

LECTURE 1 CSE610 Mon 24Jan05

CSE610 - Parallel Computer Architecture – Spr 2005

Lectures: SocialBehaviorial N107 **Mon, Wed** 15.50-17:10 (3:50-5:10 pm)

Professor: Larry Wittie **Office meetings** : 1308 Comp Sci Bldg

Email: ldw@cs.sunysb.edu **Office hours:** MW 17:30-18:15 (maybe later)
(1308 CSB)

Phones:(Messages only)**2-8456** (1308)**2-8750** **Web:**

<http://www.cs.sunysb.edu/~cse610>

Required Text (price estimates from web 1/23/05): "In Search of Clusters"
2nd Edition, by Gregory Pfister, Prentice Hall, 1998 \$45 (\$25 used at
<http://getttextbooks.com/search/?isbn=0-13-899709-8>) ISBN 0-13-899709-8

*****5 of 5

Recommended Text 1: "Computer Architecture: A Quantitative Approach" **THIRD Edition**, by John Hennessy and David Patterson, Morgan-

Kaufman, 2002 \$90 (\$40 used at <http://getttextbooks.com/search/?isbn=1-55860-596-7>) ISBN 1-55860-596-7 ***3.4 of 5

Recommended Text 2: "Parallel Computer Architecture: A Hardware/Software Approach", by David Culler, J.P. Singh and Anoop Gupta, Morgan-Kaufman, 1998 \$90 (\$40 used at <http://getttextbooks.com/search/?isbn=1-55860-343-3>) ISBN 1-55860-343-3
***4.3 of 5

Objectives: This course is a lecture survey of the major objectives, constraints, and techniques for creating effective parallel computing systems. This semester we will first read the Pfister text on the power and limitations of grid machines (distributed computing). The material in chapters 6 (Multicomputers) and 8 (Networks) of the Hennessy and Patterson classic will be supplemented by historical material from the Culler text, which covers mainly distributed shared memory parallel computing, culminating in the SGI Origin 2000. Material on the Cray MTA-2 (nee Tera MTA-1), the leading example of multithreaded architectural support for shared memory supercomputing will be round out the coverage of existing and historical parallel computing systems.

Most of you already own Hennessy and Patterson. I have a few copies of Culler that can be shared.

Explicitly parallel computing historically has been the answer to the next increase in computing power for each generation of technology, only to become an invisible part of future generations. The 1960s-70s had vector pipeline supercomputers. The 70s-80s, parallel machines of many shapes to minimize switch delays for data sharing. The 80s-90s, shared memory bus multiprocessors with scalar pipeline RISC mCPUs. The 90s-00s, non-shared memory grid computers from racks of commodity (RISC and shared bus) mCPUs connected by fast network switches. What is next? Multiprocessor chips. Shared-memory grid machines spanning the globe. Invisible nanotech computing grids in wall paper, with short-distance wireless links to global grid optical fibers. Unobservable parallelism via quantum computing. What else?

Central themes will be how each architecture seeks to execute certain classes of programs faster than a single processor, the difficulty in accessing memory fast enough to keep the CPUs busy, the de-emphasis of network topologies as propagation delays dominate switch delays in both ultra-fast and Earth-grid

computers, theoretical limits on parallel program speedup, and memory sharing via caches for fewer than 500 CPUs and multi-threading for massively parallel computers. Roughly 30% of the course material will come from the Pfister essay, 30% of the course material will come from the Hennessy and Patterson text, 20% of the course material will come from the Culler text, mainly chapters 1, 8, and 10; the rest from newer articles and my own research.

Credits: 3

Homework, Project, and Examination: Homework problems will be assigned occasionally, usually due in a week and all to be done personally. There will be individual research projects with class presentations near the semester end. Some may involve hands-on work with a small grid computer. Midterm and final exam questions will resemble those for homework or will involve architectural features covered in lectures. Exams will be open books and notes.

Grading: The final grade will be determined by adding the raw scores with these weights: Homework(20%) Midterm (20%) Project(30%) Final exam(30%) . I may change a grade upwards by as much as 5%.

Handouts: This syllabus, homework and/or project assignments, will be available on the Web <http://www.cs.sunysb.edu/~cse610> for you to download and print.

Special Needs: If you have any condition, such as a physical or mental disability, which will make it difficult for you to carry out the work as I have outlined it or which will cause you to need extra time on examinations, please notify me in the first two weeks of the course so that we may make appropriate arrangements.

Tentative Schedule - Subject to Change

Week	Mon, Wed	Text Chapter
1	24,26 Jan	Course overview, Culler 1 (Intro, Parallel computing trends)
	... more soon	

University Calendar Special Dates – from links on <http://www.stonybrook.edu/sb/calacademic.shtml>

What is Parallel Architecture?

A parallel computer is a collection of processing elements that cooperate to solve large problems fast

Some broad issues:

- Resource Allocation:
 - how large a collection?
 - how powerful are the elements?
 - how much memory?
- Data access, Communication and Synchronization
 - how do the elements cooperate and communicate?
 - how are data transmitted between processors?
 - what are the abstractions and primitives for cooperation?
- Performance and Scalability
 - how does it all translate into performance?
 - how does it scale?

Why Study it Today?

History: diverse and innovative organizational structures, often tied to novel programming models

Rapidly maturing under strong technological constraints

- The “killer micro” is ubiquitous
- Laptops and supercomputers are fundamentally similar!
- Technological trends cause diverse approaches to converge

Technological trends make parallel computing inevitable

- In the mainstream

Need to understand fundamental principles and design tradeoffs, not just taxonomies

Inevitability of Parallel Computing

Application demands: Our insatiable need for computing cycles

Technology Trends

- Number of transistors on chip growing rapidly
- Clock rates expected to go up only slowly

Architecture Trends

- Instruction-level parallelism valuable but limited
- Coarser-level parallelism, as in MPs, the most viable approach

Current trends:

- Today's microprocessors have multiprocessor support
- Servers and workstations becoming MP: Sun, SGI, DEC, COMPAQ!...
- Tomorrow's microprocessors are multiprocessors - multiple CPU cores/chip

Application Trends

Demand for cycles fuels advances in hardware, and vice-versa

- Cycle drives exponential increase in microprocessor performance
- Drives parallel architecture harder: most demanding applications

Range of performance demands

- Need range of system performance with progressively increasing cost
- Platform pyramid

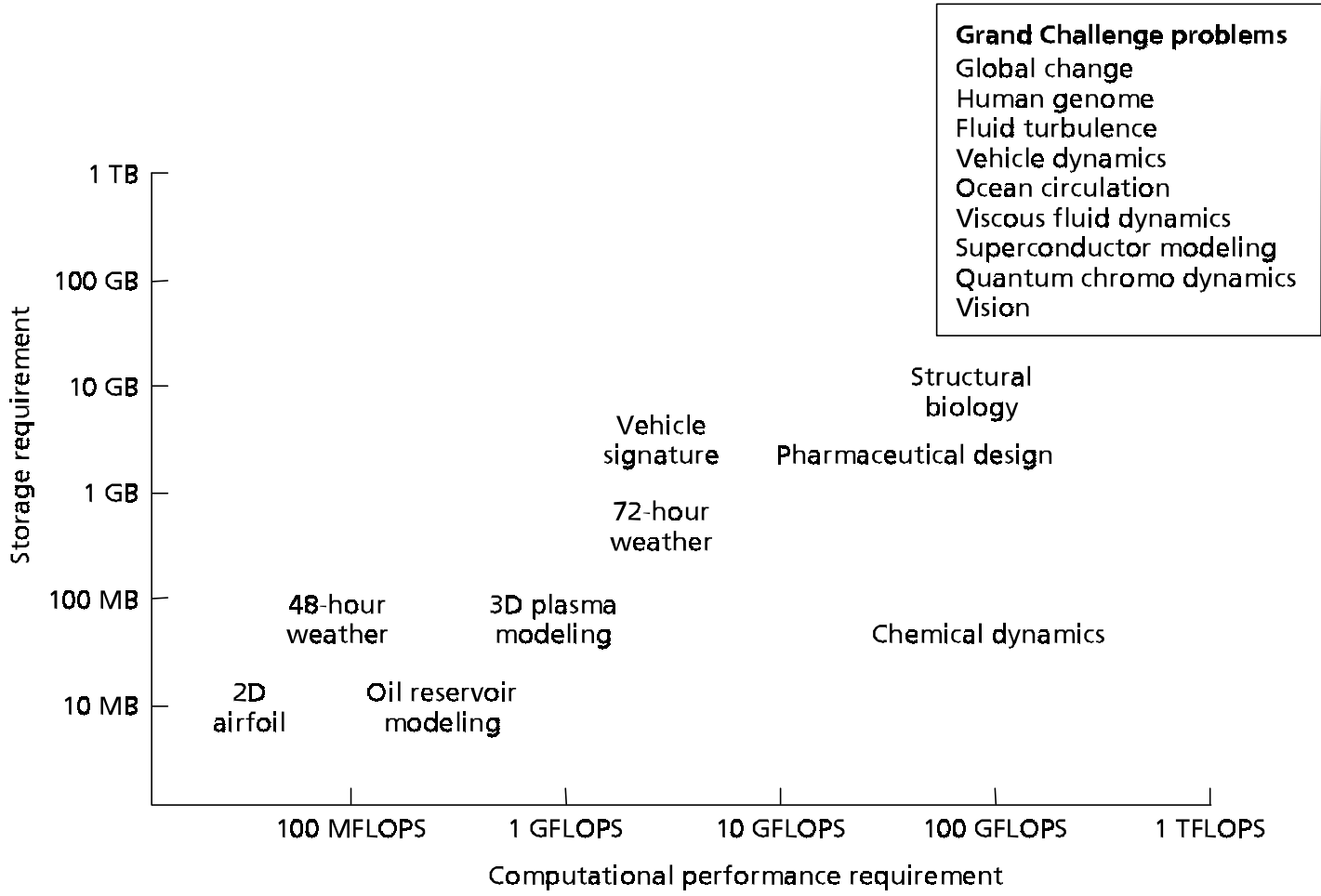
Goal of applications in using parallel machines: Speedup

$$\text{Speedup } (p \text{ processors}) = \frac{\text{Performance } (p \text{ processors})}{\text{Performance } (1 \text{ processor})}$$

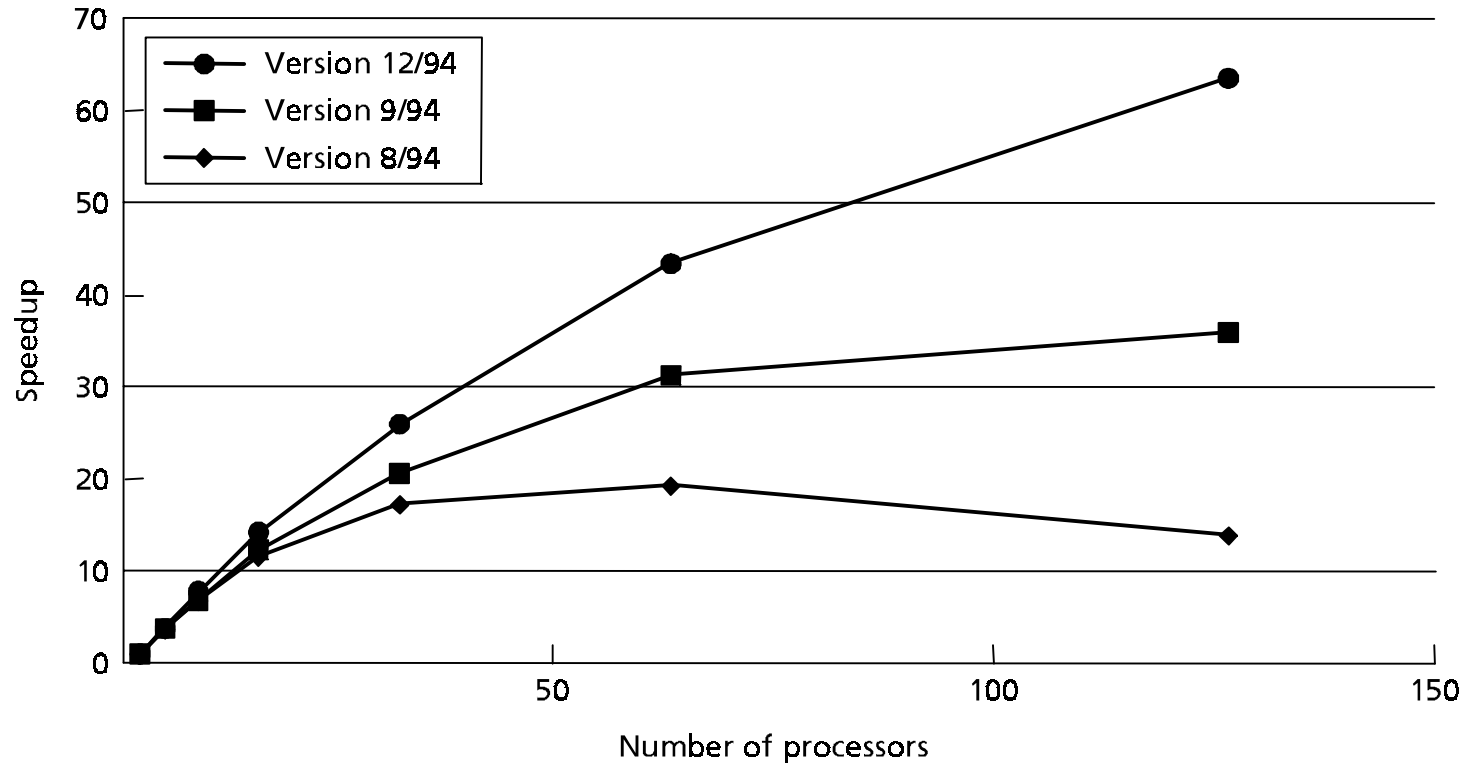
For a fixed problem size (input data set), performance = 1/time

$$\text{Speedup}_{\text{fixed problem}} (p \text{ processors}) = \frac{\text{Time } (1 \text{ processor})}{\text{Time } (p \text{ processors})}$$

Scientific Computing Demand - NanoScale Design Needs 1 PFLOPS



Learning (and Bowed Speedup) Curve for Parallel Applications



- AMBER molecular dynamics simulation program
- Starting point was vector code for Cray-1
- 145 MFLOP on Cray90, 406 for final version on 128-processor Paragon, 891 on 128-processor Cray T3D

Summary of Application Trends

Transition to parallel computing has occurred for scientific and engineering computing

In rapid progress in commercial computing

- Database and transactions as well as financial
- Usually smaller-scale, but large-scale systems also used

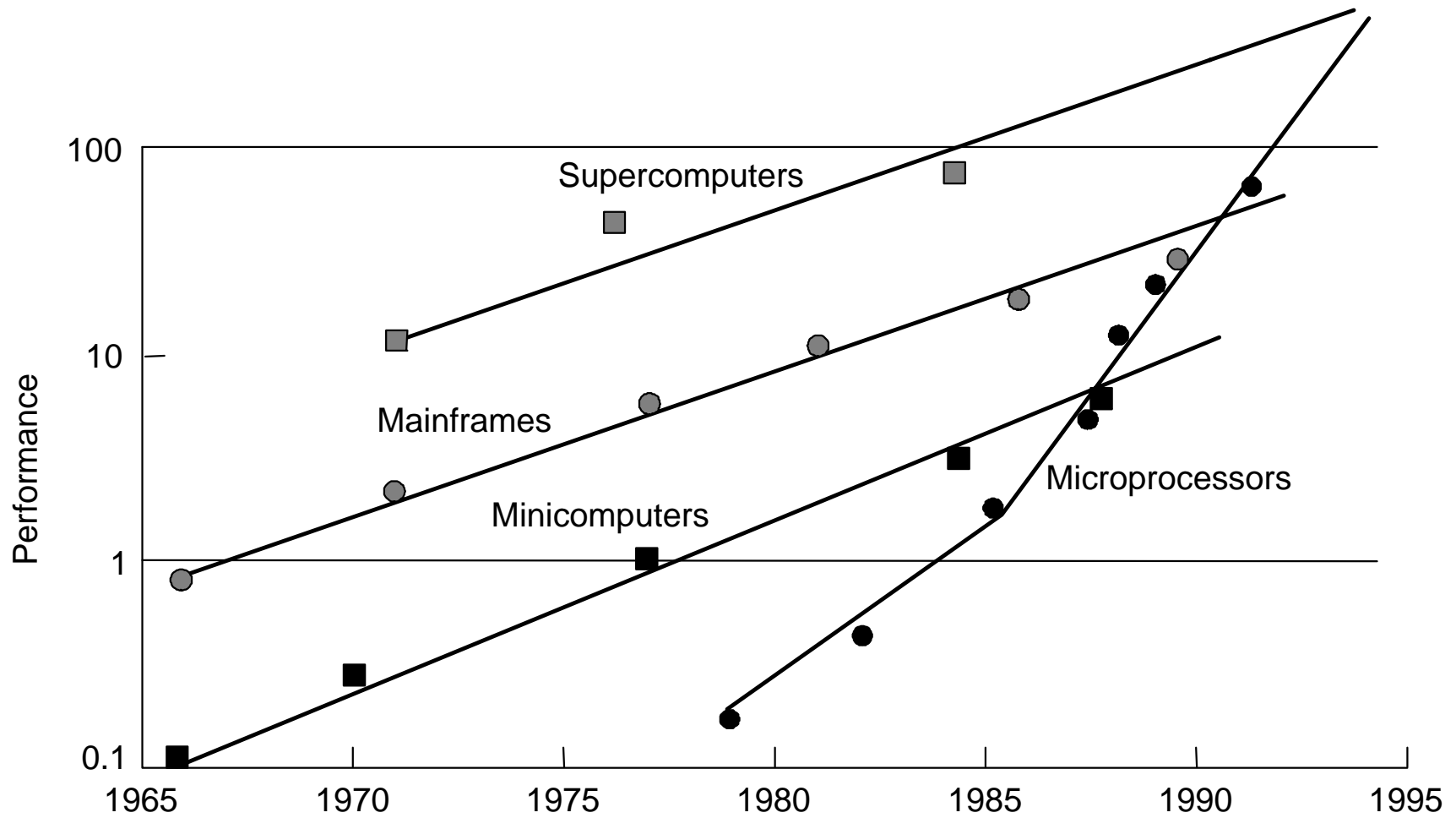
Desktop also uses multithreaded programs, which are a lot like parallel programs

Demand for improving throughput on sequential workloads

- Greatest use of small-scale multiprocessors

Solid application demand exists and will increase

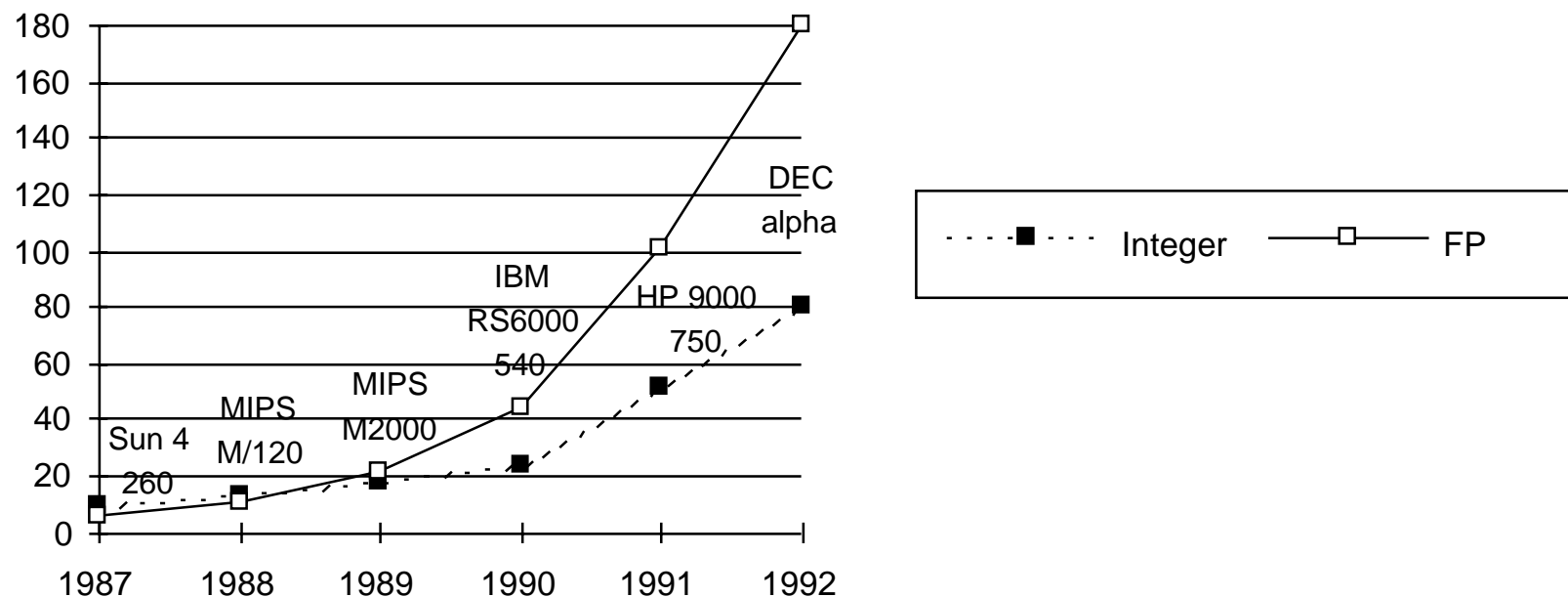
Technology Trends



The natural building block for multiprocessors is now also about the fastest!

General Technology Trends

- *Microprocessor performance* increases 50% - 100% per year
- *Transistor count* doubles every 3 years
- *DRAM size* quadruples every 3 years
- Huge investment per generation is carried by huge commodity market



- Not that single-processor performance is plateauing, but that parallelism is a natural way to improve it.

Technology: A Closer Look

Basic advance is *decreasing feature size* (λ)

- Circuits become either faster or lower in power

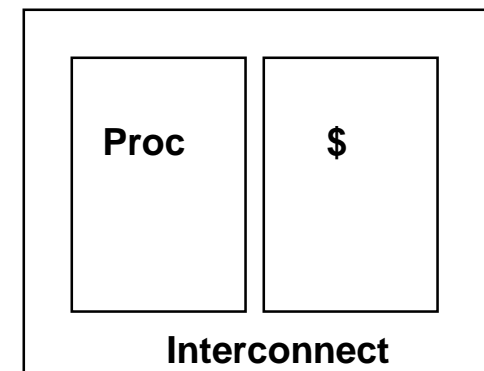
Die size is growing too

- Clock rate improves roughly proportional to improvement in λ
- Number of transistors improves like λ^2 (or faster)

Performance > 100x per decade; clock rate 10x, rest transistor count

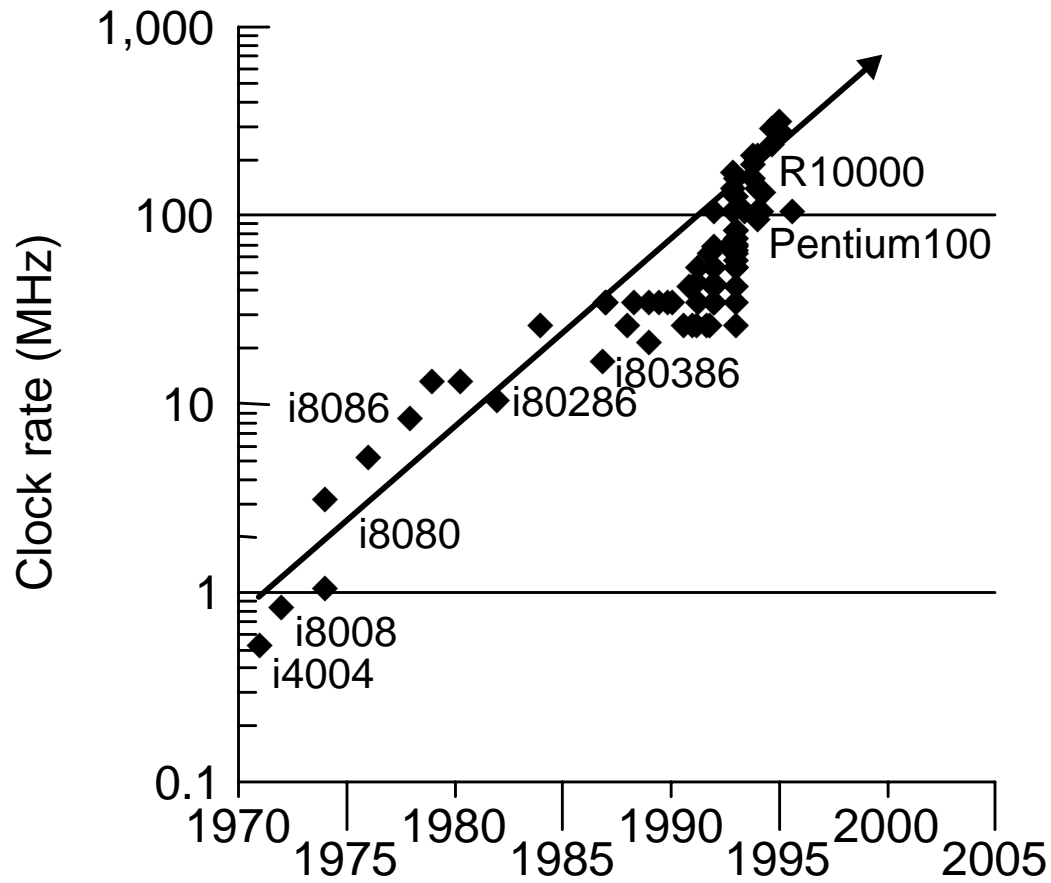
How to use more transistors?

- Parallelism in processing
 - multiple operations per cycle reduces CPI
- Locality in data access
 - avoids latency and reduces CPI
 - also improves processor utilization
- Both need resources, so tradeoff



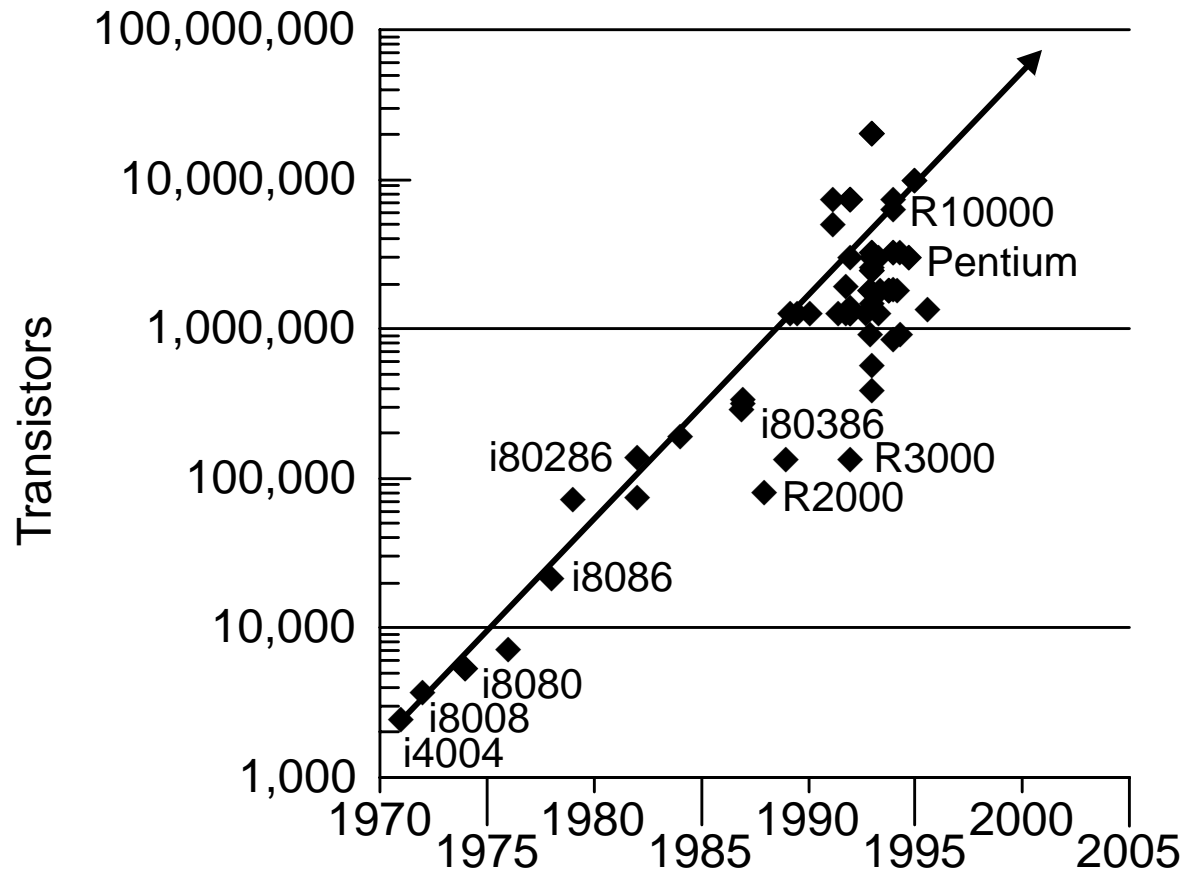
Fundamental issue is resource distribution, as in uniprocessors

Clock Frequency Growth Rate



- 30% per year

Transistor Count Growth Rate



- 100 million transistors on chip by early 2000's A.D.
- Transistor count grows much faster than clock rate
 - 40% per year, order of magnitude more contribution in 2 decades

Similar Story for Storage

Divergence between memory capacity and speed more pronounced

- Capacity increased by 1000x from 1980-95, speed only 2x
- Gigabit DRAM by c. 2000, but gap with processor speed much greater

Larger memories are slower, while processors get faster

- Need to transfer more data in parallel
- Need deeper cache hierarchies
- How to organize caches?

Parallelism increases effective size of each level of hierarchy, without increasing access time

Parallelism and locality within memory systems too

- New designs fetch many bits within memory chip; follow with fast pipelined transfer across narrower interface
- Buffer caches most recently accessed data

Disks too: Parallel disks plus caching

Architectural Trends

Architecture translates technology's gifts to performance and capability

Resolves the tradeoff between parallelism and locality

- Year 2000 microprocessor: 1/3 compute, 1/3 cache, 1/3 off-chip connect
- Tradeoffs may change with scale and technology advances

Understanding microprocessor architectural trends

- Helps build intuition about design issues or parallel machines
- Shows fundamental role of parallelism even in “sequential” computers

Four generations of architectural history: tube, transistor, IC, VLSI

- Here focus only on VLSI generation

Greatest delineation in VLSI has been in type of parallelism exploited

Architectural Trends

Greatest trend in VLSI generation is increase in parallelism

- Up to 1985: bit level parallelism: 4-bit -> 8 bit -> 16-bit
 - slows after 32 bit
 - adoption of 64-bit now under way, 128-bit far (not performance issue)
 - great inflection point when 32-bit micro and cache fit on a chip
- Mid 80s to mid 90s: instruction level parallelism
 - pipelining and simple instruction sets, + compiler advances (RISC)
 - on-chip caches and functional units => superscalar execution
 - greater sophistication: out of order execution, speculation, prediction
 - to deal with control transfer and latency problems
- Next step: thread level parallelism

Architectural Trends: ILP

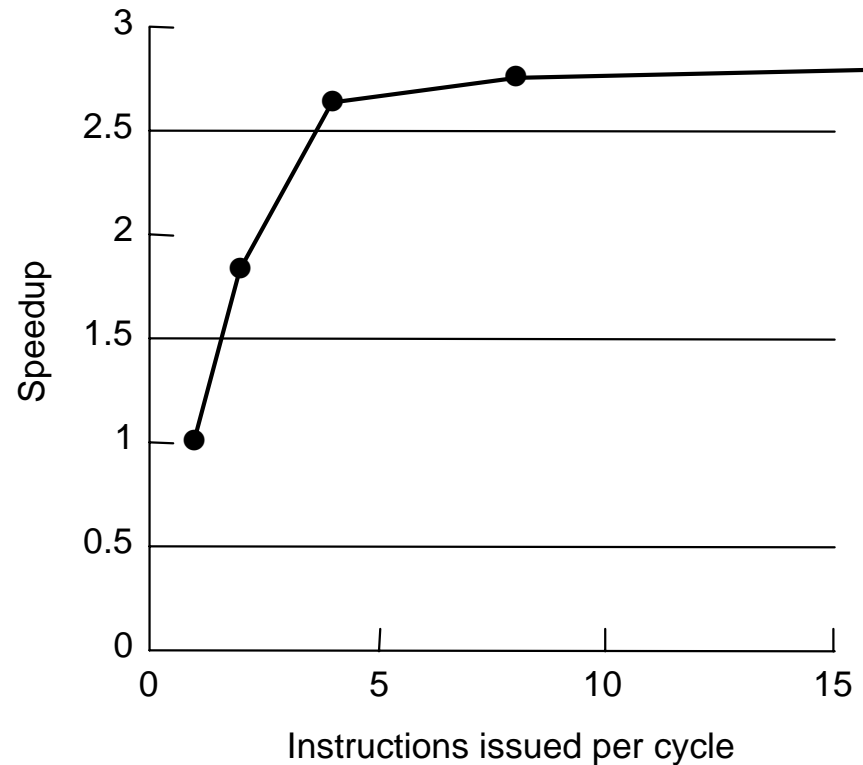
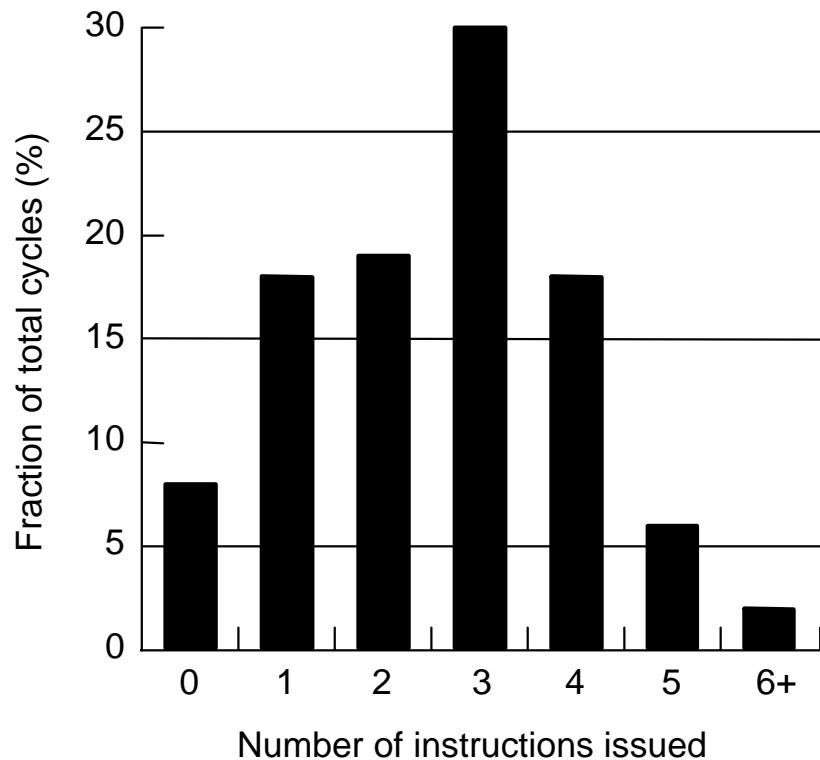
- Reported speedups for superscalar processors

• Horst, Harris, and Jardine [1990]	1.37
• Wang and Wu [1988]	1.70
• Smith, Johnson, and Horowitz [1989]	2.30
• Murakami et al. [1989]	2.55
• Chang et al. [1991]	2.90
• Jouppi and Wall [1989]	3.20
• Lee, Kwok, and Briggs [1991]	3.50
• Wall [1991]	5
• Melvin and Patt [1991]	8
• Butler et al. [1991]	17+

- Large variance due to difference in

- application domain investigated (numerical versus non-numerical)
- capabilities of processor modeled (only 2.6 or less realistic)

ILP Ideal Potential



- Infinite resources and fetch bandwidth, perfect branch prediction and renaming
 - real caches and non-zero miss latencies

Homework 1 for CSE610

Assigned Mon 24 Jan 2005

Due Wed 9 Feb 2005

Exercise 1.4 page 72, of Culler (see also p87 of 2.2.1 for help):

- 1.4 Given a histogram of available parallelism such as that shown in Figure 1.7, where f_i is the fraction of cycles on an ideal machine in which i instruction issue, derive a generalization of Amdahl's Law to estimate the potential speedup on a k -issue superscalar machine. Apply your formula to the histogram data in Figure 1.7 to produce the speedup curve shown in that figure.

Histogram shows fractions of cycles while running the program on a theoretical infinite-issue machine, so limit on number of instructions issued per cycle are those inherent to the dependencies of the program.