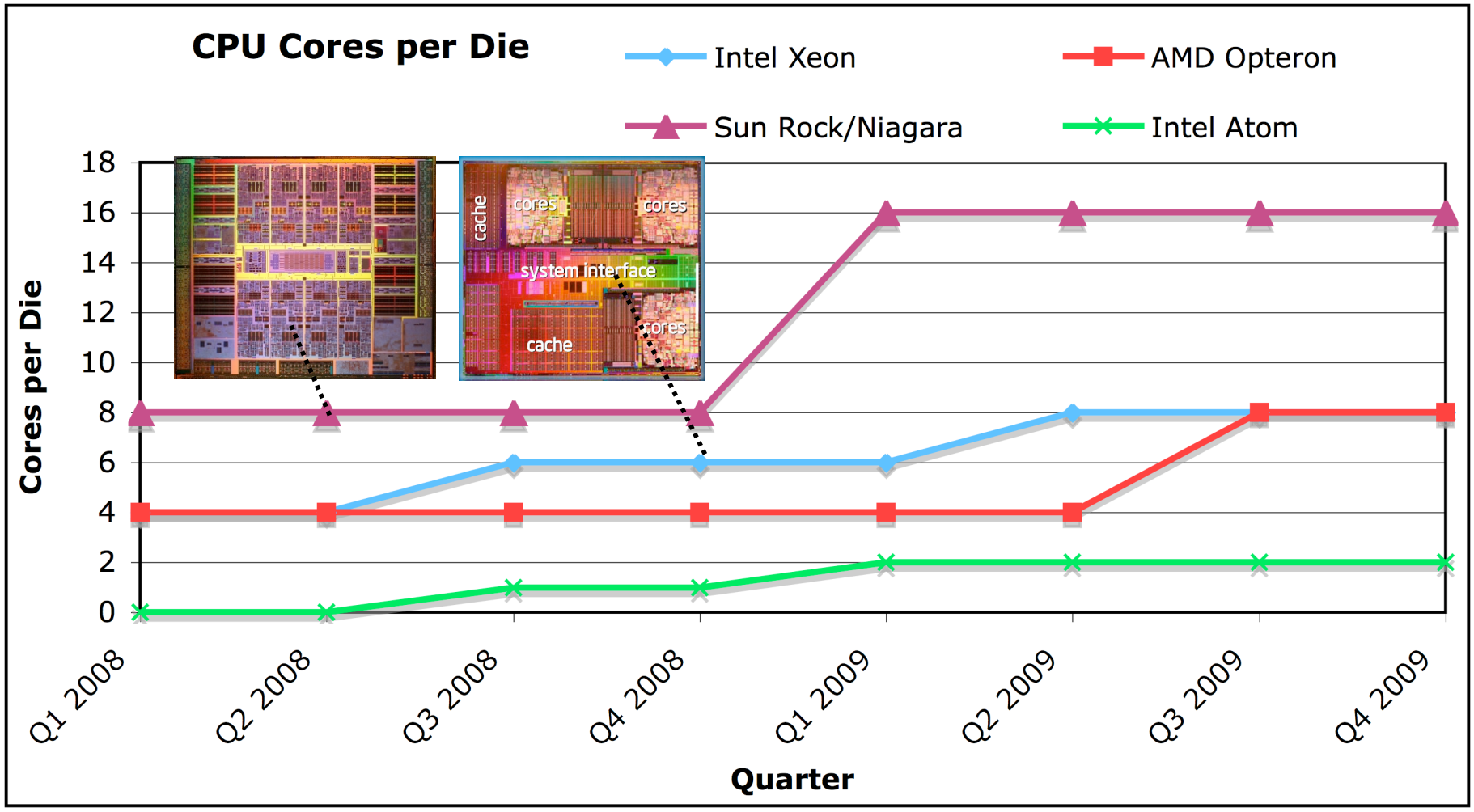
Supporting Fine-grain TLP with Selective Timesharing

John Giacomoni

Advisor: Dr. Manish Vachharajani
University of Colorado at Boulder

2008.04.26

The Rise of Multicore



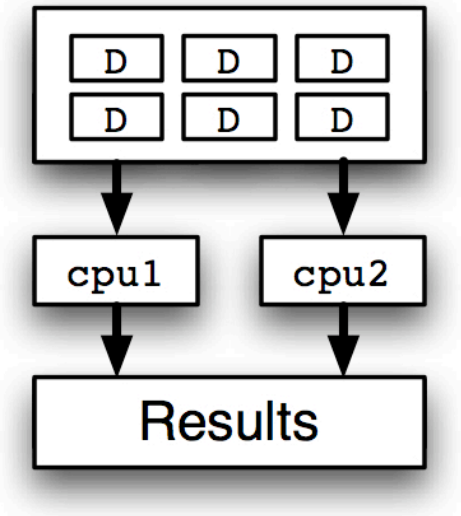
The Challenge of Multicore

- Programming
  - Finding parallelism
  - Extracting parallelism
  - Debugging
- Scalability
  - Cores scale faster than memory and I/O bandwidth
  - Low-latency synchronization for many cores
- System Design Issues
  - Isolation from interference (OS, shared resources)

Using Multi-Core

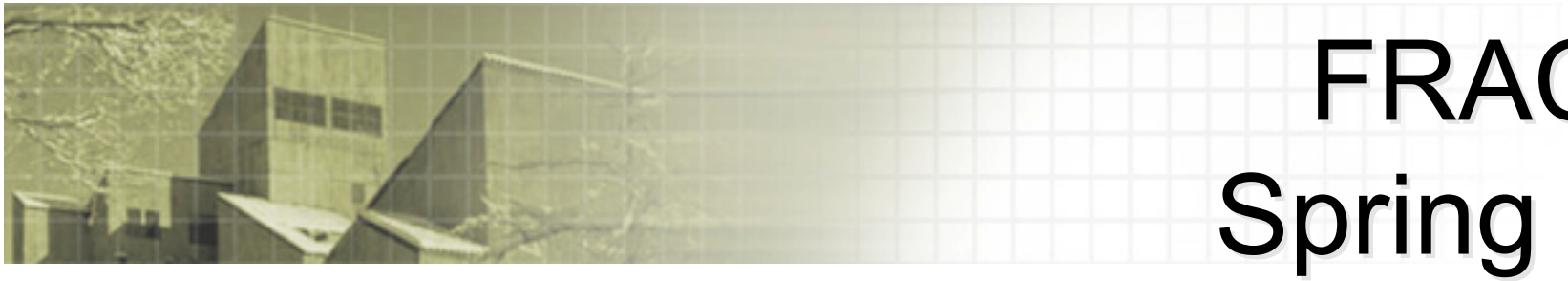
- Task Parallelism
  - Desktop - easy
- Data Parallelism
  - Web serving - “easy”

- Sequential applications
  - HARD (data dependencies)
    - Ex: Video Decoding
    - Ex: Network Processing



Problem

- Programmers are:
  - Bad at explicitly parallel programming
  - Better at sequential programming
- Need to make life easier for programmers

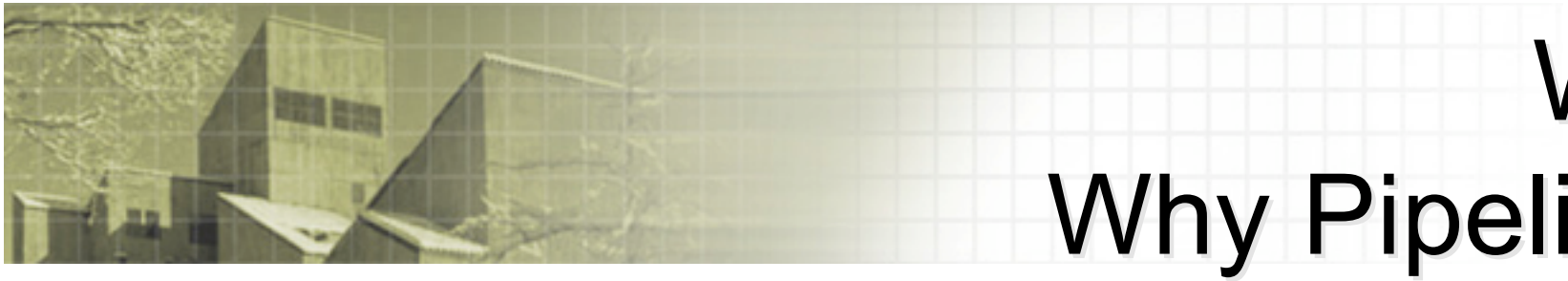


FRACTAL Spring 2007

- Development Support for Concurrent Threaded Pipelining
  - Communicating Concurrent Threads
  - Low-overhead core-to-core communication is critical
  - Need to pay attention to computer architecture
  - Need to hide computer architecture from users
  - PPoPP ‘08, ANCS ‘07

FRACTAL
Fall 2007

- Operating System Support for Fine-grain Parallelism on Multicore Architectures
  - Pipelinable System Services
  - Multi-Domain Entities
  - Gang Scheduling
    - Change Utility Function - Optimize for critical applications
    - **Want Selective Timesharing**
  - OSHMA ‘07



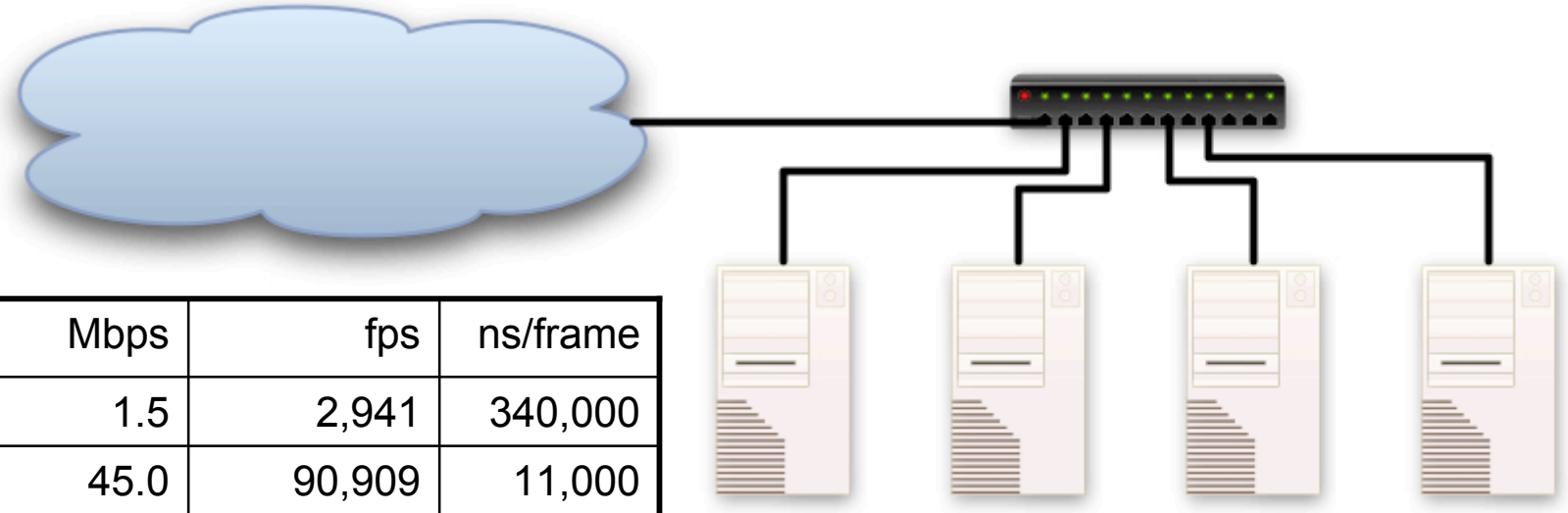
Why? Why Pipelines?

- Multicore systems are the future
- Many apps can be pipelined if the granularity is fine enough
  - ≈ < 1 µs
  - ≈ 3.5 x interrupt handler

Fine-Grain Pipelining Examples

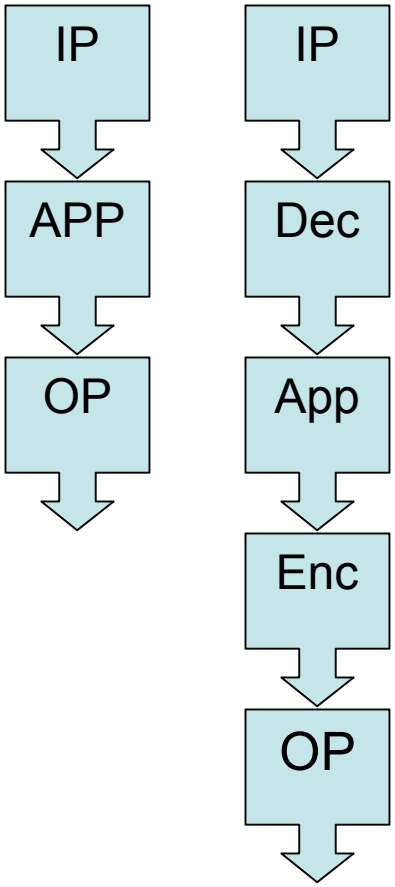
- Network processing:
  - Intrusion detection (NID)
  - Traffic filtering (e.g., P2P filtering)
  - Traffic shaping (e.g., packet prioritization)

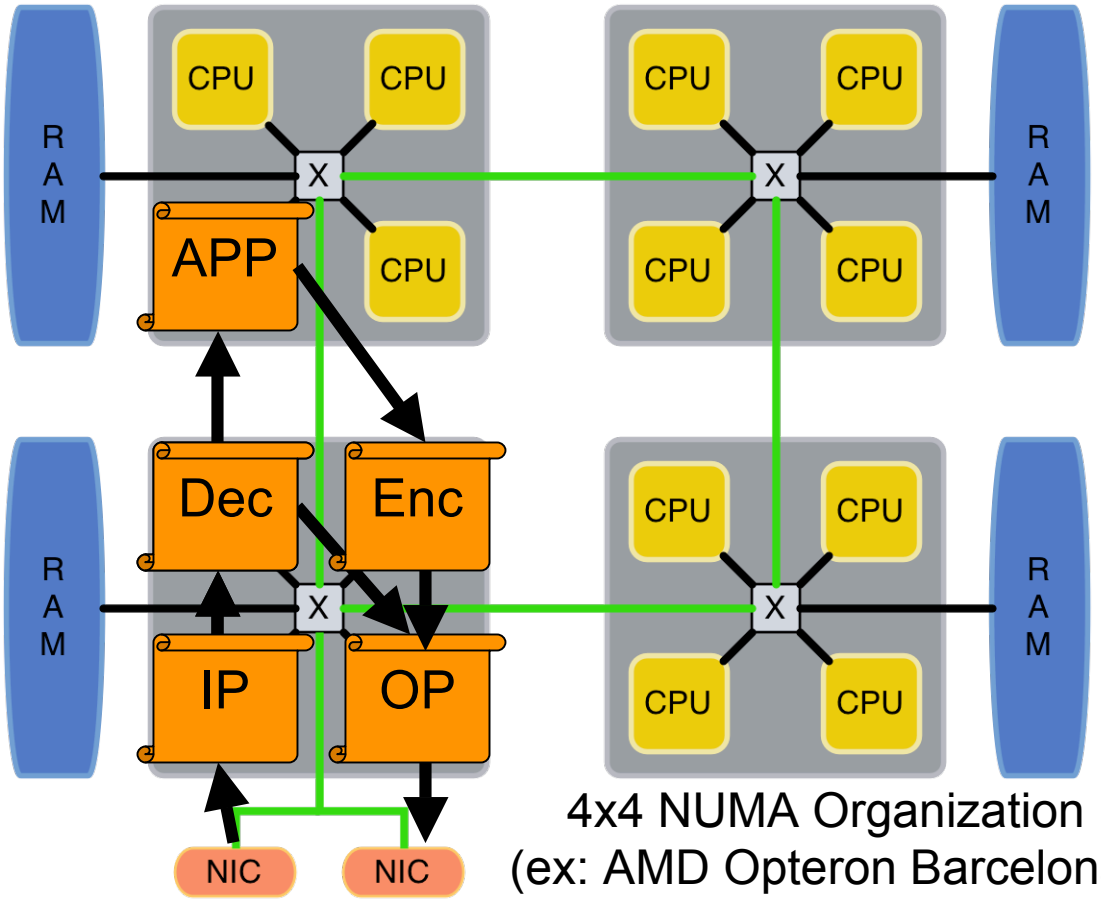
Network Processing Scenarios



Link	Mbps	fps	ns/frame
T-1	1.5	2,941	340,000
T-3	45.0	90,909	11,000
OC-3	155.0	333,333	3,000
OC-12	622.0	1,219,512	820
GigE	*1,000.0*	*1,488,095*	*672*
OC-48	2,500.0	5,000,000	200
10 GigE	10,000.0	14,925,373	67
OC-192	9,500.0	19,697,843	51

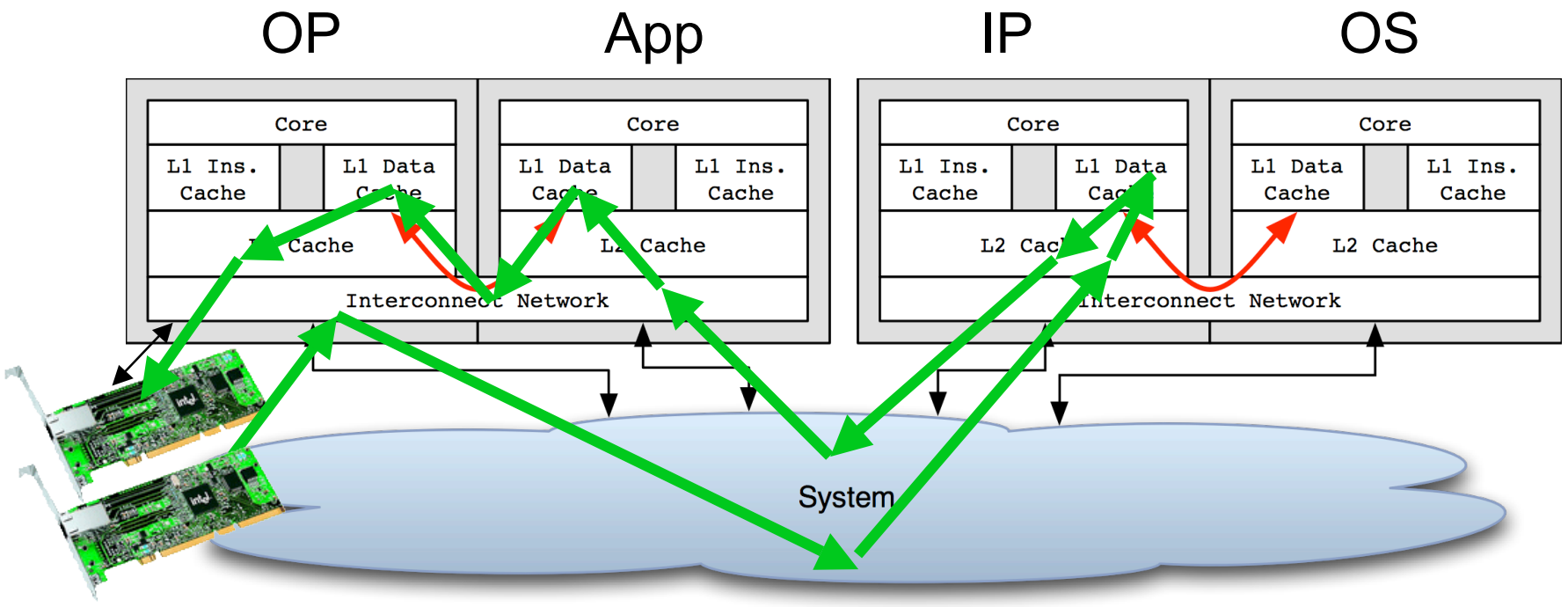
Core-Placements



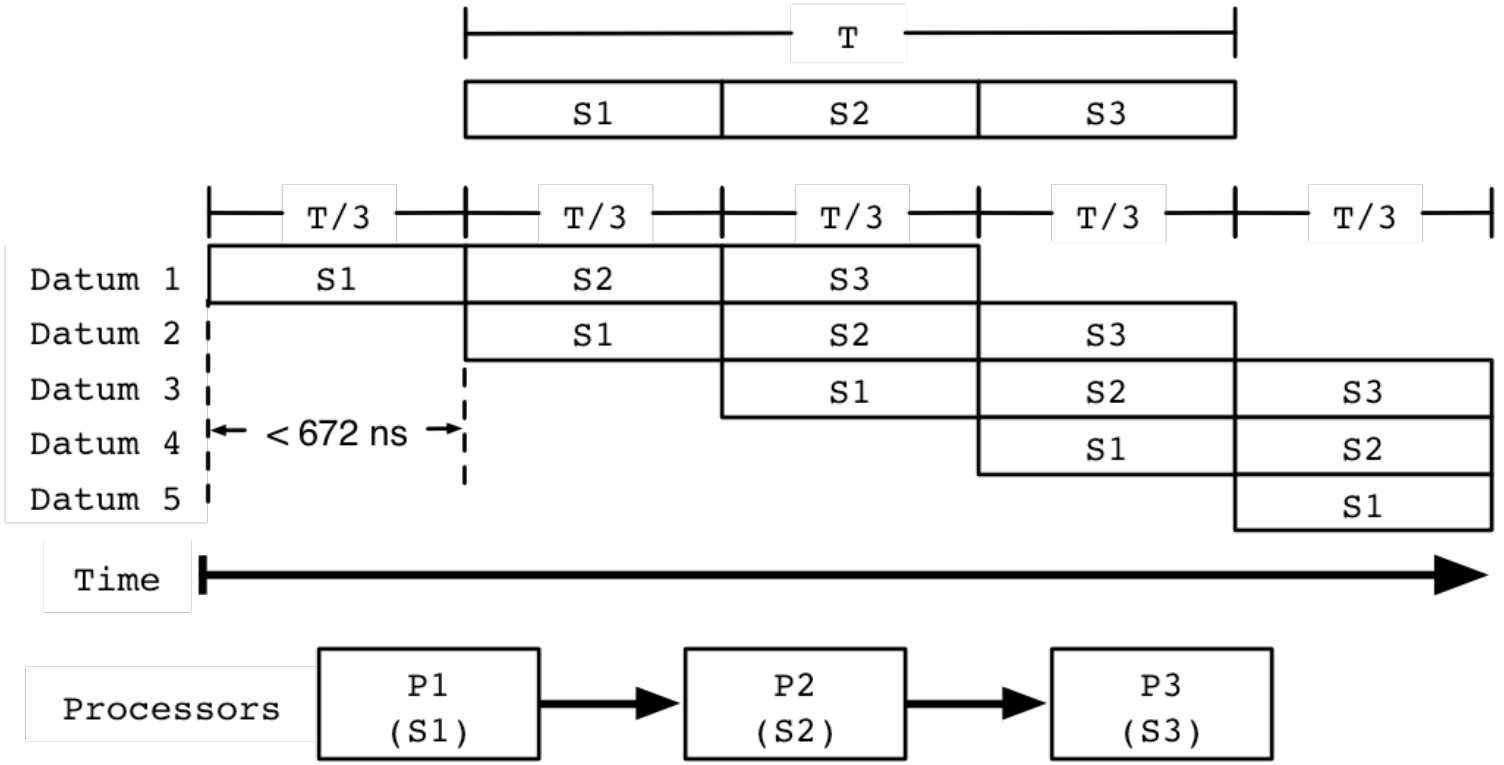


4x4 NUMA Organization
(ex: AMD Opteron Barcelona)

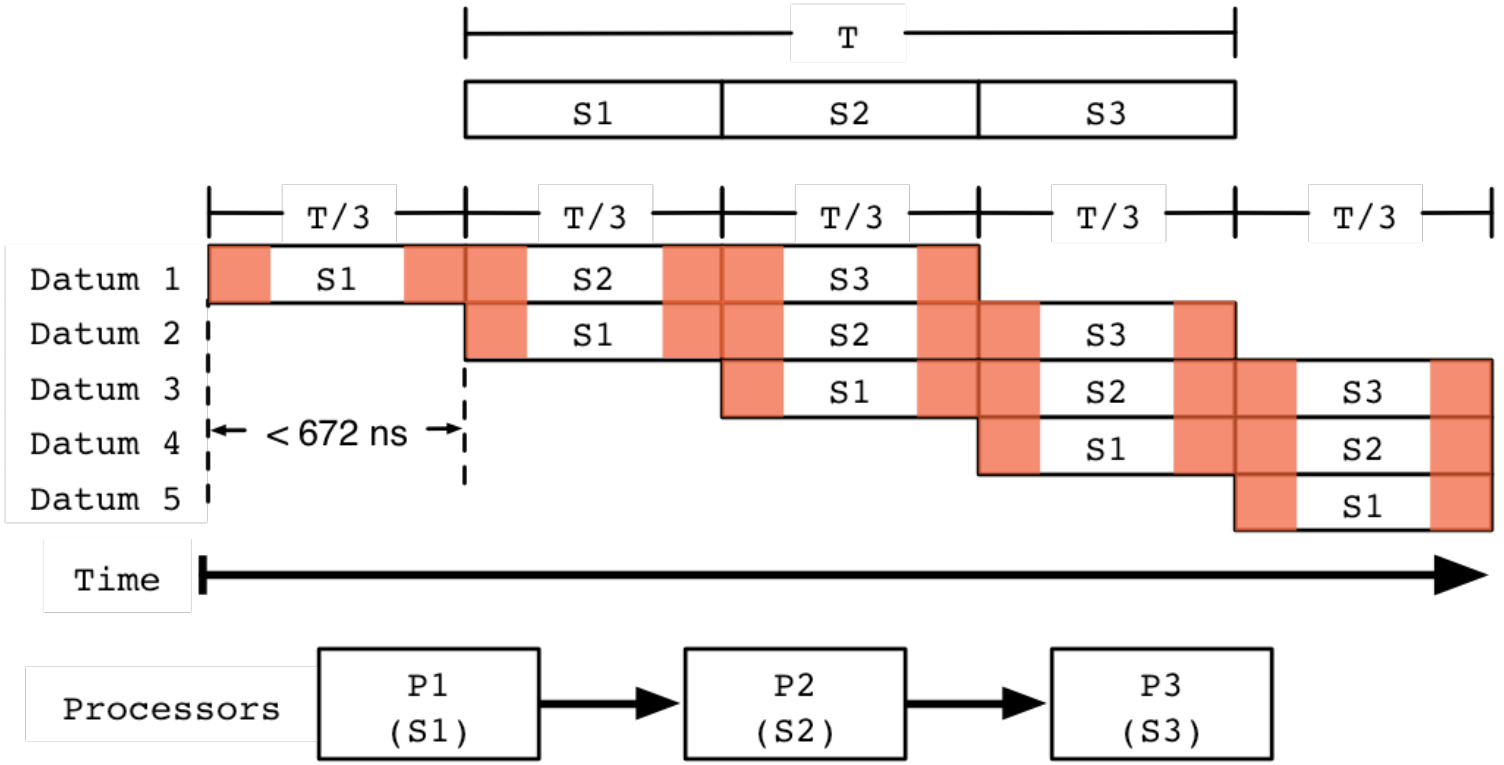
Routing/Bridge Data Flow



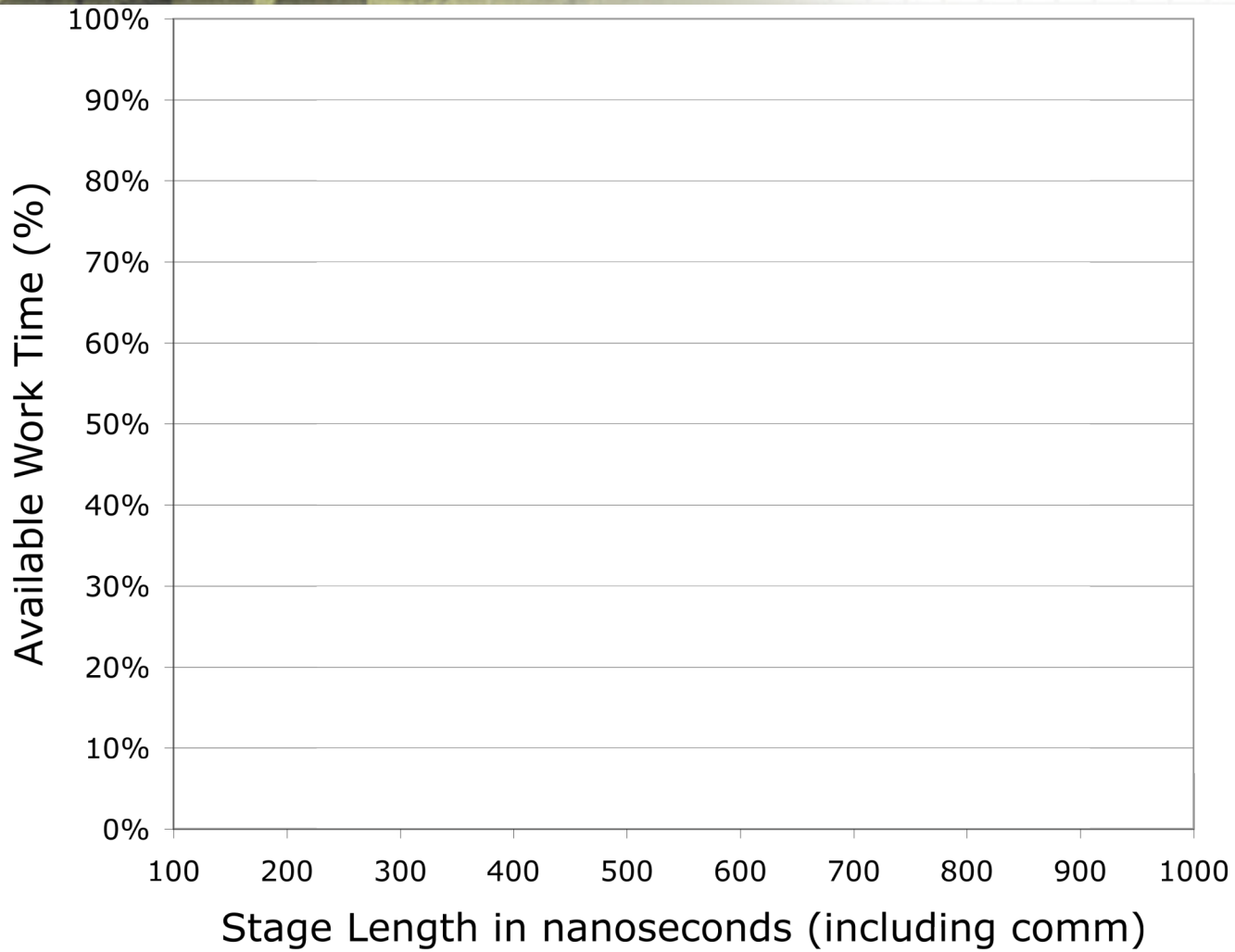
Example
3 Stage Pipeline



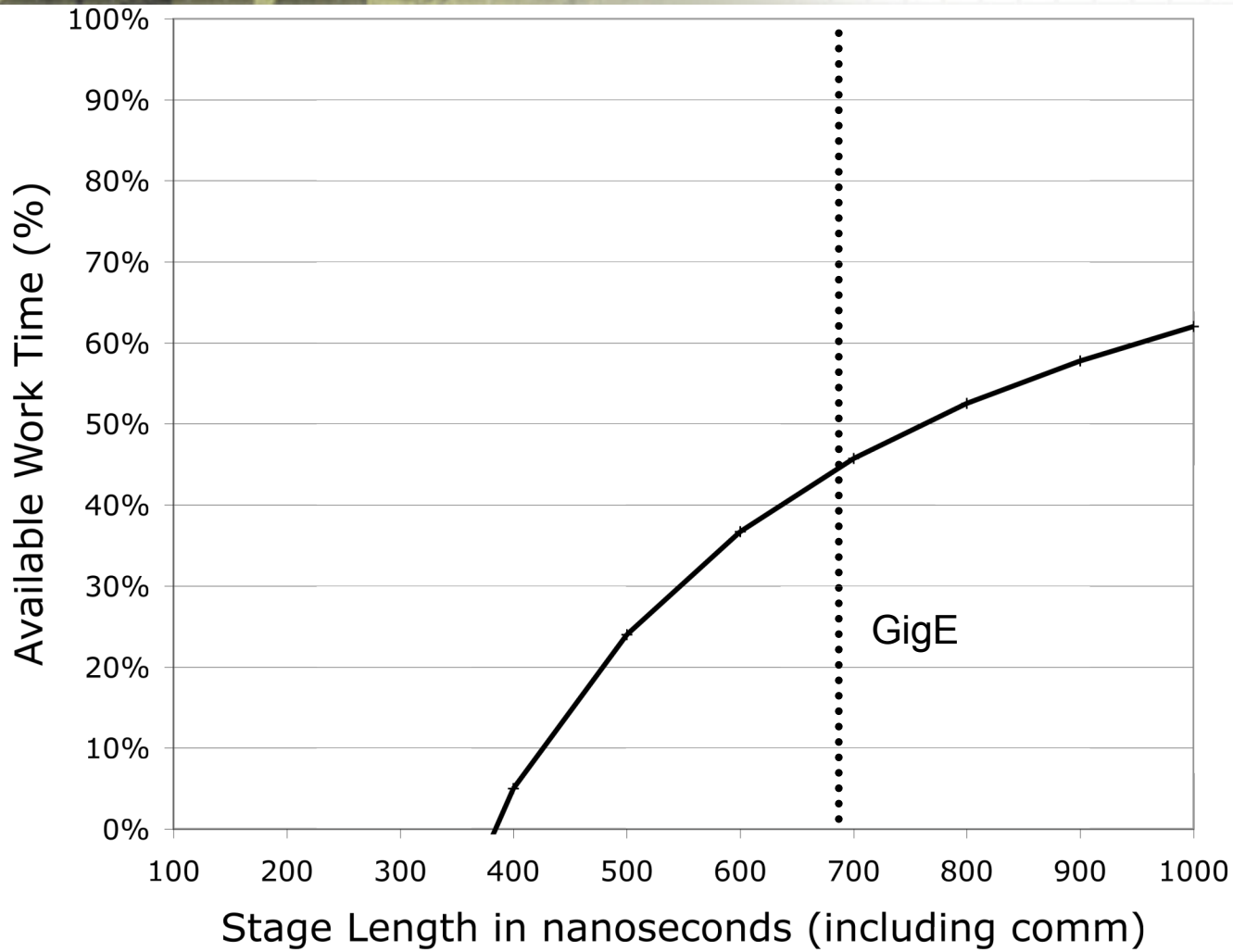
Example
3 Stage Pipeline



Communication Overhead

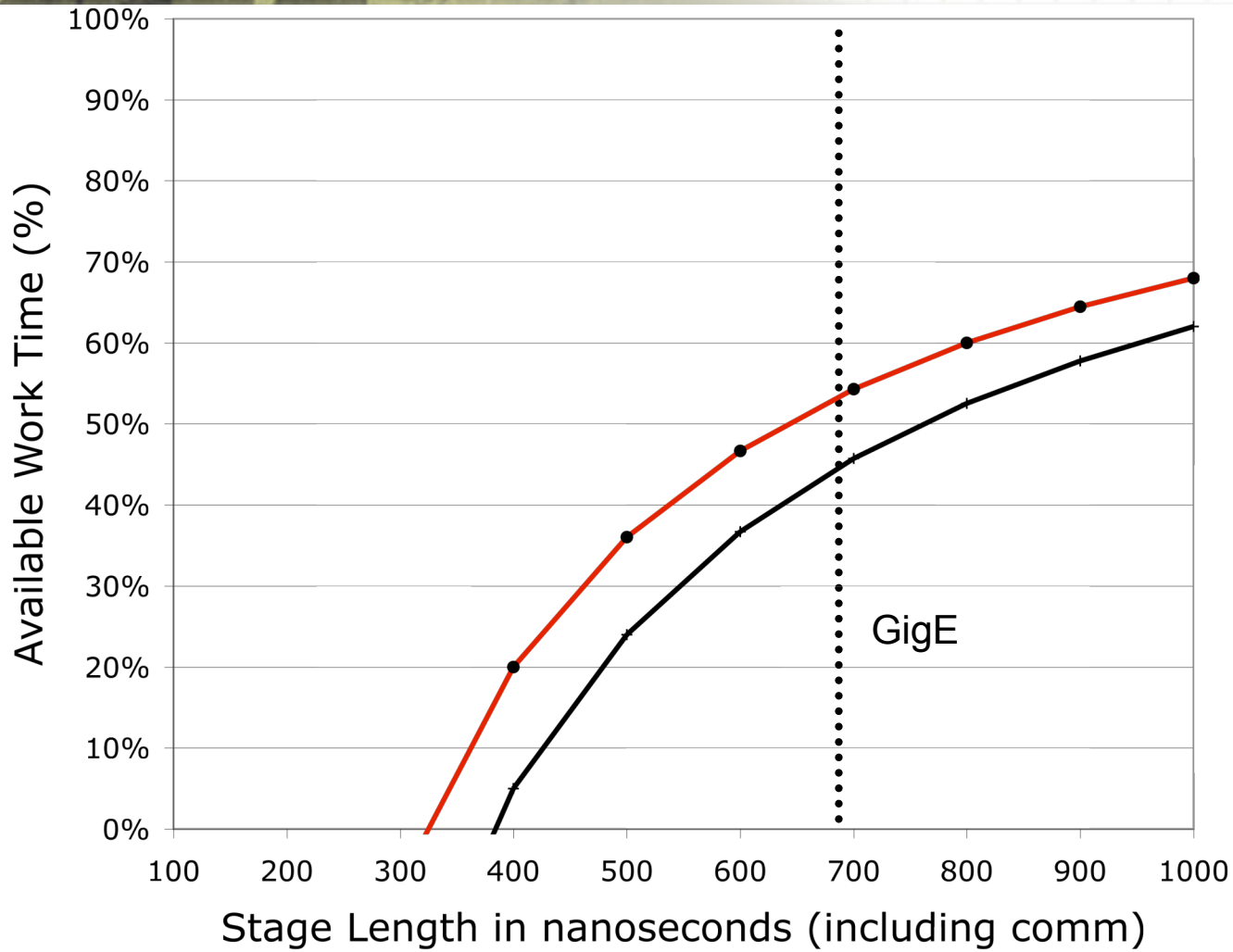


Communication Overhead



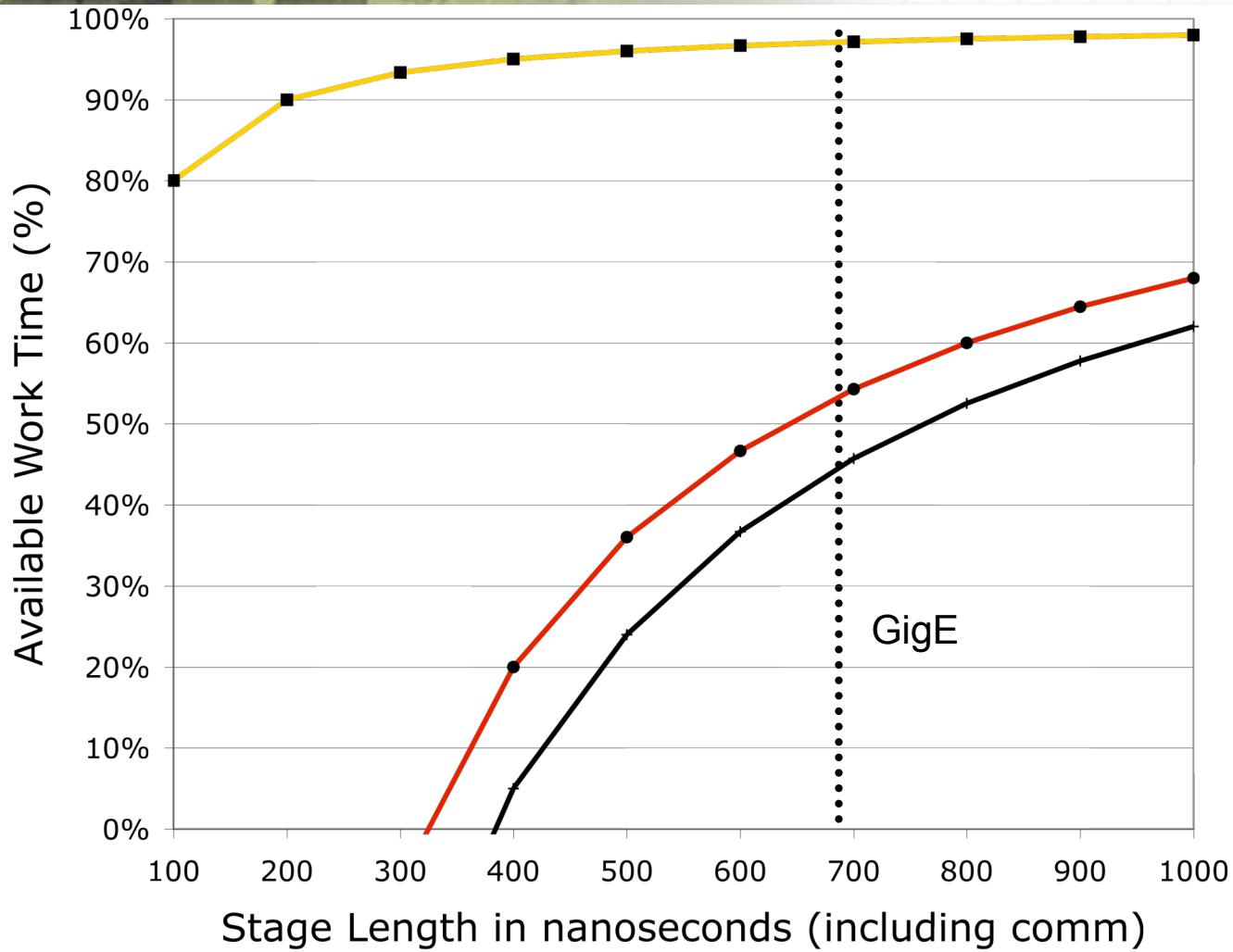
Locks ≈ 190ns

Communication Overhead



Lamport ≈ 160ns
Locks ≈ 190ns

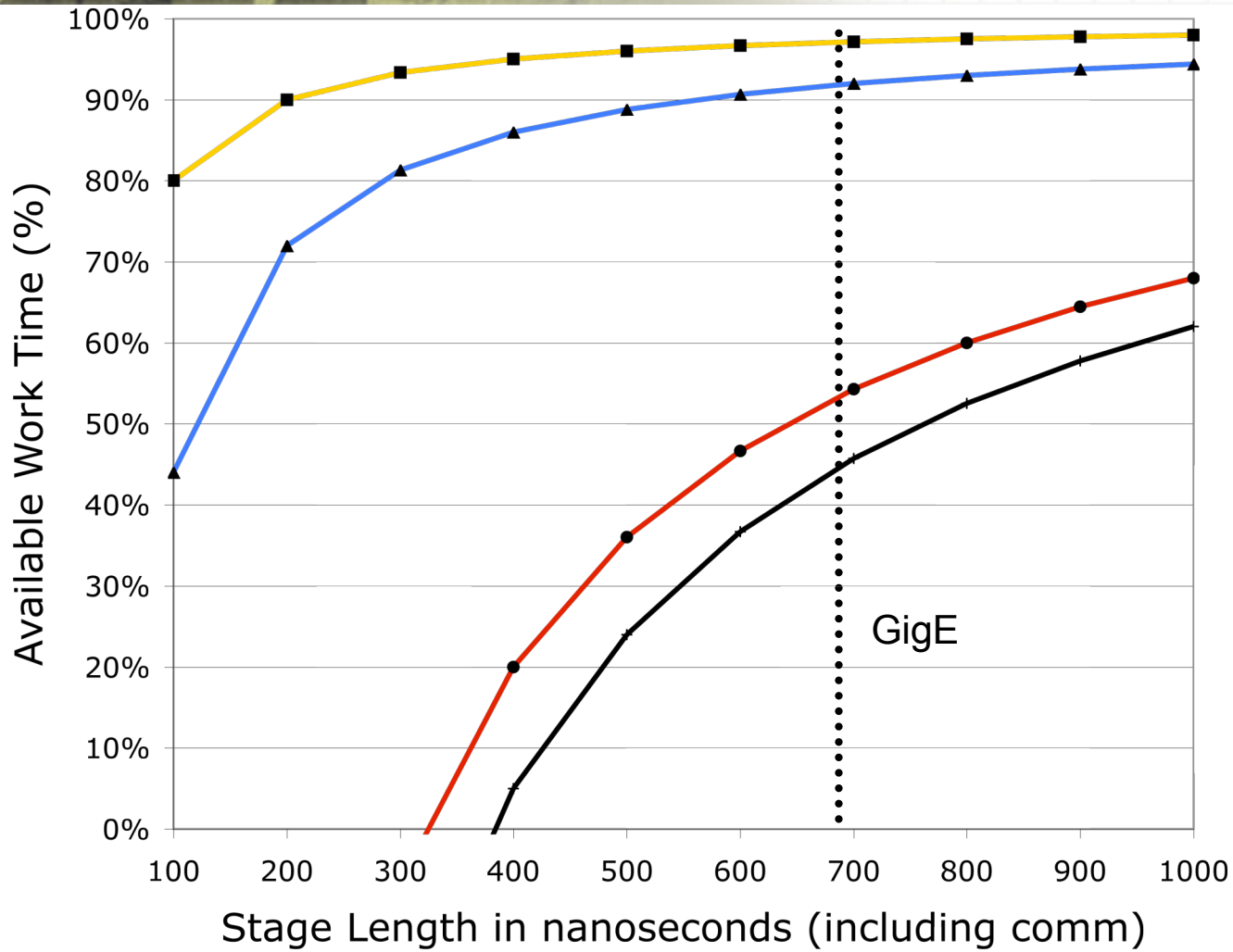
Communication Overhead



Hardware ≈ 10ns

Lamport ≈ 160ns
Locks ≈ 190ns

Communication Overhead



Hardware ≈ 10ns
FastForward ≈ 28ns

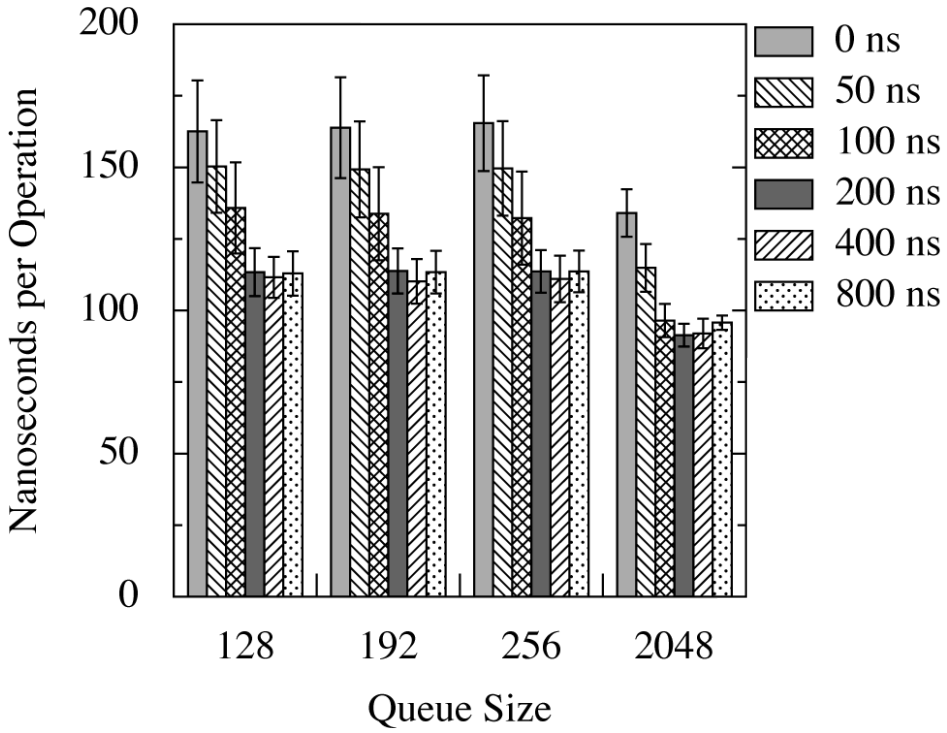
Lamport ≈ 160ns
Locks ≈ 190ns

More Fine-Grain Pipelining Examples

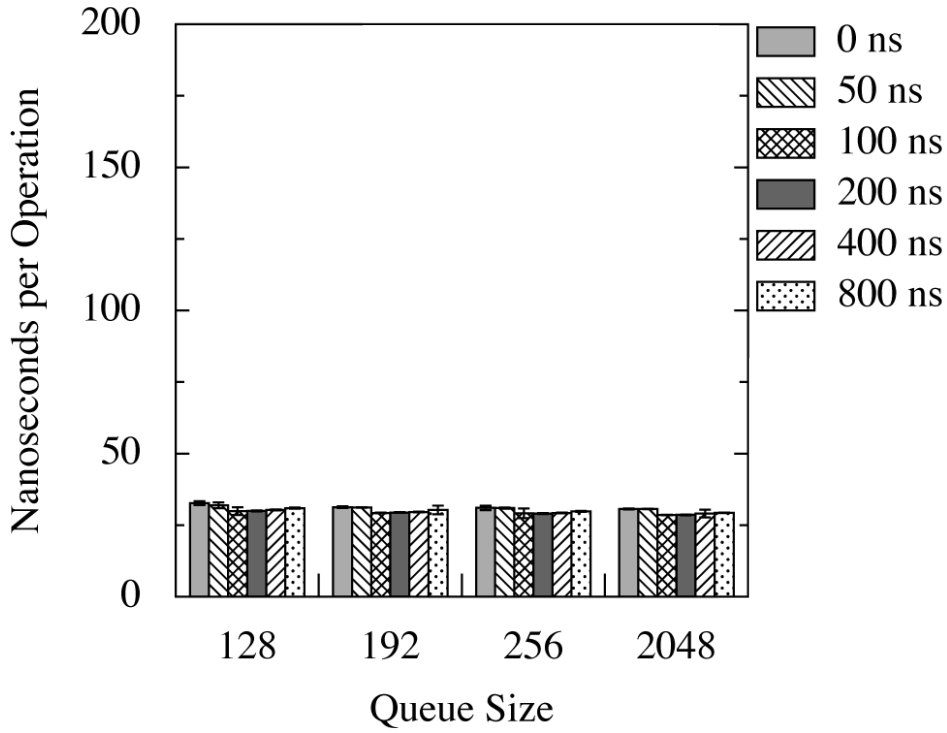
- Signal Processing
  - Media transcoding/encoding/decoding
  - Software Defined Radios
- Encryption
  - Triple-DES
  - Counter-Mode AES
- Other Domains
  - ODE Solvers
  - Fine-grain kernels extracted from sequential applications

Comparative Performance

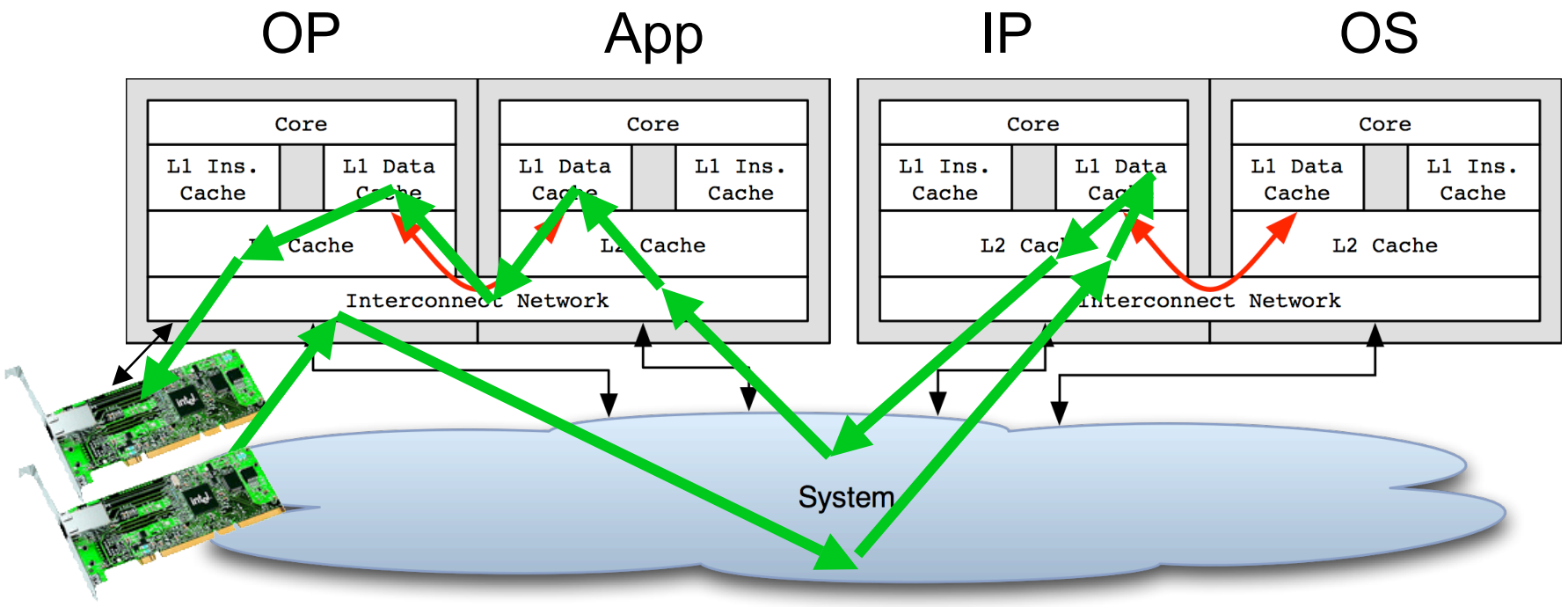
Lamport



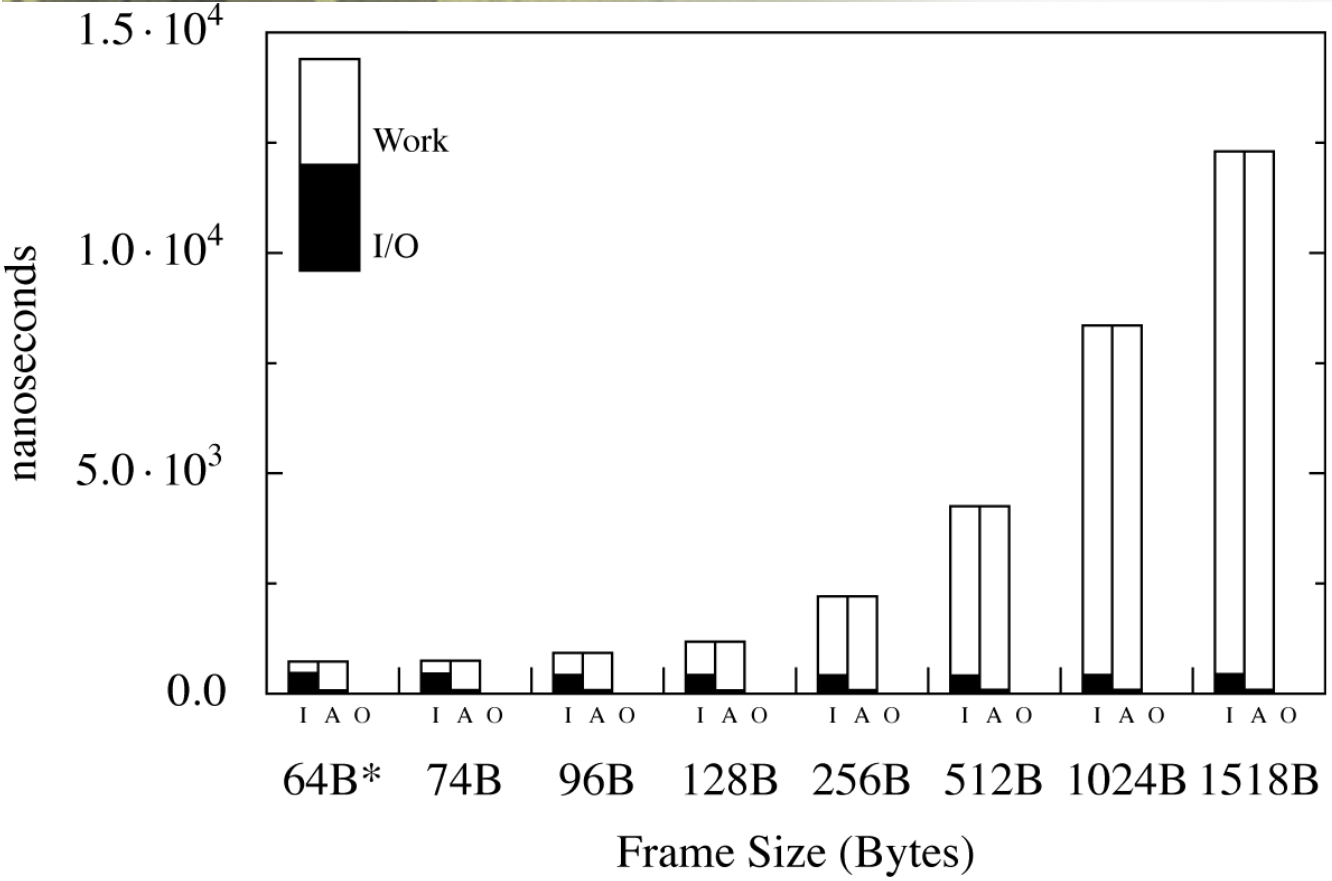
FastForward



Routing/Bridge Data Flow



FShm Forward (Bridge)



–AES encrypting filter
- •Link layer encryption
- •~10 lines of code

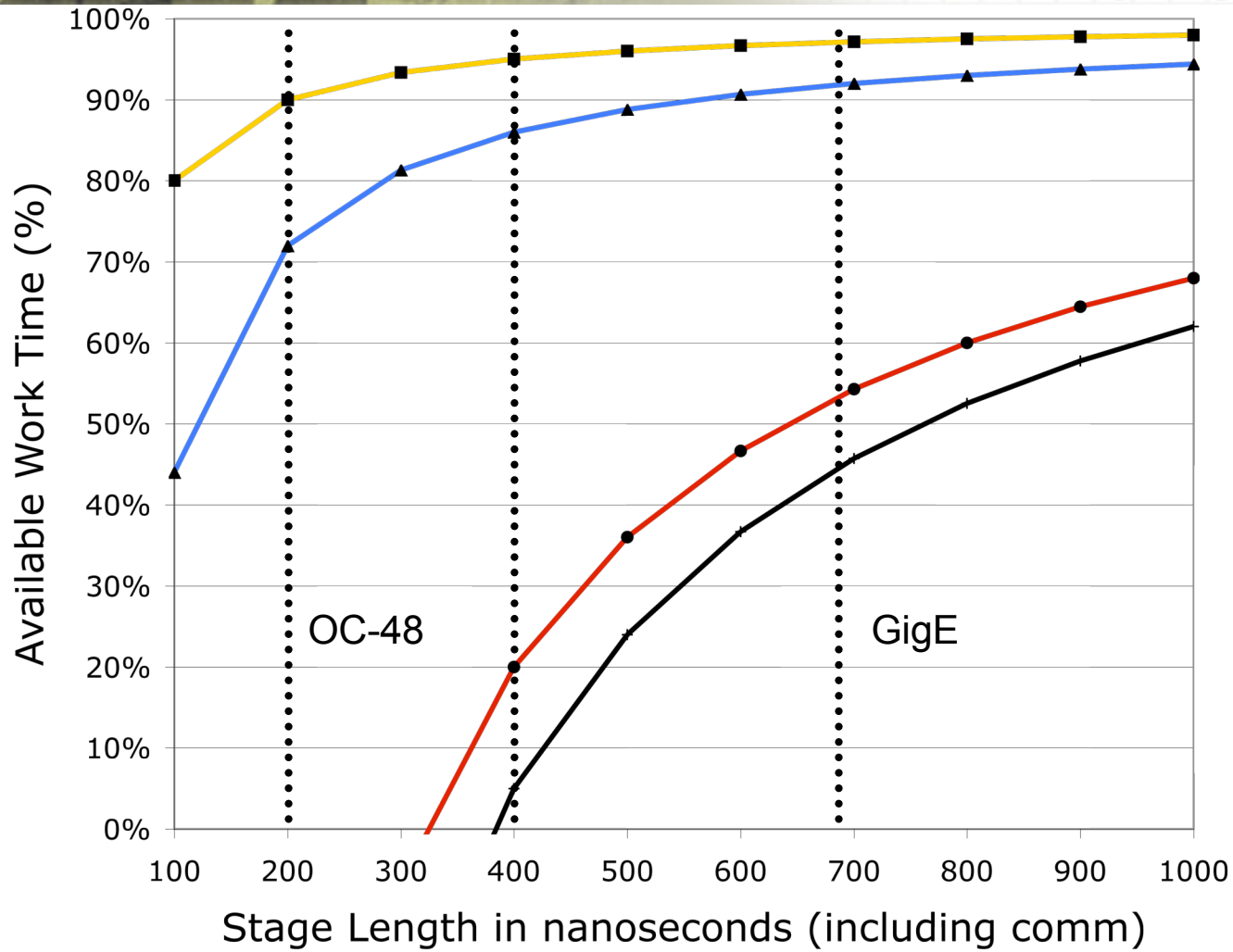
•IDS
- •Complex Rules

•IPS
- •DDoS

•Data Recorders
- •Traffic Analysis
- •Forensics
- •CALEA

64B* ≈ 1.36 Mfps

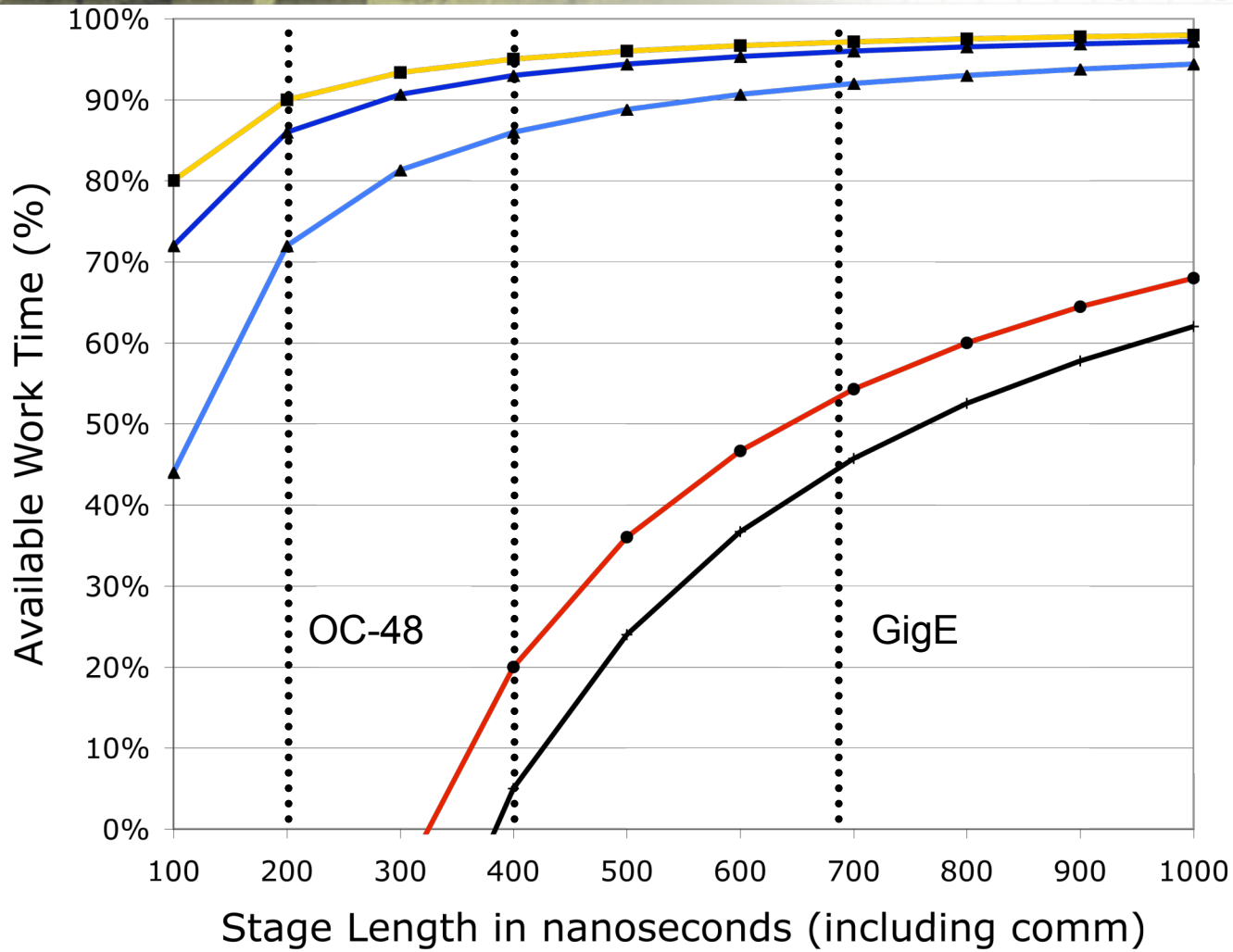
Gazing into the Crystal Ball



Hardware ≈ 10ns
FastForward ≈ 28ns

Lamport ≈ 160ns
Locks ≈ 190ns

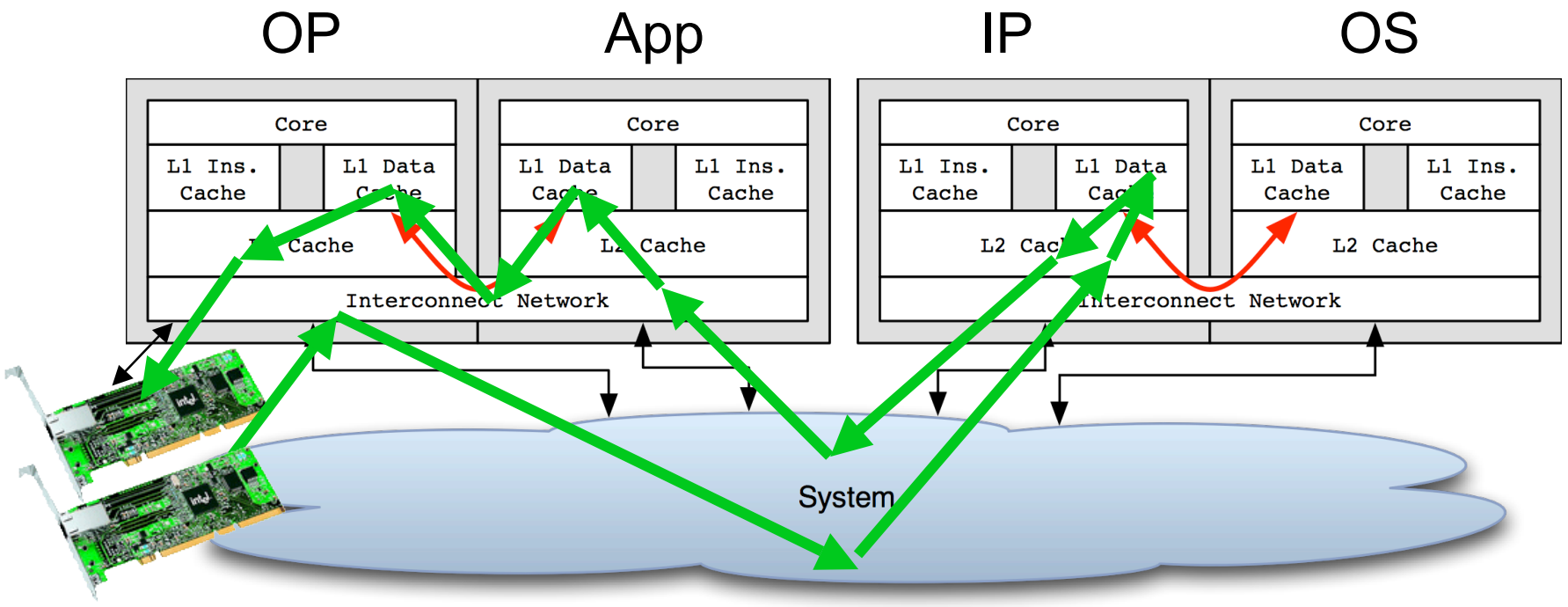
Gazing into the Crystal Ball



Hardware ≈ 10ns
FastForward ≈ 14ns
FastForward ≈ 28ns

Lamport ≈ 160ns
Locks ≈ 190ns

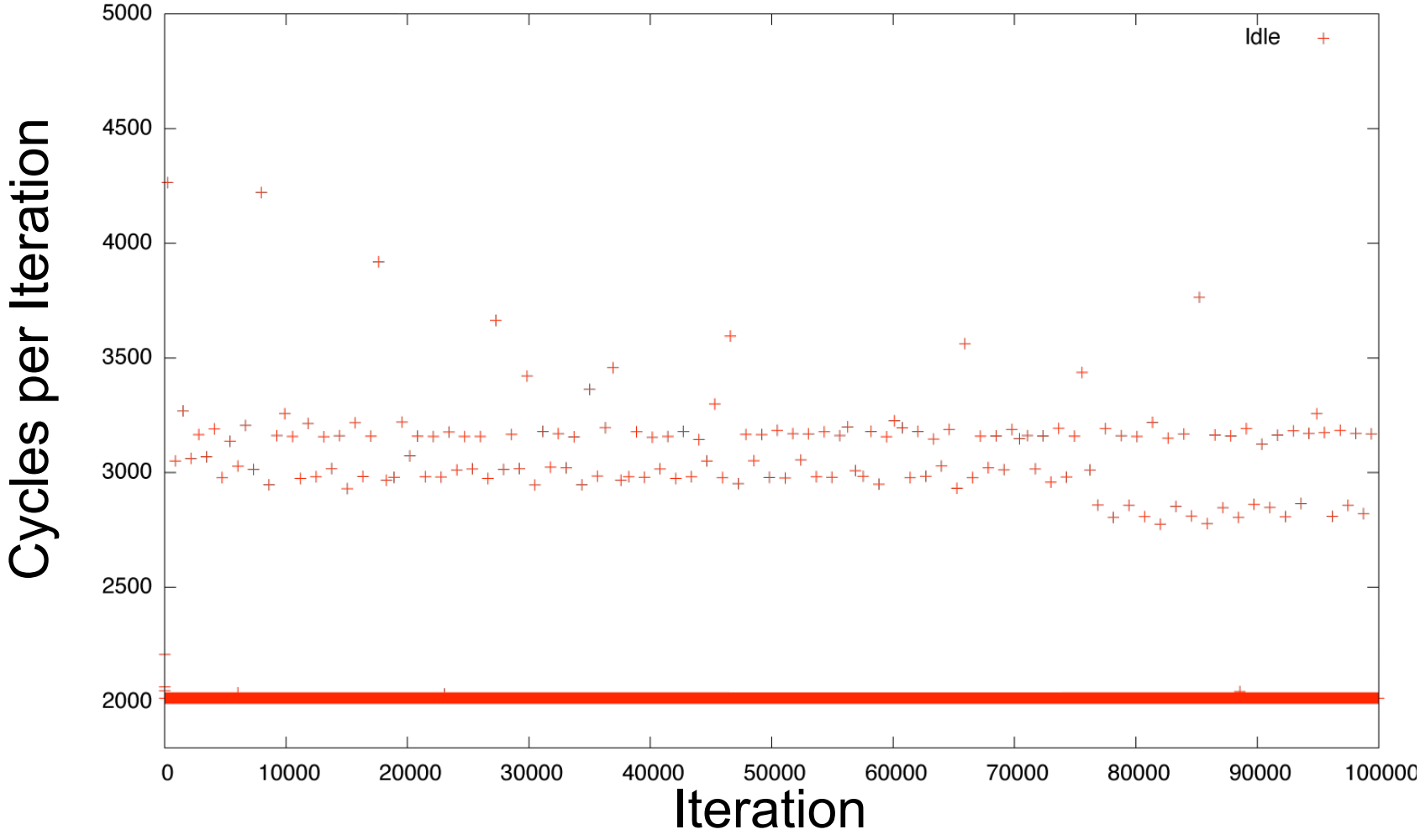
Routing/Bridge Data Flow



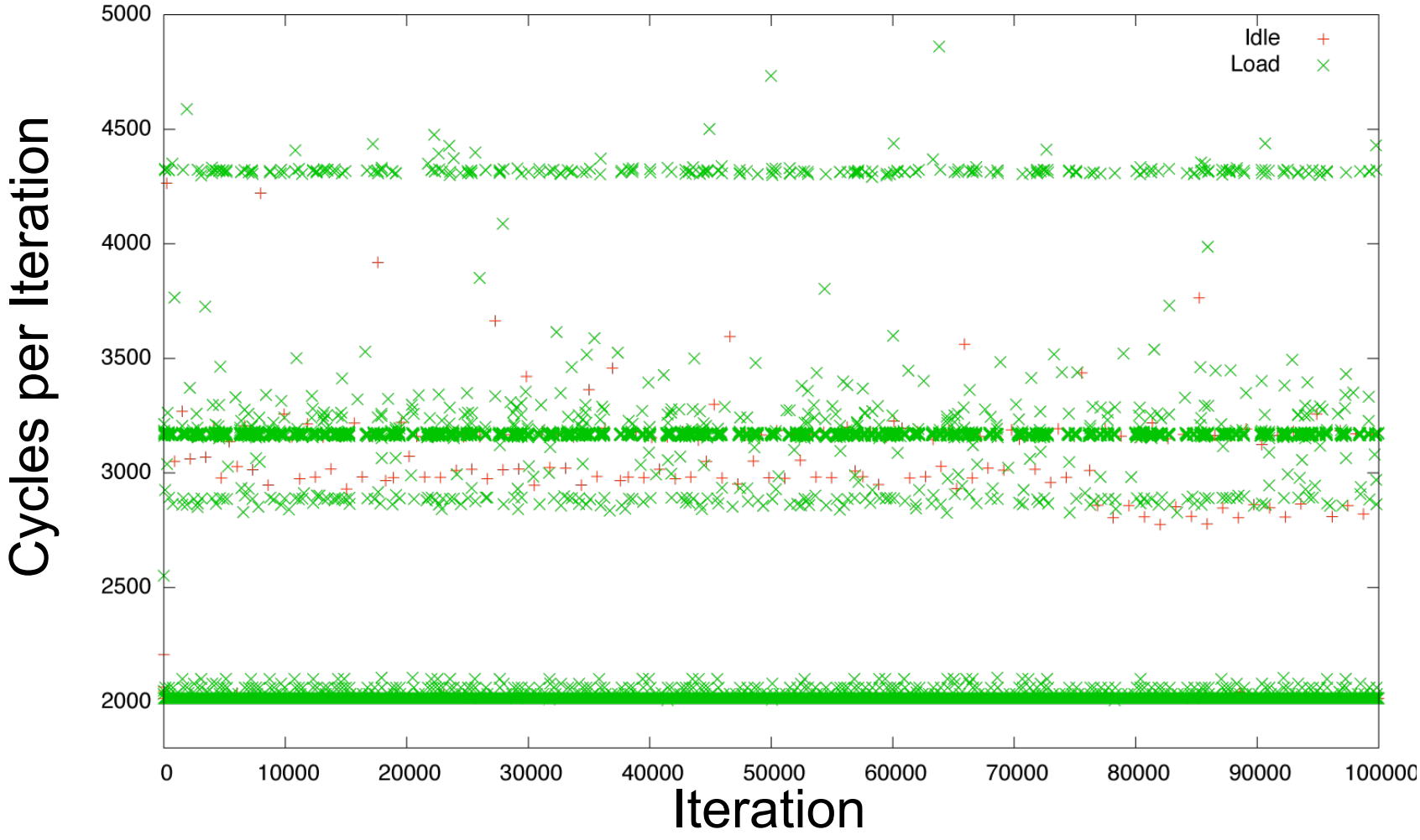
Implementing Fine-Grain Parallelism

- Software Only
  - FastForward/FShm
  - Streamware
- Hardware Support
  - Stream-It
  - DSWP

Register Bound Performance



Register Bound Performance



Effect of Jitter on Pipelines

	Stages	Free	Bind	Bind++	Hard Bind
Idle	2	2838	2844	2843	2842
	3	2842	2845	2841	2841
Load	2	2995 (106%)	3008	2998	2863 (101%)
	3	3075 (108%)	3067	3072	2848 (100%)

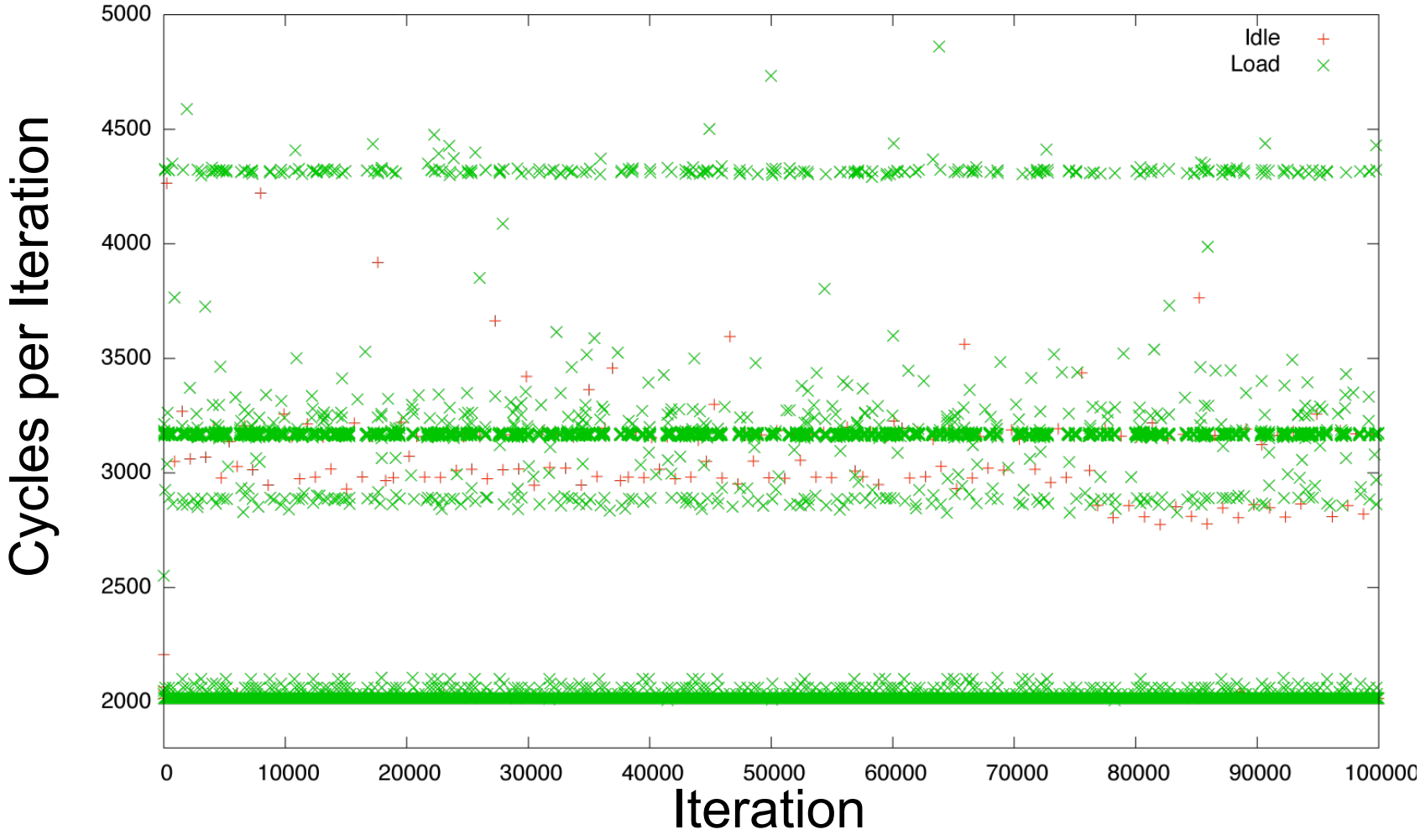
The Real World

- System Jitter - Symmetric & Asymmetric
  - Timer Interrupts
  - Scheduler
  - I/O
  - TLB updates

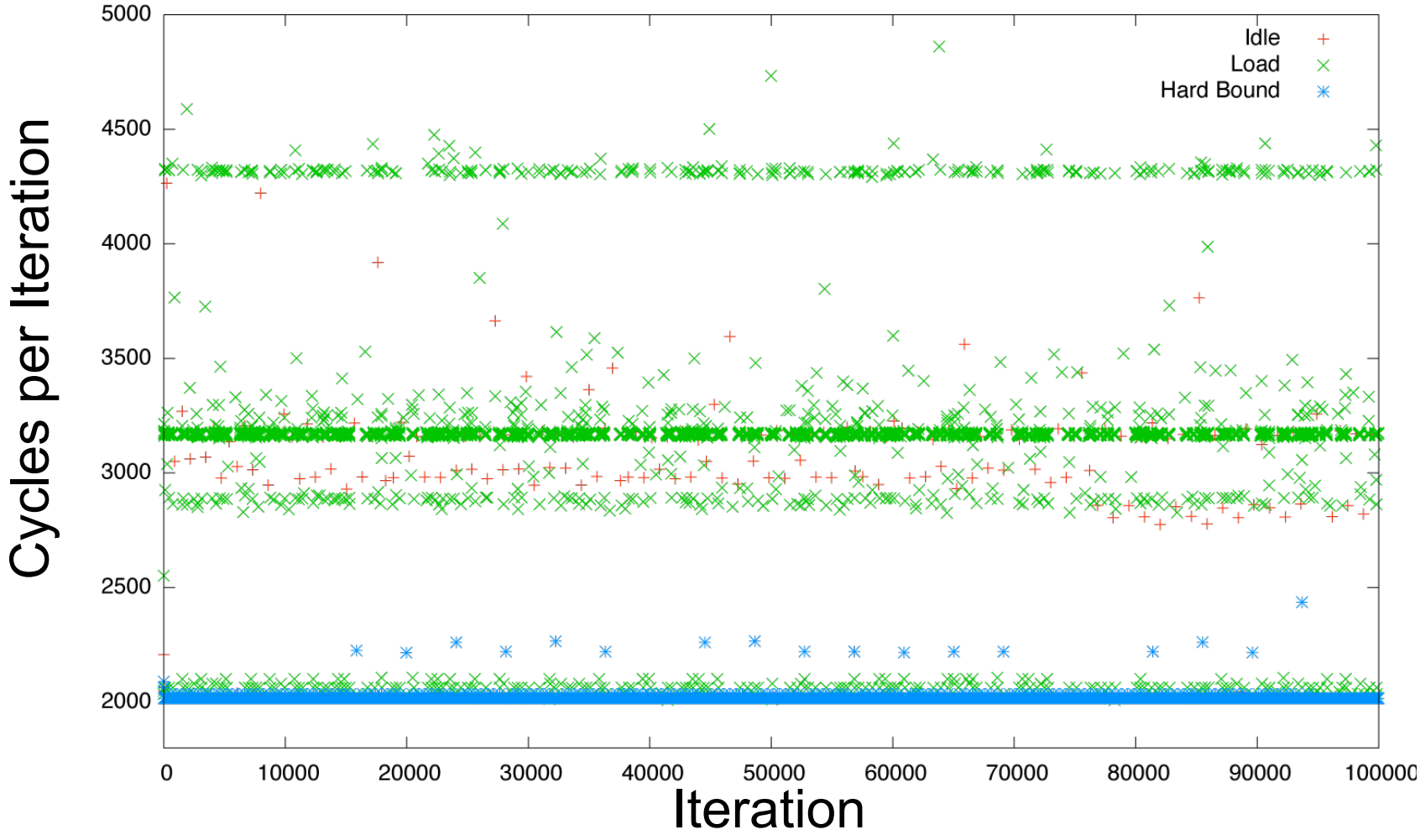
Selective Timesharing

- Partitioning the system
  - Isolate performance critical applications
    - Processors
    - Memory (NUMA)
    - Interconnects
    - Lasts until application releases resources or user changes priorities
  - Timeshare normal applications
    - Normal applications see a system with fewer resources

Register Bound Performance



System with Selective Timesharing



Shared Memory Accelerated Queues
Now Available!

http://ce.colorado.edu/core

Questions?

john.giacomoni@colorado.edu