Parallel Computer Architecture

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 Parallel Architecture?

Role of a computer architect:

> To design and engineer the various levels of a computer system to maximize *performance* and *programmability* within limits of *technology* and *cost*.

Parallelism:

- Provides alternative to fasterclock for perfomance
- Applies at all levels of system design
- Is a fascinating perspective from which to view architecture
- Is increasingly central in information processing

Why Study it Today?

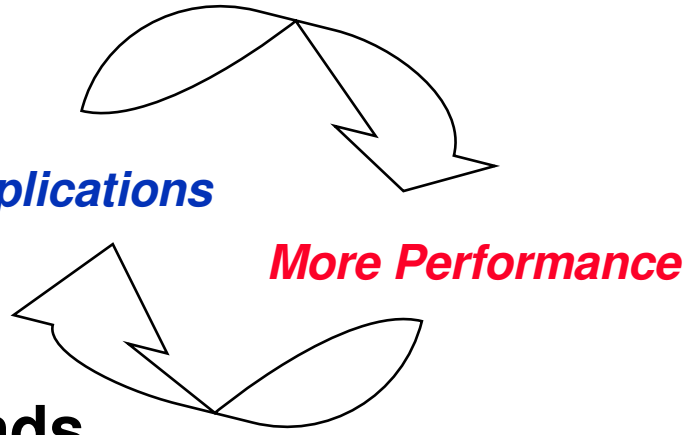
- History: diverse and innovative organizational structures, often tied to novel programming models
- Rapidly maturing under strong technological constraints
  - The “killer micro” is ubiquitous
  - Laptops and supercomputers are fundamentally similar!
  - Technological trends cause diverse approaches to converge
- Technological trends make parallel computing inevitable
- Need to understand fundamental principles and design tradeoffs, not just taxonomies
  - Naming, Ordering, Replication, Communication performance

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, COMPAQ!...
  - 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**

Speedup

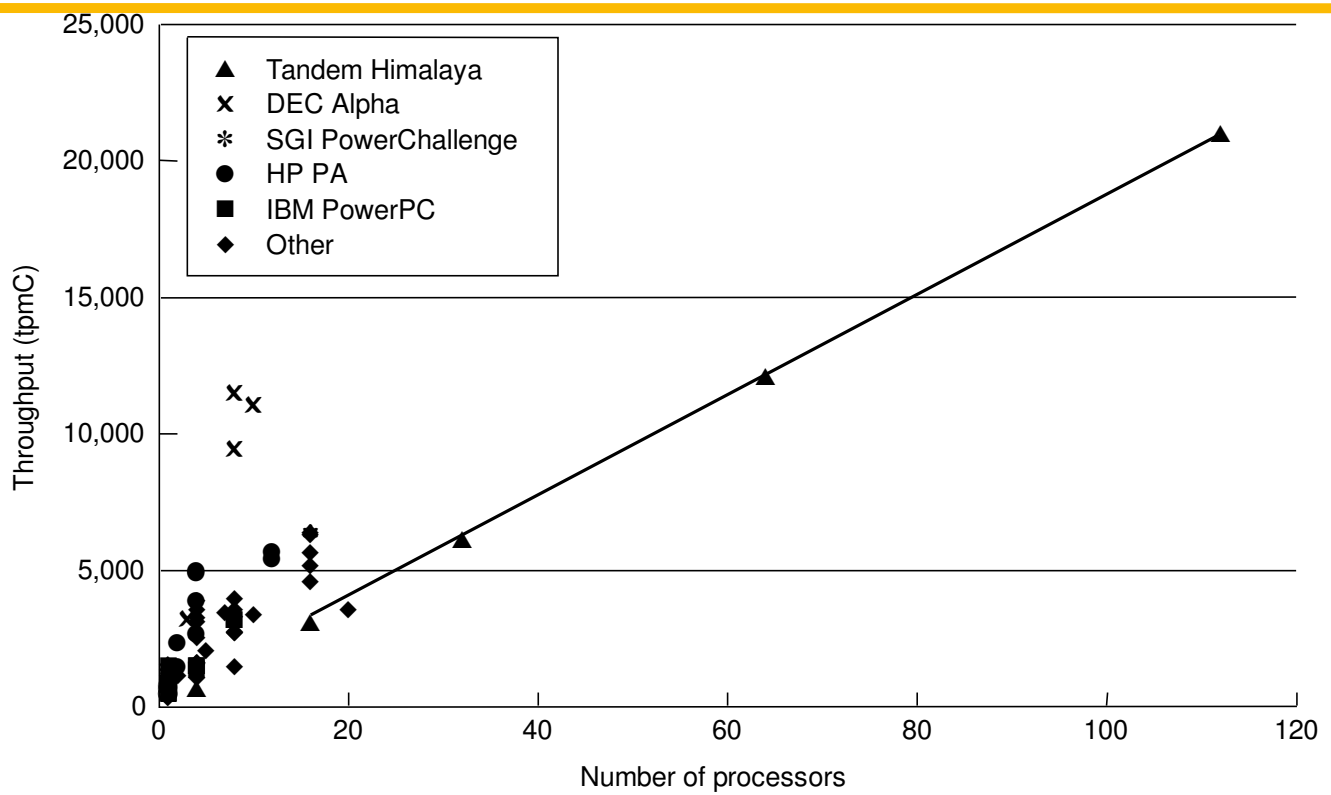
- **Speedup (p processors) =** $\dfrac{\textit{Performance (p processors)}}{\textit{Performance (1 processor)}}$
- **For a fixed problem size (input data set), performance = 1/time**
- **Speedup fixed problem (p processors) =**

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

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 (TPC-C order entry, TPC-D decision support)
  - 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**

Engineering Computing Demand

- **Large parallel machines a mainstay in many industries**
  - **Petroleum (reservoir analysis)**
  - **Automotive (crash simulation, drag analysis, combustion efficiency),**
  - **Aeronautics (airflow analysis, engine efficiency, structural mechanics, electromagnetism),**
  - **Computer-aided design**
  - **Pharmaceuticals (molecular modeling)**
  - **Visualization**
    - » **in all of the above**
    - » **entertainment (films like Toy Story)**
    - » **architecture (walk-throughs and rendering)**
  - **Financial modeling (yield and derivative analysis)**
  - **etc.**

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

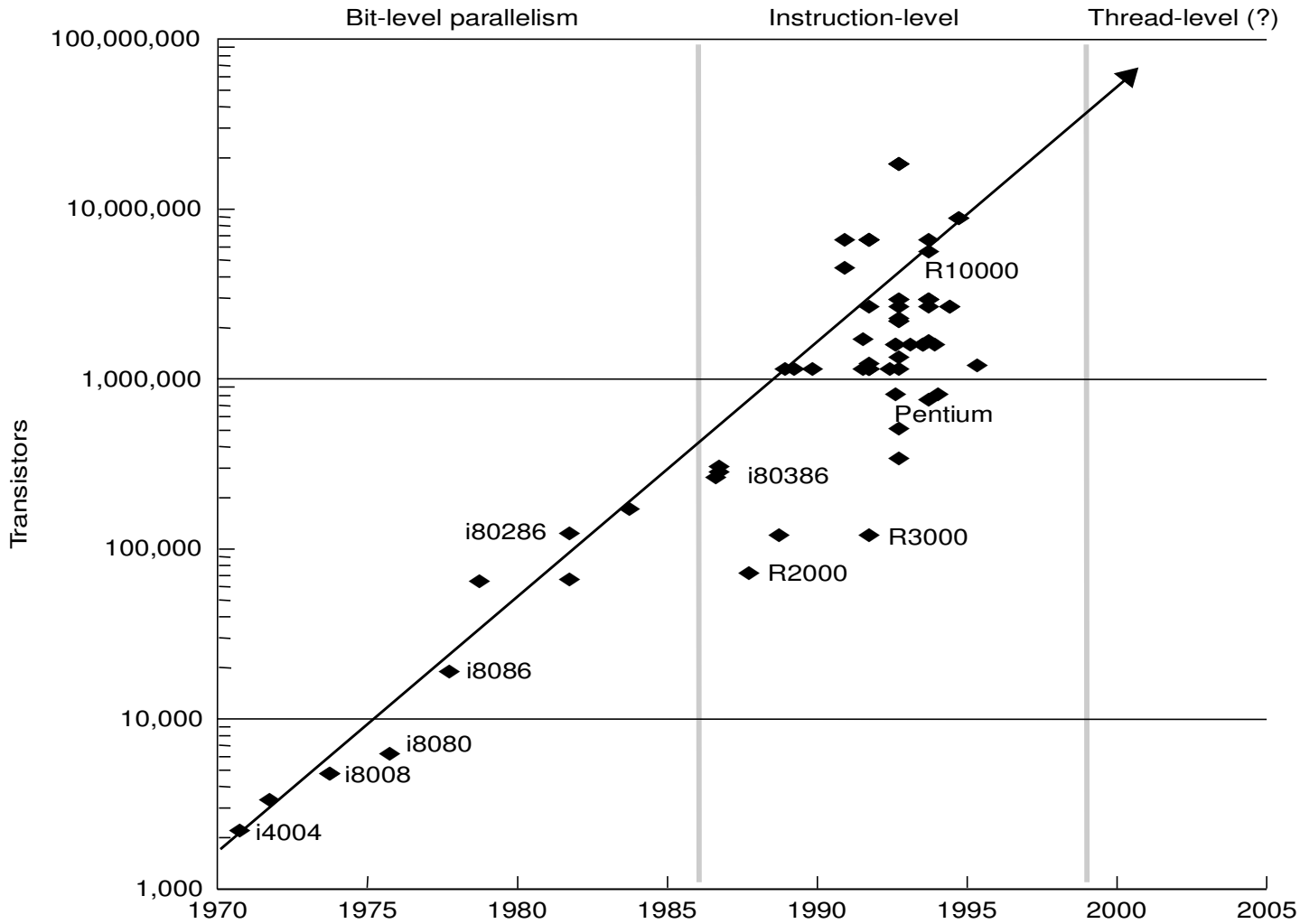
Architectural Trends

- **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

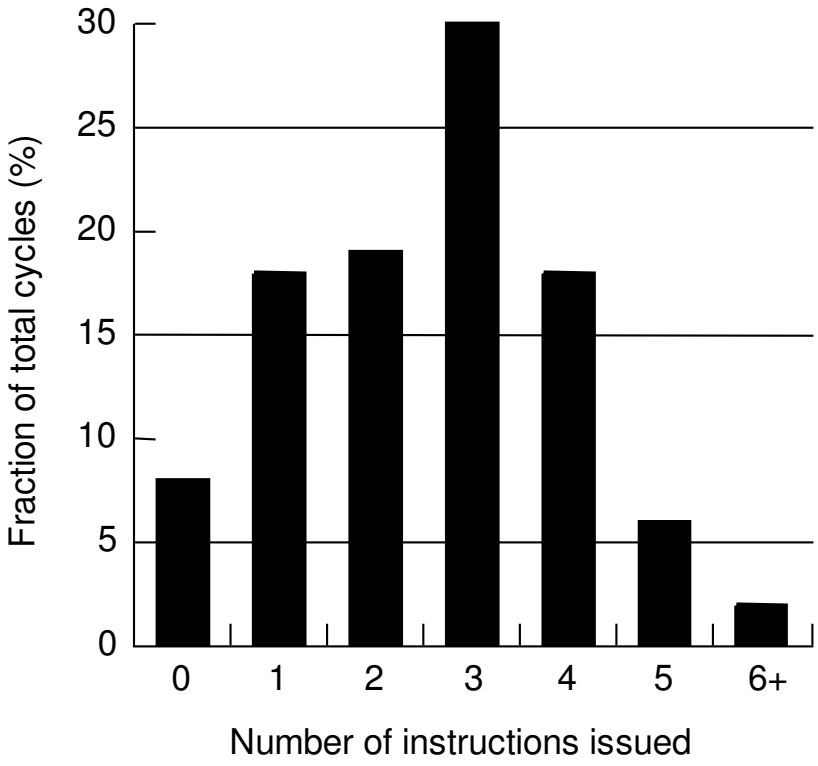
Phases in "VLSI" Generation

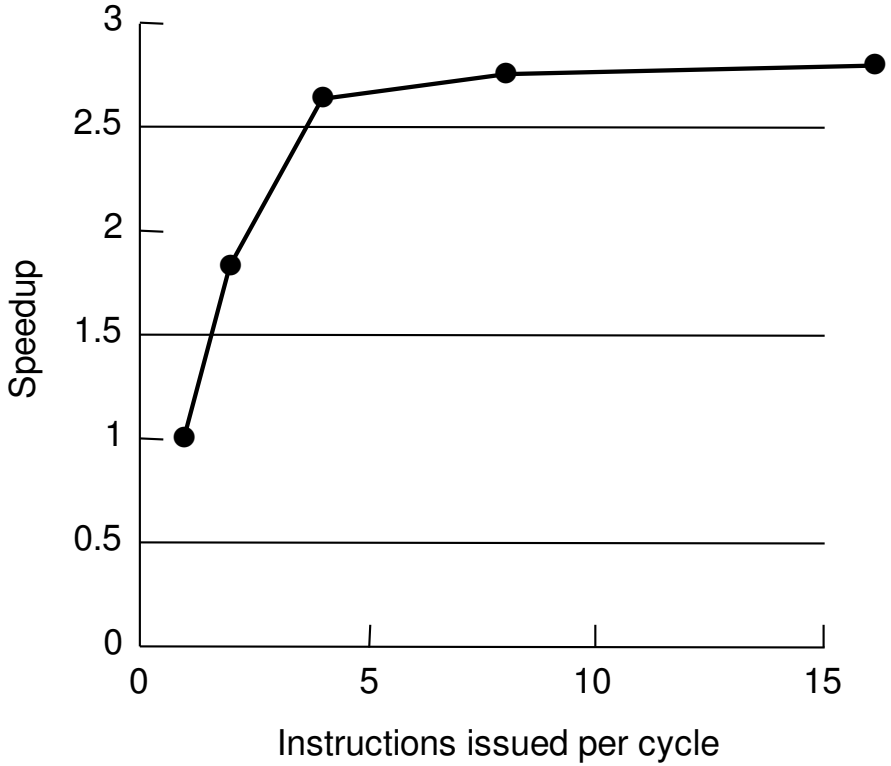


Architectural Trends

- **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**