ECE 669

Parallel Computer Architecture

Lecture 1

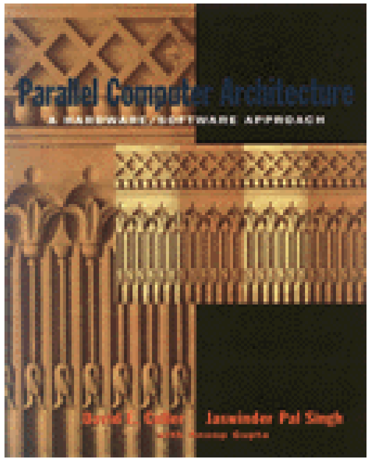
Course Introduction

Prof. Russell Tessier

Department of Electrical and Computer Engineering

Welcome to ECE 669/CA720-A

- Parallel computer architectures
- Interactive: your questions welcome!
- Grading
  - 6 Homework assignments (30%)
  - Mid-term exam (35%)
  - Final exam (35%)
- Experiments with network and cache simulators
- Culler text – research papers
- Acknowledgment
  - Prof Anant Agarwal (MIT)
  - Prof David Culler (California – Berkeley)



Parallel Architectures

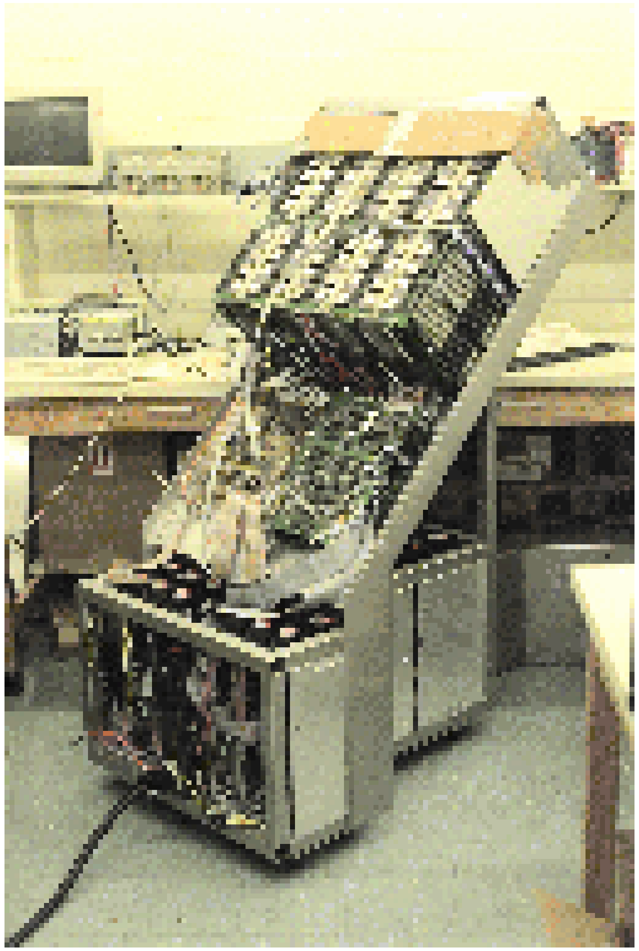
Why build parallel machines?

- To help build even bigger parallel machines
- To help solve important problems
  - Speed – more trials, less time
  - Cost
  - Larger problems
  - Accuracy

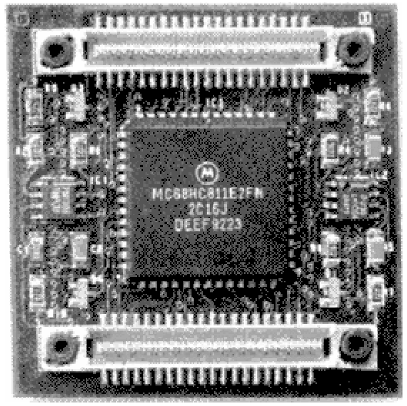
Must understand typical problems

MIT Computer Architecture Group – early 1990's

Alewife Machine



J-Machine



NuMesh

Applications ---> Requirements

- Processing
- Communication
- Memory
- I/O
- Synchronization

Numeric
Symbolic
Combinatorial

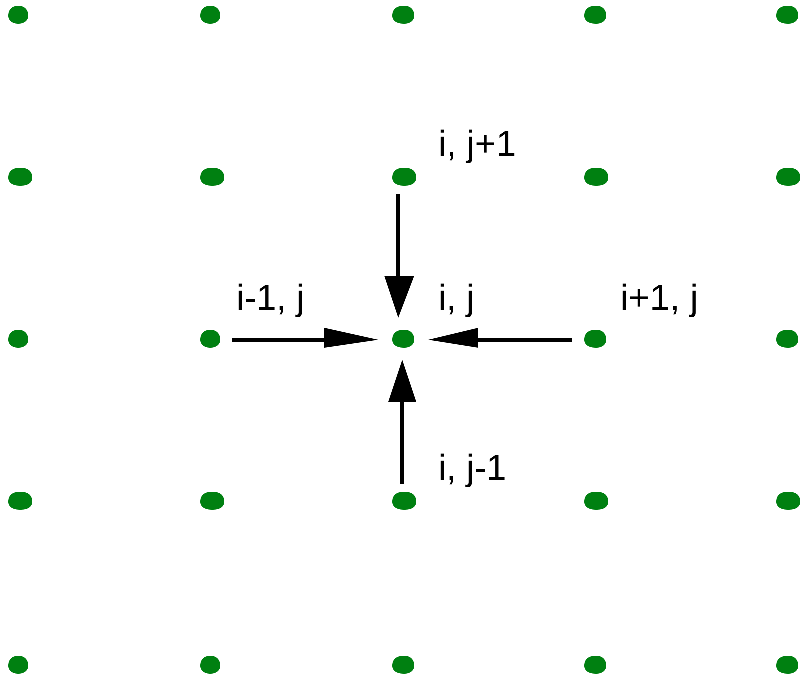
° **Architect must provide all of the above**

Requirements ---> Examples

- Relaxation - near-neighbor communication
- Multigrid - 1, 2, 4, 8, ... communication
- Numeric comp - Floating point
- Symbolic - Tags, branches
- Database - I/O
- Data parallel - Barrier synchronicity
- Dictionary - Memory, I/O

Communication requirements ---> Example

- Relaxation



Only near-neighbor communication

- So, let's build a special machine!
- But ... pitfalls!

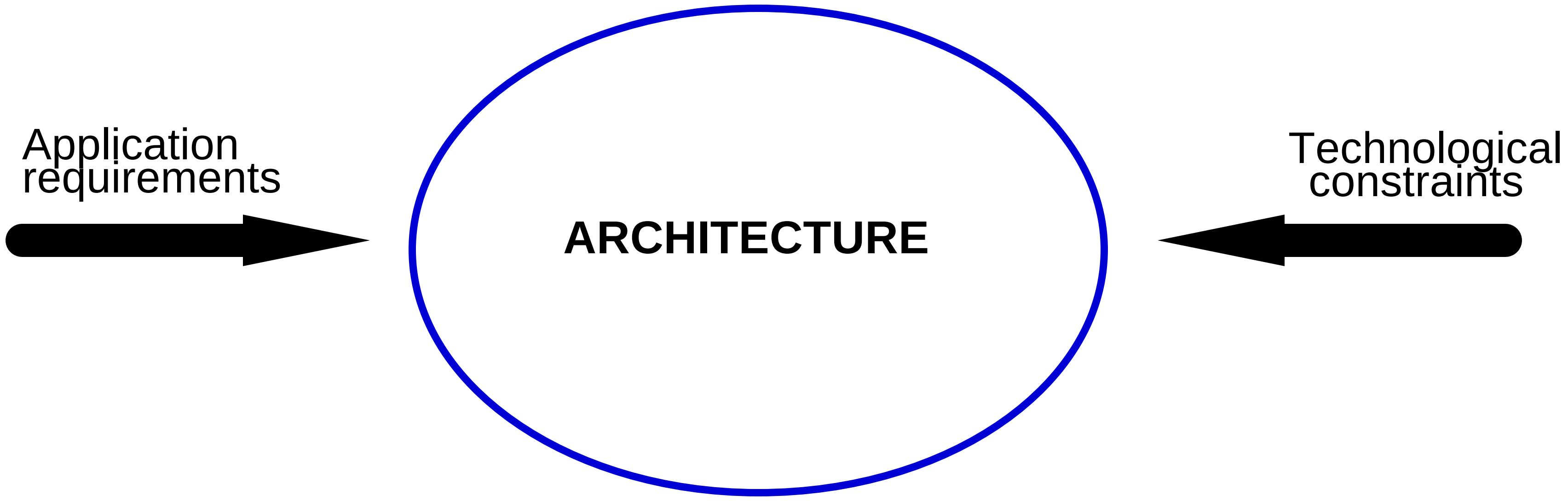
Specialized machines: Pitfalls

- **Faster algorithms appear … with different communication requirements**
- **Cost effectiveness**
  - **Economies of scale**
- **Simpler hardware & software mechanisms**
  - **More flexible**
  - **May even be faster!**
    - e.g. Specialized support for synchronization across multiple processors

Technology ---> Limitations & Opportunities

- Wires
  - Area
  - Propogation speed
- Clock
- Power
- VLSI
  - I/O pin limitations
  - Chip area
  - Chip crossing delay
  - Power
- Can not make light go any faster
- Three dimensions max
- KISS rule

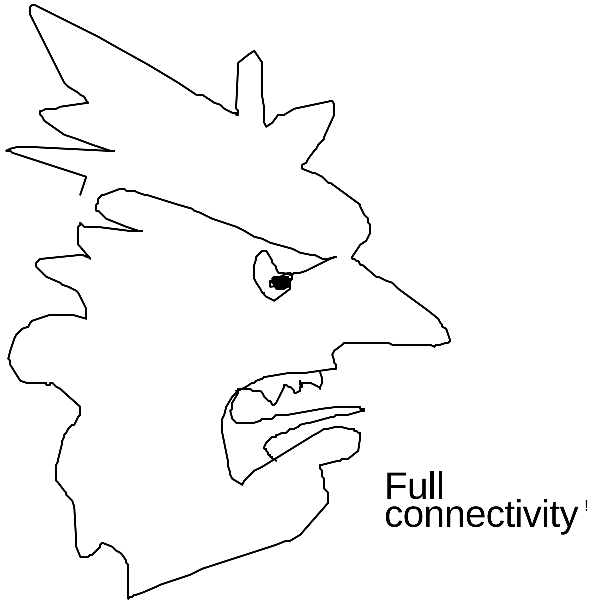
Major theme



- **Look at typical applications**
- **Understand physical limitations**
- **Make tradeoffs**

Unfortunately

- **Requirements and constraints are often at odds with each other!**



- **Architecture ---> making tradeoffs**

Putting it all together

- The systems approach
  - Lesson from RISCs
  - Hardware software tradeoffs
  - Functionality implemented at the right level
    - Hardware
    - Runtime system
    - Compiler
    - Language, Programmer
    - Algorithm

What will you get out of this course?

- In-depth understanding of the design and engineering of modern parallel computers
  - Technology forces
  - Fundamental architectural issues
    - naming, replication, communication, synchronization
  - Basic design techniques
    - cache coherence, protocols, networks, pipelining, …
  - Underlying engineering trade-offs
  - Case studies
  - Influence of applications
- From moderate to very large scale
- Across the hardware/software boundary

Speedup

- **Speedup (p processors) =**

$$\frac{\textit{Performance (p processors)}}{\textit{Performance (1 processor)}}$$

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

- **Speedup fixed problem (p processors) =**

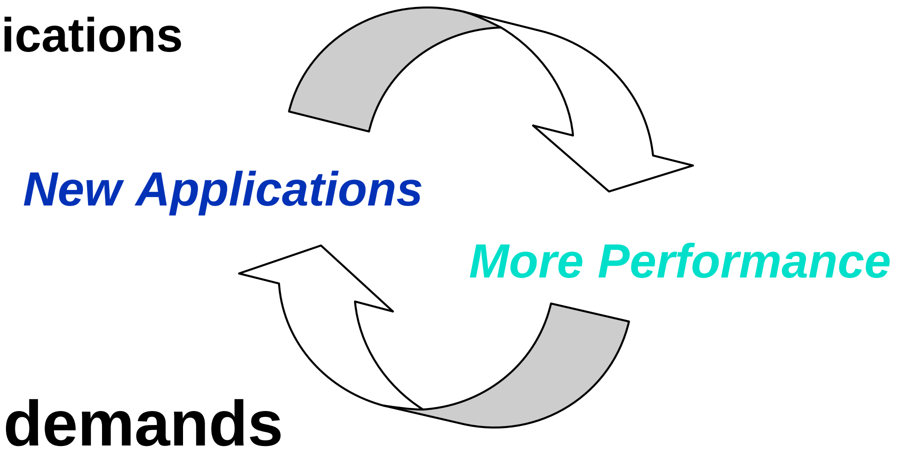
$$\frac{\textit{Time (1 processor)}}{\textit{Time (p processors)}}$$

Is Parallel Computing Inevitable?

- Application demands: Our insatiable need for computing cycles
- Technology Trends
- Architecture Trends
- Economics
- Current trends:
    - Today's microprocessors have multiprocessor support
    - Servers and workstations becoming MP: Sun, SGI, DEC, HP!...
    - Tomorrow's microprocessors are multiprocessors

Application Trends

- **Application demand for performance fuels advances in hardware, which enables new appl'ns, which...**
  - **Cycle drives exponential increase in microprocessor performance**
  - **Drives parallel architecture harder**
    - **most demanding applications**

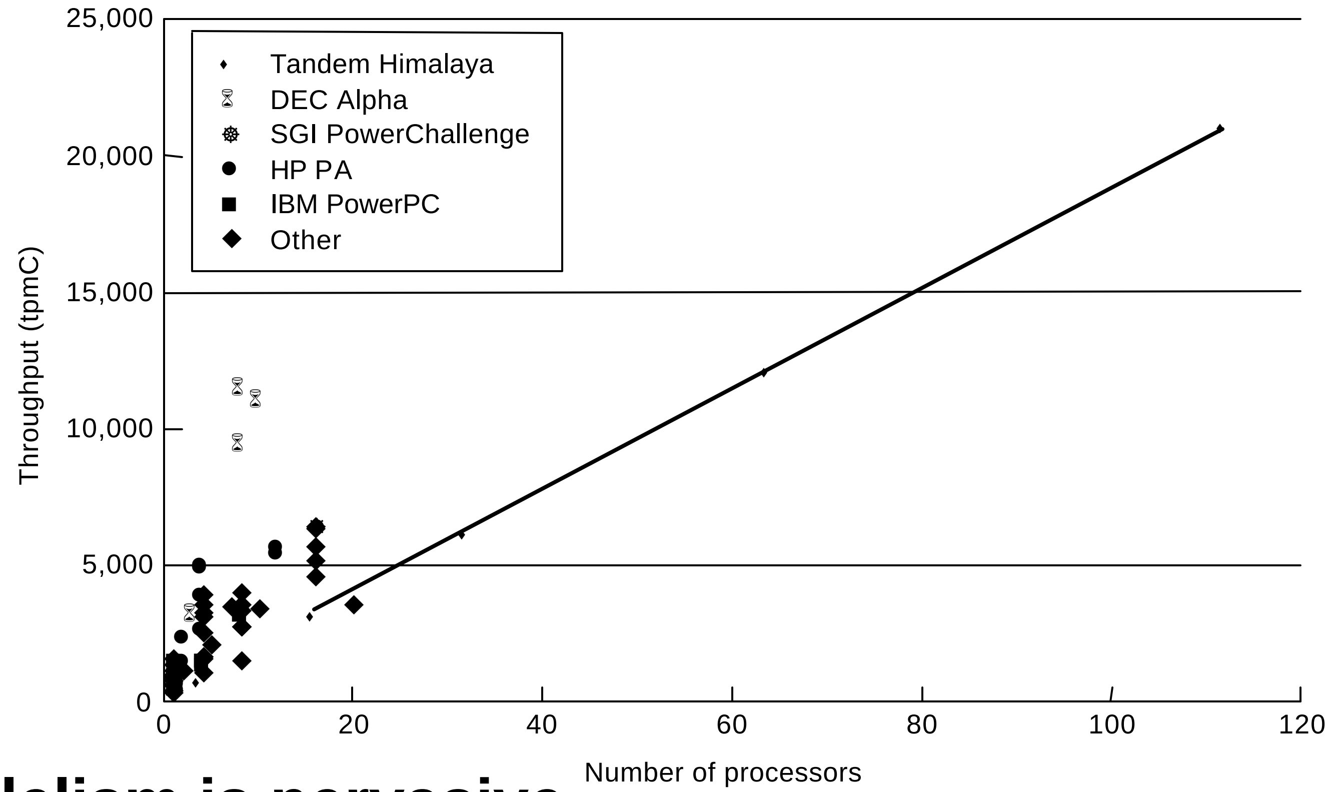


- **Range of performance demands**
  - **Need range of system performance with progressively increasing cost**

Commercial Computing

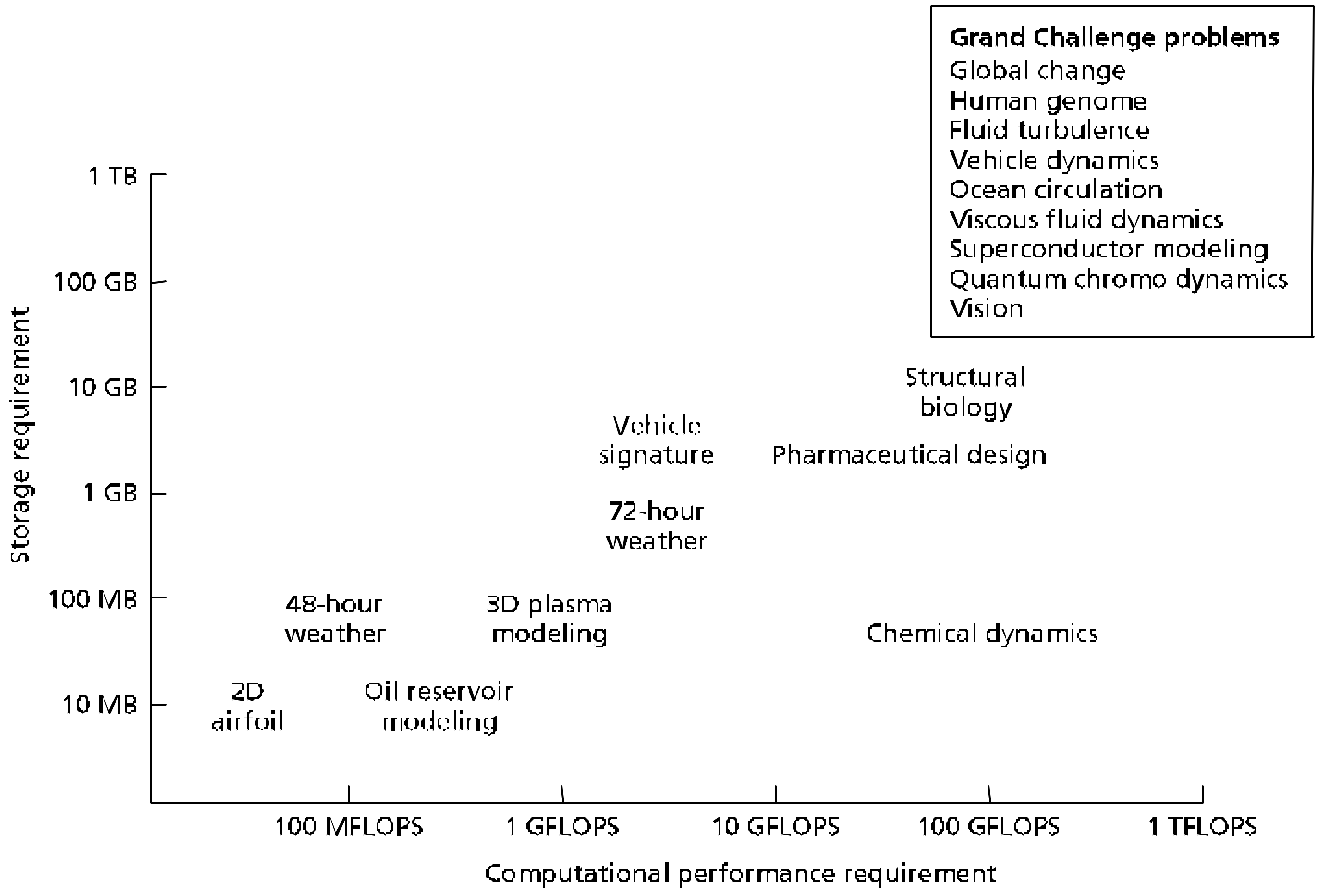
- Relies on parallelism for high end
  - Computational power determines scale of business that can be handled
- Databases, online-transaction processing, decision support, data mining, data warehousing ...
- TPC benchmarks Explicit scaling criteria provided
  - Size of enterprise scales with size of system
  - Problem size not fixed as p increases.
  - Throughput is performance measure (transactions per minute or tpm)

TPC-C Results for March 1996

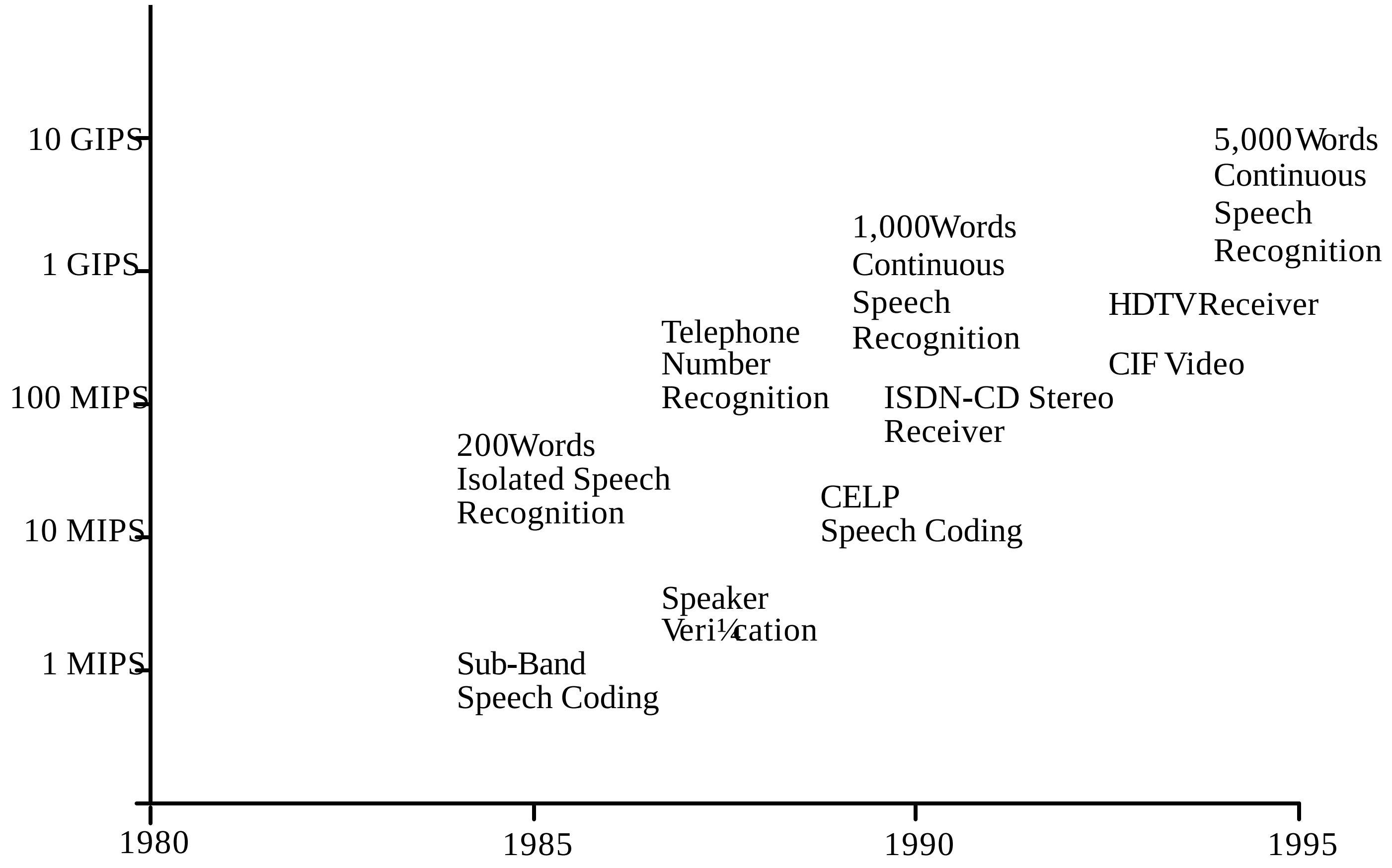


- **Parallelism is pervasive**
- **Small to moderate scale parallelism very important**
- **Difficult to obtain snapshot to compare across vendor platforms**

Scientific Computing Demand

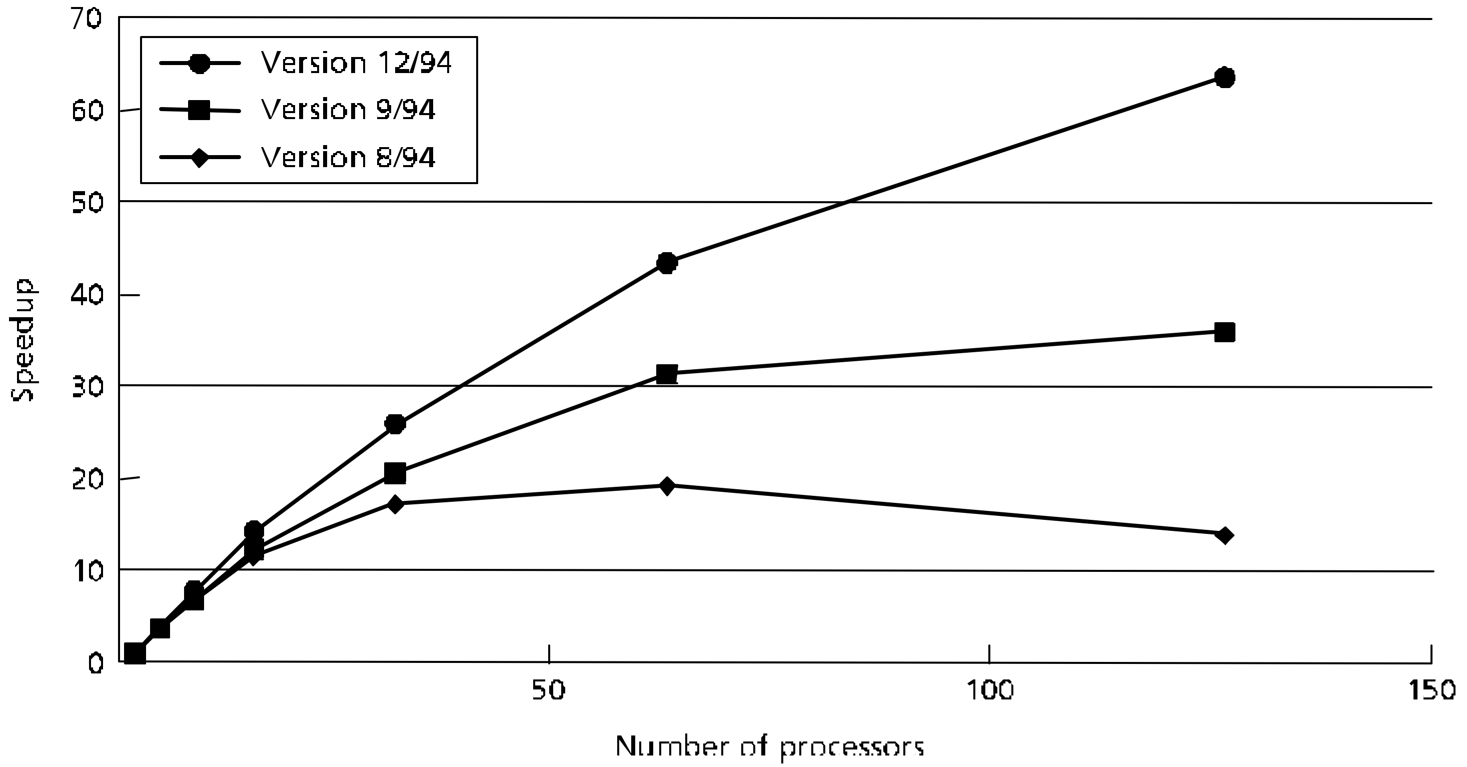


Applications: Speech and Image Processing



- Also CAD, Databases, . . .
- *100 processors gets you 10 years, 1000 gets you 20* !

Is better parallel arch enough?

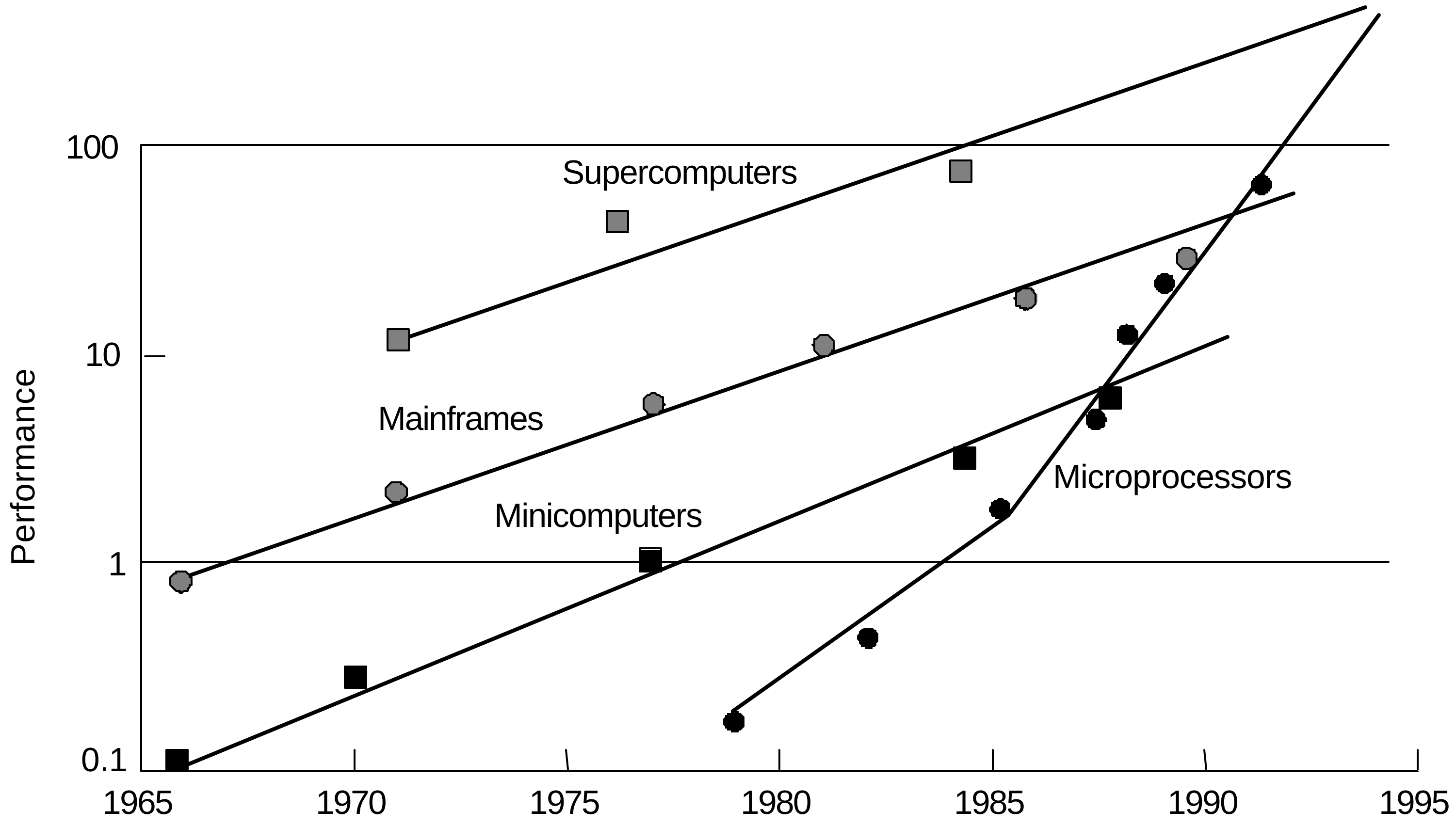


- AMBER molecular dynamics simulation program
- Starting point was vector code for Cray-1
- 145 MFLOP on Cray90, 406 for final version on 128-processor Paragon, 891 on 128-processor Cray T3D

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**

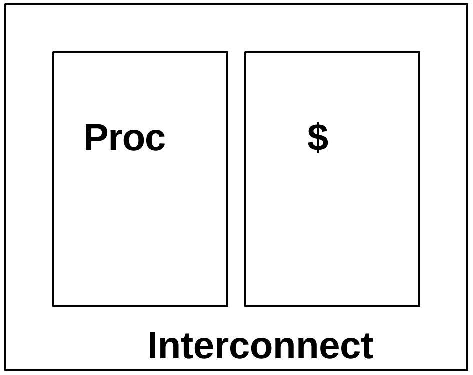
Technology Trends



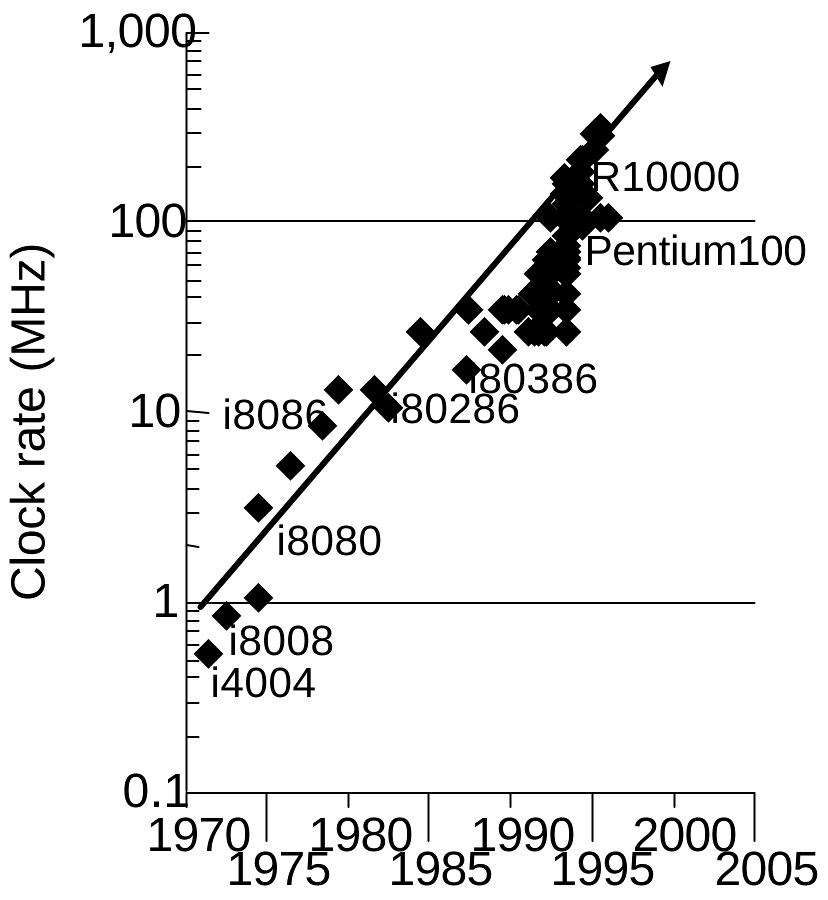
- **Today the natural building-block is also fastest!**

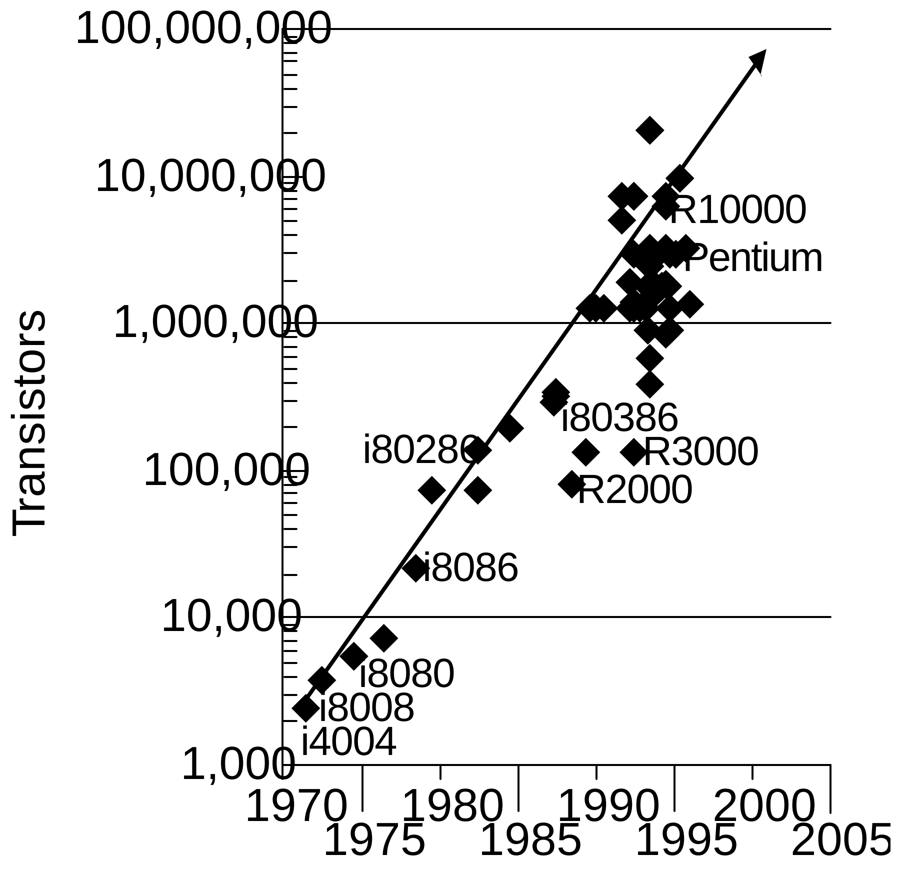
Technology: A Closer Look

- **Basic advance is *decreasing feature size* (l)**
  - **Circuits become either faster or lower in power**
- **Die size is growing too**
  - **Clock rate improves roughly proportional to improvement in l**
  - **Number of transistors improves like l^2 (or faster)**
- **Performance > 100x per decade**
  - **clock rate < 10x, rest is 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***



Growth Rates





- 30% per year

40% per year

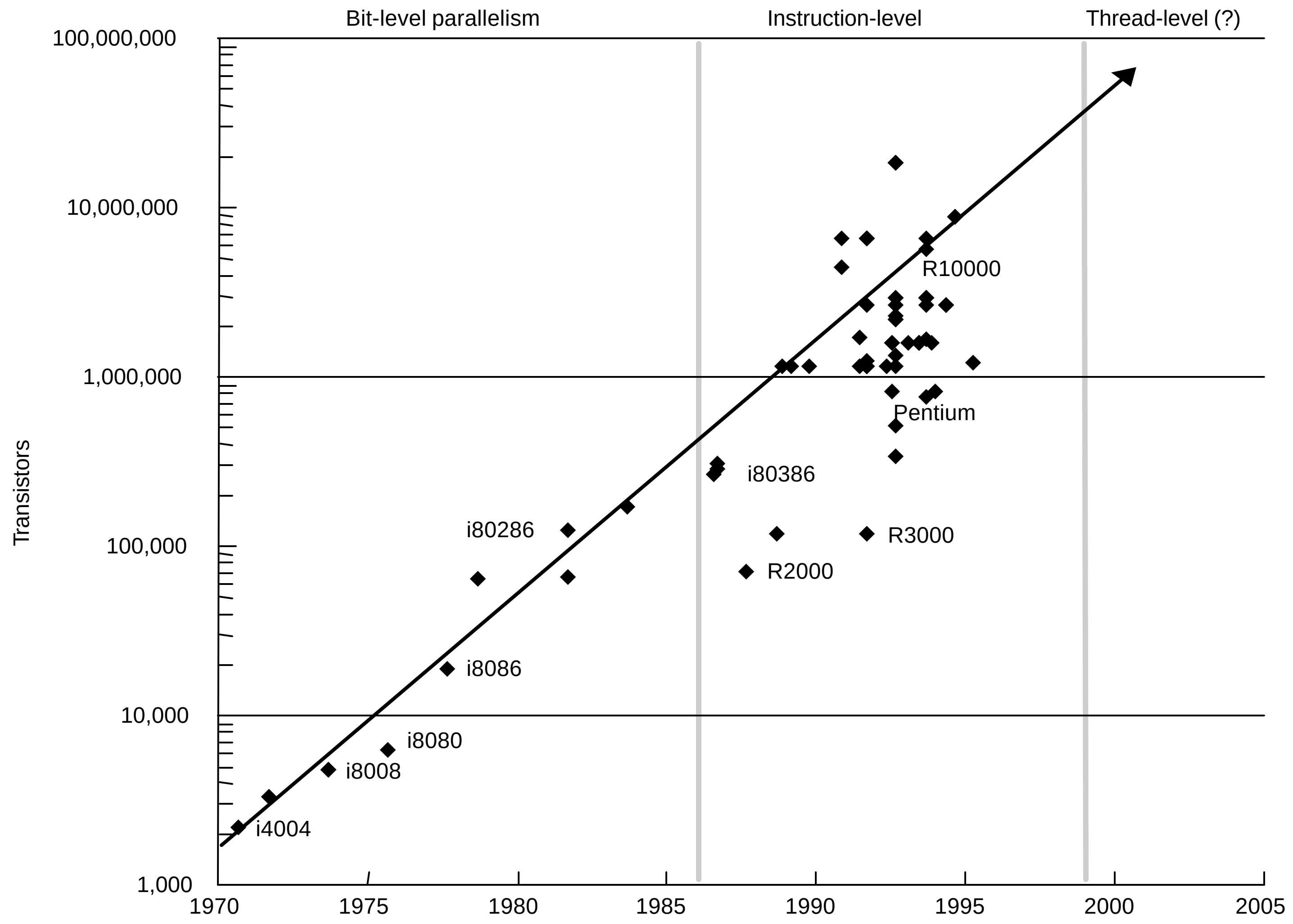
Architectural Trends

- Architecture translates technology's gifts into performance and capability
- Resolves the tradeoff between parallelism and locality
  - Current 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

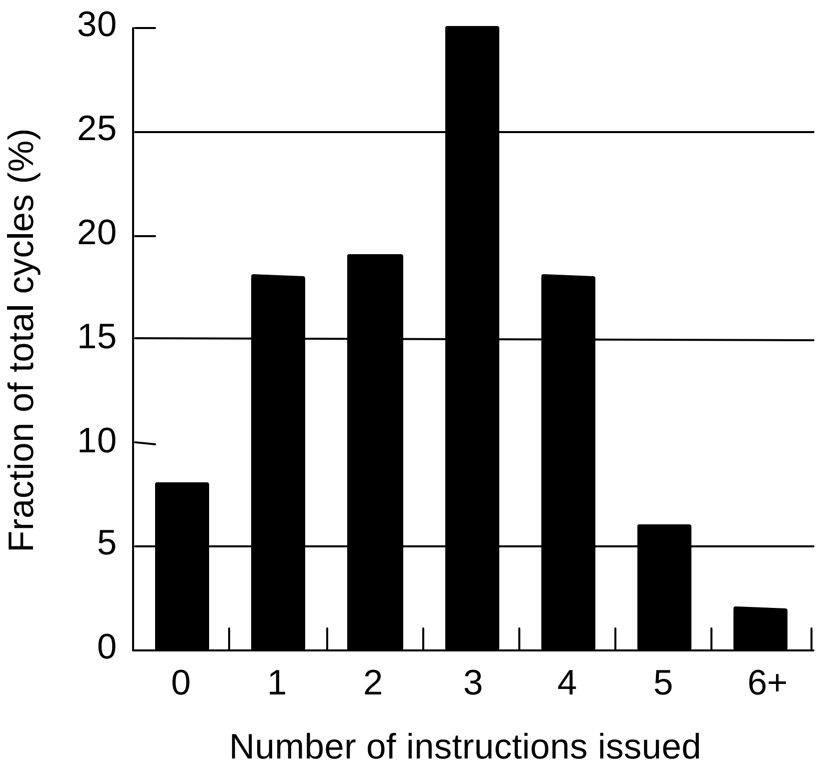
Phases in “VLSI” Generation

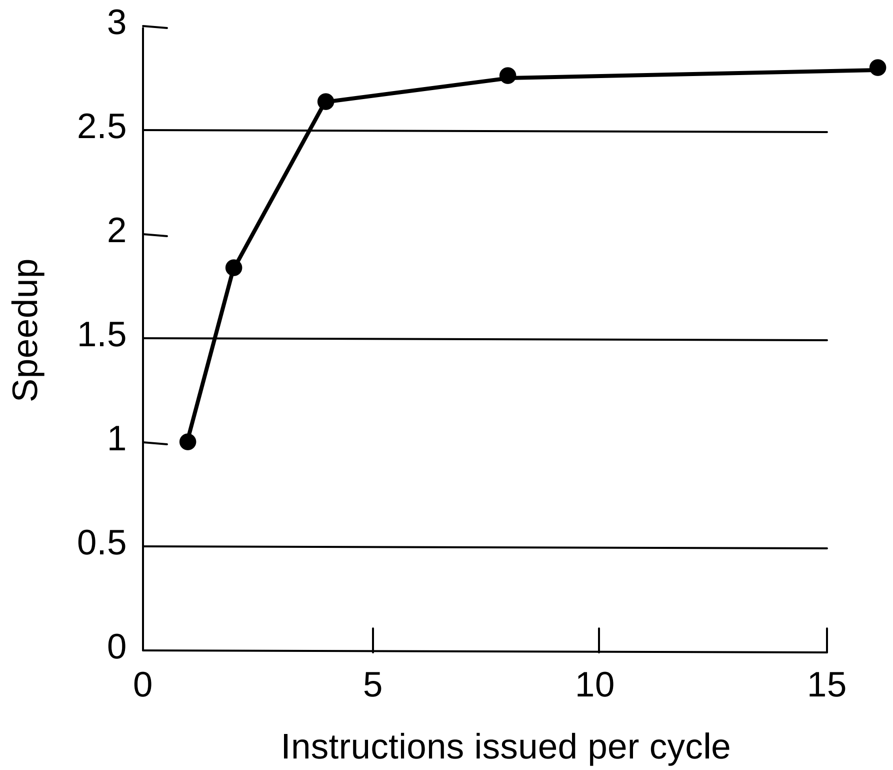


Architectural Trends

- **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**

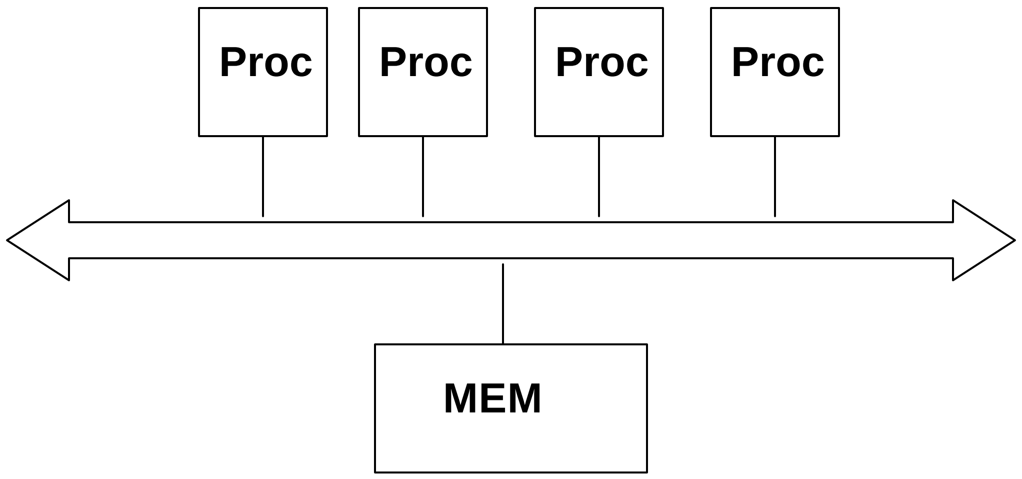
How far will ILP go?





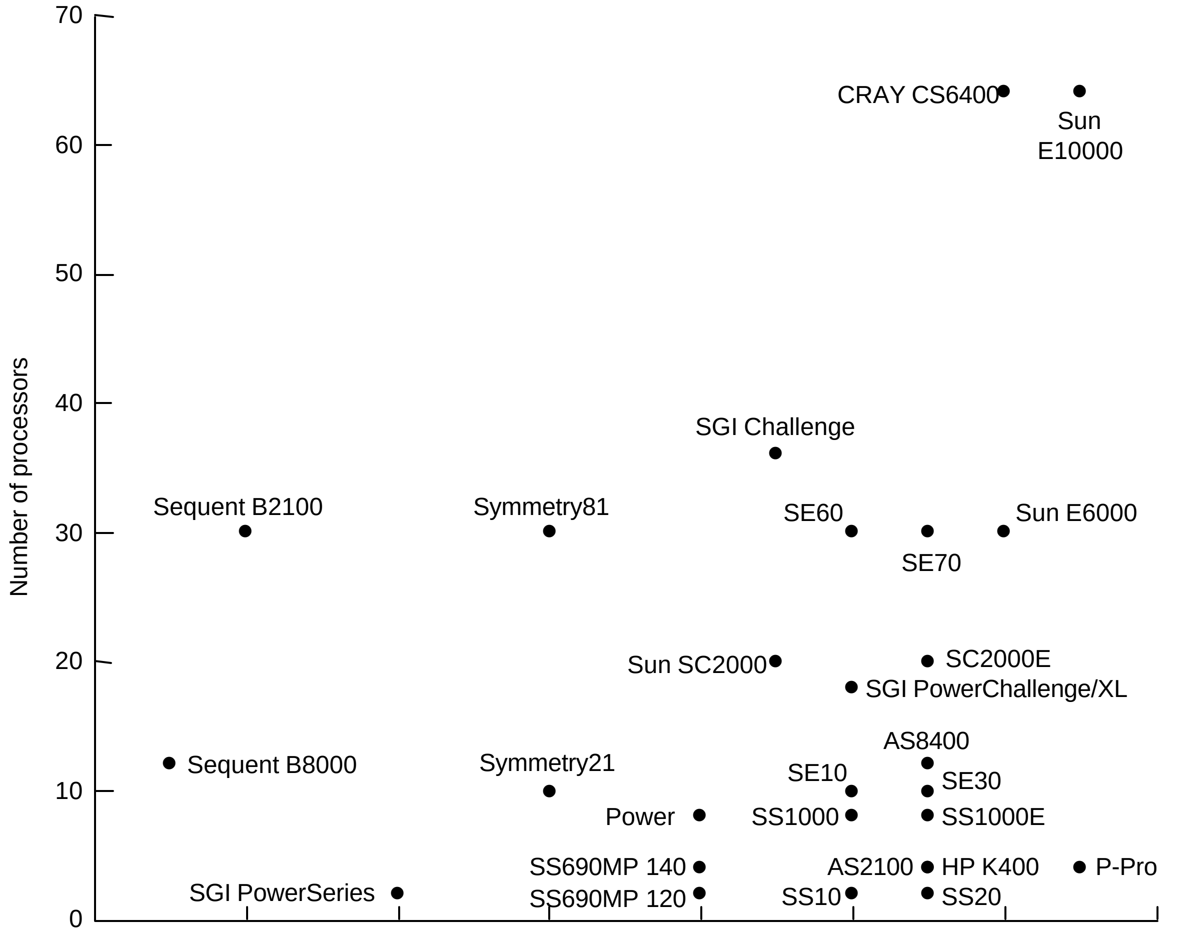
- **Infinite resources and fetch bandwidth, perfect branch prediction and renaming**
  - **real caches and non-zero miss latencies**

Threads Level Parallelism “on board”



- Micro on a chip makes it natural to connect many to shared memory
  - dominates server and enterprise market, moving down to desktop
- Faster processors began to saturate bus, then bus technology advanced
  - today, range of sizes for bus-based systems, desktop to large servers

What about Multiprocessor Trends?



What about Storage Trends?

- Divergence between memory capacity and speed even 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

Economics

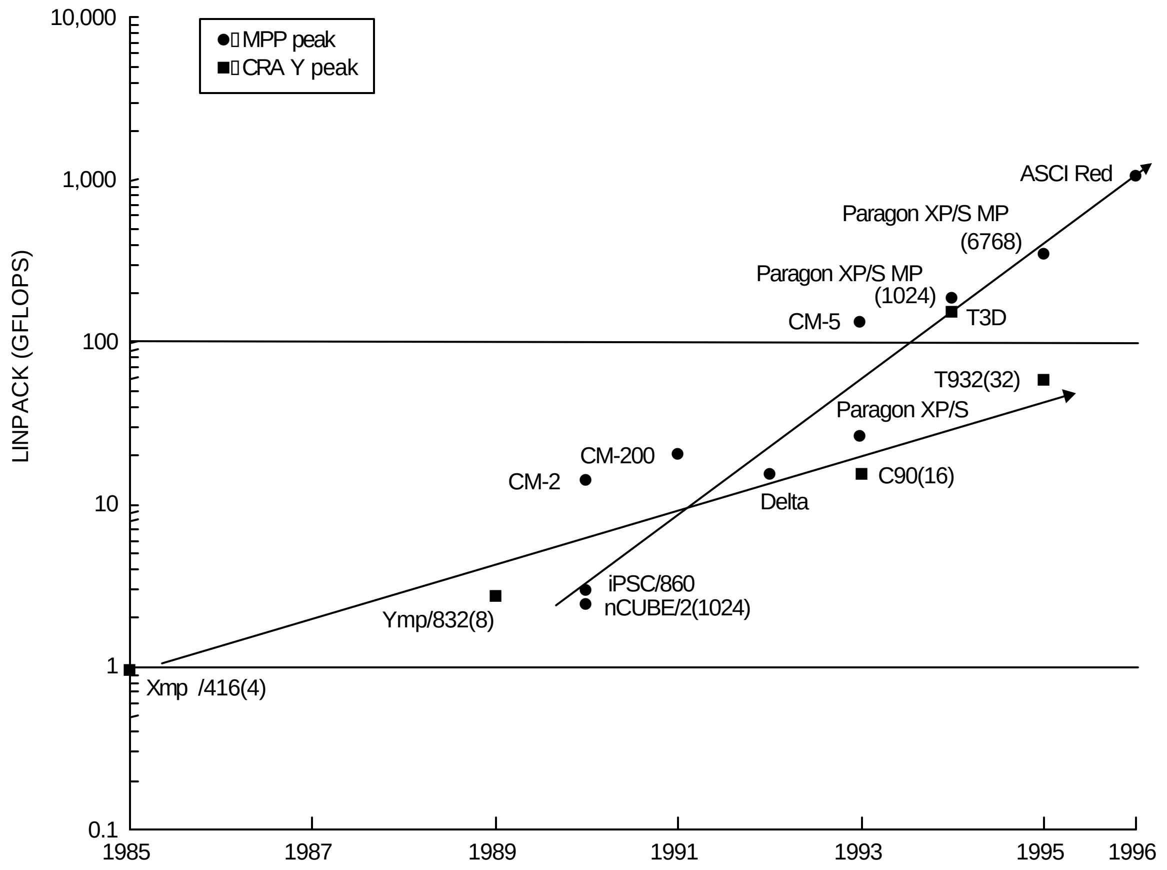
- Commodity microprocessors not only fast but CHEAP
  - Development costs tens of millions of dollars
  - BUT, many more are sold compared to supercomputers
  - Crucial to take advantage of the investment, and use the commodity building block
- Multiprocessors being pushed by software vendors (e.g. database) as well as hardware vendors
- Standardization makes small, bus-based SMPs commodity
- Desktop: few smaller processors versus one larger one?
- Multiprocessor on a chip?

Consider Scientific Supercomputing

- **Proving ground and driver for innovative architecture and techniques**
  - **Market smaller relative to commercial as MPs become mainstream**
  - **Dominated by vector machines starting in 70s**
  - **Microprocessors have made huge gains in floating-point performance**
    - **high clock rates**
    - **pipelined floating point units (e.g., multiply-add every cycle)**
    - **instruction-level parallelism**
    - **effective use of caches (e.g., automatic blocking)**
  - **Plus economics**

- **Large-scale multiprocessors replace vector supercomputers**

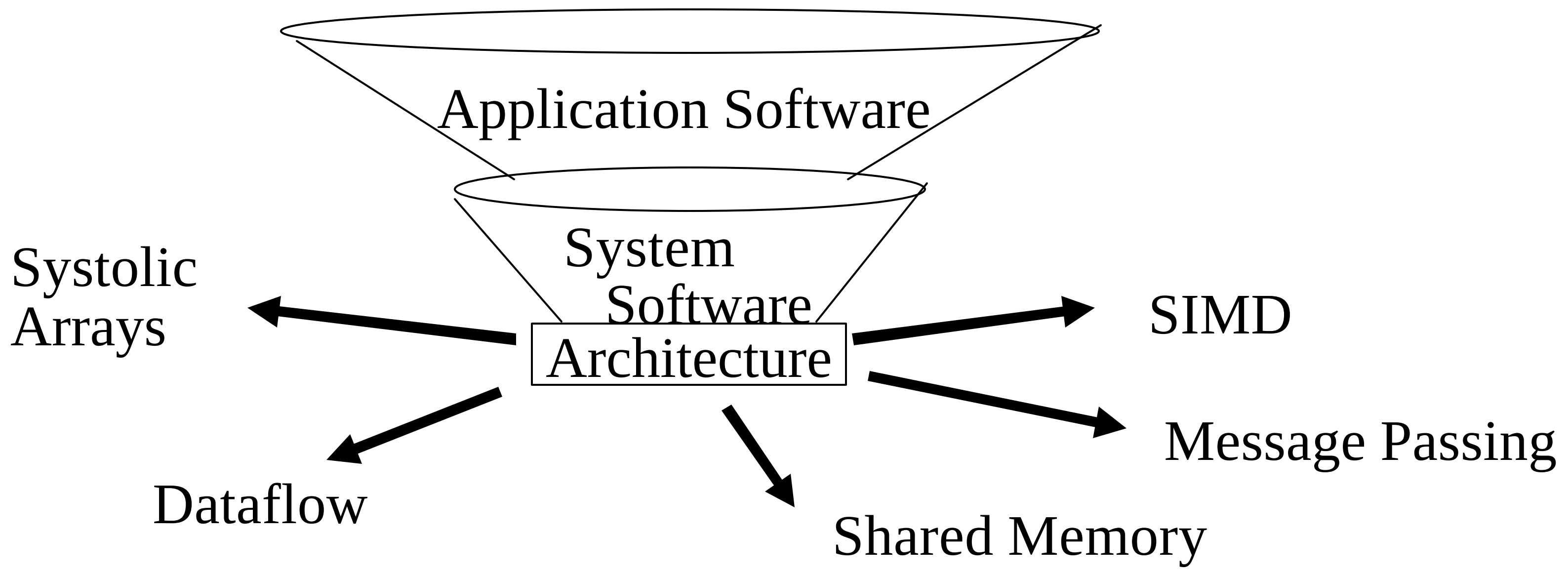
Raw Parallel Performance: LINPACK



- Even vector Crays became parallel
  - X-MP (2-4) Y-MP (8), C-90 (16), T94 (32)
- Since 1993, Cray produces MPPs too (T3D, T3E)

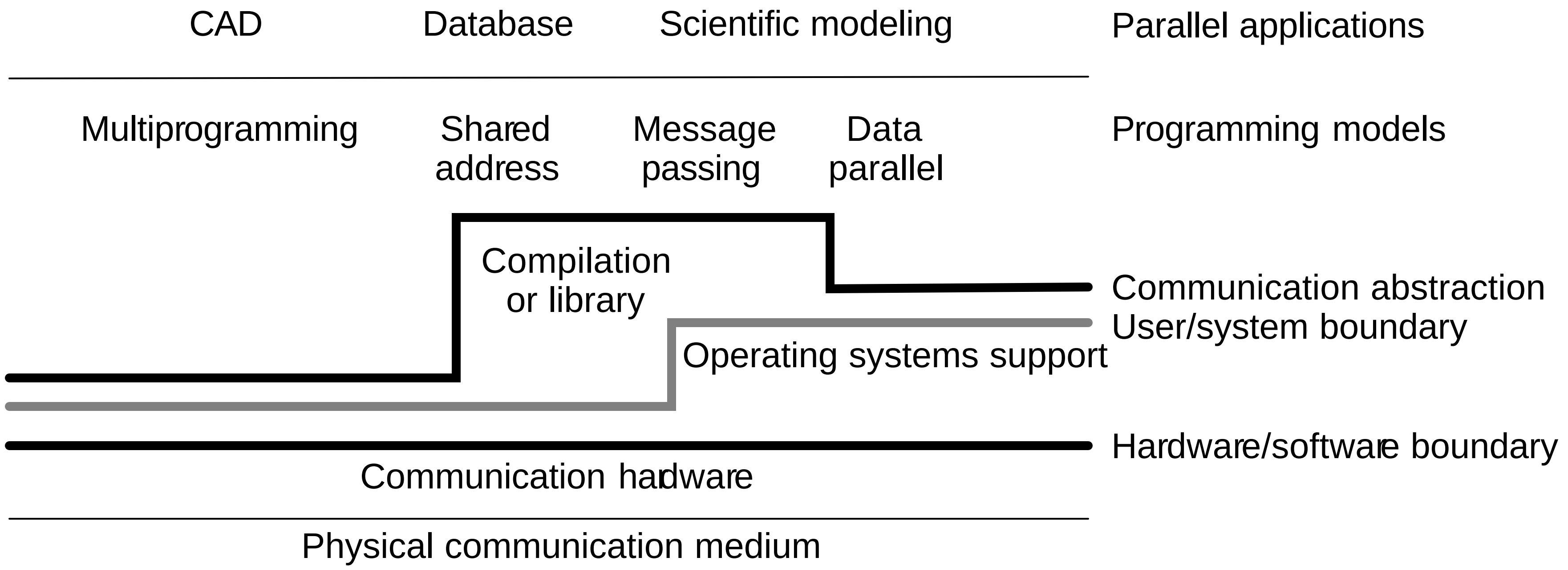
Where is Parallel Arch Going?

Old view: Divergent architectures, no predictable pattern of growth.



• Uncertainty of direction paralyzed parallel software development!

Modern Layered Framework



Summary: Why Parallel Architecture?

- Increasingly attractive
  - Economics, technology, architecture, application demand
- Increasingly central and mainstream
- Parallelism exploited at many levels
  - Instruction-level parallelism
  - Multiprocessor servers
  - Large-scale multiprocessors (“MPPs”)
- Focus of this class: multiprocessor level of parallelism
- Same story from memory system perspective
  - Increase bandwidth, reduce average latency with many local memories
- Spectrum of parallel architectures make sense
  - Different cost, performance and scalability