power goes up by a factor of x^4 !
    - Typically most programs run faster by a factor of x^3 without any change!

Source: Kathy Yelick and Jim Demmel, UC Berkeley

Unicore Performance: Manufacturing Issues

- Manufacturing cost goes up as feature size decreases
  - Cost of a semiconductor fabrication plant doubles every 4 years (Rock's Law)
- CMOS feature size is limited to 5 nm (at least 10 atoms)



Source: Kathy Yelick and Jim Demmel, UC Berkeley

Unicore Performance: Physical Limits

Execute the following loop on a serial machine in 1 second:

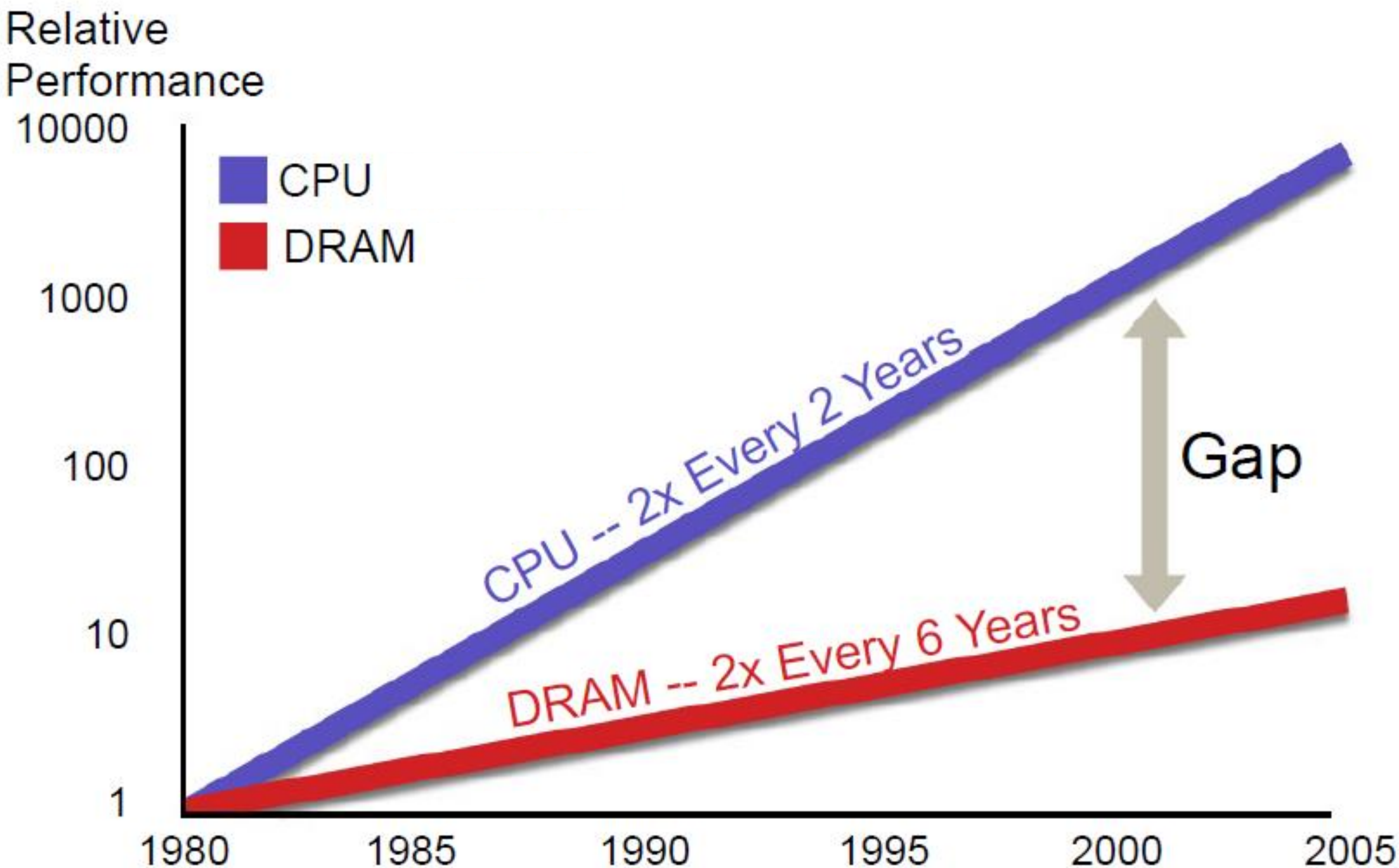
$$\textit{for}\ (\ i = 0;\ i < 10^{12};\ {+}{+}i\ )$$
$$z[\ i\ ] = x[\ i\ ] + y[\ i\ ];$$

- We will have to access $3{\times}10^{12}$ data items in one second
- Speed of light is, $c \approx 3{\times}10^{8}$ m/s
- So each data item must be within $c\ /\ 3{\times}10^{12} \approx 0.1$ mm from the CPU on the average
- All data must be put inside a 0.2 mm × 0.2 mm square
- Each data item (≥ 8 bytes) can occupy only 1 $Å^2$ space!

(size of a small atom!)

Source: Kathy Yelick and Jim Demmel, UC Berkeley

Unicore Performance: Memory Wall



Source: Sun World Wide Analyst Conference Feb. 25, 2003

Source: Rick Hetherington, Chief Technology Officer, Microelectronics, Sun Microsystems

Unicore Performance Has Hit a Wall!

Some Reasons

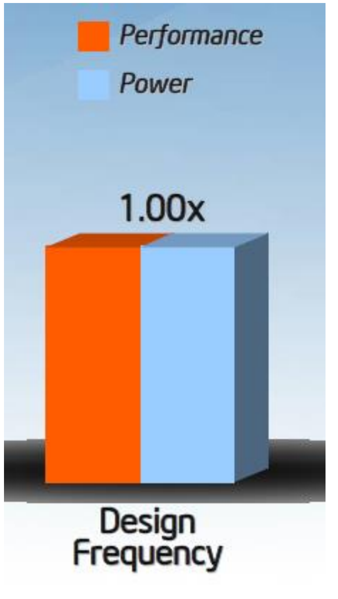
- Lack of additional ILP
  (Instruction Level Hidden Parallelism)
- High power density
- Manufacturing issues
- Physical limits
- Memory speed

"Oh Sinnerman, where you gonna run to?"

— Sinnerman (recorded by Nina Simone)

Where You Gonna Run To?

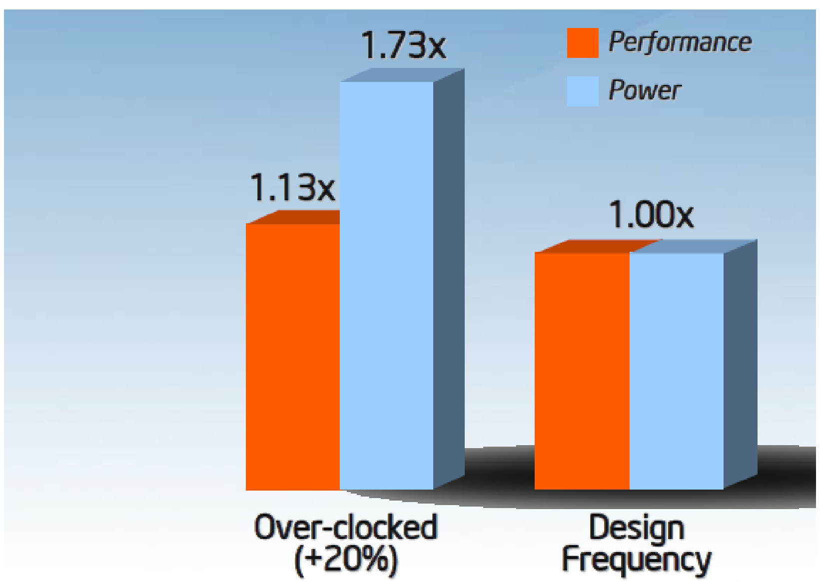
- Changing f by 20% changes performance by 13%
- So what happens if we overclock by 20%?



Source: Andrew A. Chien, Vice President of Research, Intel Corporation

Where You Gonna Run To?

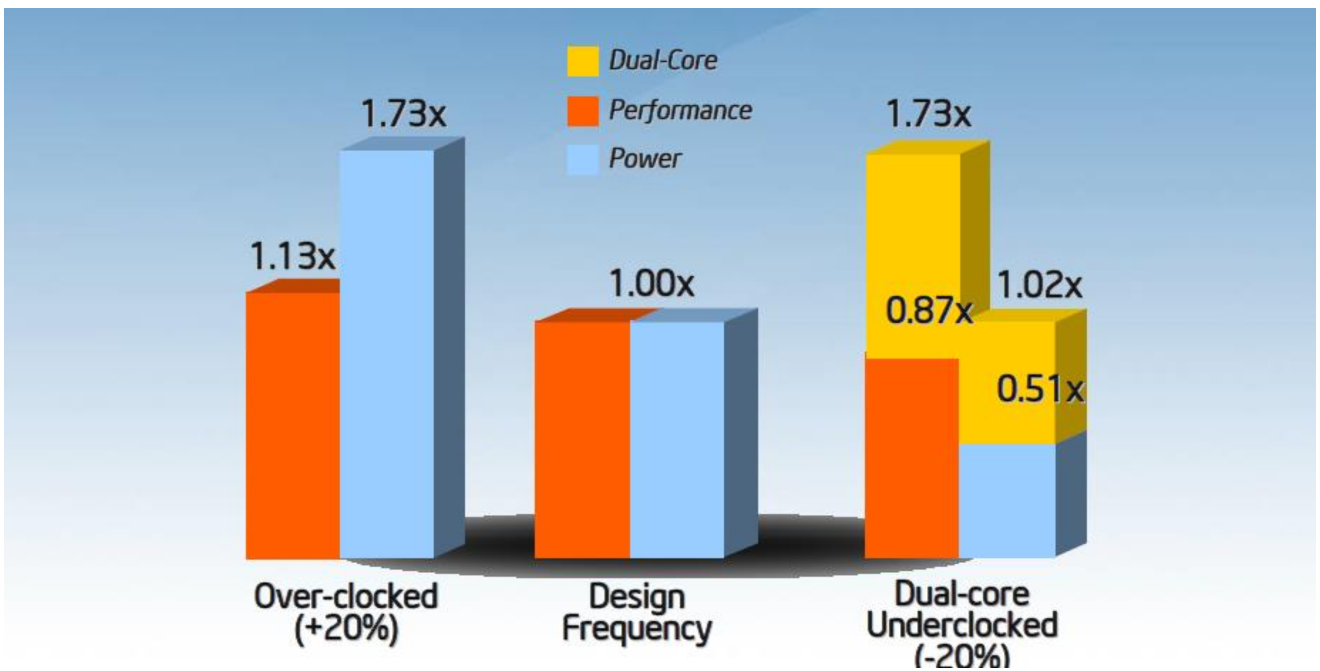
- Changing f by 20% changes performance by 13%
- So what happens if we overclock by 20%?
- And underclock by 20%?



Source: Andrew A. Chien, Vice President of Research, Intel Corporation

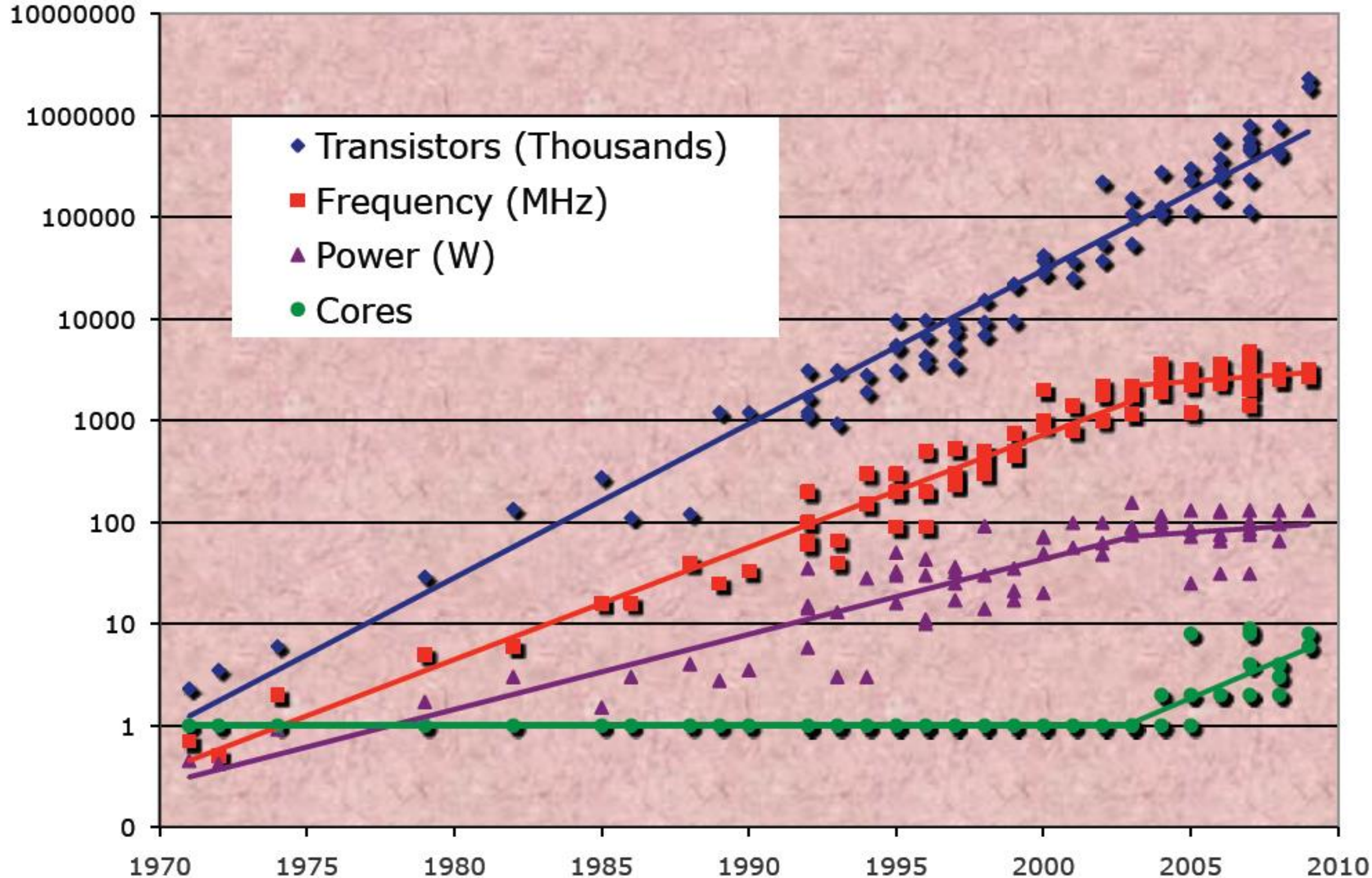
Where You Gonna Run To?

- Changing f by 20% changes performance by 13%
- So what happens if we overclock by 20%?
- And underclock by 20%?



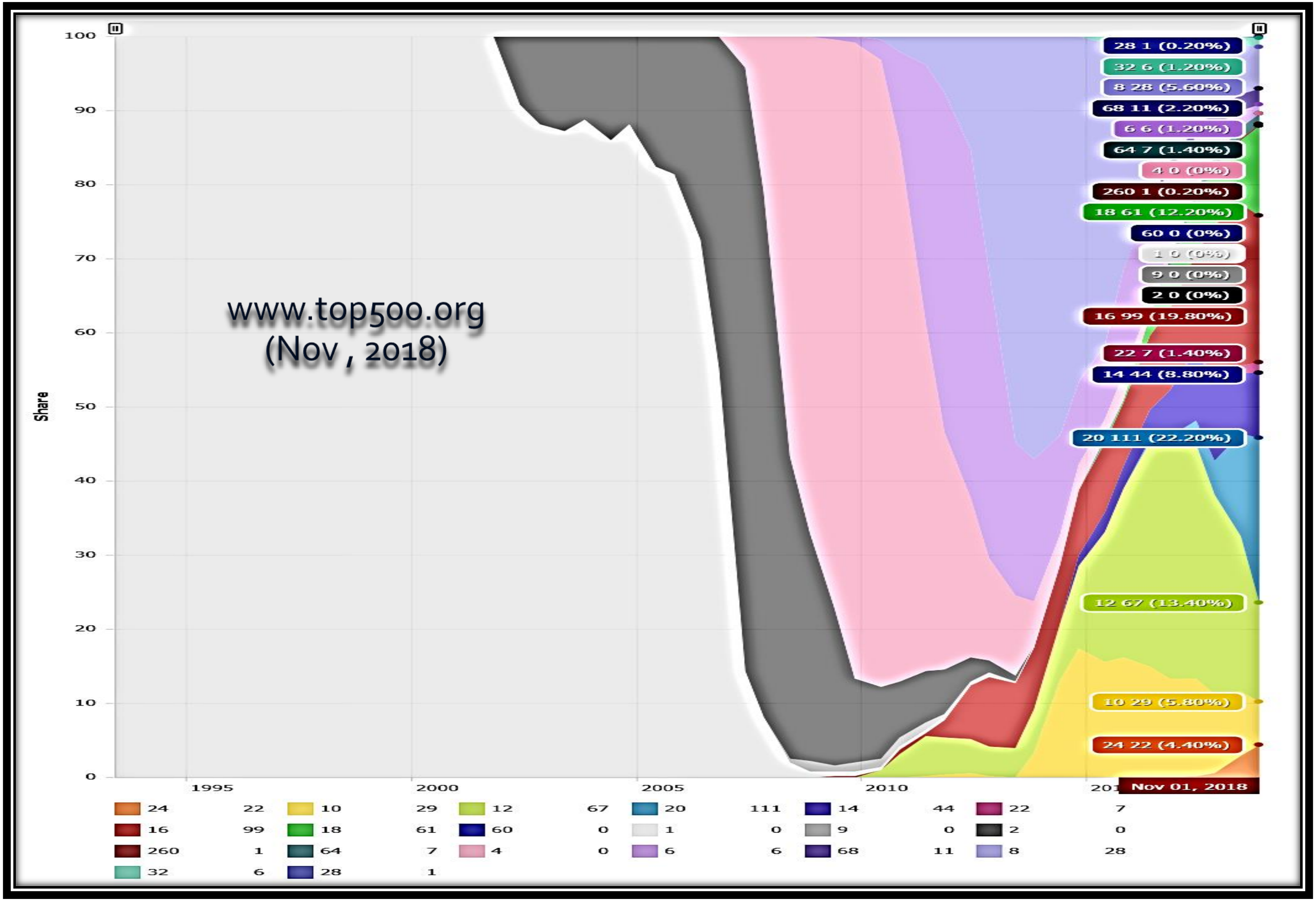
Source: Andrew A. Chien, Vice President of Research, Intel Corporation

Moore's Law Reinterpreted

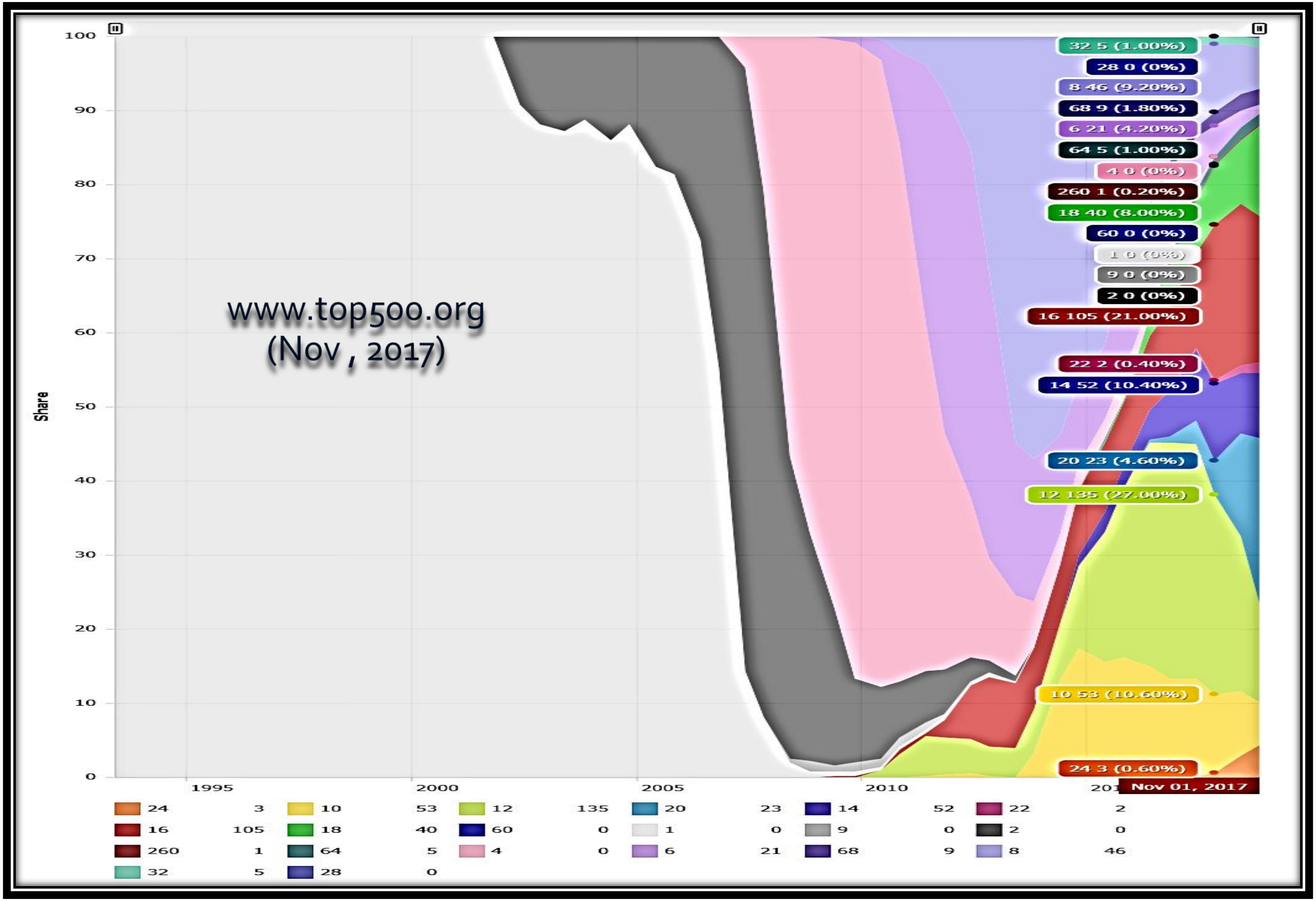


Source: Report of the 2011 Workshop on Exascale Programming Challenges

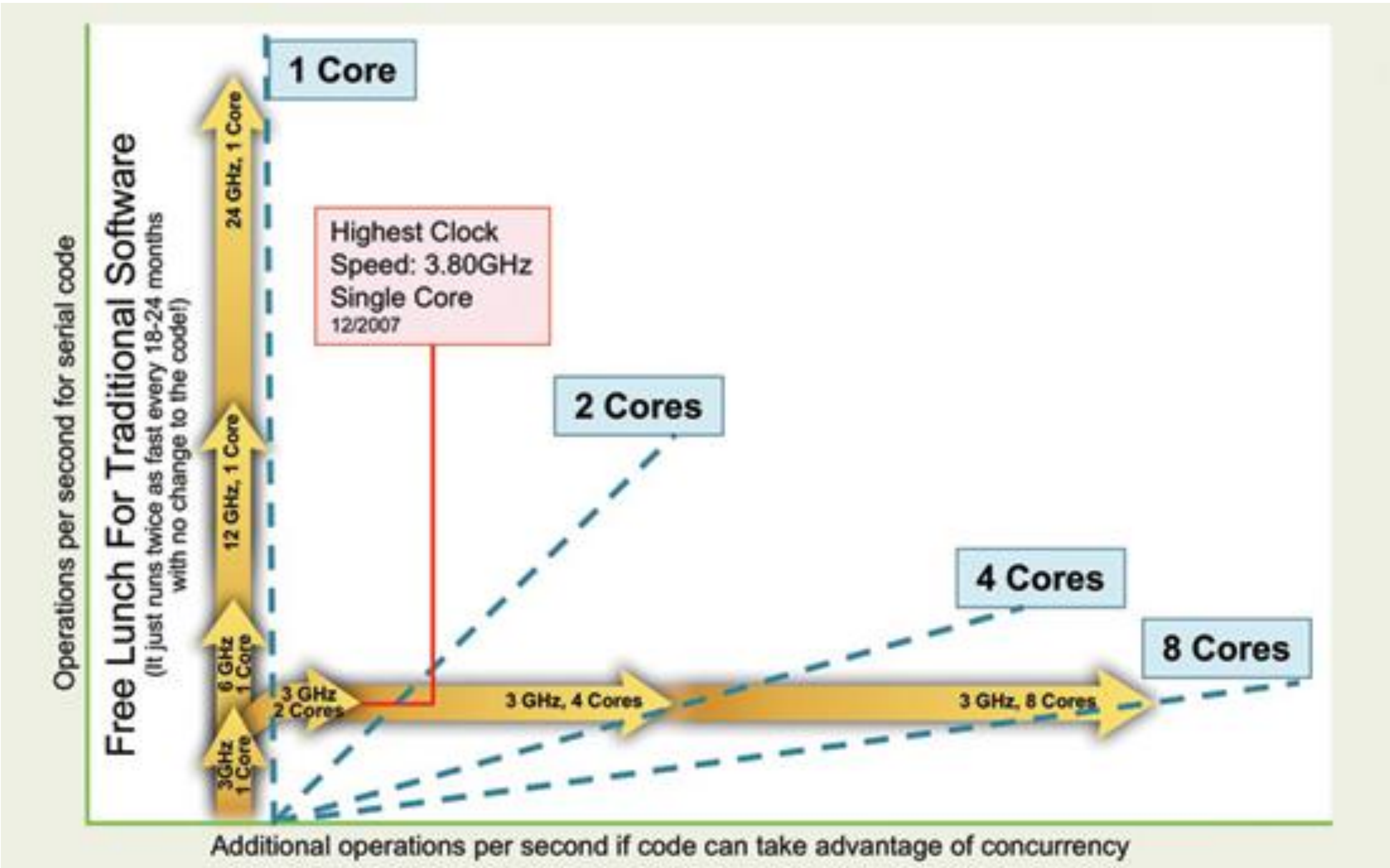
Top 500 Supercomputing Sites (Cores / Socket)



Top 500 Supercomputing Sites (Cores / Socket)



No Free Lunch for Traditional Software



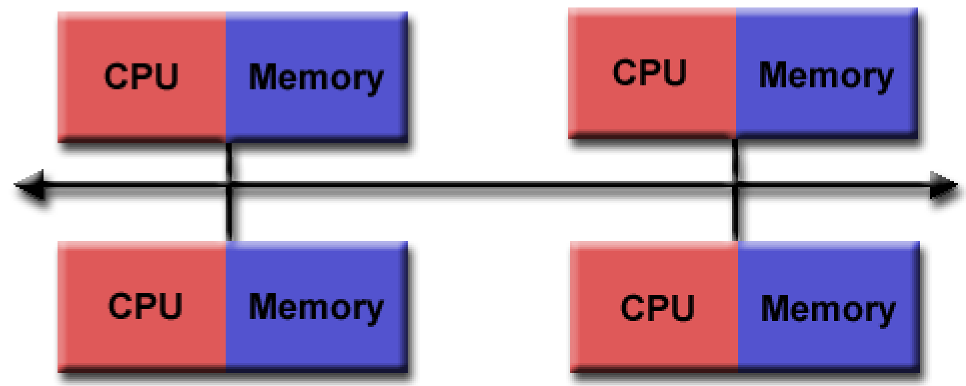
Source: Simon Floyd, Workstation Performance: Tomorrow's Possibilities (Viewpoint Column)

Insatiable Demand for Performance

Source: Patrick Gelsinger, Intel Developer Forum, 2008

Some Useful Classifications of Parallel Computers

Parallel Computer Memory Architecture (Distributed Memory)

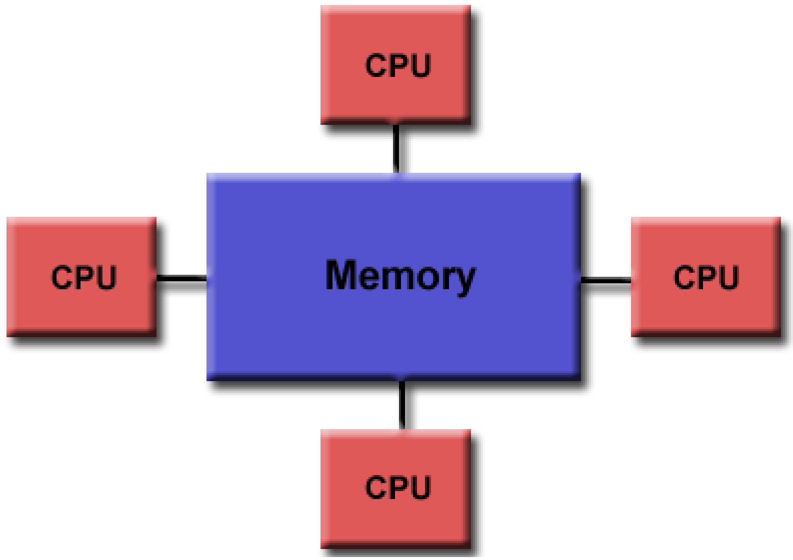


Source: Blaise Barney, LLNL

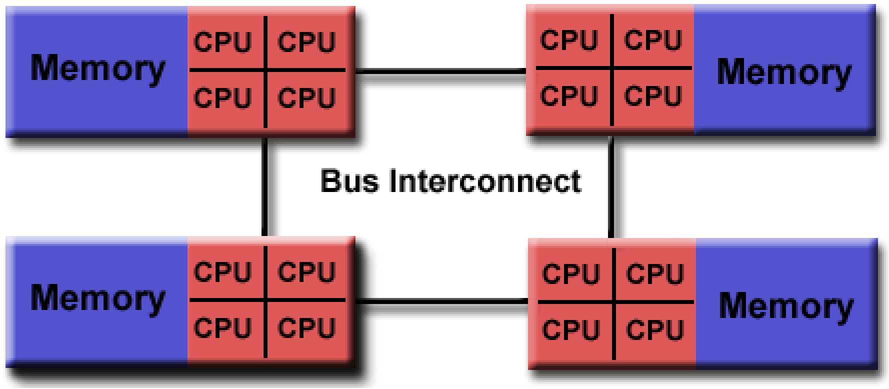
- Each processor has its own local memory — no global address space
- Changes in local memory by one processor have no effect on memory of other processors
- Communication network to connect inter-processor memory
- Programming
  - Message Passing Interface (MPI)
  - Many once available: PVM, Chameleon, MPL, NX, etc.

Parallel Computer Memory Architecture (Shared Memory)

- All processors access all memory as global address space
- Changes in memory by one processor are visible to all others
- Two types
  - Uniform Memory Access (UMA)
  - Non-Uniform Memory Access (NUMA)
- Programming
  - Open Multi-Processing (OpenMP)
  - Cilk/Cilk++ and Intel Cilk Plus
  - Intel Thread Building Block (TBB), etc.



UMA

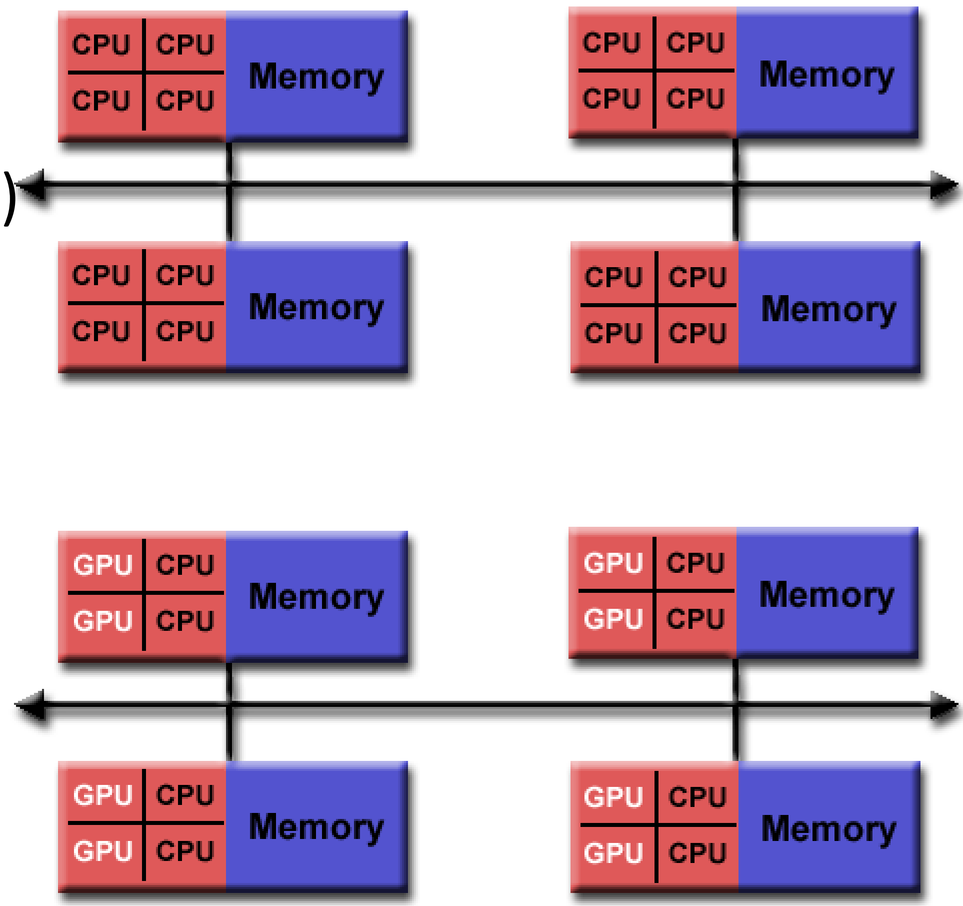


NUMA

Source: Blaise Barney, LLNL

Parallel Computer Memory Architecture (Hybrid Distributed-Shared Memory)

- The share-memory component can be a cache-coherent SMP or a Graphics Processing Unit (GPU)
- The distributed-memory component is the networking of multiple SMP/GPU machines
- Most common architecture for the largest and fastest computers in the world today
- Programming
  - OpenMP / Cilk + CUDA / OpenCL + MPI, etc.



Source: Blaise Barney, LLNL

Flynn's Taxonomy of Parallel Computers

Flynn's classical taxonomy (1966):

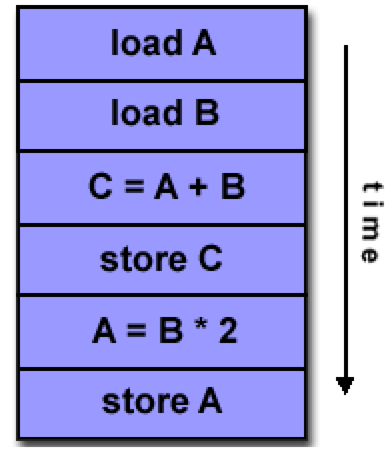
Classification of multi-processor computer architectures along two independent dimensions of *instruction* and *data*.

	Single Data (SD)	Multiple Data (MD)
Single Instruction (SI)	SISD	SIMD
Multiple Instruction (MI)	MISD	MIMD

Flynn's Taxonomy of Parallel Computers

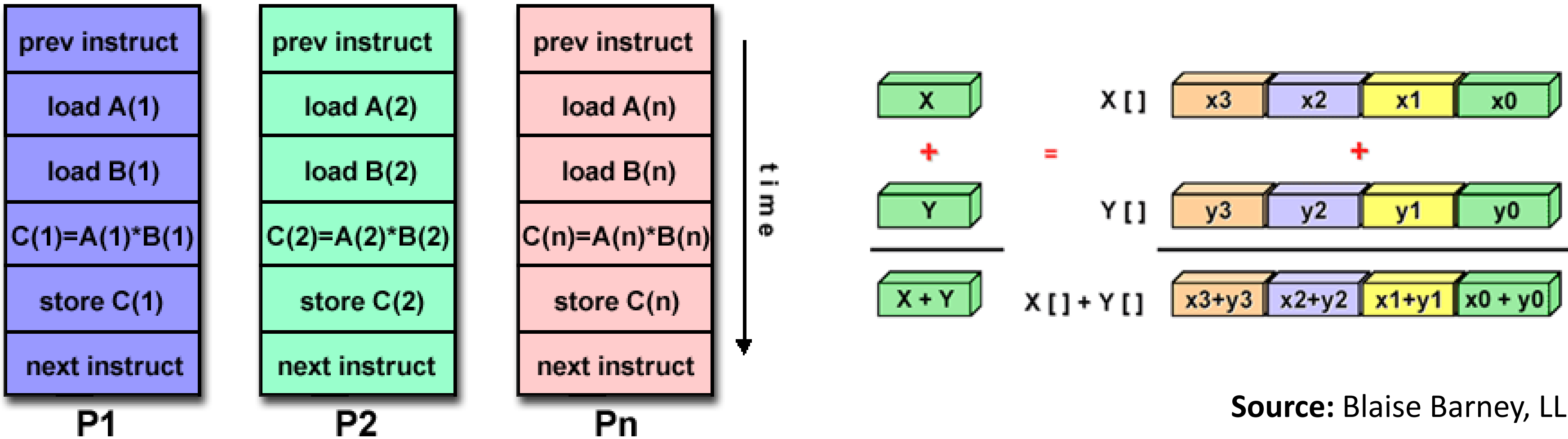
SISD

- A serial (non-parallel) computer
- The oldest and the most common type of computers
- Example: Uniprocessor unicore machines



Source: Blaise Barney, LLNL

Flynn's Taxonomy of Parallel Computers



Source: Blaise Barney, LLNL

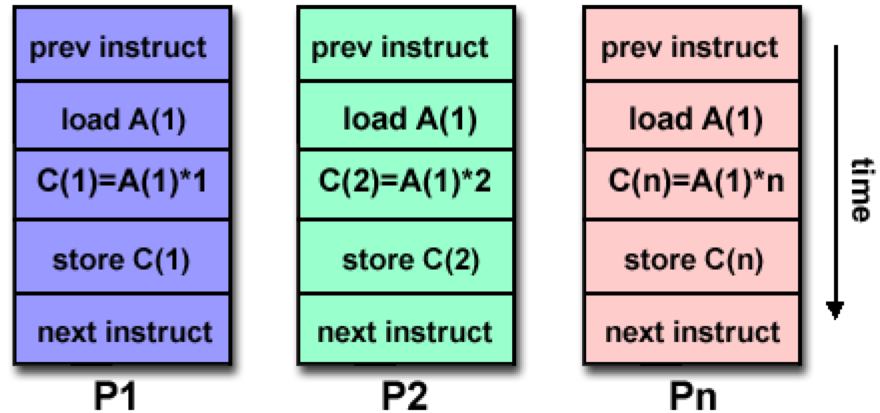
SIMD

- A type of parallel computer
- All PU's run the same instruction at any given clock cycle
- Each PU can act on a different data item
- Synchronous (lockstep) execution
- Two types: processor arrays and vector pipelines
- Example: GPUs (Graphics Processing Units)

Flynn's Taxonomy of Parallel Computers

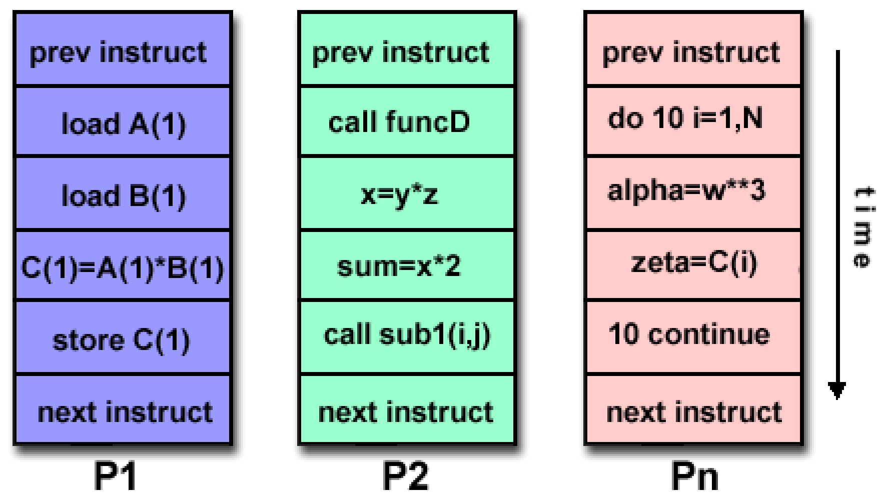
MISD

- A type of parallel computer
- Very few ever existed



MIMD

- A type of parallel computer
- Synchronous /asynchronous execution
- Examples: most modern supercomputers, parallel computing clusters, multicore PCs



Source: Blaise Barney, LLNL

Parallel Algorithms Warm-up

"The way the processor industry is going, is to add more and more cores, but nobody knows how to program those things. I mean, two, yeah; four, not really; eight, forget it."

— Steve Jobs, NY Times interview, June 10 2008

Parallel Algorithms Warm-up (1)

Consider the following loop:

$$\textit{for } i = 1 \textit{ to } n \textit{ do}$$

$$C[\,i\,] \leftarrow A[\,i\,] \times B[\,i\,]$$

- Suppose you have an infinite number of processors/cores
- Ignore all overheads due to scheduling, memory accesses, communication, etc.
- Suppose each operation takes a constant amount of time
- How long will this loop take to complete execution?

Parallel Algorithms Warm-up (1)

Consider the following loop:

$$\textit{for } i = 1 \textit{ to } n \textit{ do}$$

$$C[\,i\,] \leftarrow A[\,i\,] \times B[\,i\,]$$

– Suppose you have an infinite number of processors/cores

– Ignore all overheads due to scheduling, memory accesses, communication, etc.

– Suppose each operation takes a constant amount of time

– How long will this loop take to complete execution?

– $O(\,1\,)$ time

Parallel Algorithms Warm-up (2)

Now consider the following loop:

$$c \leftarrow 0$$

$$\mathit{for}\ i = 1\ \mathit{to}\ n\ \mathit{do}$$

$$\quad c \leftarrow c + A[\,i\,] \times B[\,i\,]$$

– How long will this loop take to complete execution?

Parallel Algorithms Warm-up (2)

Now consider the following loop:

$$c \leftarrow 0$$

$$\textit{for } i = 1 \textit{ to } n \textit{ do}$$

$$\quad c \leftarrow c + A[\ i\ ] \times B[\ i\ ]$$

- How long will this loop take to complete execution?
  - $O(\log n)$ time

Parallel Algorithms Warm-up (3)

Now consider quicksort:

$$QSort(\ A\ )$$

$$if\ |A| \leq 1\ return\ A$$

$$else\quad p \leftarrow A[\ rand(\ |A|\ )\ ]$$

$$\text{return}\ QSort(\ \{\ x \in A: x < p\ \}\ )$$

$$\#\ \{\ p\ \}\ \#$$

$$QSort(\ \{\ x \in A: x > p\ \}\ )$$

– Assuming that A is split in the middle everytime, and the two recursive calls can be made in parallel, how long will this algorithm take?

Parallel Algorithms Warm-up (3)

Now consider quicksort:

$$QSort(\ A\ )$$

$$\text{if } |A| \leq 1 \text{ return } A$$

$$\text{else} \quad p \leftarrow A[\ rand(\ |A|\ )\ ]$$

$$\text{return } QSort(\ \{\ x \in A : x < p\ \}\ )$$

$$\#\ \{\ p\ \}\ \#$$

$$QSort(\ \{\ x \in A : x > p\ \}\ )$$

– Assuming that A is split in the middle everytime, and the two recursive calls can be made in parallel, how long will this algorithm take?

  – $O(\ \log^2 n\ )$ (if partitioning takes logarithmic time)
  – $O(\ \log n\ )$ (but can be partitioned in constant time)