



Parallelism Challenges: Rewind and Fast Forward

Hesham El-Rewini, Ph.D., P.E.
Department of Computer Science
and Engineering
SMU

rewini@engr.smu.edu
www.engr.smu/~rewini



Talk Outline

- 1. Why Parallel Computing? Major Highlights**
- 2. Ups and Downs of Parallel Computing**
- 3. Recent Trends and Challenges for the future**



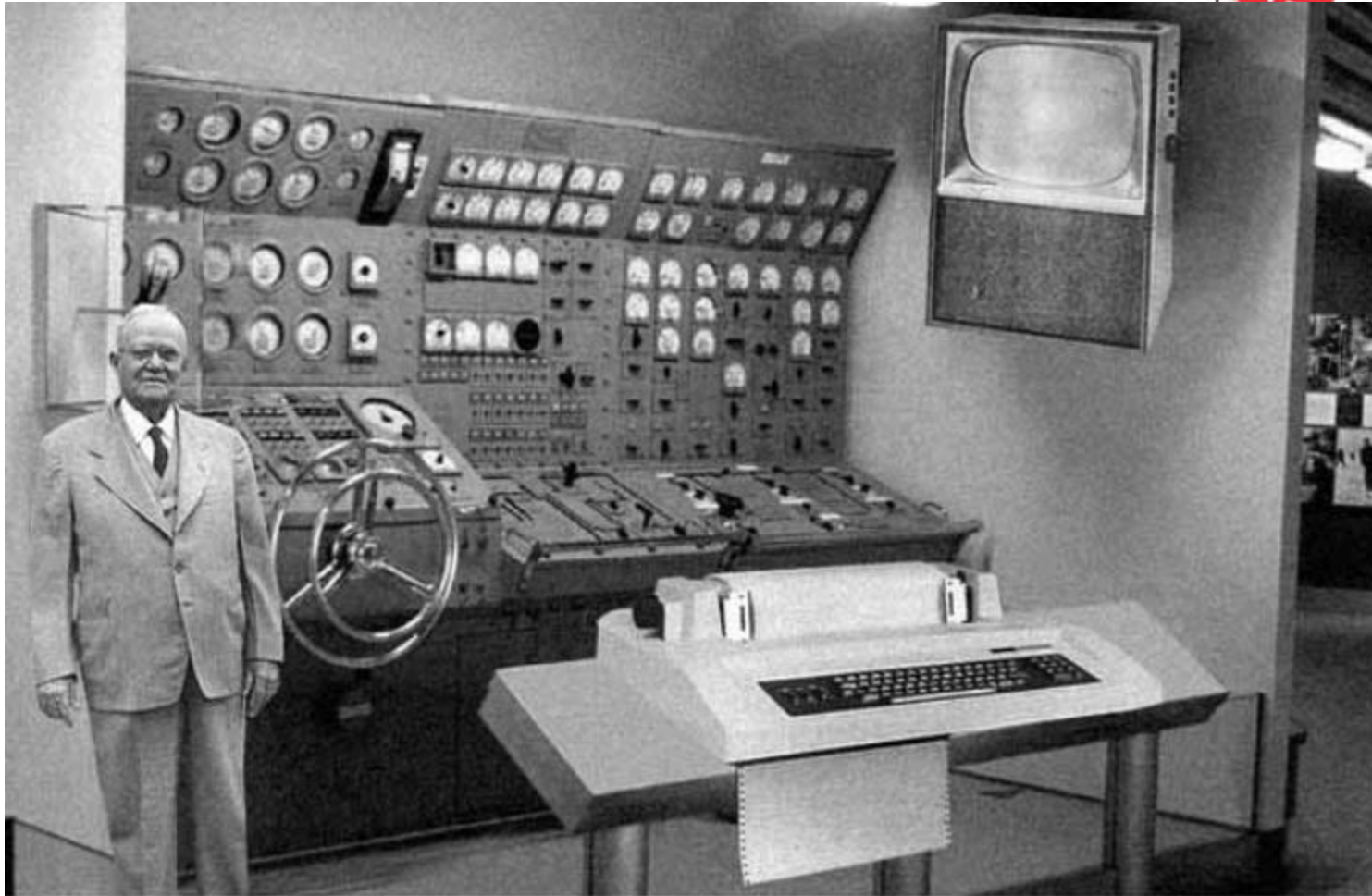
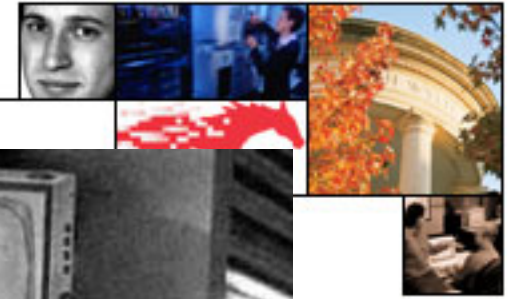
Let's Start with a Couple of Interesting Pictures!!



Picture 1 --- Home Computer as imagined 50 years ago.

In 1954, scientists from the RAND Corporation have created this model (shown in the next slide) to illustrate how a home computer could look like in the year 2004. However, the needed technology will not be economically feasible for the average home. Also the scientists readily admit that the computer will require not yet invented technology to actually work, but 50 years from now scientific progress is expected to solve these problems. With the teletype interface and the Fortran language, the computer will be easy to use.

From 1954 popular mechanics magazine

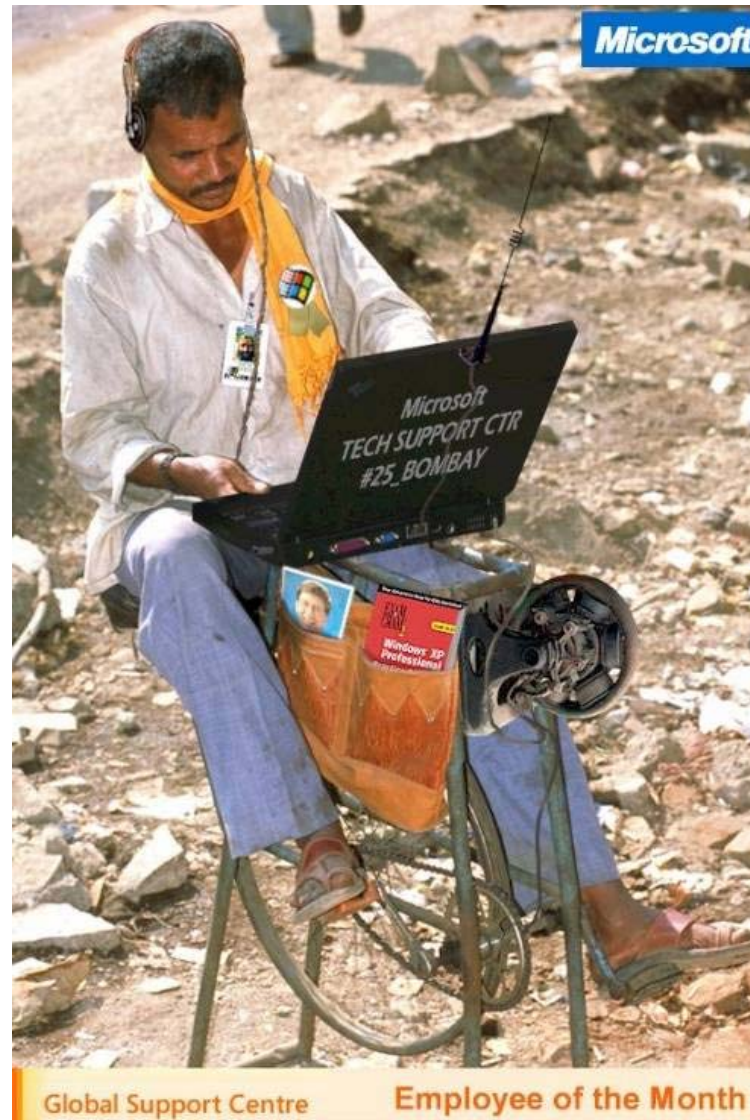


Scientists from the RAND Corporation have created this model to illustrate how a "home computer" could look like in the year 2004. However the needed technology will not be economically feasible for the average home. Also the scientists readily admit that the computer will require not yet invented technology to actually work, but 50 years from now scientific progress is expected to solve these problems. With teletype interface and the Fortran language, the computer will be easy to use.



Picture 2 --- Mobile Computer in the Microsoft Era

Tech Support CTR in the middle of nowhere!





Picture 3 --- Future Personal Computer

Wonder Computer Technology



AICCSA-06

Hesham El-Rewini, March 9, 2006

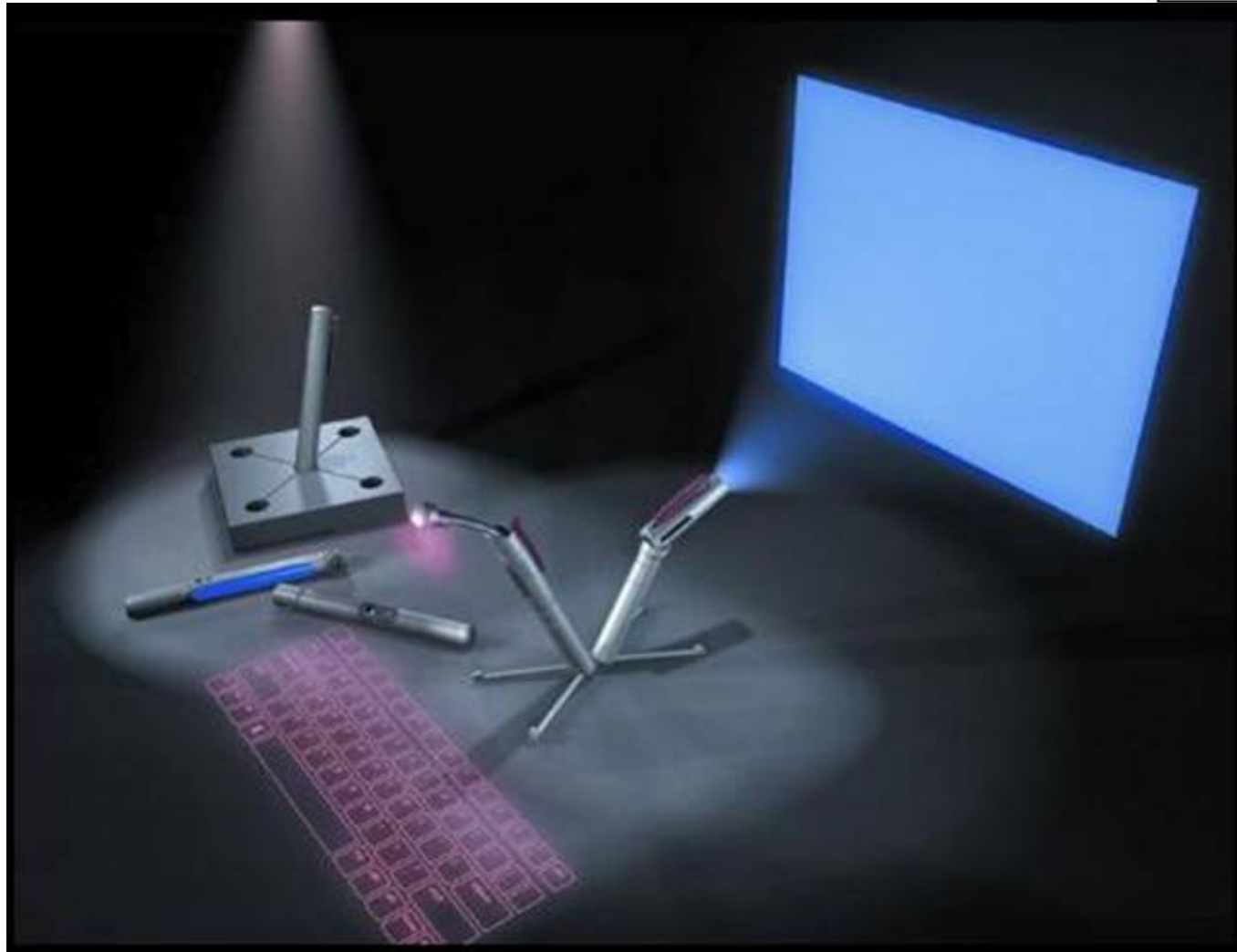
Computer Science and Engineering



AICCSA-06

Hesham El-Rewini, March 9, 2006

Computer Science and Engineering



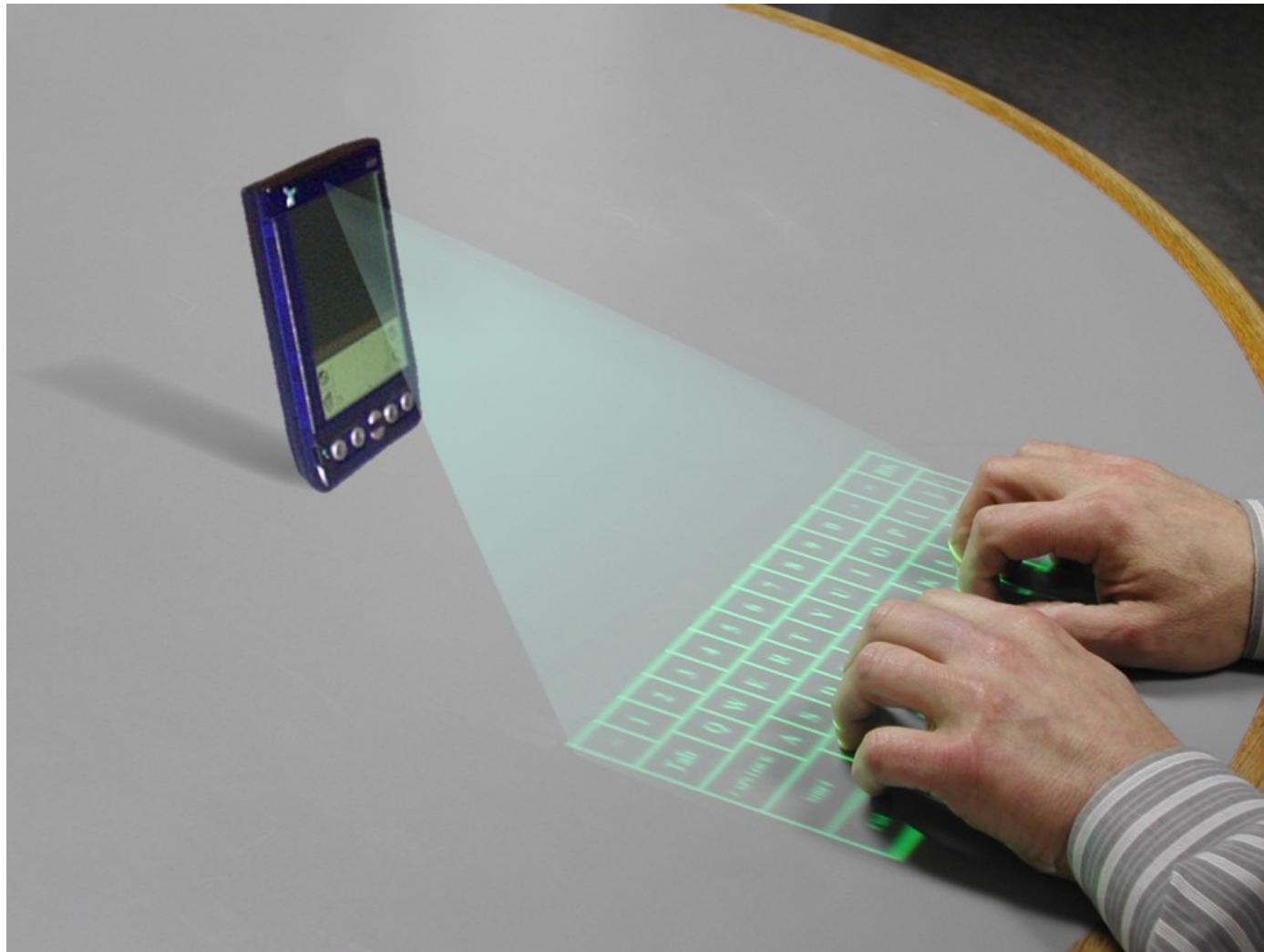
AICCSA-06

Hesham El-Rewini, March 9, 2006

Computer Science and Engineering

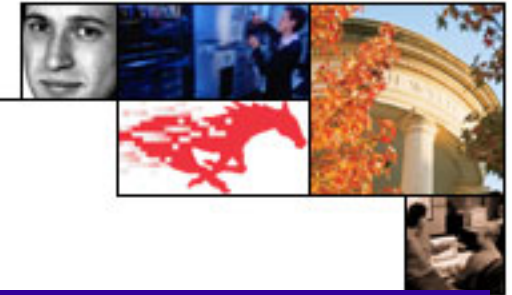








1- Why Parallel Computing? Major Highlights



Four Eras of Computing

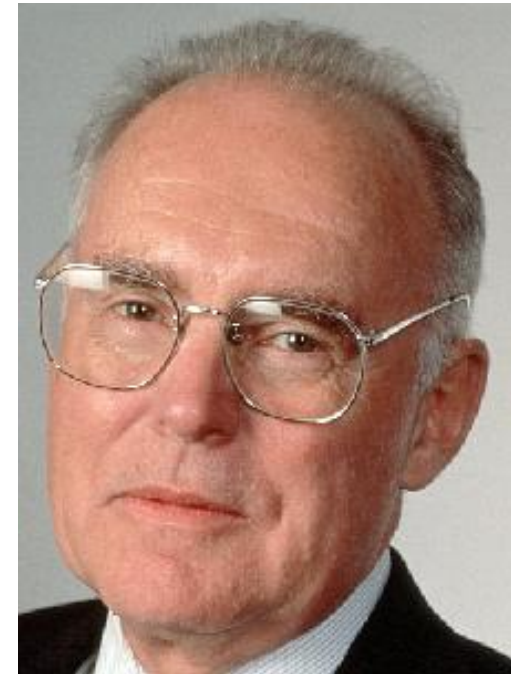
Feature	Batch	Time Sharing	Desktop	Shared Network
Time Period	1960s	1970s	1980s	1990s/2000s
Location	Computer room	Terminal room	Desktop	Mobile
Users	Experts	Specialists	Individuals	Groups
Data	Alphanumeric	Text, numbers	Font, Graphs	Multimedia
Objective	Calculate	Access	Present	Communicate
Interface	Punched cards	Keyboard, CRT	See and point	Ask and Tell
Operation	Process	Edit	Layout	Orchestrate
Connectivity	None	Peripheral cable	LAN	Internet/Wireless
Owners	Corporate computer centers	Divisional IS shops	Departments/individuals	Everyone



Moore's Law

1965 prediction by Intel founder Gordon Moore:

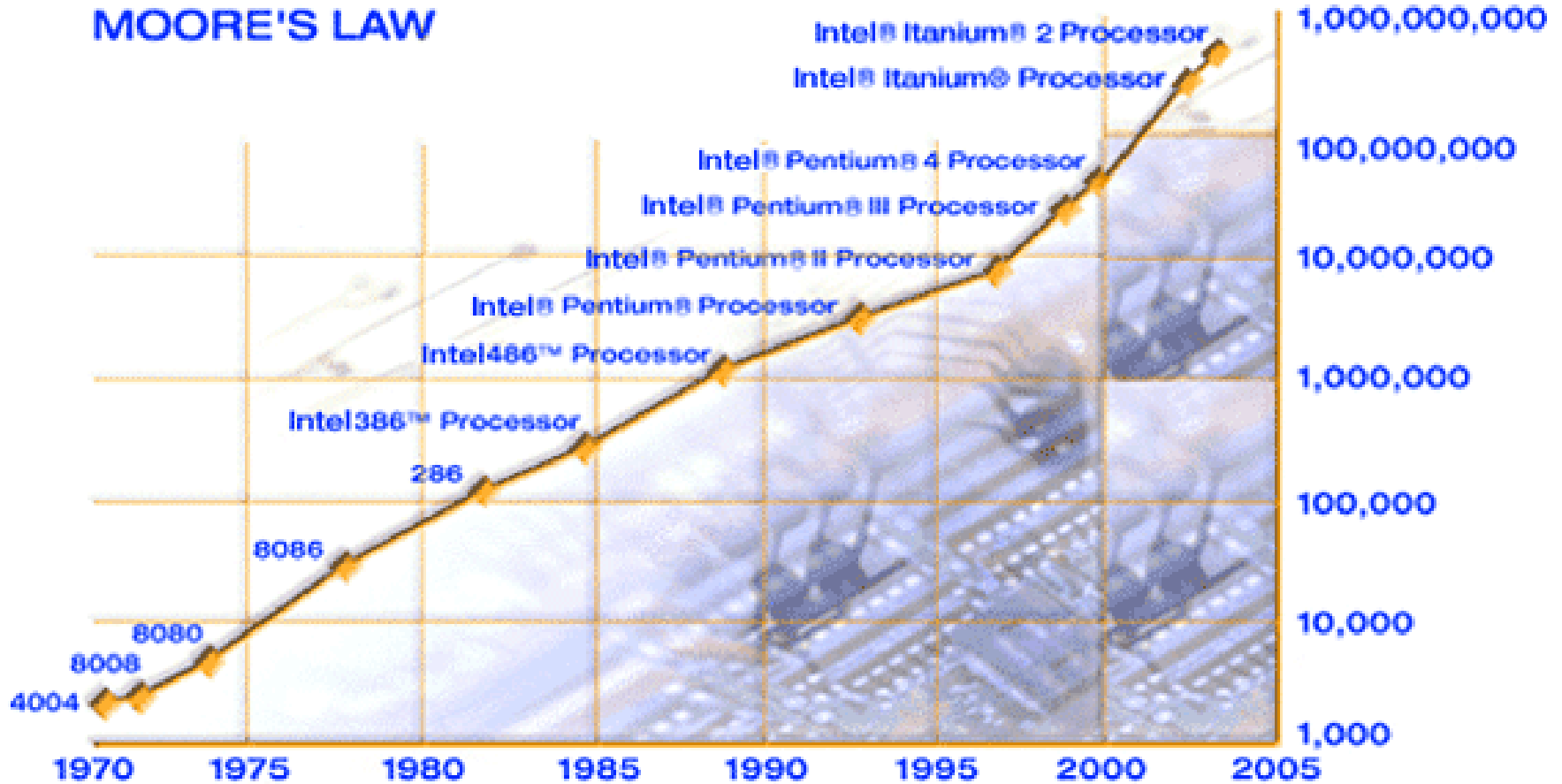
The number of transistors that can be built on the same size piece of silicon will double every 18 months

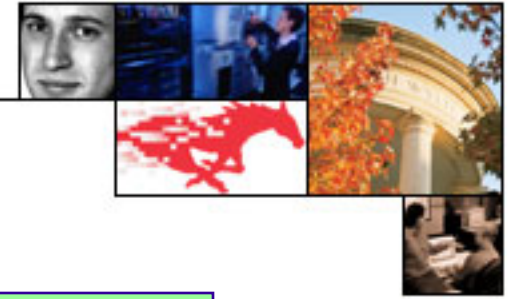




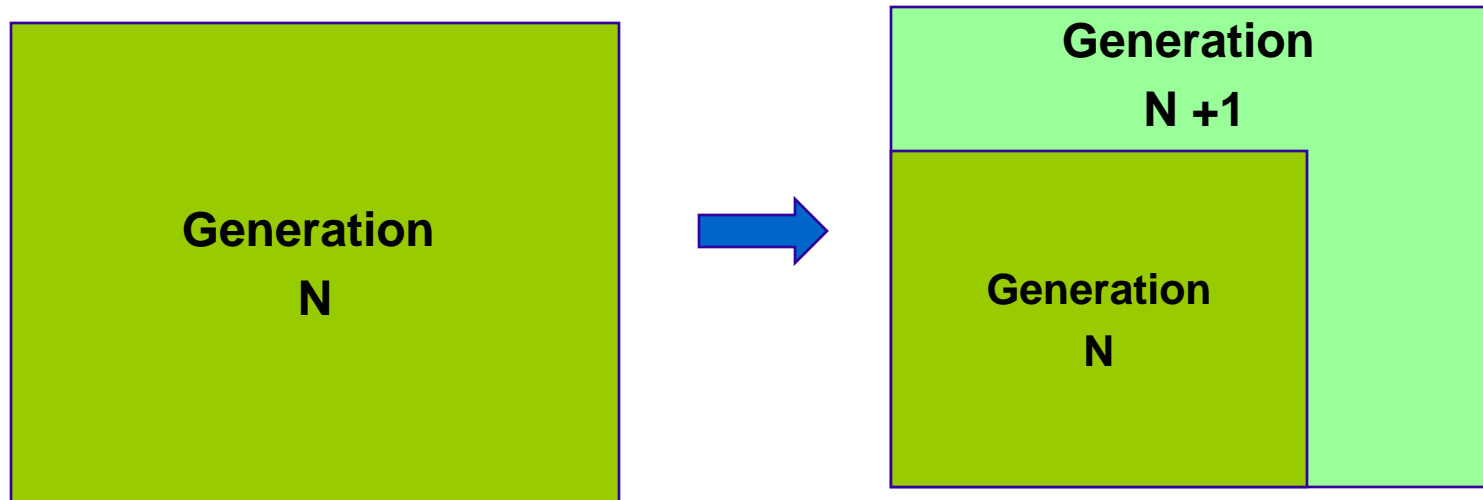
transistors

MOORE'S LAW





Processor Evolution



- Gate delay reduces by $1/\sqrt{2}$ (frequency up by $\sqrt{2}$)
- Number of transistors in a constant area goes up by 2
- Additional transistors enable an additional $\sqrt{2}$ increase in performance
- Result: 2x performance at roughly equal cost



Can this Growth be sustained forever?

- Speed of Light Argument
(Most people)
- The Vanishing Electrons Argument
(Joel Birnbaum, HP Labs, ACM 97)
- The FM Radio Analogy
(Erik P. DeBenedictis, Sandia National Labs, 2005)



Speed of Light Limit

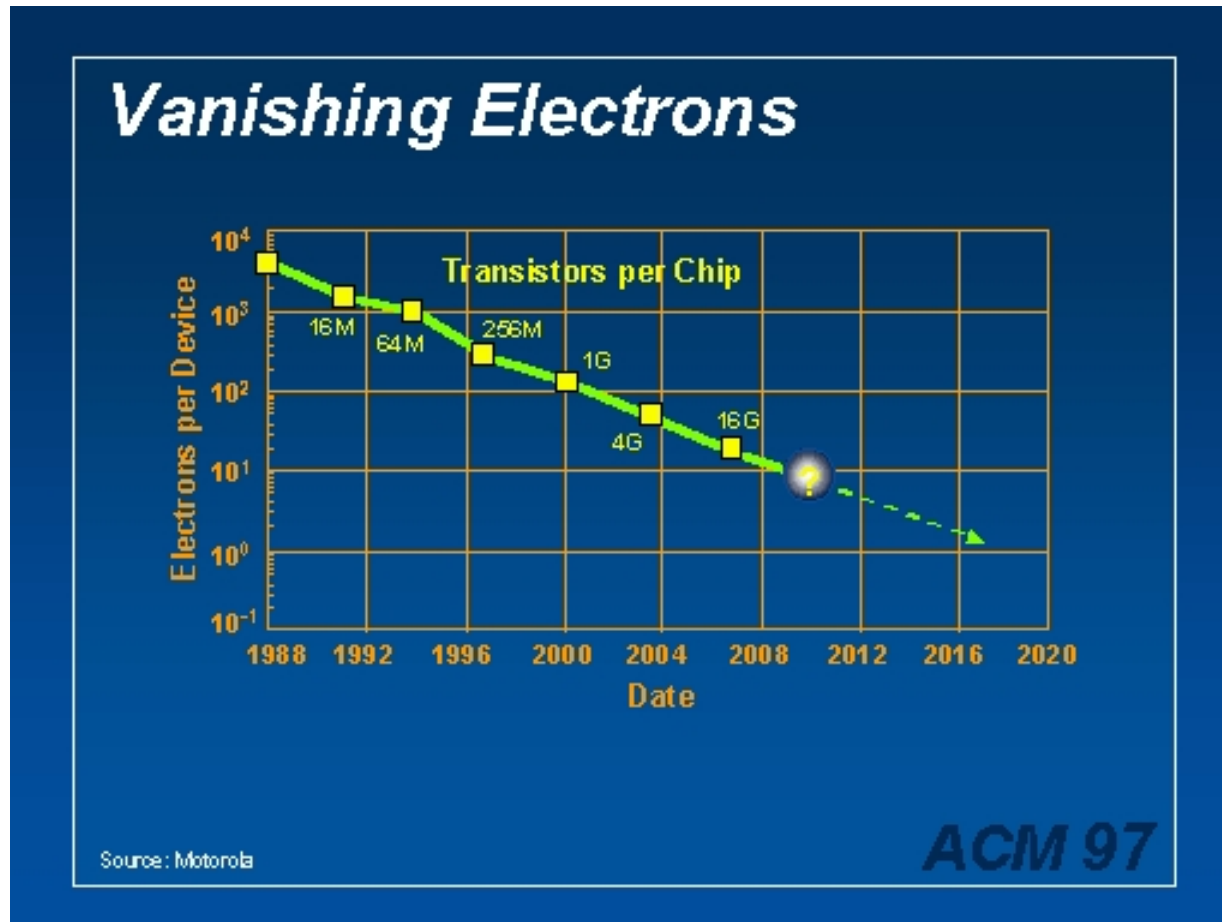
Light travels 1 cm in $\frac{1}{30}$ nanosecond

What is the speed if a signal must travel 1 cm during the execution of an instruction?





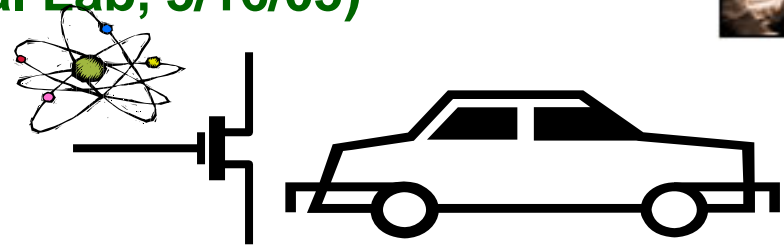
Joel Birnbaum, HP Labs, 1997



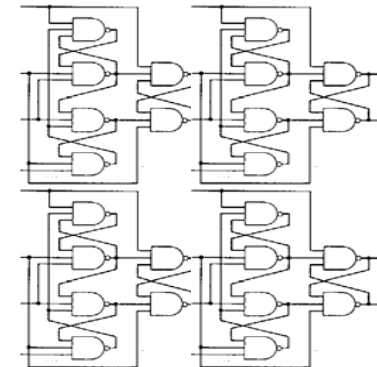
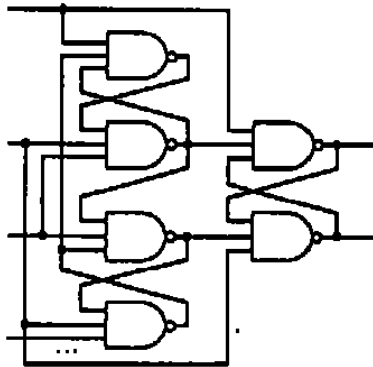


FM Radio and End of Moore's Law

(Erik P. DeBenedictis, Sandia National Lab, 5/16/05)



Driving away from FM transmitter → less signal
Noise from electrons → no change



Increasing numbers of gates → less signal power
Noise from electrons → no change



Grand Challenges need Computing Power

Vision, Human Genome
Climate Model, Ocean Circulation
Fluid Turbulence, Viscous Flow
Quantum Chromodynamics
Superconductor Model
Vehicle Dynamics
Weather Prediction
Chemical Dynamics
3D plasma
Oil Reservoir Model

Parallelism is an obvious answer!!



Parallelism

Multiple processors cooperate to jointly execute a single computational task in order to speed up its execution.

- Solve Problems Faster (Speedup)
- Solve More Problems (Higher Throughput)
- Solve Larger Problems (Computational Power)
- Enhance Solutions' Quality (Quality Up)



Types of Parallelism

	Single Data Stream	Multiple Data Stream
Single Instruction Stream	SISD Uniprocessors	SIMD Array Processors Vector
Multiple Instruction Stream	MISD	MIMD Multiprocessors Multicomputers

Flynn's Taxonomy



MIMD Categories

	Shared Variables	Message Passing
Global Memory	GMSV Shared Memory Multiprocessors	GMMP
Distributed Memory	DMSV Distributed Shared Memory	DMMP Distributed Memory Multicomputers

Johnson's Expansion



SMU School of Engineering



Main Parallel Architecture

AICCSA-06

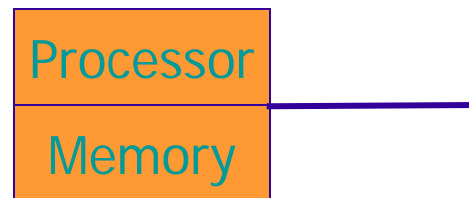
Hesham El-Rewini, March 9, 2006

Computer Science and Engineering

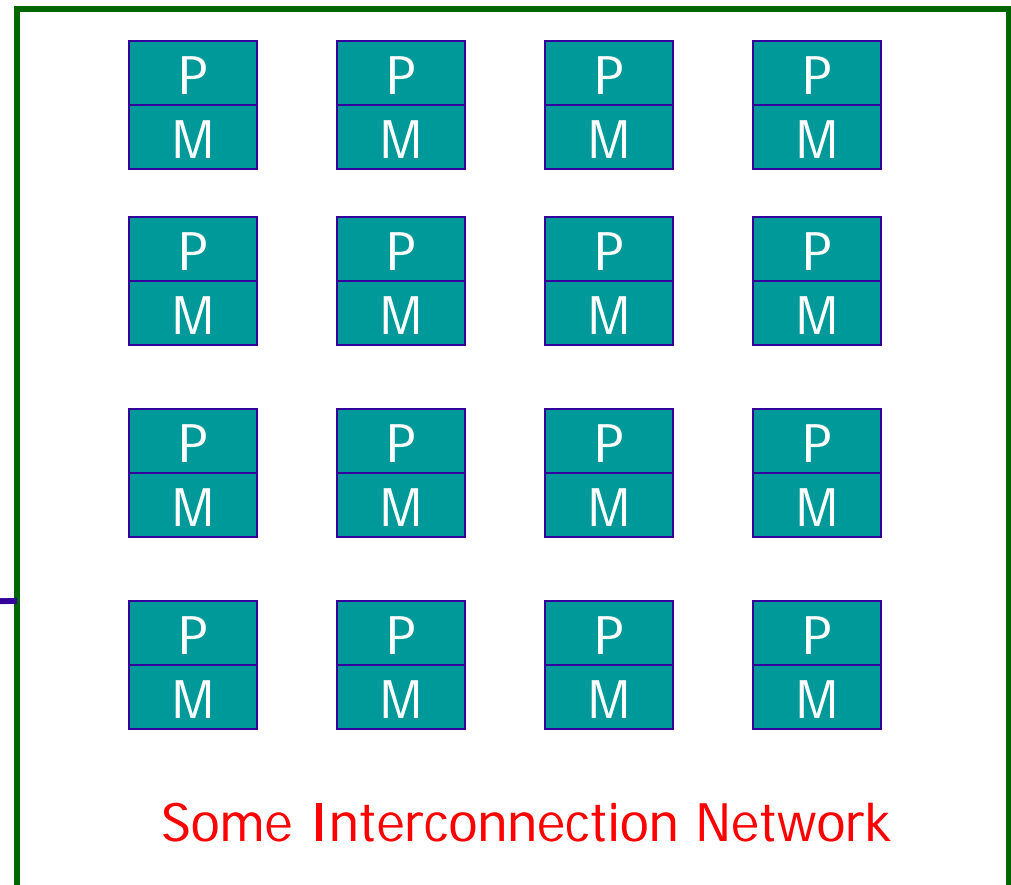


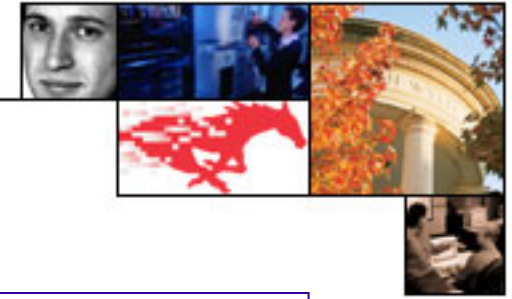
SIMD Systems

One control unit
Lockstep
All Ps do the same or nothing

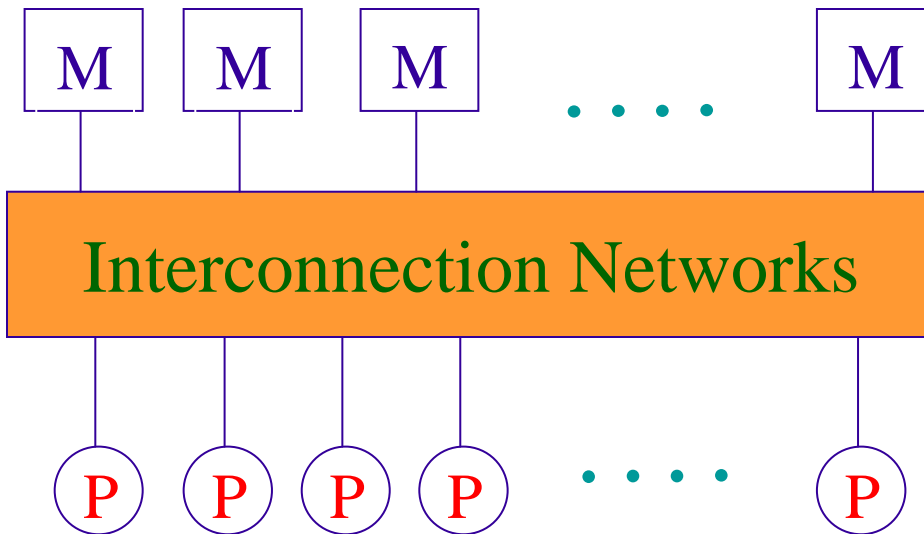


von Neumann Computer

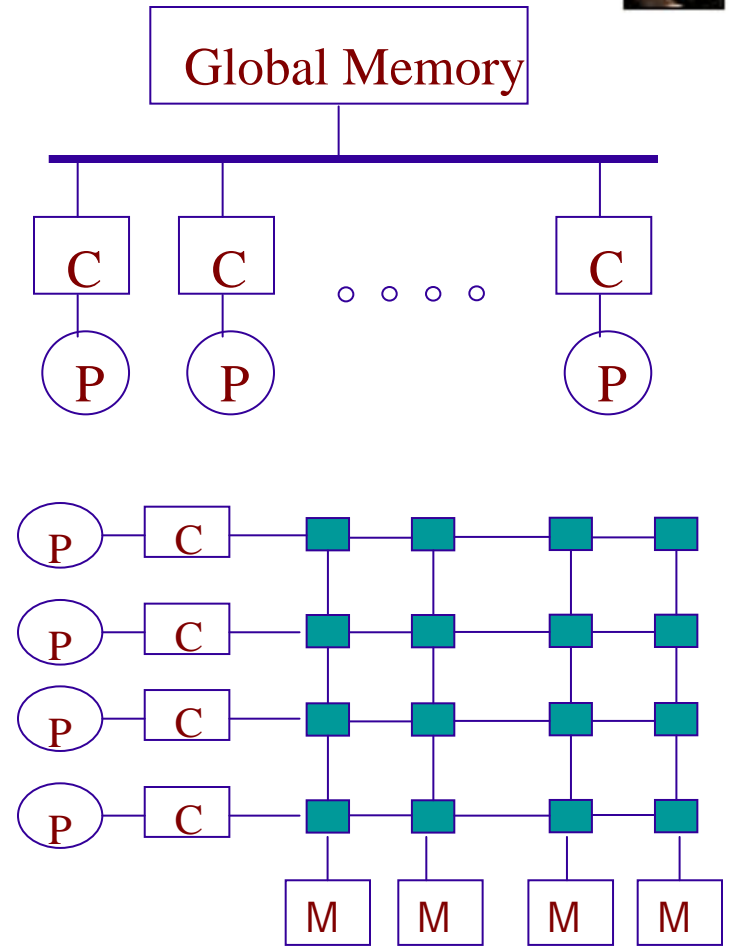




MIMD Shared Memory Systems

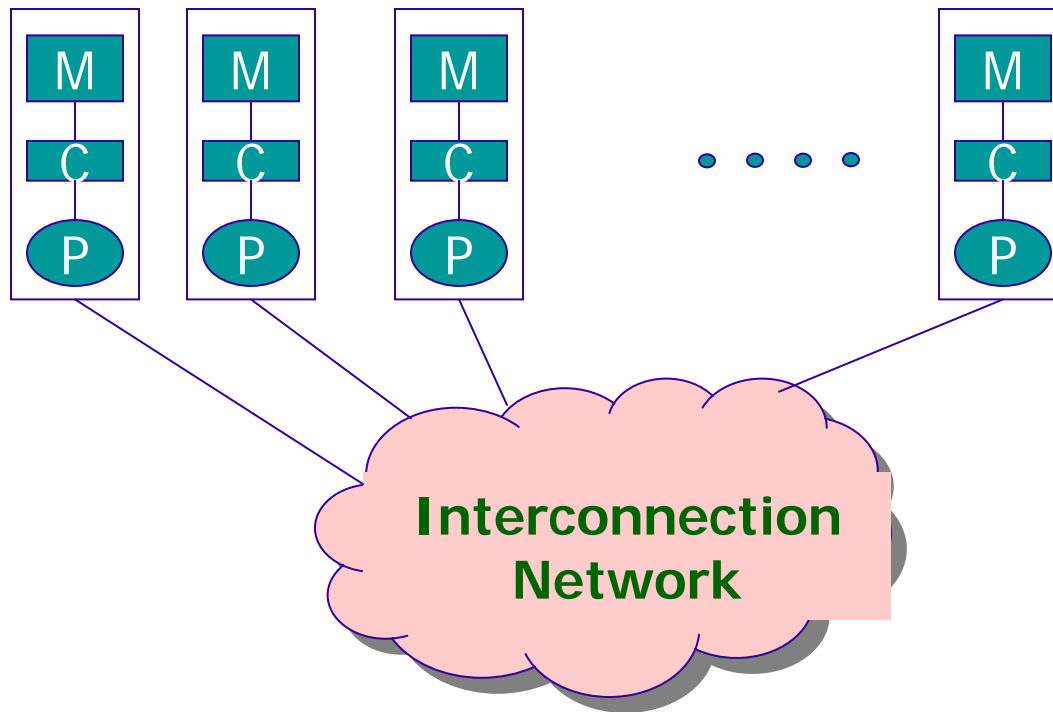


One global memory
 Cache Coherence
 All Ps have equal access to memory





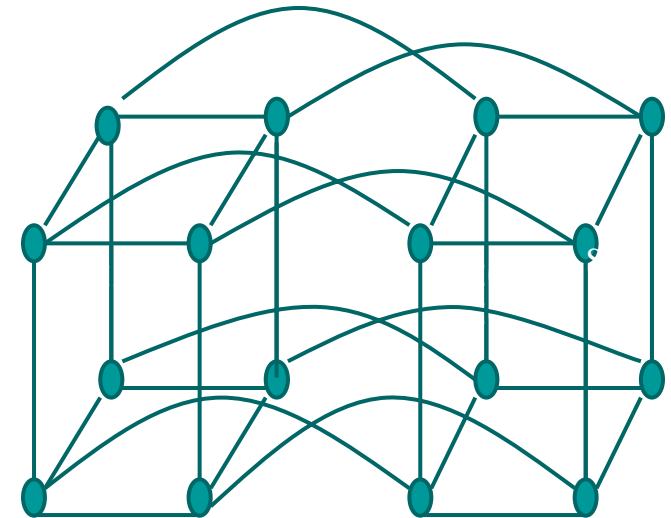
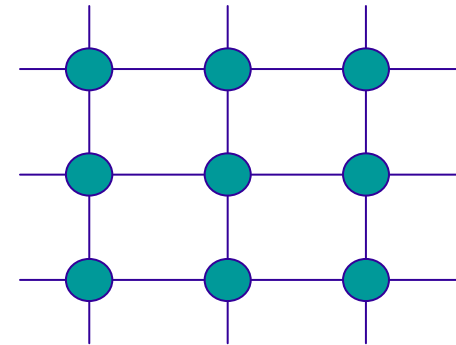
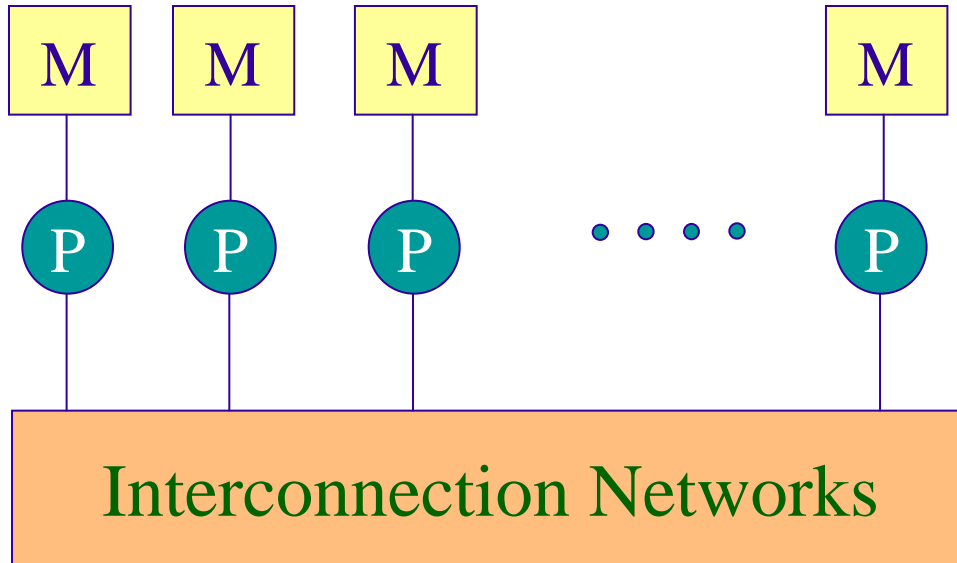
Cache Coherent NUMA



Each P has part of the shared memory
Non uniform memory access



MIMD Distributed Memory Systems



No shared memory
 Message Passing
 Topology



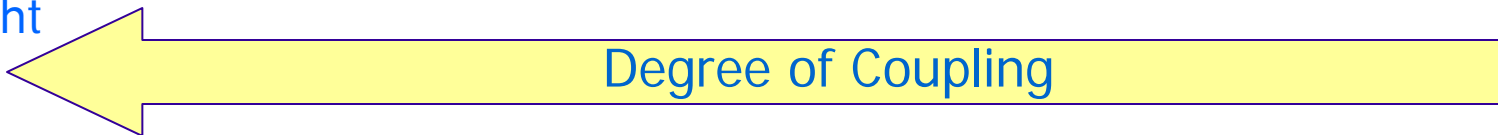


Parallel and Distributed Architecture (Leopold, 2001)

SIMD SMP CC-NUMA DMPC Cluster Grid



tight



loose

fine



coarse

fast



slow



SMU School of Engineering



Parallel Programming

AICCSA-06

Hesham El-Rewini, March 9, 2006

Computer Science and Engineering



Parallel Programming is Difficult

- Multiple threads of control
- Partitioning for concurrent execution
- Task Scheduling/resource allocation
- Communication and Sharing
- Synchronization
- Debugging
- Different Architecture



Explicit versus Implicit Parallel Programming

Applications

Parallel Tools

Smart Compiler

Architecture



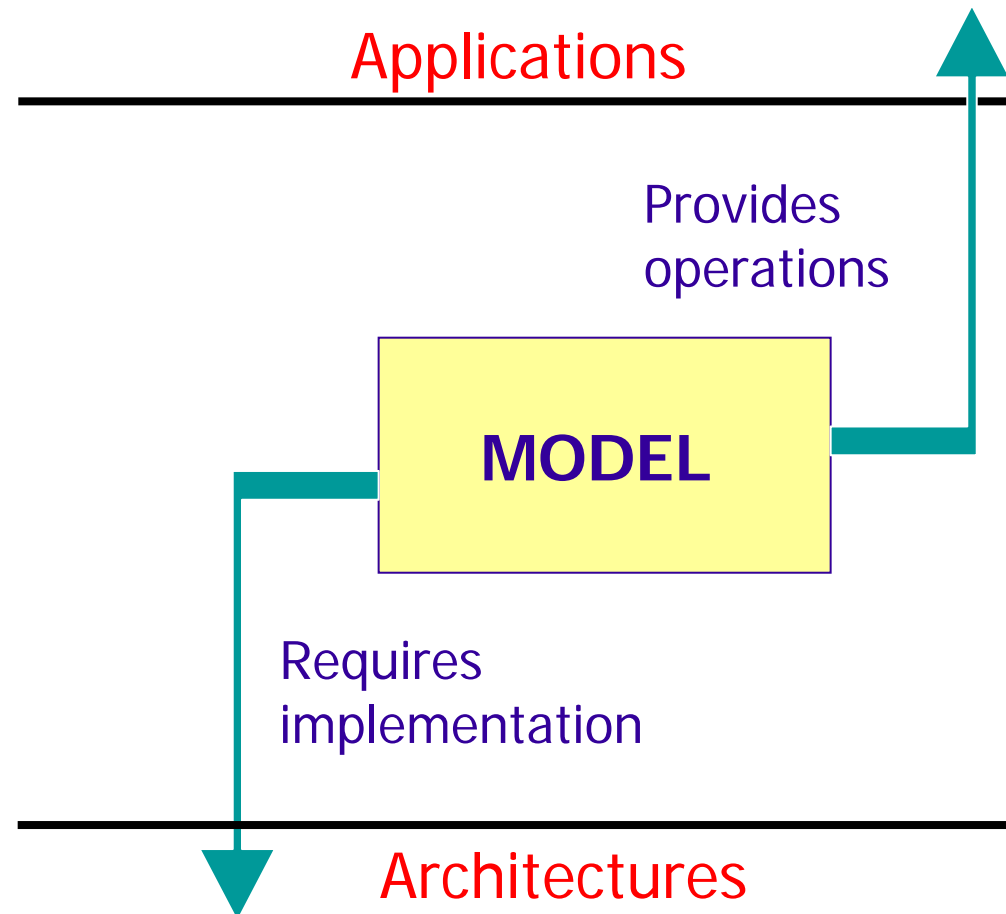
Programmer's Responsibilities

Class	Programmer Responsibility
1	Implicit Parallelism (nothing much)
2	Identification of Parallelism Potential
3	Decomposition (potential), placement
4	Decomposition, high level coordination
5	Decomposition, high level coord, placement
6	Decomposition, low level coordination
7	Decomposition, low level coord, placement



Models

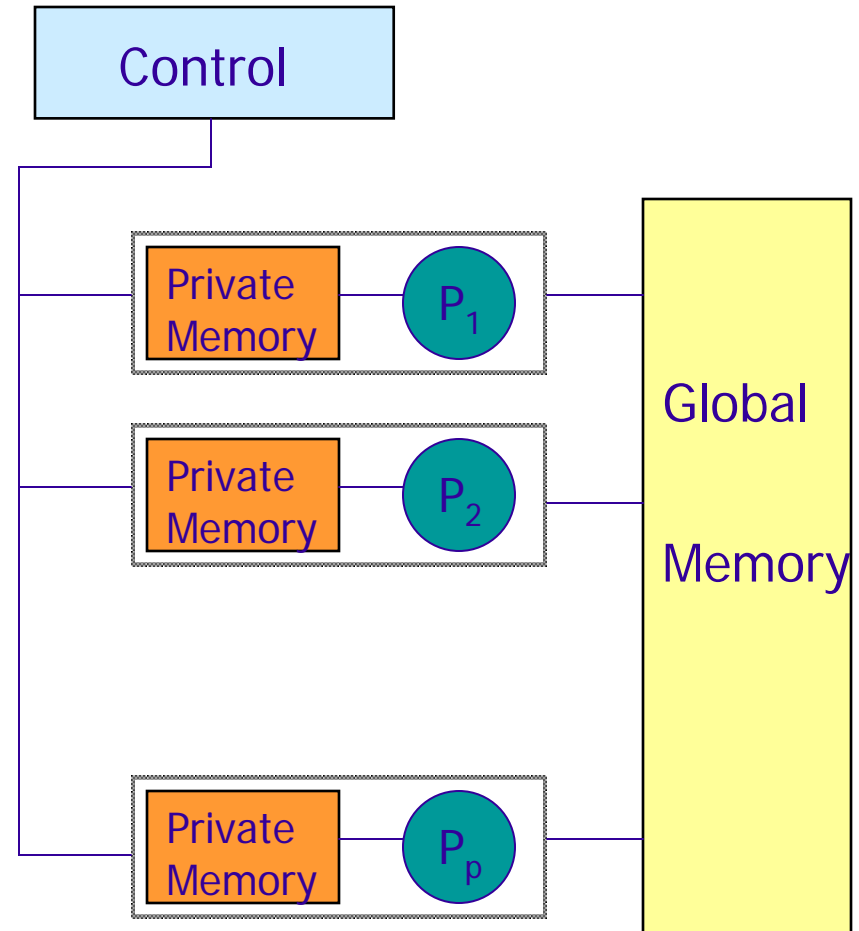
A model is an interface separating high level properties from low level ones

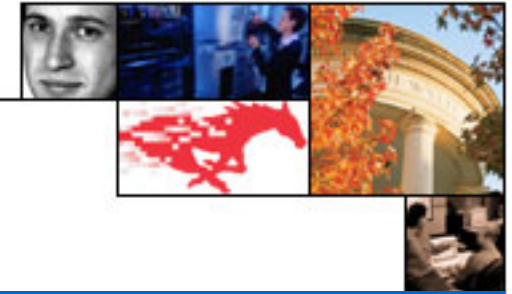




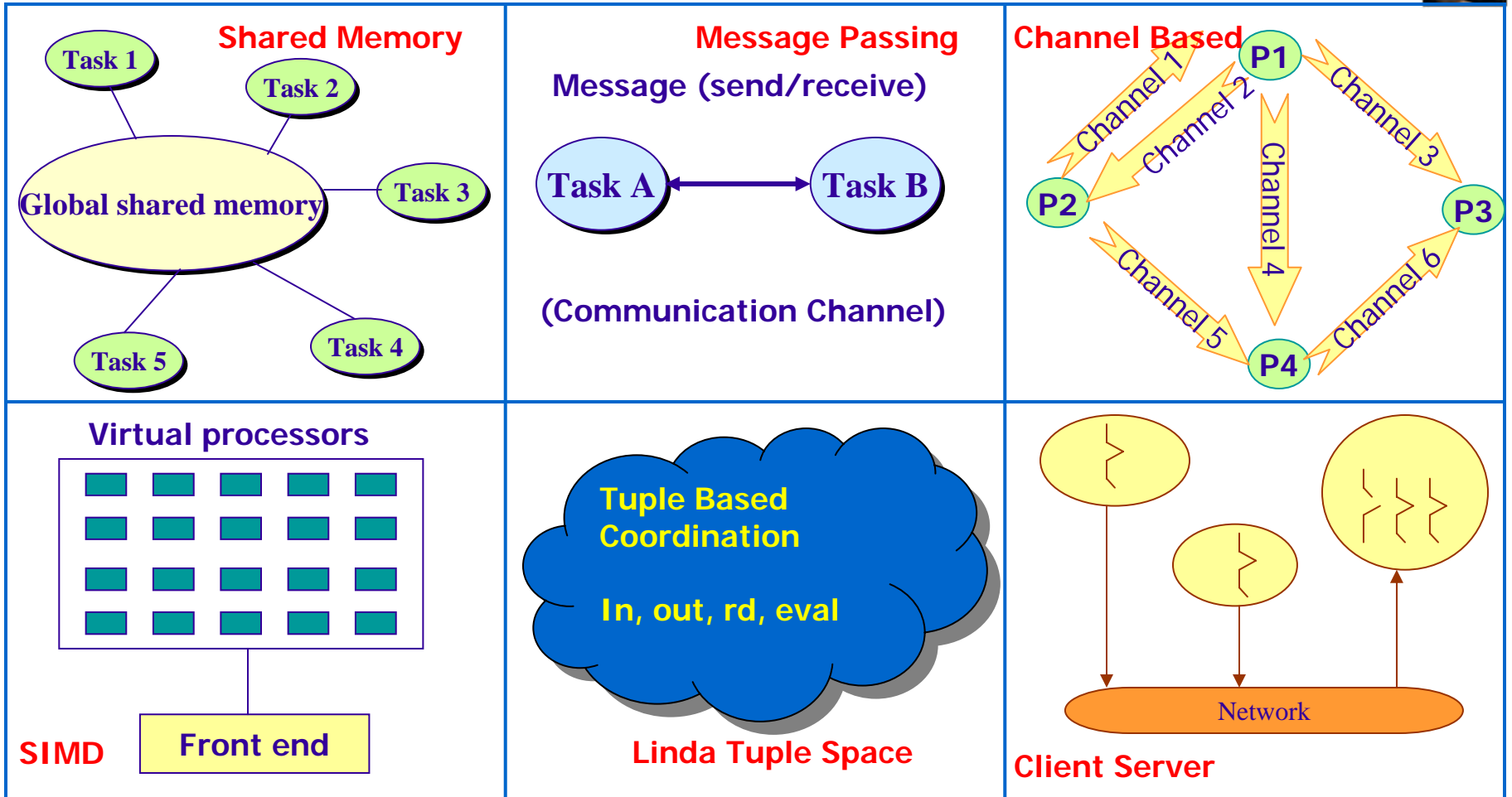
PRAM Theoretical Model

- Synchronized Read Compute Write Cycle
- EREW
- ERCW
- CREW
- CRCW
- Complexity:
 $T(n), P(n), C(n)$





Programming Models



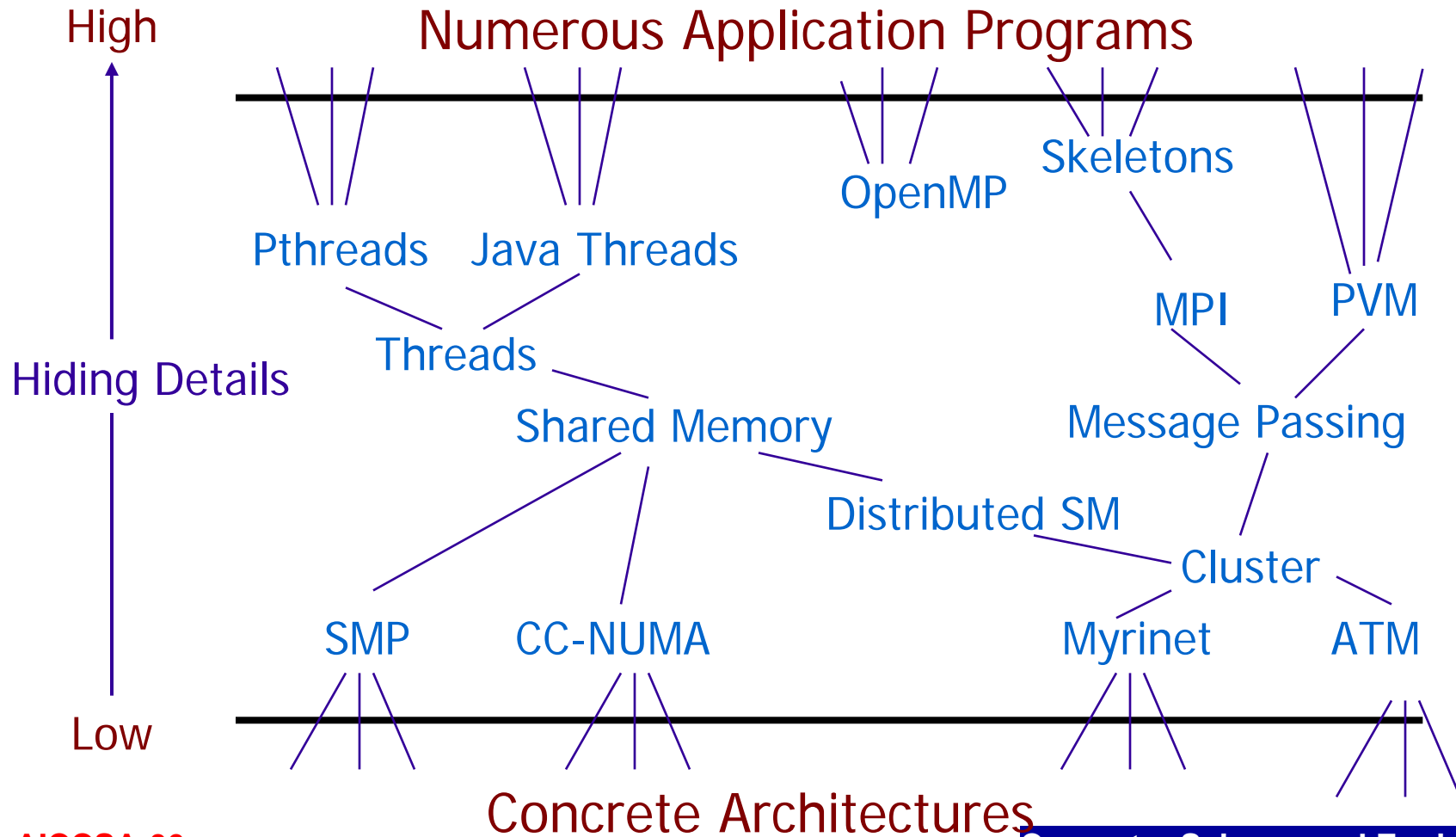


Programming Approaches So Far

- Automatic Parallelizing Compilers
- Compiler Directives
- Add on to sequential languages
- Standard Parallel Features in Existing Languages
- New Languages
- Programming Tools



Leopold's View of the Field





Two Important Laws Influenced Parallel Computing



Argument Against Massively Parallel Processing. Gene Amdahl, 1967.

For over a decade prophets have voiced the contention that the organization of a single computer has reached its limits and that truly significant advances can be made only by interconnection of multiplicity of computers in such a manner as to permit cooperative solution .. The nature of this overhead (in parallelism) appears to be sequential so that it is unlikely to be amenable to parallel processing techniques. Overhead alone would then place an upper limit on throughput of five to seven times the sequential processing rate, even if the housekeeping were done in a separate processor... At any point in time it is difficult to foresee how the previous bottlenecks in a sequential computer will be effectively overcome.



What does that mean?

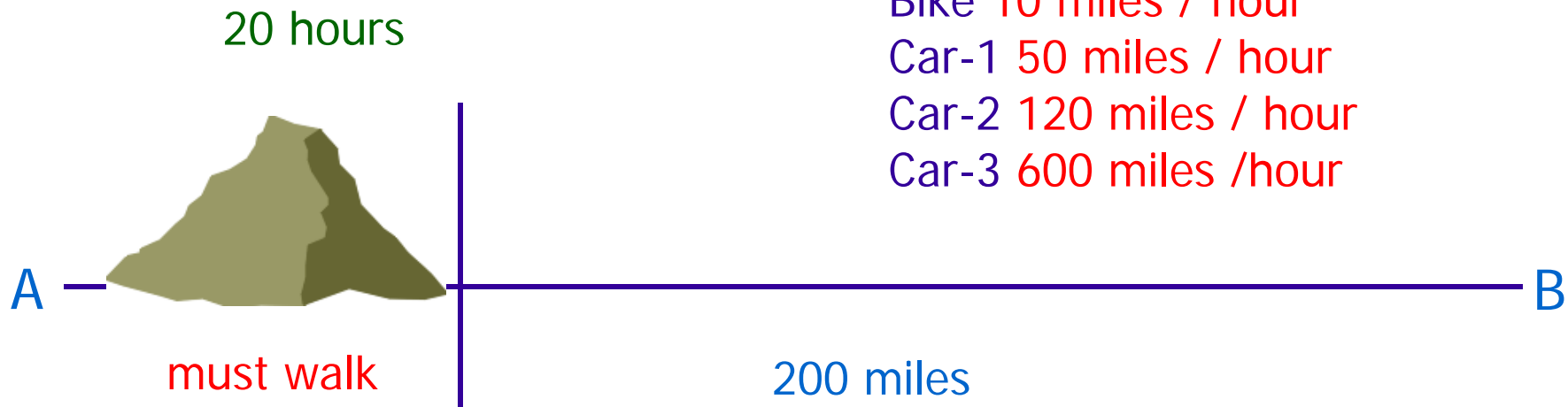
The performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode cannot be used.

Unparallelizable part of the code severely limits the speedup.



Trip Analogy

- Walk 4 miles / hour
- Bike 10 miles / hour
- Car-1 50 miles / hour
- Car-2 120 miles / hour
- Car-3 600 miles / hour





Speedup Analysis



rvrtech.com

(4 miles /hour) Time = 70 hours



(10 miles / hour) Time = 40 hours

S = 1.8



(50 miles / hour) Time = 24 hours

S = 2.9



(120 miles / hour) Time = 21.67 hours

S = 3.2



(600 miles /hour) Time = 20.33 hours

S = 3.4



Amdahl's Law

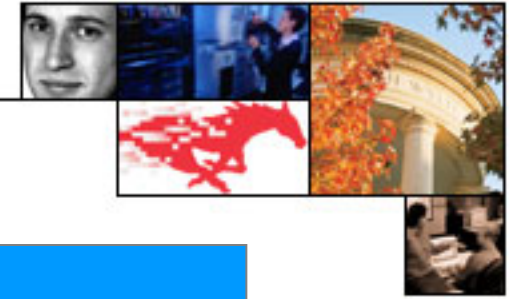
β : The fraction of the program that is naturally serial

$(1 - \beta)$: The fraction of the program that is naturally parallel

$$S = T(1)/T(N)$$

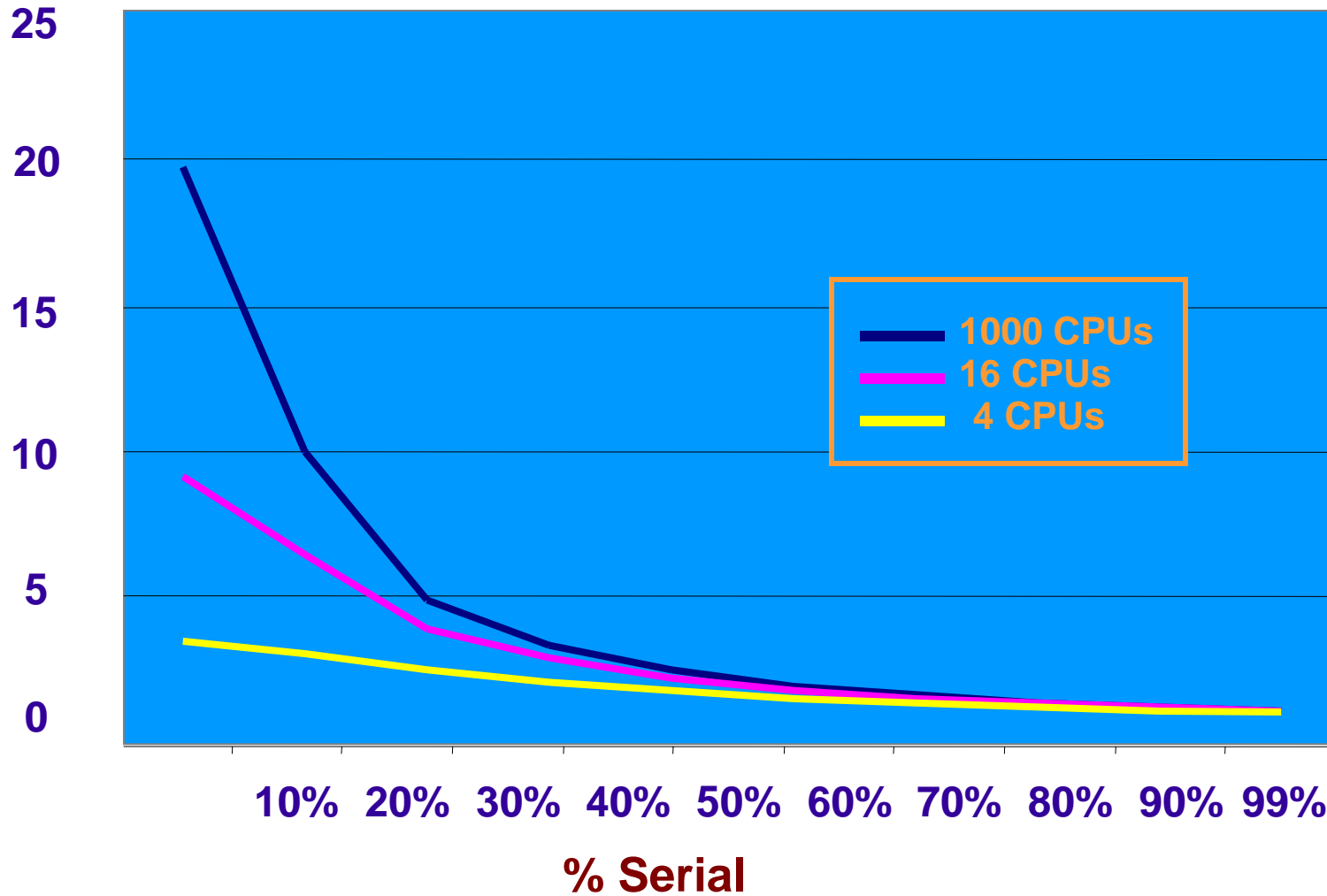
$$T(N) = T(1)\beta + \frac{T(1)(1 - \beta)}{N}$$

$$S = \frac{1}{\beta + \frac{(1 - \beta)}{N}} = \frac{N}{\beta N + (1 - \beta)}$$



Speedup

Amdahl's Law





Gustafson – Barsis Law (1988)

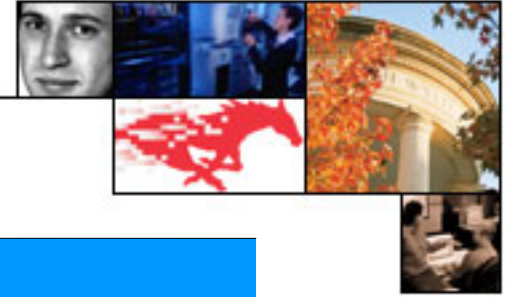
- Gordon Bell Prize
- Overcoming the conceptual barrier established by Amdahl's law
- Scale the problem to the size of the parallel system
- No fixed size problem

α : The fraction of the program that is naturally serial

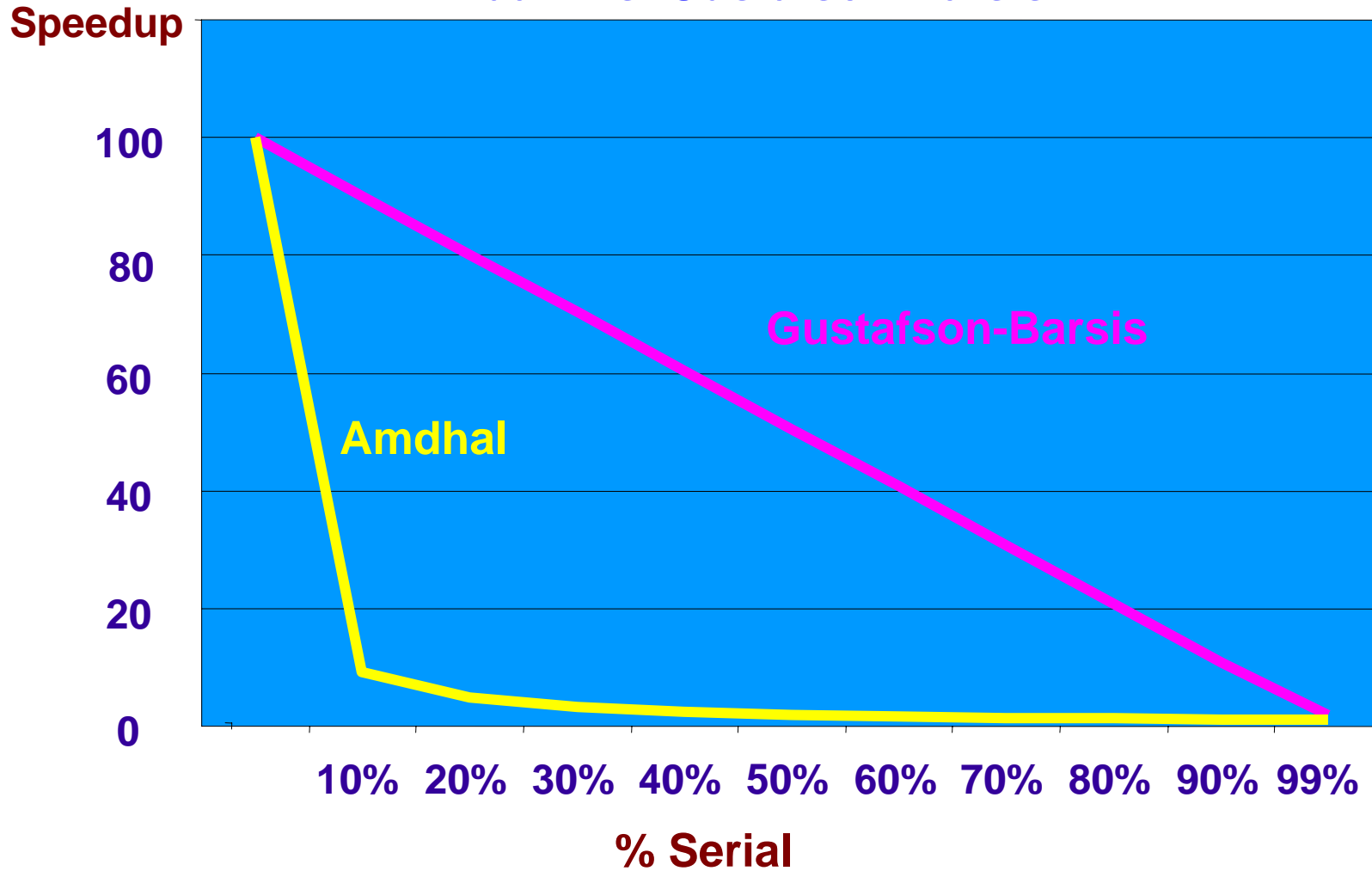
$$T(N) = 1$$

$$T(1) = \alpha + (1 - \alpha) N$$

$$S = N - (N - 1) \alpha$$



Amdahl vs. Gustafson-Barsis





Data Parallelism – Scale up

- Parallelism is in the data, not the control portion of the application
- Problem size scales up to the size of the system
- Data Parallelism is to the 1990's what vector parallelism was to the 1970's
- Supercomputer → data parallel



2-The Ups and Downs of Parallel Computing



Commercial Interest in 1980s

- Technology was driven by government funding
- Custom designed components
- A large number of competitors entered the market
- R&D time (2-3 years)
- Expensive systems
- Quickly becoming outdated

Many of the companies disappeared in the mid-1990s



1980s Parallel Systems (some)

- Alliant FX/8
- Encore
- Sequent – Balance and Symmetry
- Cray
- Thinking Machines – CM-1, CM-2, (later CM-5)
- MasPar – Mp-1, MP-2
- nCube (hypercube)
- Intel iPSC (hypercube)
- Wavetracer (3-D Mesh)
- Transputers
- Cogent



1980s Parallel Programming Languages (Some)

*LISP

C*

data parallel C,
parallexis

CM FORTRAN

LGDF

unity

Split-C

HPF

OCCAM

Strand

C-Linda, etc.

Parlog

Several flavors of C

Mostly Forgotten



Message Passing Interface (MPI)

- By 1990, there were several different ways for parallel programming.
- There was not a clear choice of which approach to be use.
- Early vendor systems were not portable.
- Early portable systems (PVM, p4) were mainly research efforts
- The MPI Forum organized in 1992 with broad participation by vendors, library writers, and end users
- MPI Standard (1.0) released June, 1994; many implementation efforts
- MPI-2 Standard (1.2 and 2.0) released July, 1997
- 1998, MPI became the choice for many people (standard)
- Standard Library of Routines for writing portable and efficient parallel programs
- It is not a language , it is a specification of a library of routines that can be called form a program.



Computer Clusters of the 1990s

- Advances in commodity processors and network technology
- Network of PCs and workstations connected via LAN or WAN forms a Parallel System
- Compete favorably (cost/performance)

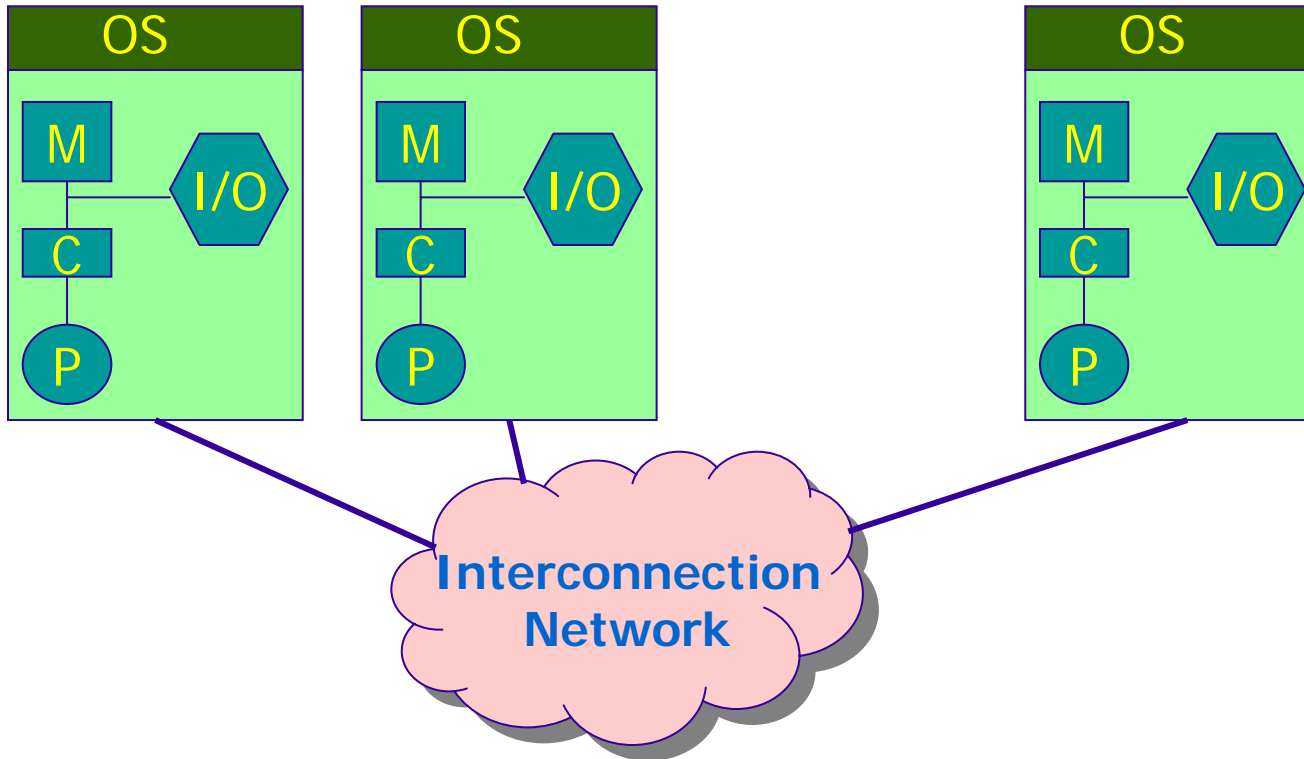




Cluster Architecture

Programming Environment

Middleware

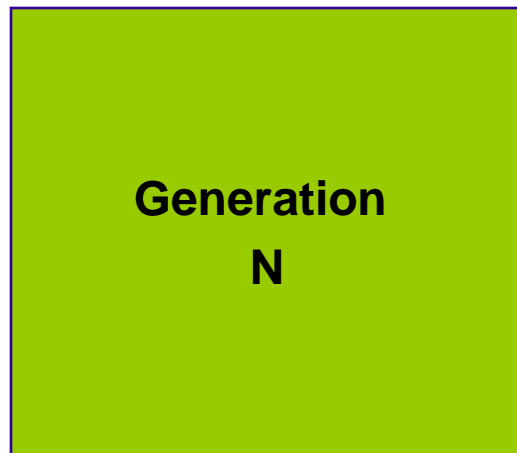


Home cluster

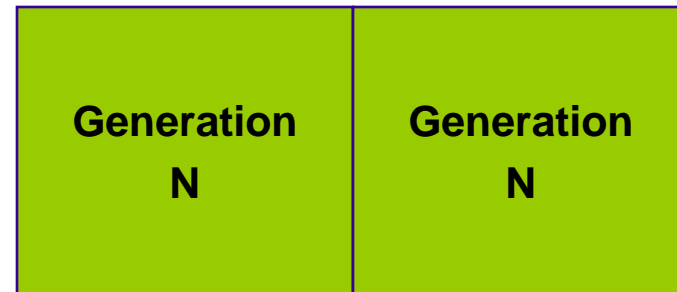


Multi-core of the 2000s

Technology Generation N



Technology Generation N+1



- Gate delay does not reduce much
- The frequency and performance of each core is the same or a little less than previous generation



Four Eras of Parallelism


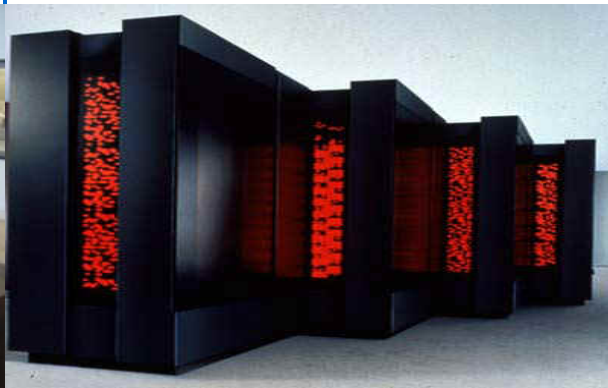

	1970	1980	1990	2000	Beyond 2000
Parallelism Level	Processor level	Machine level (In box)	LAN level	WAN level	Chip level
Architecture	Vector	SMP / MPP	Cluster	Grid	Multi-Core
Threads	One	Multiple	Multiple	Multiple	Multiple
Interconnection Network	None	Bus, switch, mesh, hypercube	Ethernet, Switch	Internet	On Chip
System	Custom	Custom	Commodity	Combination	SoC
Programming	Vector Fortran	C*, C-Linda, Occam, many others	PVM, MPI, HPF, ...	MPI, OpenMP, ...	?



The Fastest Machines in the last 20 years



1985-1995

Cray-2 (1985-1990)	CM-5/1024 MPP (1992-1993)	Numerical Wind Tunnel (1995)
		
1.2 Gflops	59.7 Gflops	170.4 Gflops
4 vector processors	1024 SPARC processors	140 vector processors (Fujitsu 105 MHz)
Shared memory	Fat tree Network	Crossbar switch
Cray	TMC	Fujitsu



Earth Simulator MPP (2002-2004)



- ✓ 35.86 Tflops, 5120 processors
- ✓ 640 nodes with 8 vector processors each
- ✓ Single stage crossbar switches (2400 km of cables)
- ✓ NEC



IBM BlueGene/L MPP (2005)



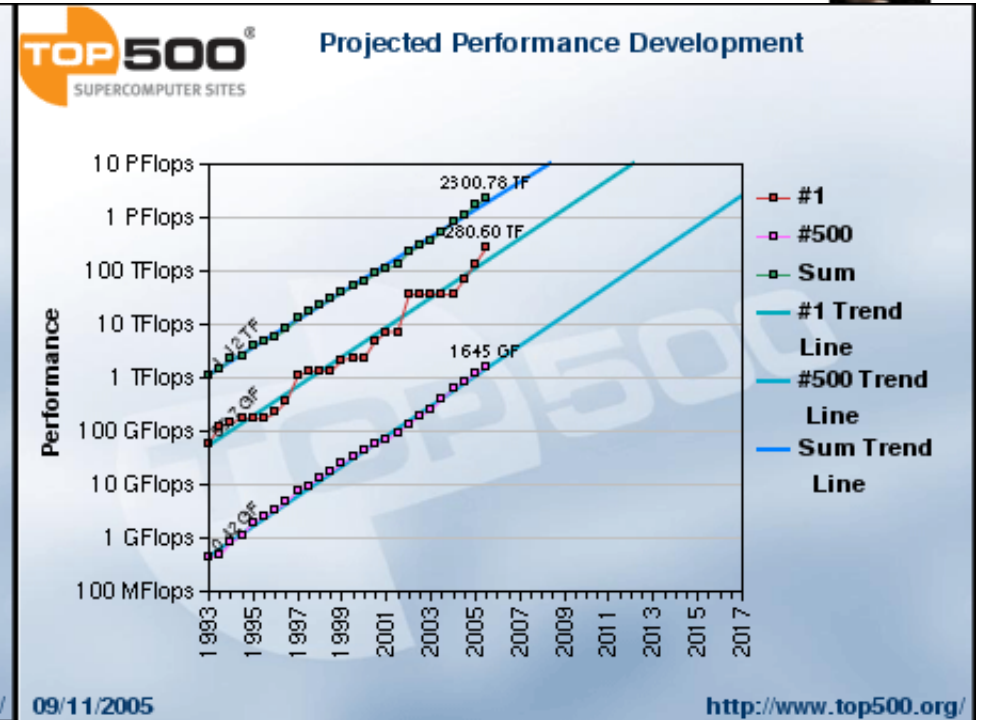
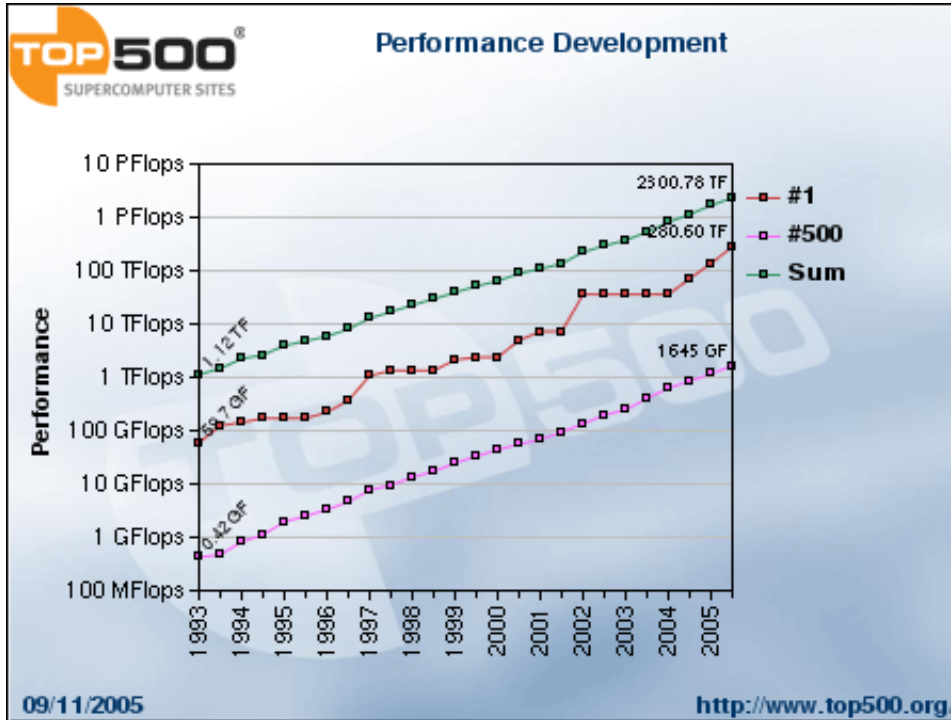
- ✓ 280 Tflops, 131,072 processors
- ✓ Each node has two 700 Mhz PowerPC 440
- ✓ 3D toroidal network for peer-to peer communication
- ✓ IBM



3- Recent Trends and Challenges for the Future



Top 500 Performance Development



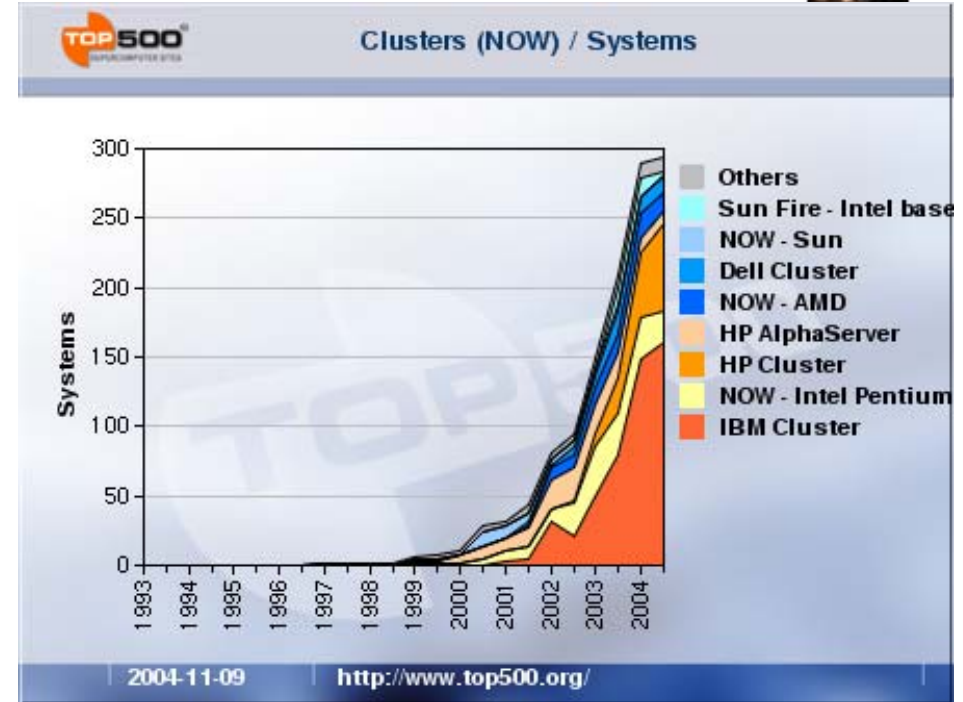
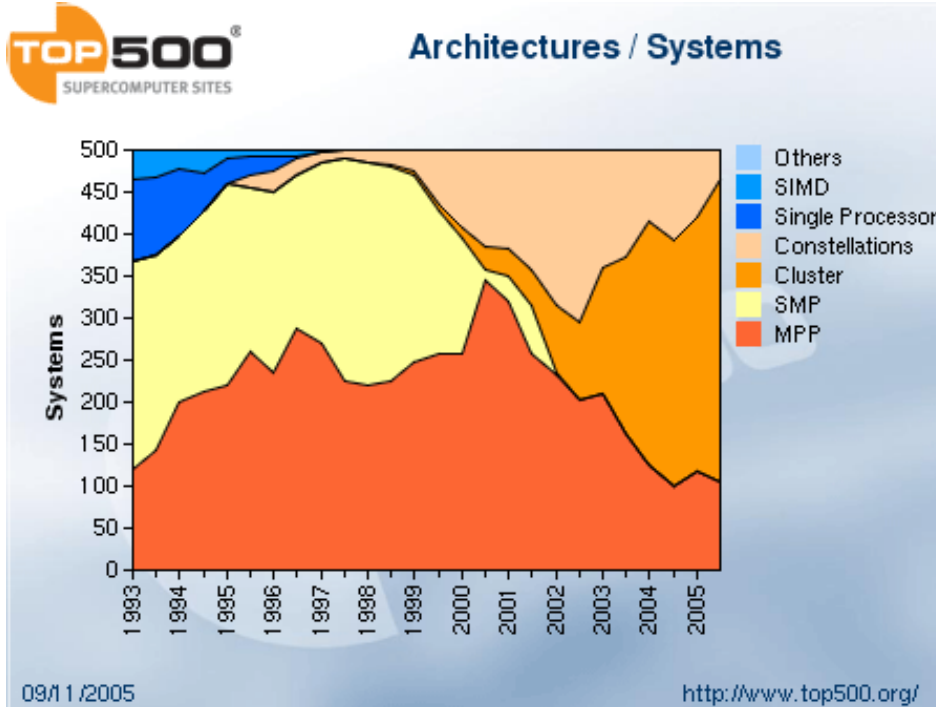
2005 → 1 Tflops – 280 Tflops

1993 → 1 Gflops – 100 Gflops

Pflops → 2009 - 2015



Clusters win

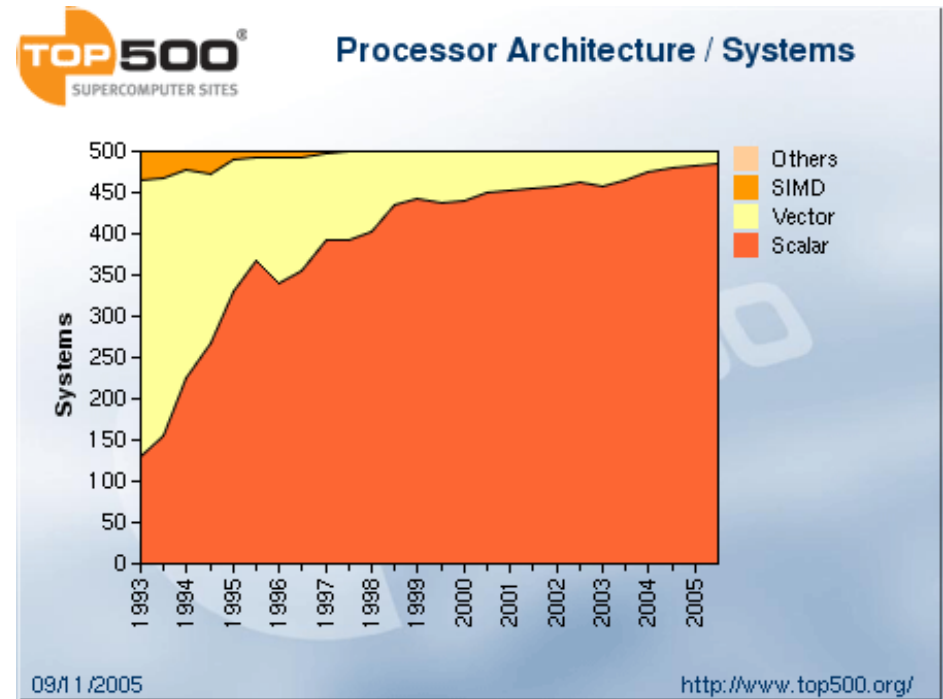
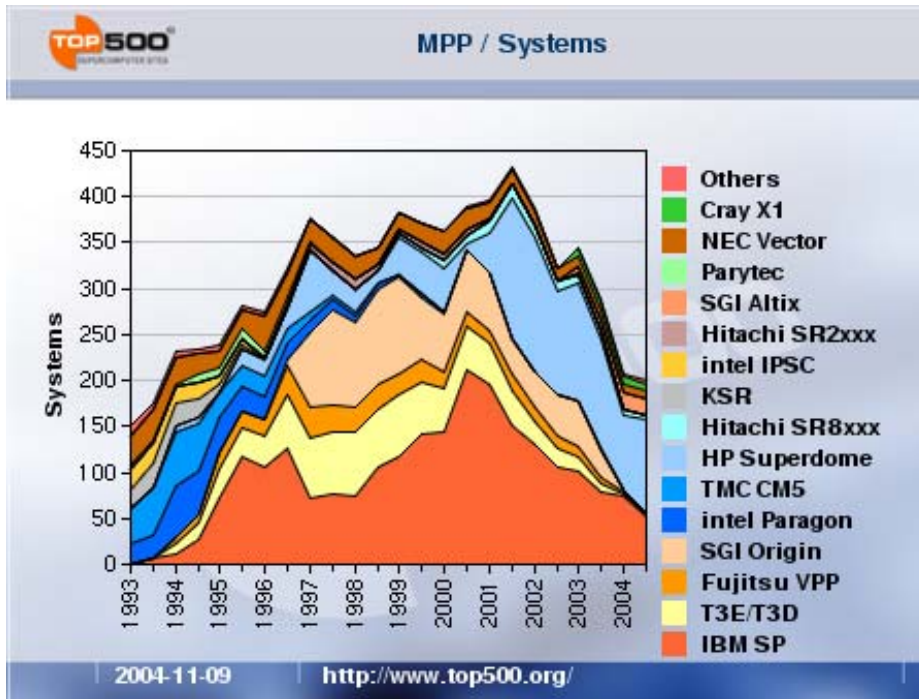


More than 60% of 2005 systems are clusters

IBM clusters dominate 2005



MPP Machines and Processor Architecture

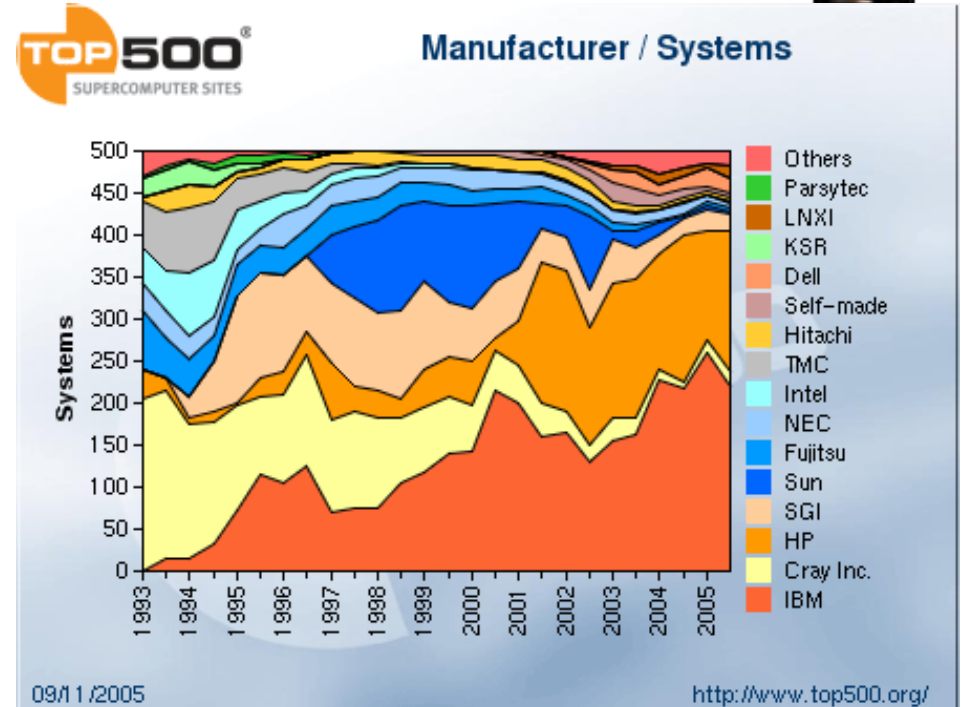
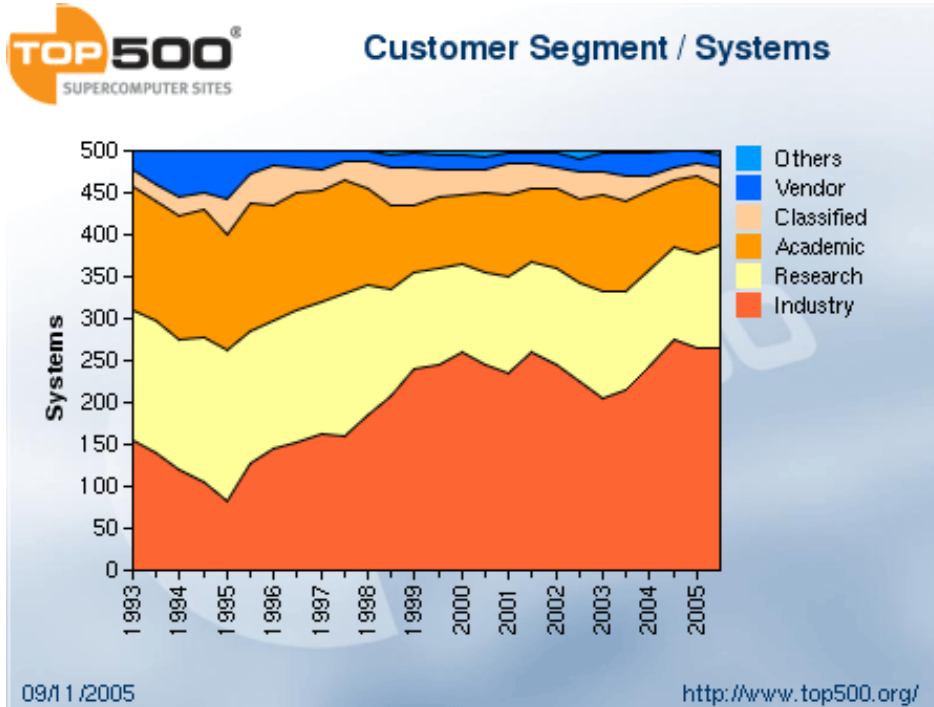


The use of vector processors is declining

MPP is going down



Main Customers and Manufacturers

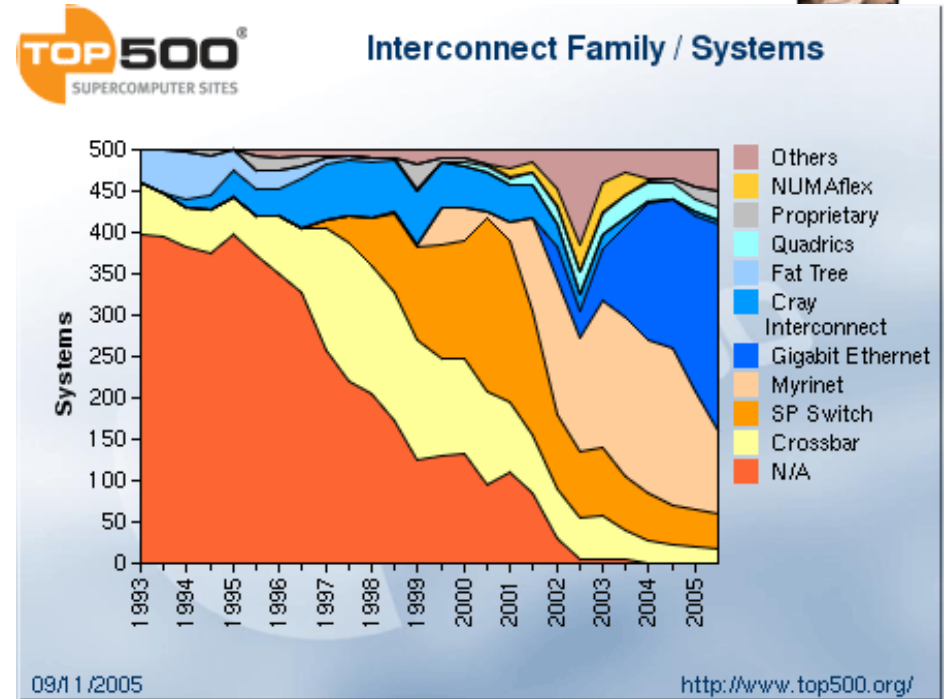
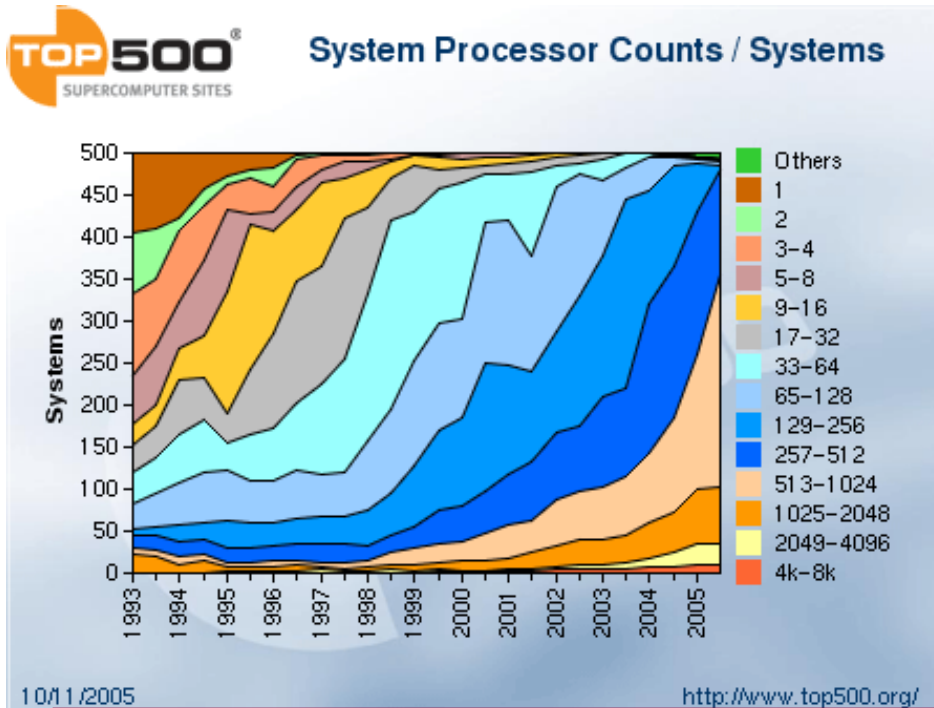


Industry use of supercomputers is increasing

IBM and HP dominate the manufacturers' list



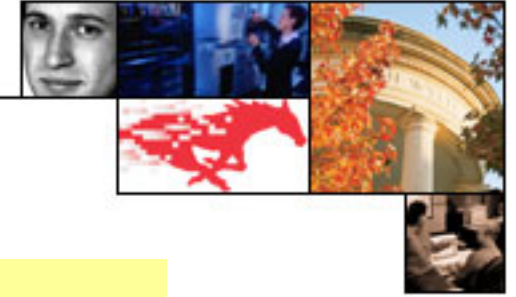
Processor Count and Interconnect



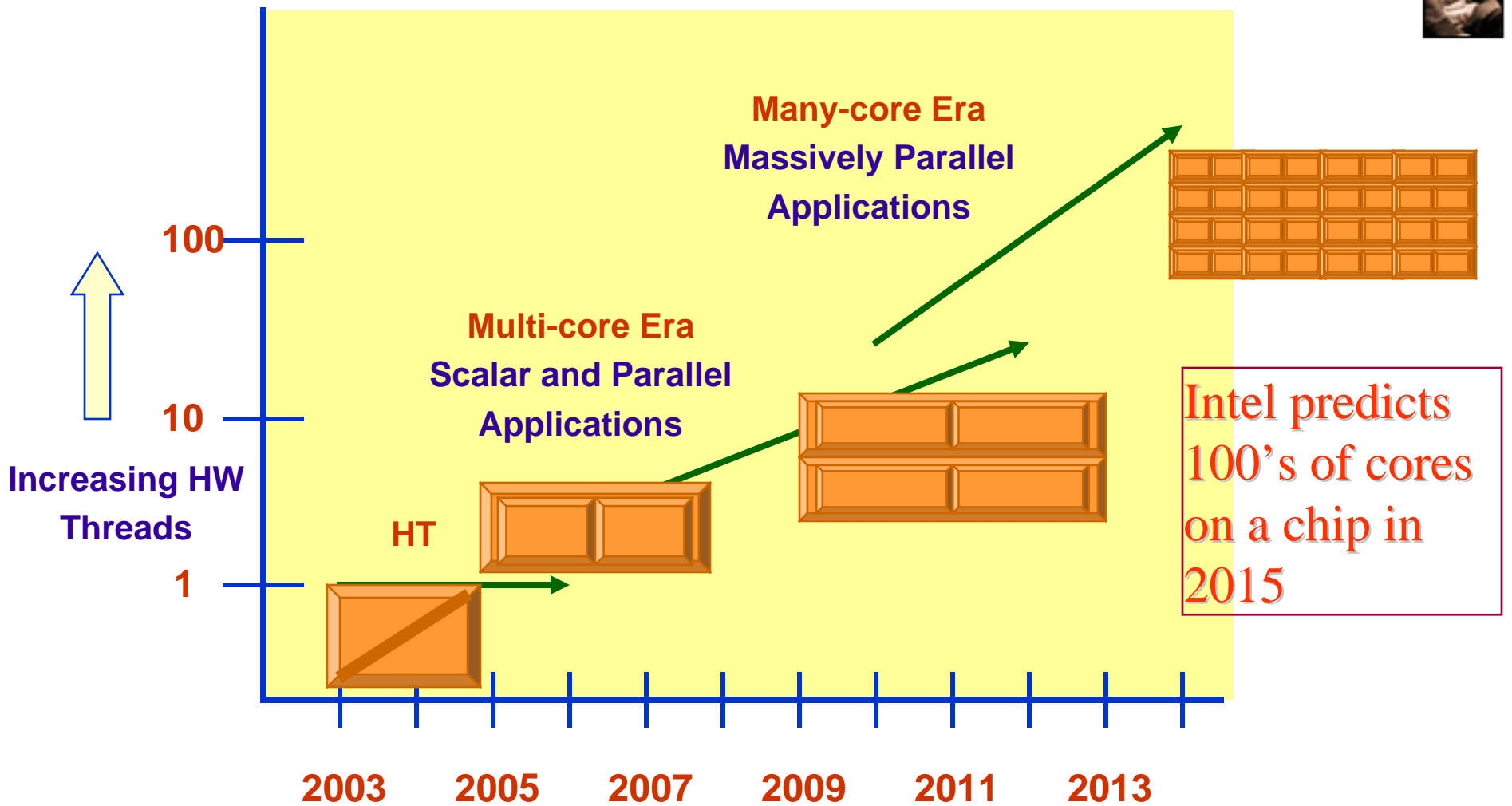
Majority of 2005 systems → 513-1024 procs

Until 2004 → less than 10,000 procs per system

Gigabit Ethernet is the winning interconnect in 2005



From HT to Many-Core





What shall we expect next? Challenges for the Future



Hierarchy of Parallelism

Multiple processors at all levels: Chip, Machine, Cluster, and Grid

- *Chip-level* → coping with the flattening of Moore's Law will drive the **Many-Core** alternative.
- *Machine level* → high performance desktop. Ease of use and application development are major challenges.
- *Cluster level* → will continue to be a dominant force in the market but the **number of processors** is not expected to increase significantly.
- *Grid level* → islands of computing and data resources. Dealing with **complexity** is a major challenge.



Reaching the Petaflop Performance

- Petaflop systems will continue to exceed the 100,000 processor mark
- An alternative to the cluster architecture will have to emerge to cope with larger number of processors.
(Research in parallel architecture)
- Scaling is a major challenge
 - *How will algorithms, tools and system software scale to hundreds of thousands of processors?*
(Research in parallel algorithms and tools)



Software Paradigm Shift

- How long more can MPI survive?
- Many-core architecture will drive more research in compiler technology and programming languages.
- With differences in architecture at the different levels, a programming model will emerge with the following properties:
 - *Simplicity and Ease of understanding*
 - *Moderate level of abstraction*
 - *Stability in face of technological changes*
 - *Use with wide range of parallel computers*



Mainstream Parallelism

- Popular Moore's Law will soon end
- In addition to Grid, Cluster, and machine level parallelism, Chip level multiprocessors will flourish.
- Researchers and developers of compiler, algorithms, programming language, and software tools will have to react with innovations.
- Parallel Computer → Computer
Parallel Program → Program
Parallel Algorithm → Algorithm
(this time it will happen)



SMU School of Engineering



Thank you!!

AICCSA-06

Hesham El-Rewini, March 9, 2006

Computer Science and Engineering