Programming for Performance

Spring 2022

Suresh Purini

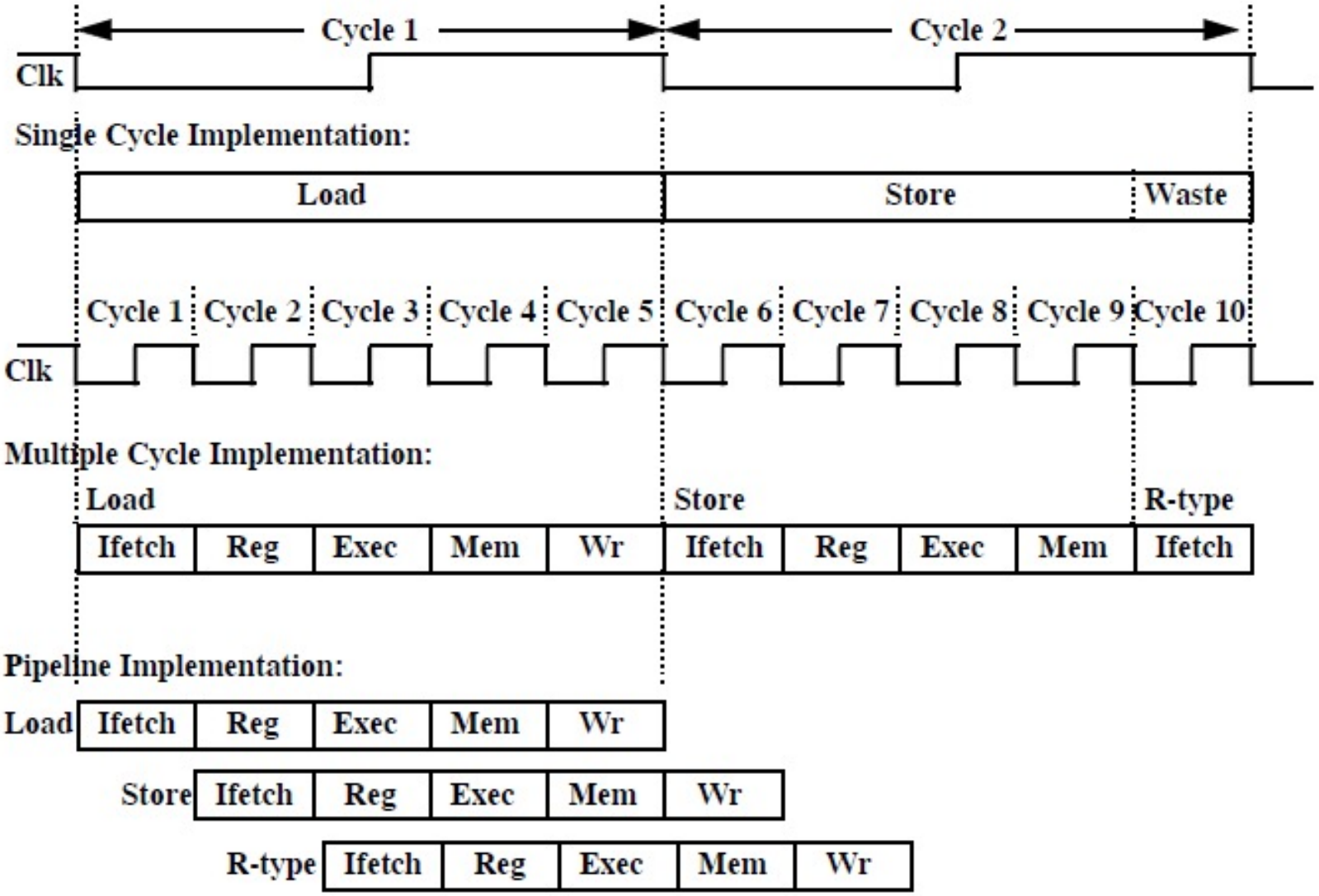
Computer Systems Group, IIIT , Hyderabad

Evolution of Processors

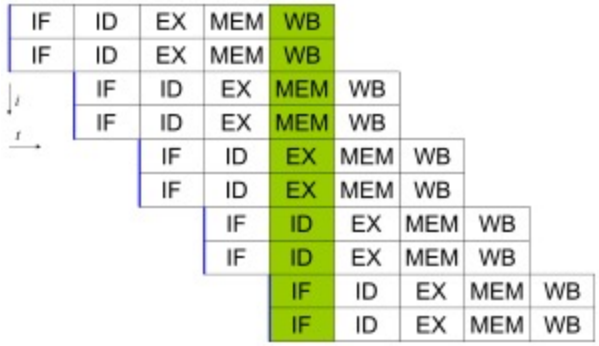
- Single Cycle
- Multi Cycle
- Pipelined
- Super Pipelined
- Super Scalar Processors
- …….

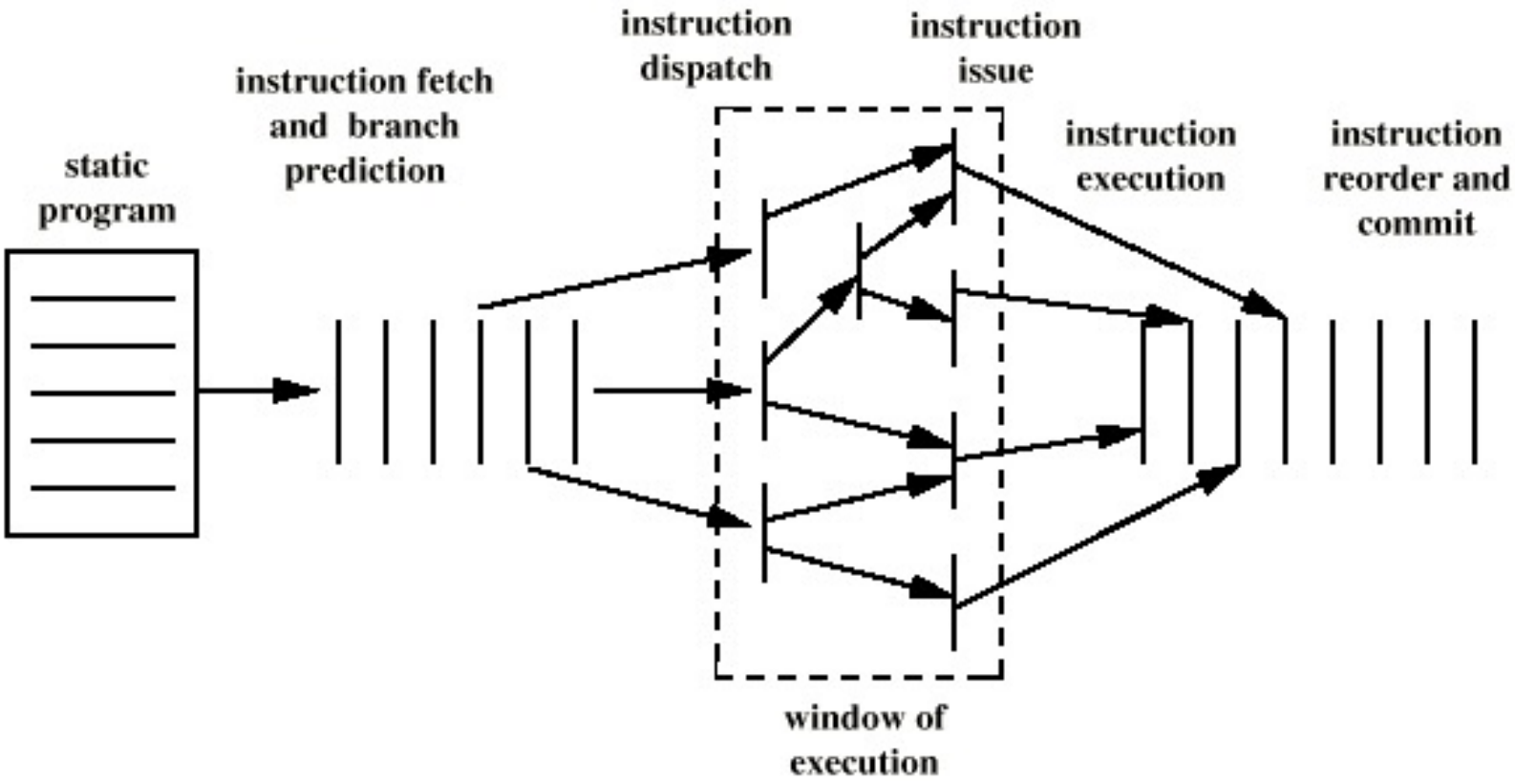
Instruction level parallelism (ILP)

Single Cycle, Multiple Cycle, Vs. Pipeline

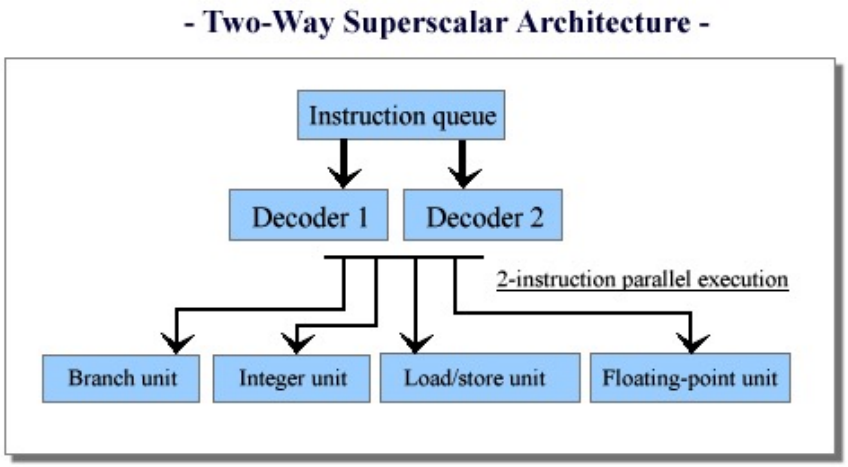


Superscalar Processors

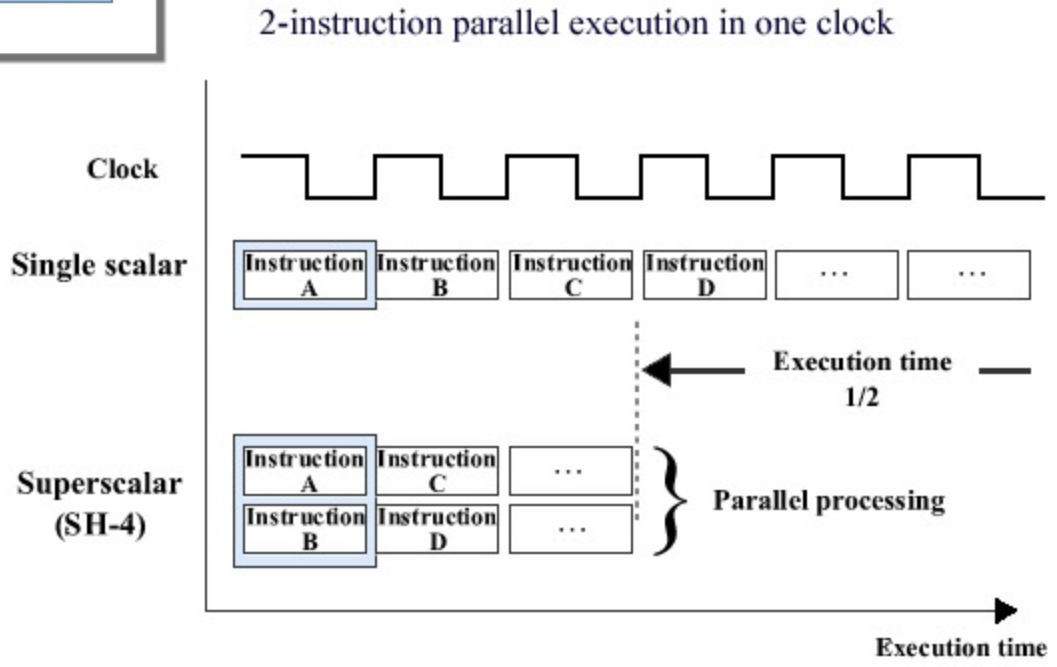




Superscalar Processors

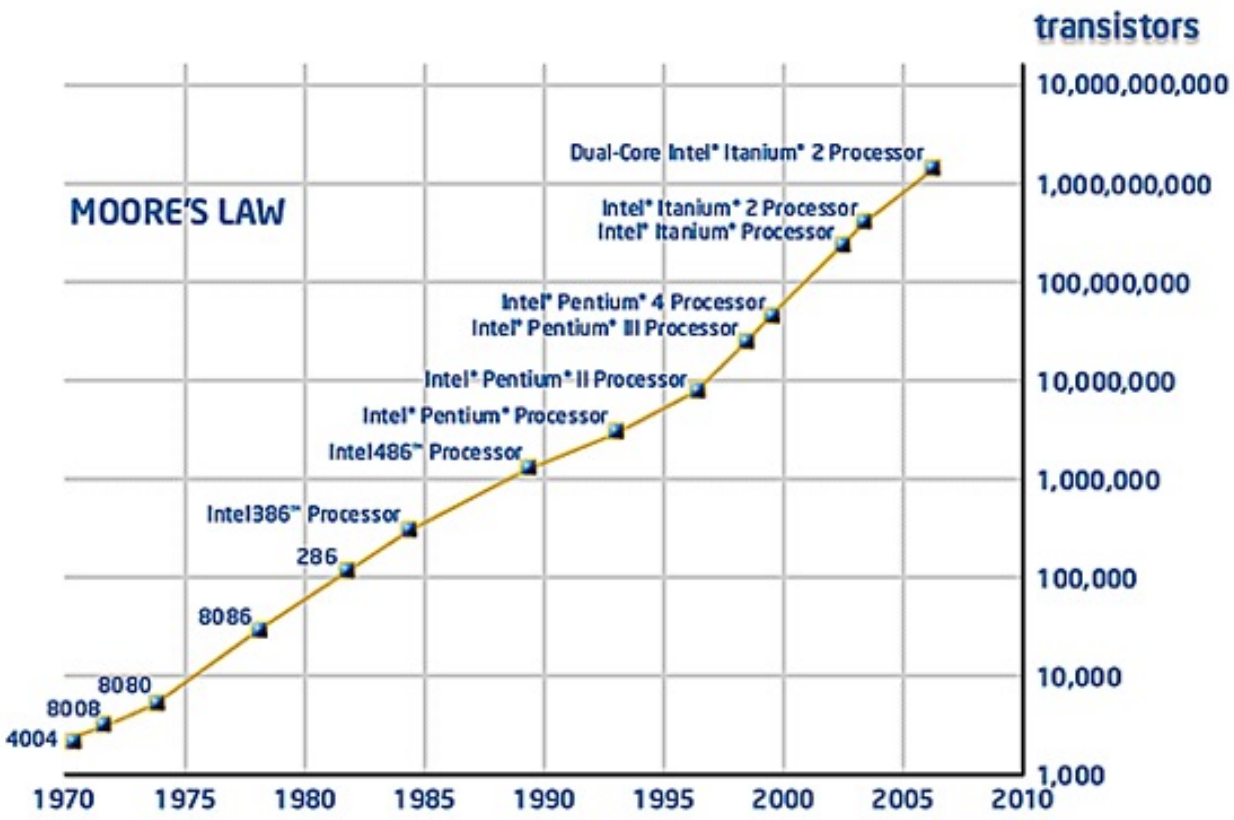


For all the fancy features of superscalar processors where do we get the chip real estate?



Moore's law (1965): The number of transistors that can be placed inexpensively on an integrated circuit doubles every two years.

- Gordon Moore, Intel co-founder



Moore's Law and Processor Performance

Question: Why increased transistor density should translate to improved processor performance?

- **Increased transistor density means scope for more complex Micro Architecture for processors.**
- **Processor performance kept improving by exploiting Instruction Level Parallelism in programs in Superscalar processors**
  - Multiple Instruction Issue
  - Register renaming to eliminate false data dependencies
  - Deeper pipelines and value forwarding between pipeline stages
  - Out of order instruction execution
  - Logic for precise exceptions

ILP Wall

You can only extract the available (and hidden) ILP parallelism

but not more than that.

David W. Wall. 1991. Limits of instruction-level parallelism. *SIGARCH Comput. Archit. News*19, 2 (April 1991), 176-188.

Clock Speed and Processor Performance

Question: Does a 2x increase in Clock Speed imply 2x improvement in a program performance?

- Not really! It depends on the amount of work done in each clock cycle.

Nevertheless, an increase in Clock Speed usually gives improved application performance.

Why can't we keep increasing the Clock Speed for Higher Performance?

- Amount of logic work that can be done in each clock cycle goes down.
- We shall hit a Power Wall! (Power is directly proportional to V^2f)
- Number of CPU cycles to access memory also increases. Memory Wall!

Clock Speed and Processor Performance

Why can't we keep increasing the Clock Speed for Higher Performance?

- **Amount of logic work that can be done in each clock cycle goes down.**
- **Distance a signal can travel gets curtailed.**
- **We shall hit a Power Wall!**
  - $P \propto V^2 f$
  - $V \propto f$
  - $P \propto f^3$

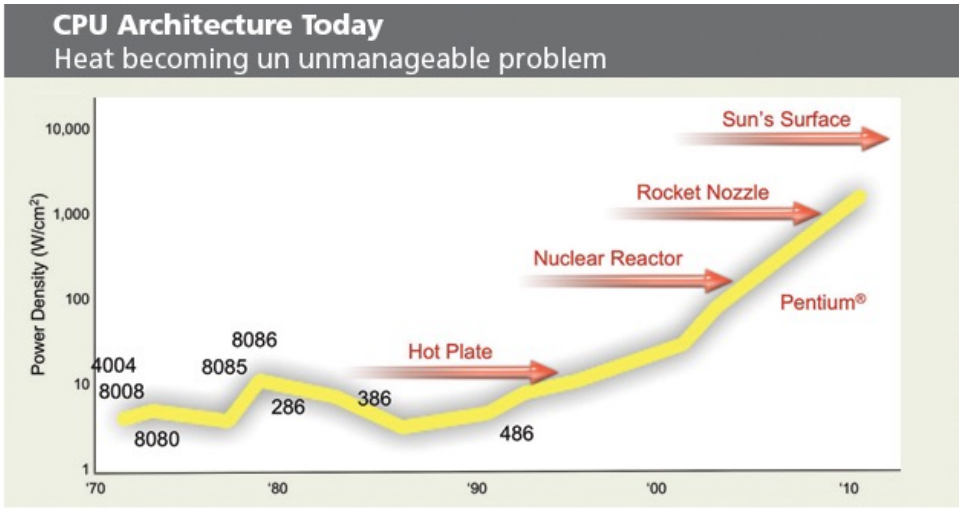
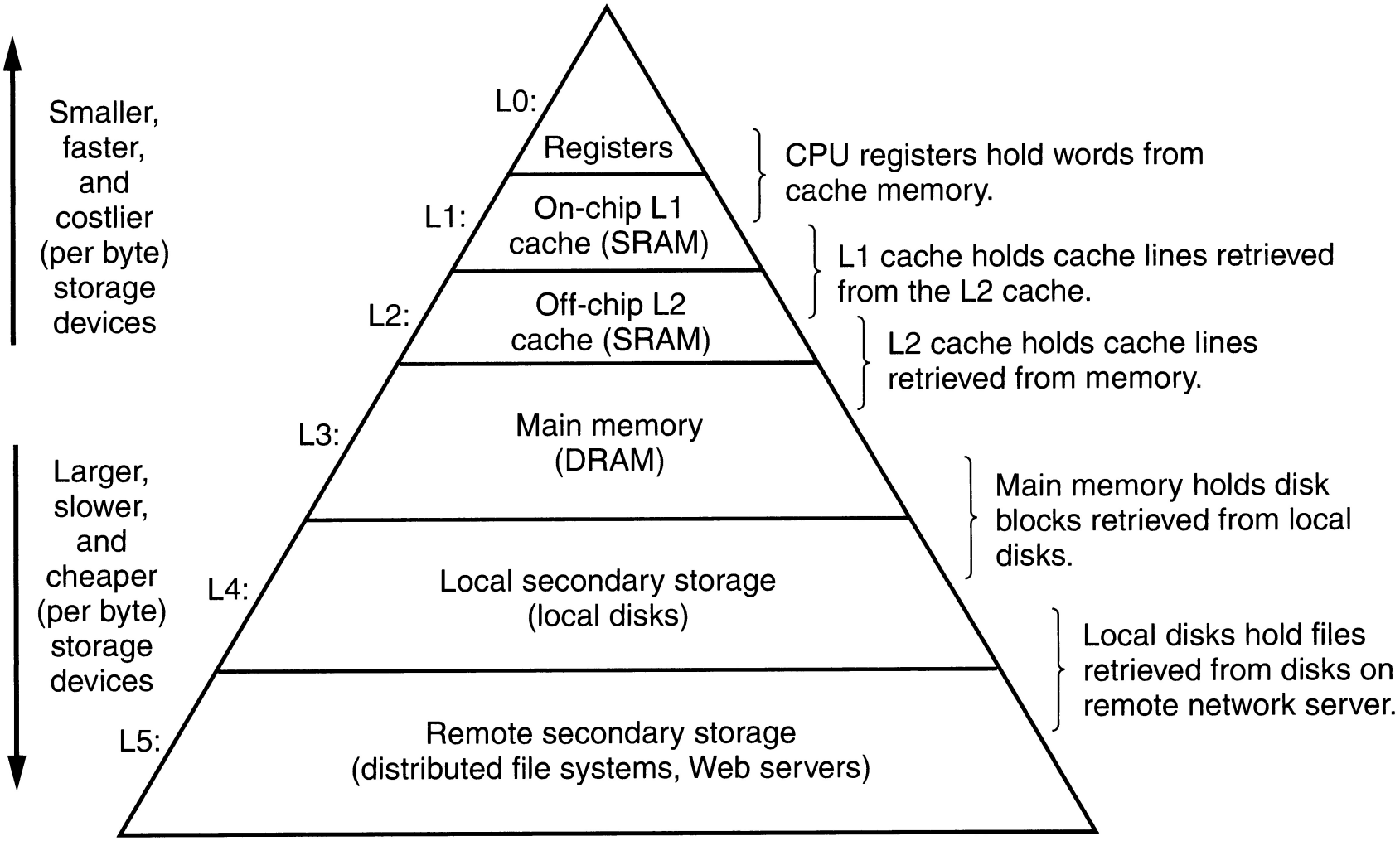


Figure 1. In CPU architecture today, heat is becoming an unmanageable problem. (Courtesy of Pat Gelsinger, Intel Developer Forum, Spring 2004)

- **Number of CPU cycles to access memory also increases. Memory Wall!**

Memory Hierarchy



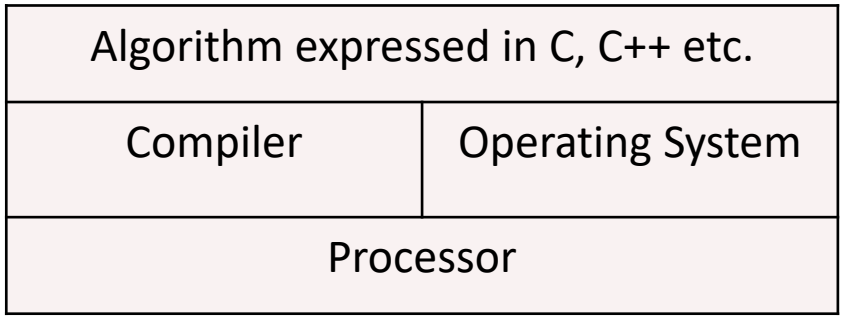
Moore's again rescues us by providing larger on chip caches.

Context – Early 2000

- **Saturation in processor performance**
  - measured in Millions of Instructions per Second (MIPS)

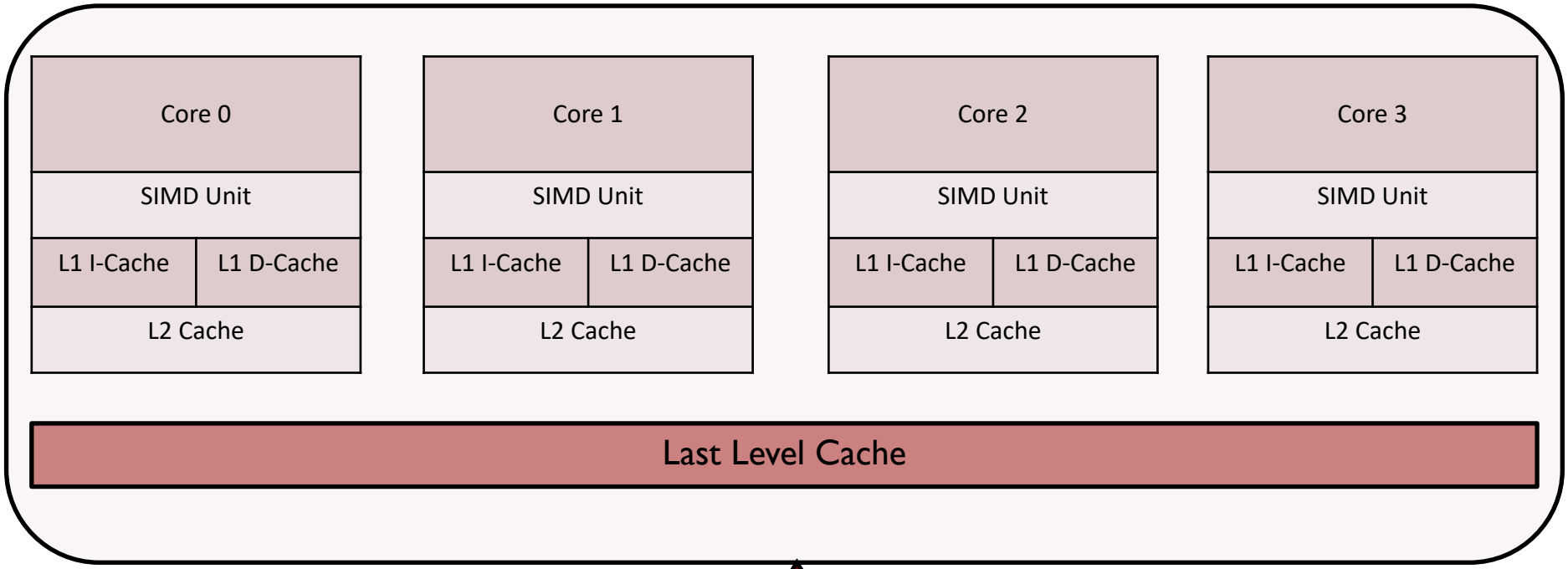
- **Reasons**
  - Instruction Level Parallelism (ILP) Wall
    - Limitations on the inherent ILP available in programs
  - Memory Wall
    - Main memory is not able to supply instructions and data to processor at the rate it can process
  - Power Wall
    - Cannot increase a processor's operational clock frequency just like that

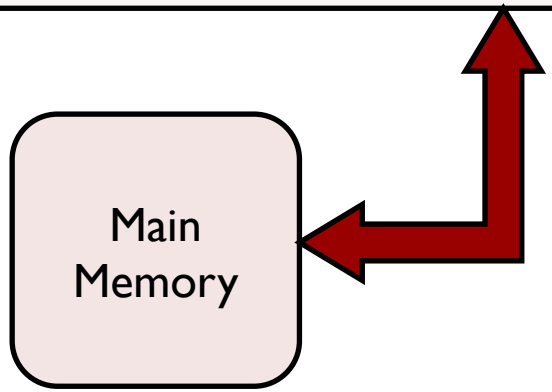
Till Early 2000: Optimizing Application Performance



- **Processor Designer:** Better Microarchitecture
- **Compiler Writer:** Better Optimization Passes, Instruction Scheduling and Register Allocation Algorithms
- **Programmer:** Better Algorithms

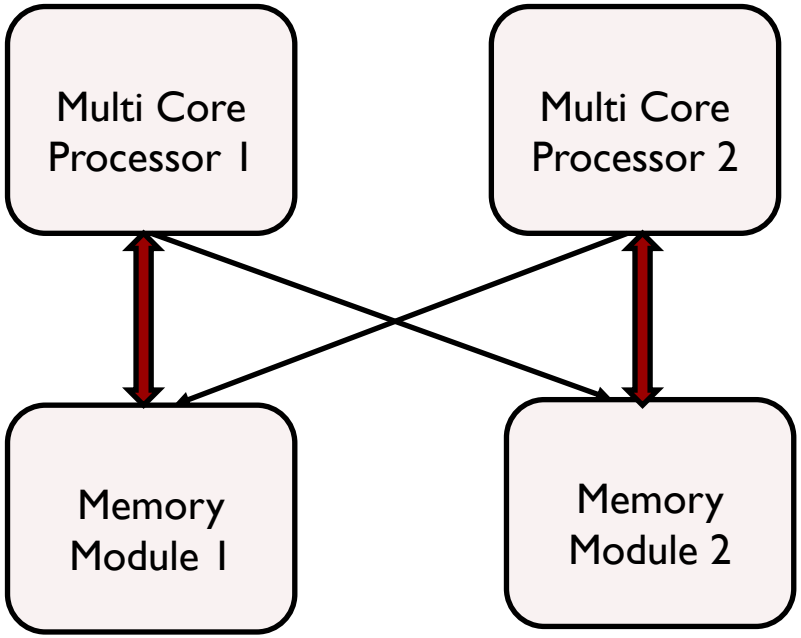
Modern Computing Systems – Late 2000 Onwards





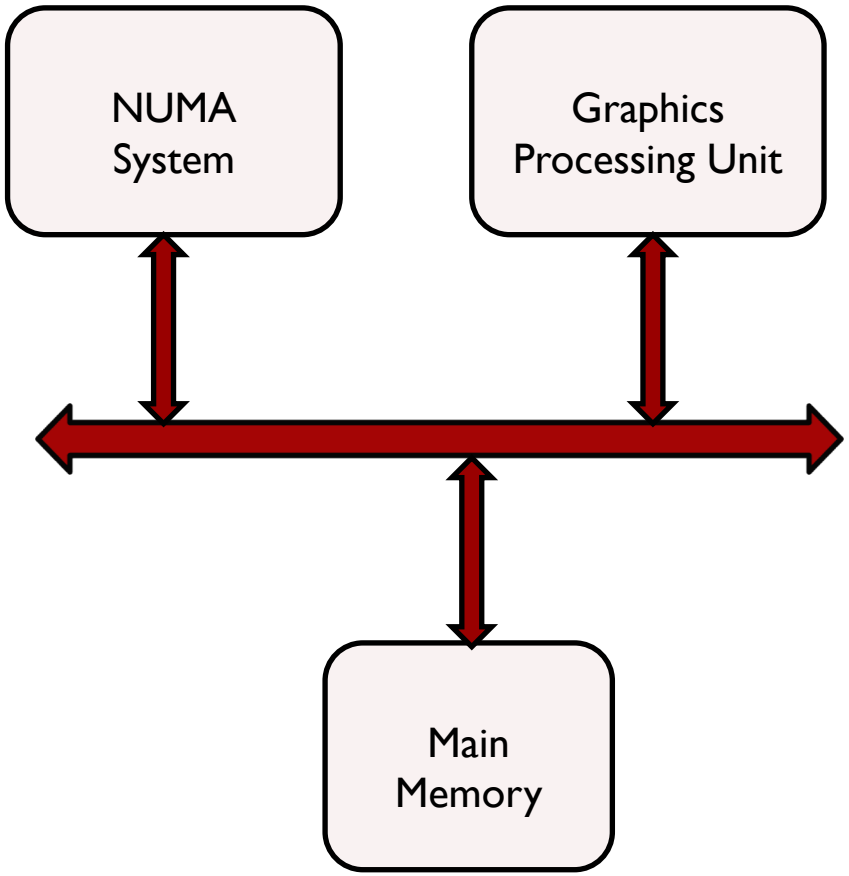
Dimensions for Optimization	
Code Vectorization	Memory Locality
Thread Level Parallelism	Cache Interference
Algorithms	

MCS – Late 2000 Onwards



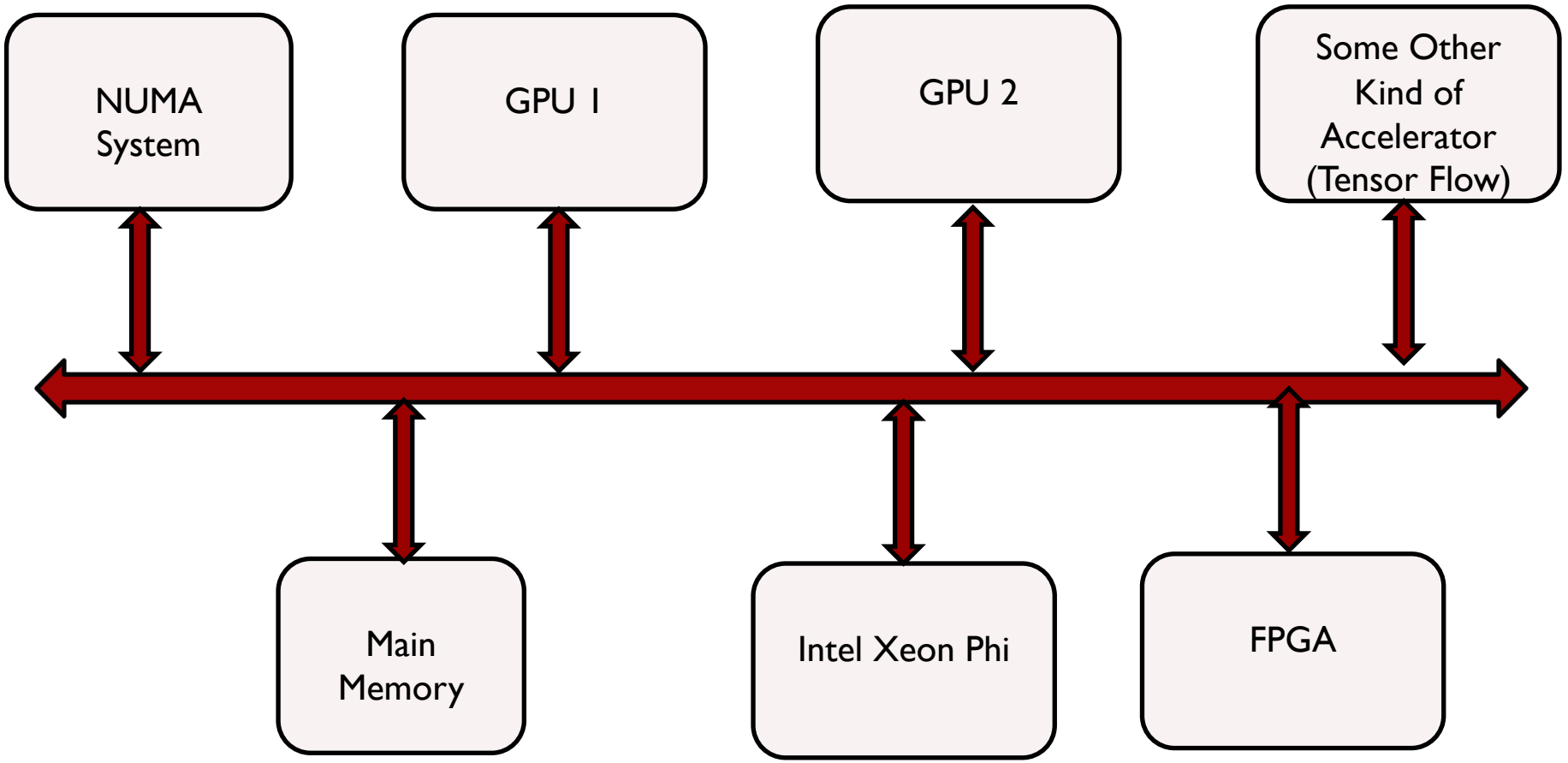
Dimensions for Optimization	
Code Vectorization	Memory Locality
Thread Level Parallelism	Cache Interference
Data Distribution (NUMA)	Thread Scheduling
Algorithms	

MCS – Late 2000 Onwards



Dimensions for Optimization	
Code Vectorization	Memory Locality
Thread Level Parallelism	Cache Interference
Data Distribution (NUMA)	Thread Scheduling
Block and Grid Dimensions	Memory Coalescing
Reduce Branch Divergence	Shared Memory Usage
Communication Latency	Limited Global Memory
Pipelined Parallelism	Load Balancing
Algorithms	

MCS – Late 2000 Onwards

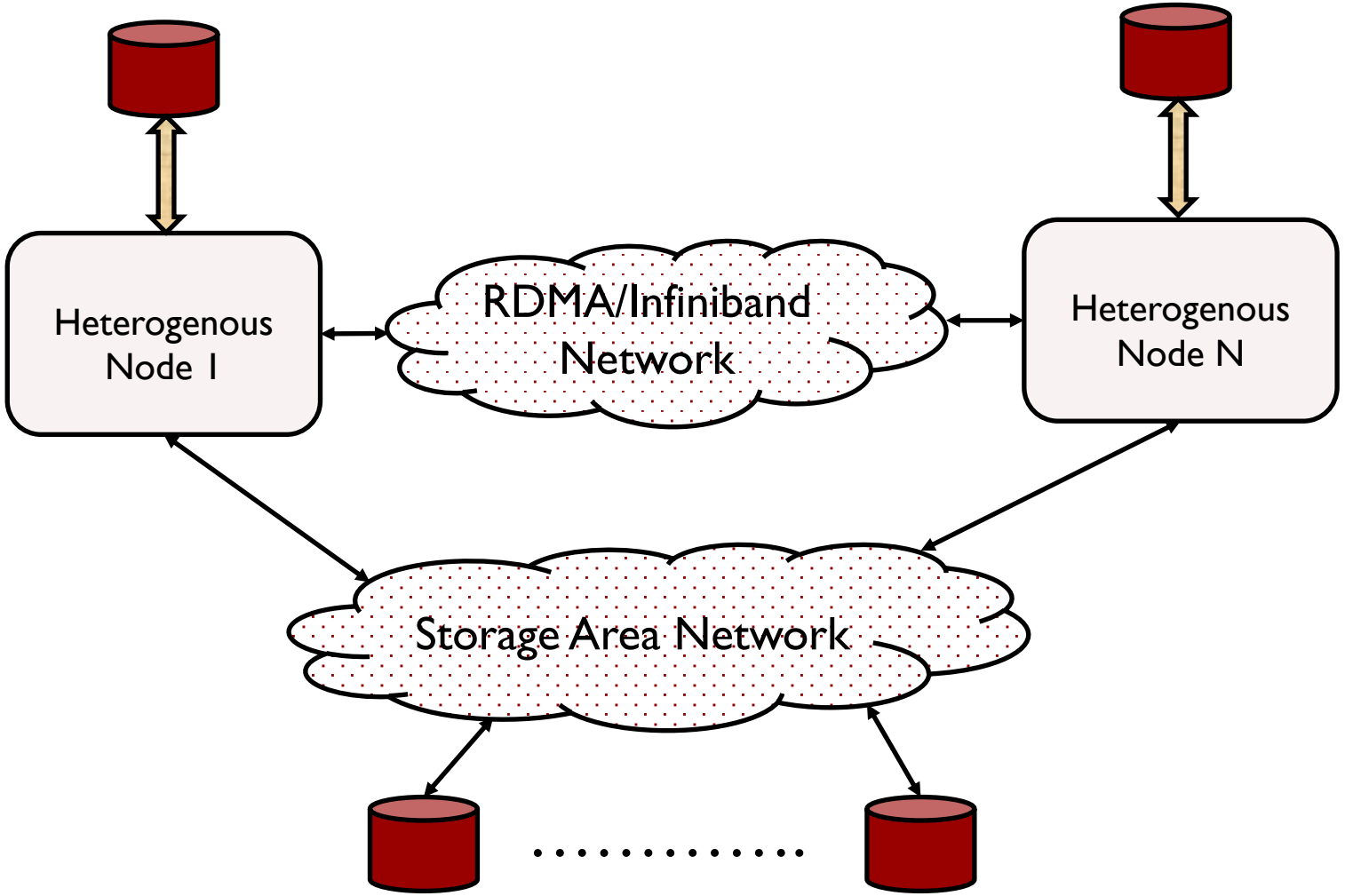


Challenges

- Beyond the single core computing structure, Von Neumann model of Sequential Computation doesn't hold.
- Every accelerator is a beast by itself and requires an expert to optimize an application for it.
- And then we would like to perform Heterogeneous Cross-accelerator Optimizations!

And the Story Doesn't End There!

Heterogeneous Collection of Heterogenous Nodes!



Varied Computational Frameworks

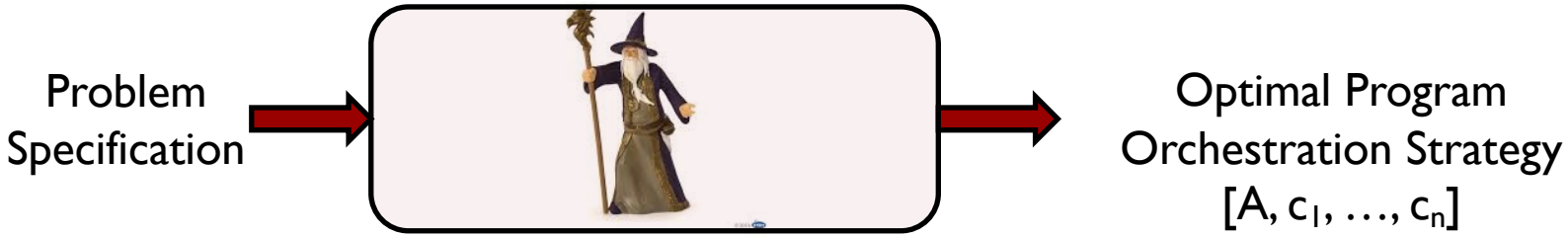
- Pthreads
- OpenMP
- Thread Building Blocks
- OpenCl
- CUDA
- OpenACC
- Remote Procedure Calls
- Message Passing MPI
- Map-Reduce
- Spark
- Pregel/Giraph (Graph based)
- Akka Actor Framework
- X11 PGAS
- ……

Problems to Solve Across the Stack

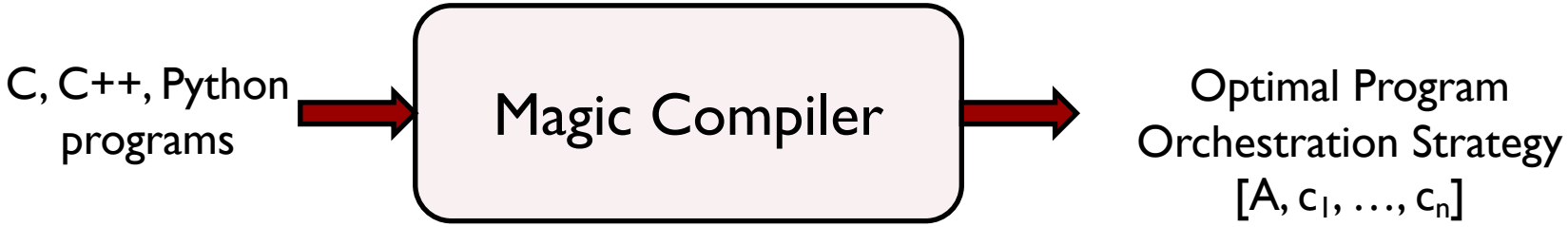
How to translate the raw computing power to application performance gains?

- **Algorithms (Parallel)**
- **Programming language abstractions**
- **Compilers**
- **Runtime systems**
- **Operating system support**
- **Architectural support**
  - Better memory consistency models
  - Cache coherence protocols
  - Synchronization primitives

Problem Statement



Ideally, we expect



Design Space Explosion

- Exponentially many program orchestration strategies
- Finding even a near-optimal orchestration strategy is equivalent to finding a needle in the haystack
- Two Approaches
  - Analytical Modelling Techniques
    - Works for well-contained problems such as Tiling strategies
    - Modern systems are too complicated to build meaningful models
  - Heuristic Search Techniques
    - Design space too large even to sample a small sub-space

Design Space Explosion

- Exponentially many program orchestration strategies
- Finding even a near-optimal orchestration strategy is equivalent to finding a needle in the haystack
- Two Approaches
  - Analytical Modelling Techniques
    - Works for well-contained problems such as Tiling strategies
    - Modern systems are too complicated to build meaningful models
  - Heuristic Search Techniques
    - Design space too large even to sample a small sub-space

THANK YOU