The Impact of Memory Subsystem Resource Sharing on Datacenter Applications

Lingjia Tang, Jason Mars

Neil Vachharajani, Robert Hundt, Mary Lou Soffa

Modern Datacenter

- Datacenters: important computing domain

- Host large-scale internet service
- Expensive

Lingjia Tang: lt8f@cs.virginia.edu

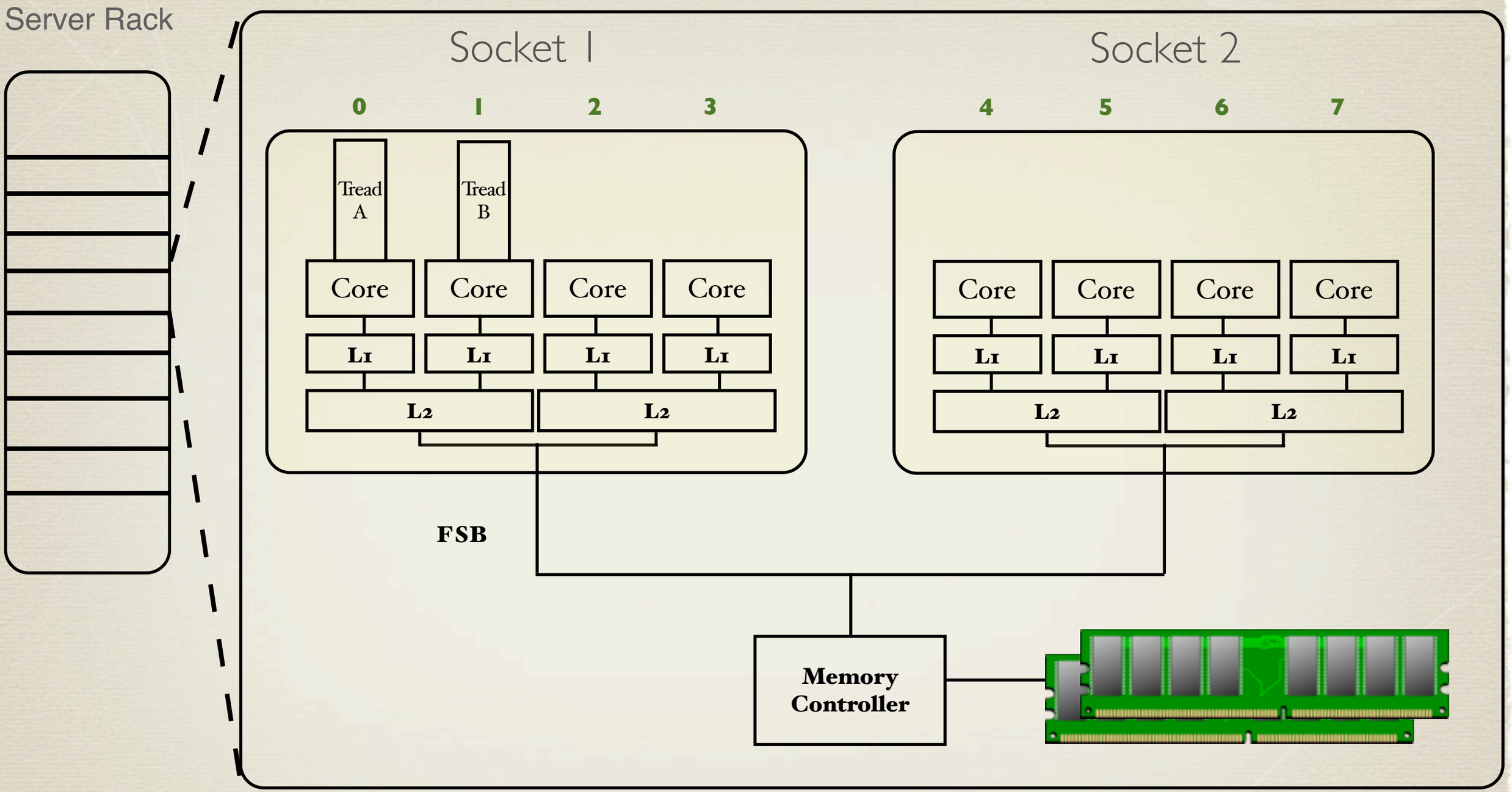
Problem

- Lack of understanding of the datacenter workloads
  - Emerging workloads (different than standard benchmarks SPEC, PARSEC, etc)
  - How workloads interact with commodity multicore hardware (shared memory resources)
  - How workloads interact with each other on multicores

- Lack of understanding leads to inefficient systems
  - 1 % performance improvement is huge at Google's scale

Lingjia Tang: lt8f@cs.virginia.edu

Memory Resource Sharing



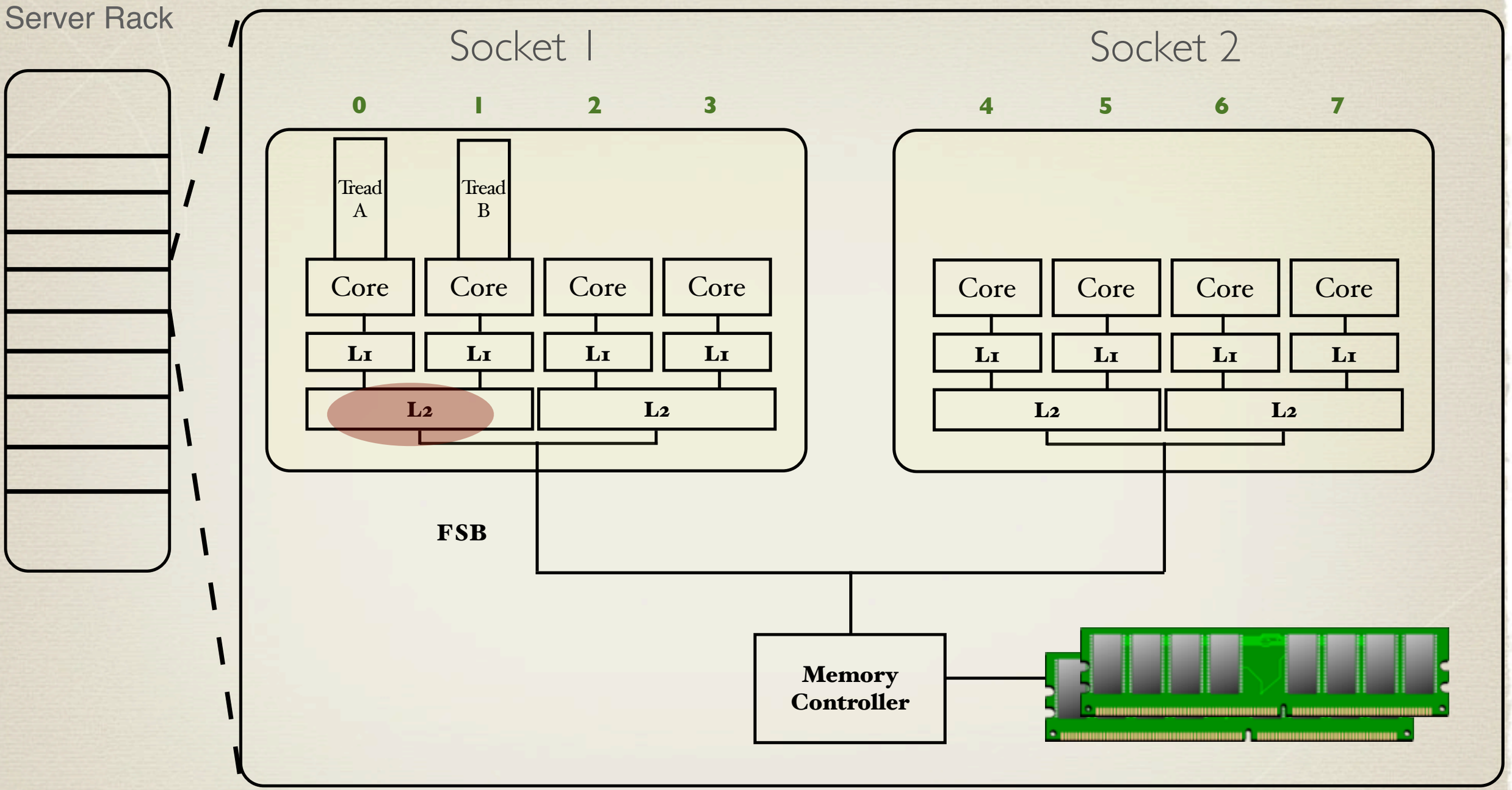
* Sharing: constructive and destructive

* Thread-to-core mapping

Intel Xeon(clovertown)

Lingjia Tang: lt8f@cs.virginia.edu

Memory Resource Sharing

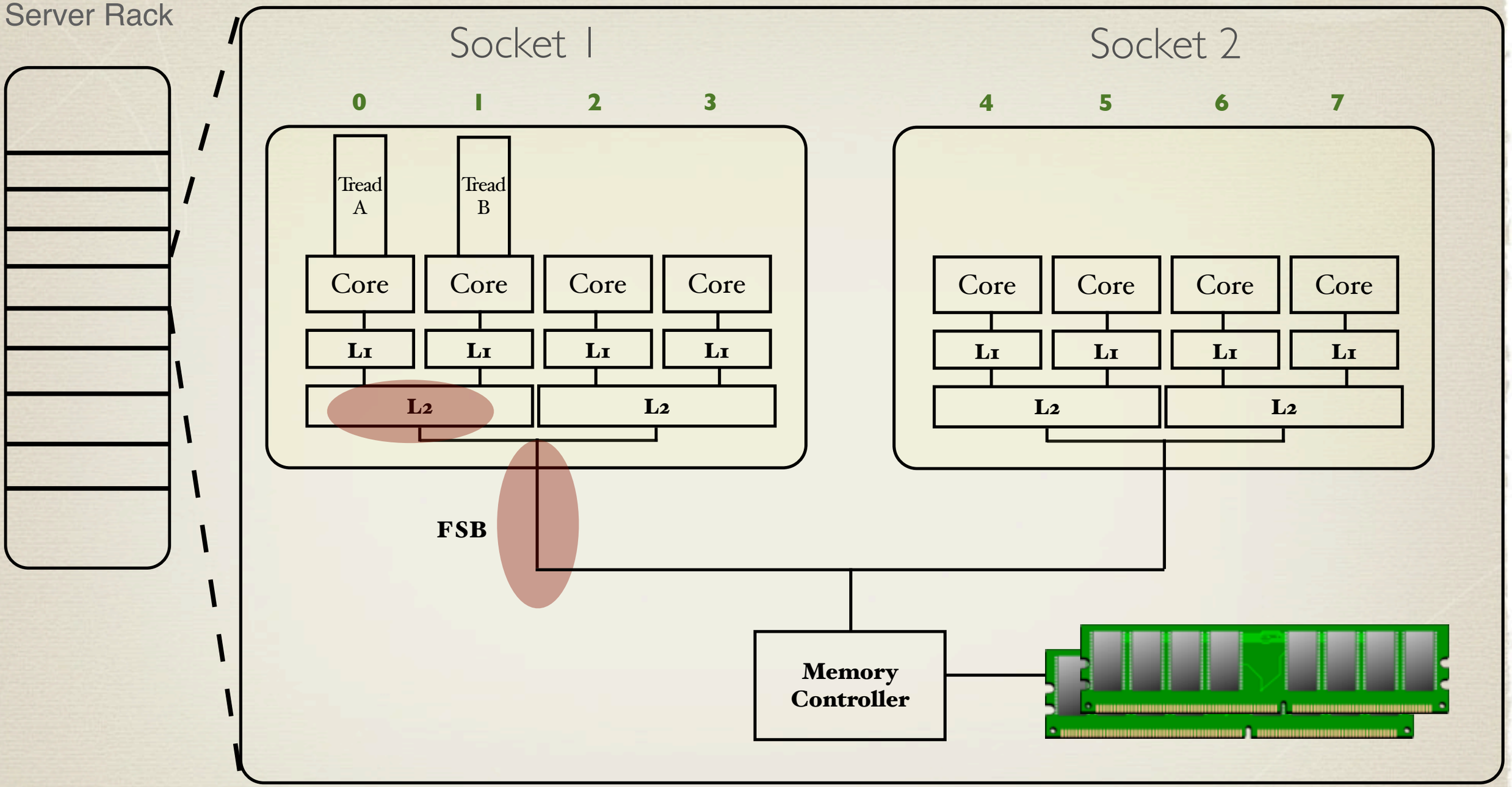


Intel Xeon(clovertown)

* Sharing: constructive and destructive
* Thread-to-core mapping

Lingjia Tang: lt8f@cs.virginia.edu

Memory Resource Sharing

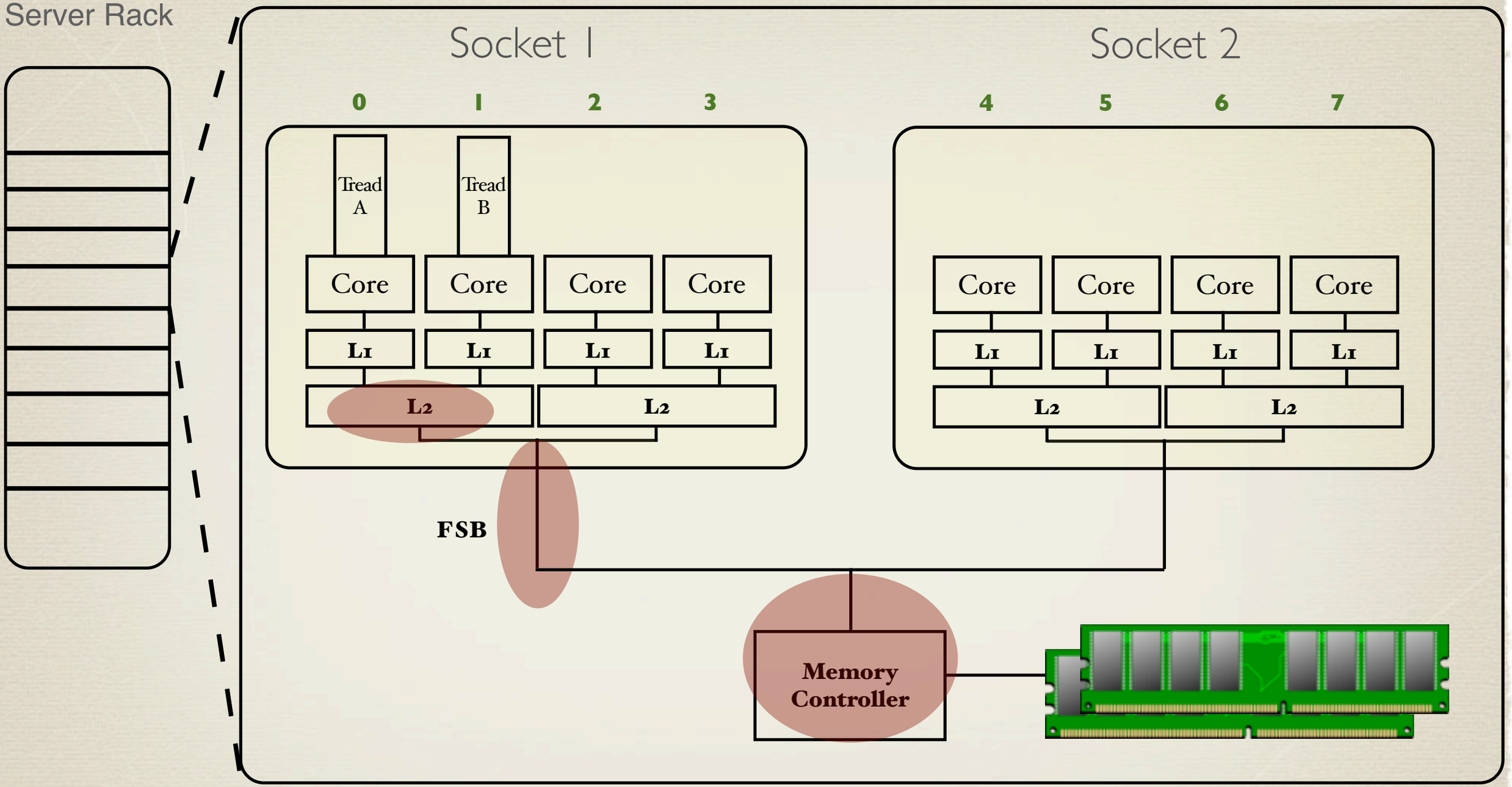


Intel Xeon(clovertown)

* Sharing: constructive and destructive
* Thread-to-core mapping

Lingjia Tang: lt8f@cs.virginia.edu

Memory Resource Sharing

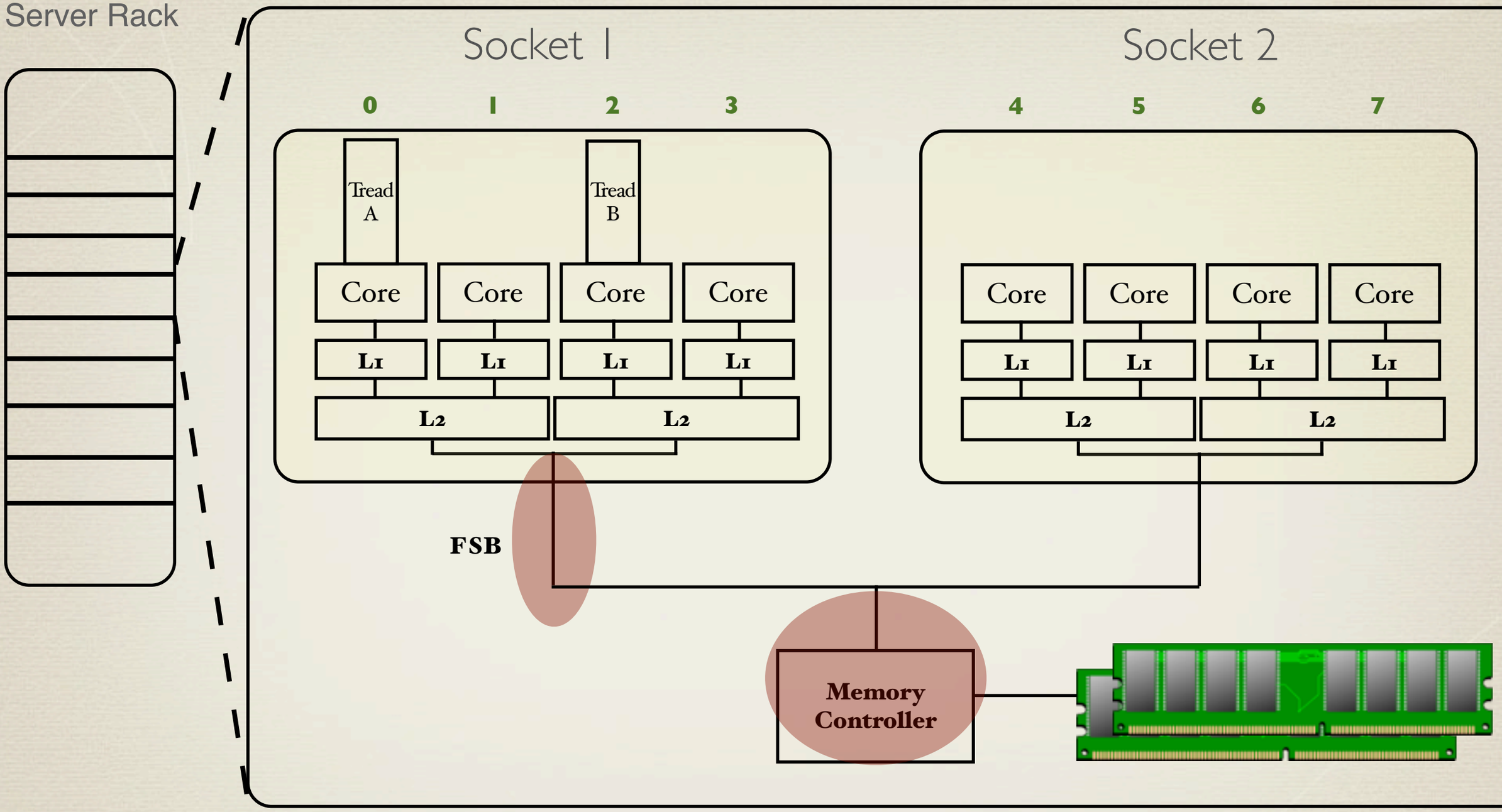


Intel Xeon(clovertown)

* Sharing: constructive and destructive
* Thread-to-core mapping

Lingjia Tang: lt8f@cs.virginia.edu

Memory Resource Sharing

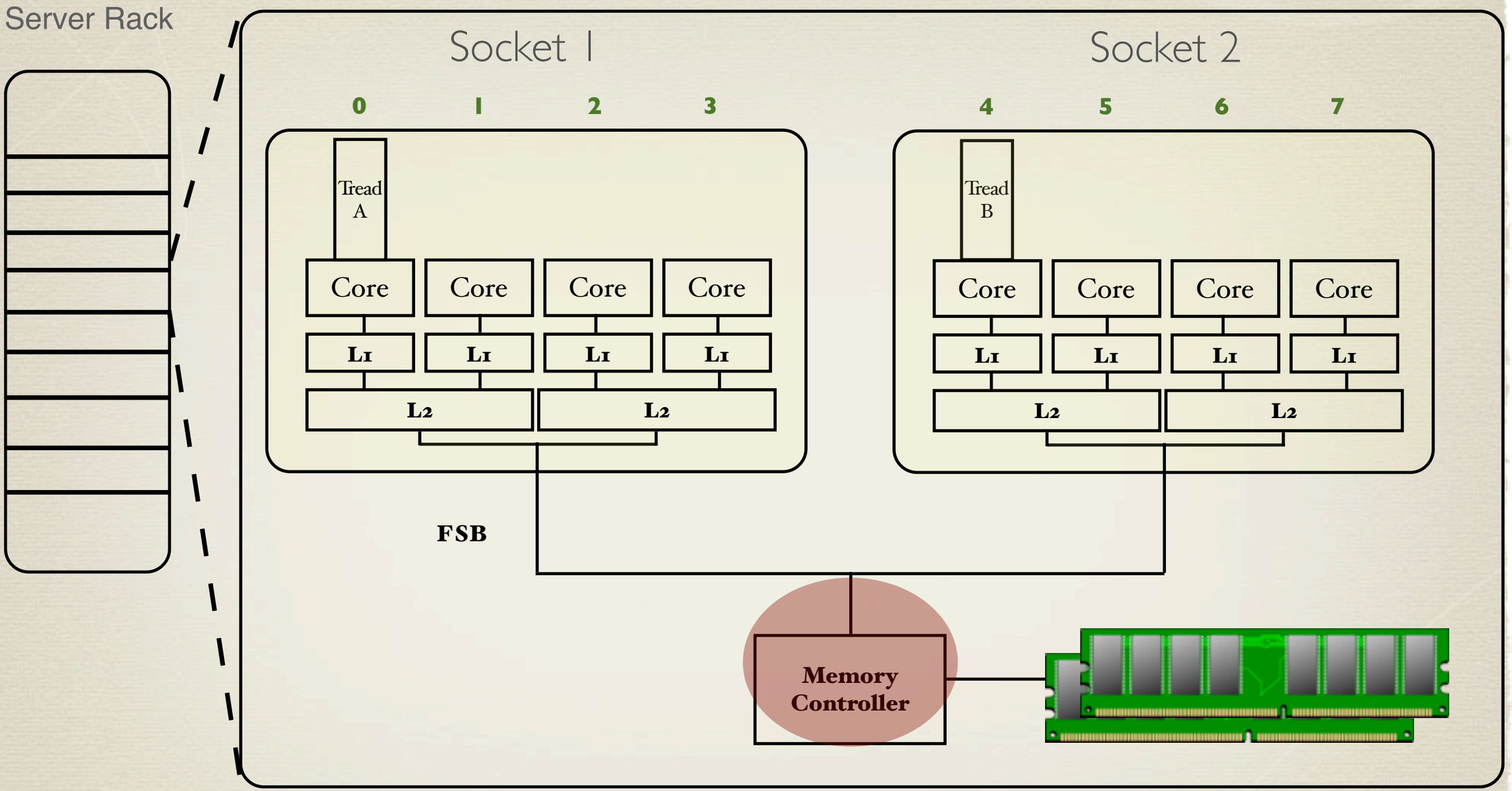


Intel Xeon(clovertown)

* Sharing: constructive and destructive
* Thread-to-core mapping

Lingjia Tang: lt8f@cs.virginia.edu

Memory Resource Sharing



Intel Xeon(clovertown)

* Sharing: constructive and destructive
* Thread-to-core mapping

Lingjia Tang: lt8f@cs.virginia.edu

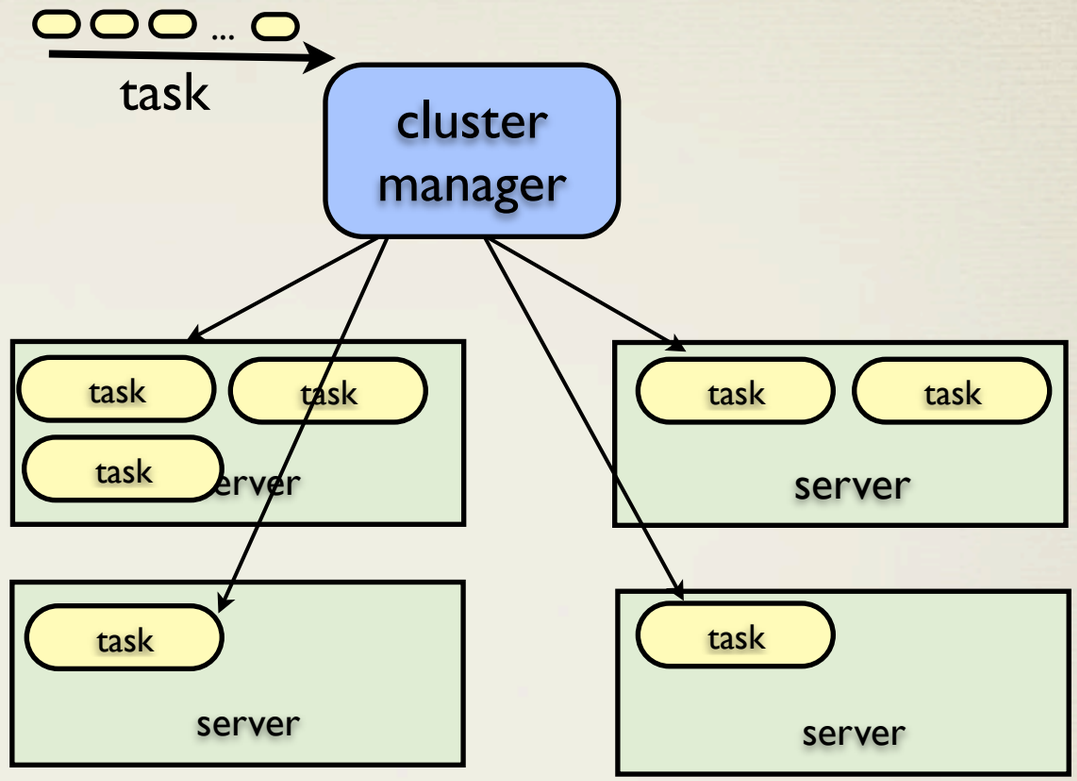
Goals

- Characterize datacenter workloads
  - Impact of memory resource sharing
  - On-chip shared caches, memory bandwidth
- Intelligent Thread-to-Core mapper:
  - Exploit workload characteristics
  - Arrive at efficient TTC mapping

Lingjia Tang: lt8f@cs.virginia.edu

Background: Datacenter scheduling

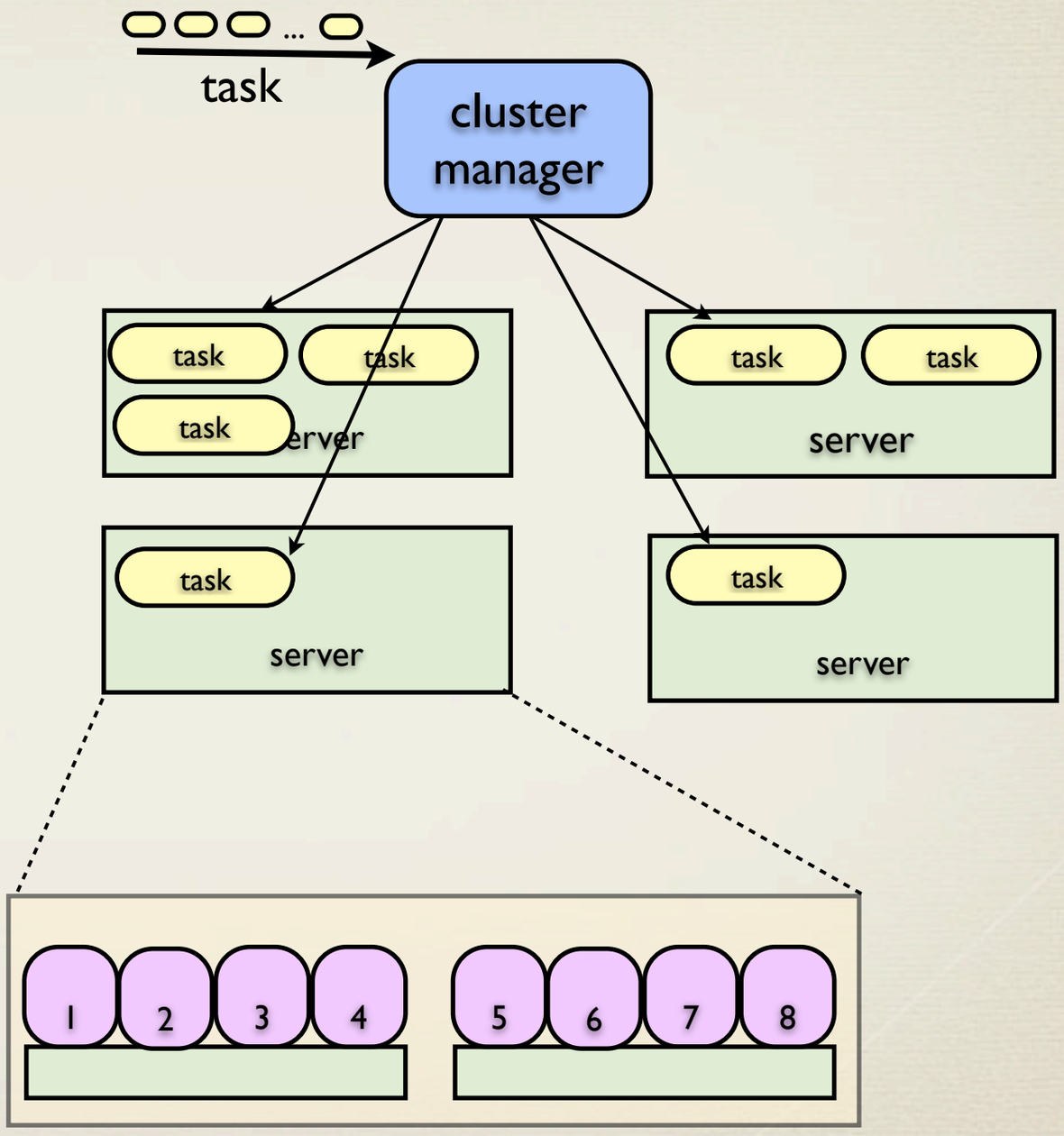
- Based on basic resource requirements.
- Applications
  - may run alone
  - may co-located with others
- Cache and bandwidth topology oblivious



Lingjia Tang: lt8f@cs.virginia.edu

Background: Datacenter scheduling

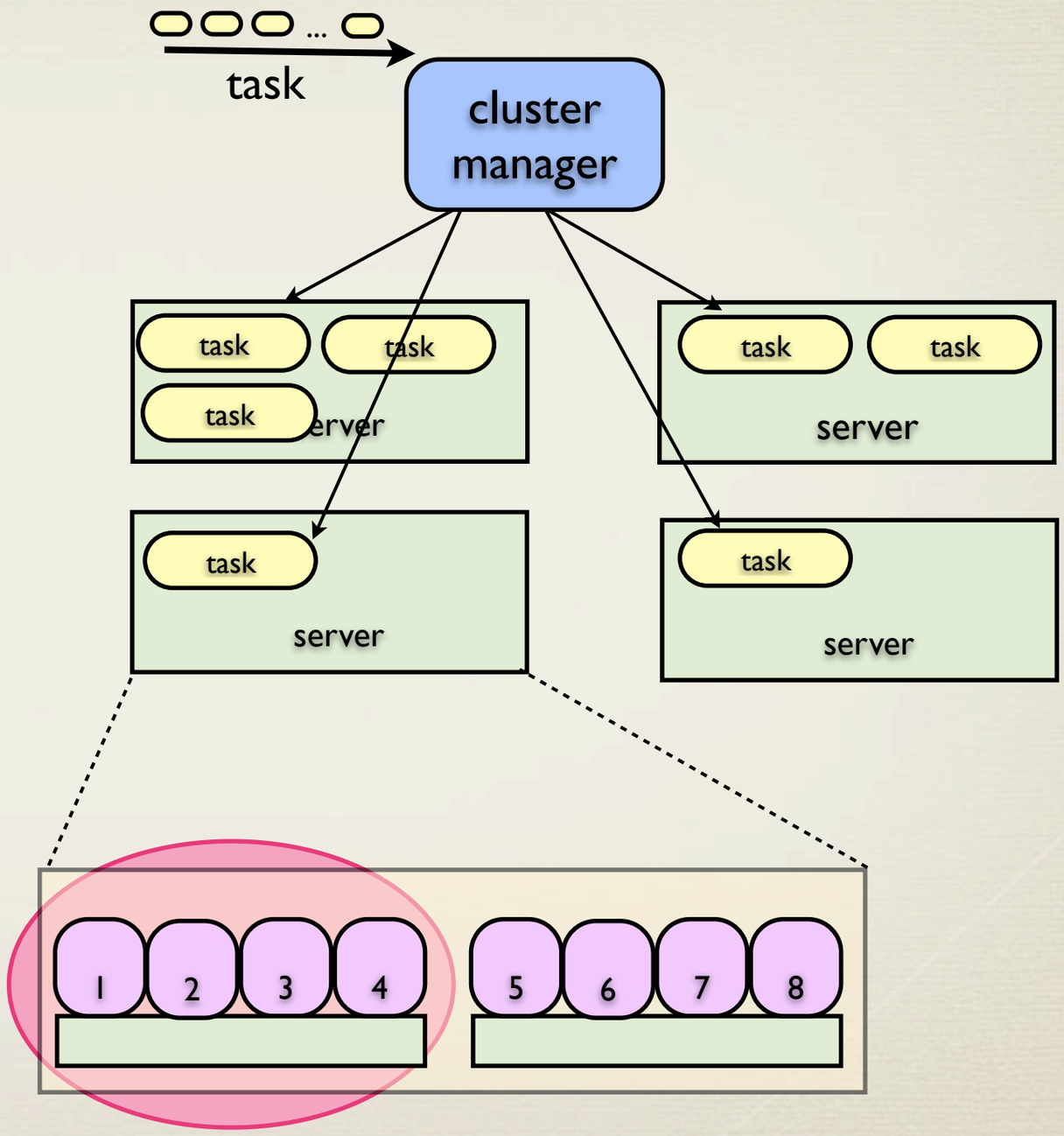
- Based on basic resource requirements.
- Applications
  - may run alone
  - may co-located with others
- Cache and bandwidth topology oblivious



Lingjia Tang: lt8f@cs.virginia.edu

Background: Datacenter scheduling

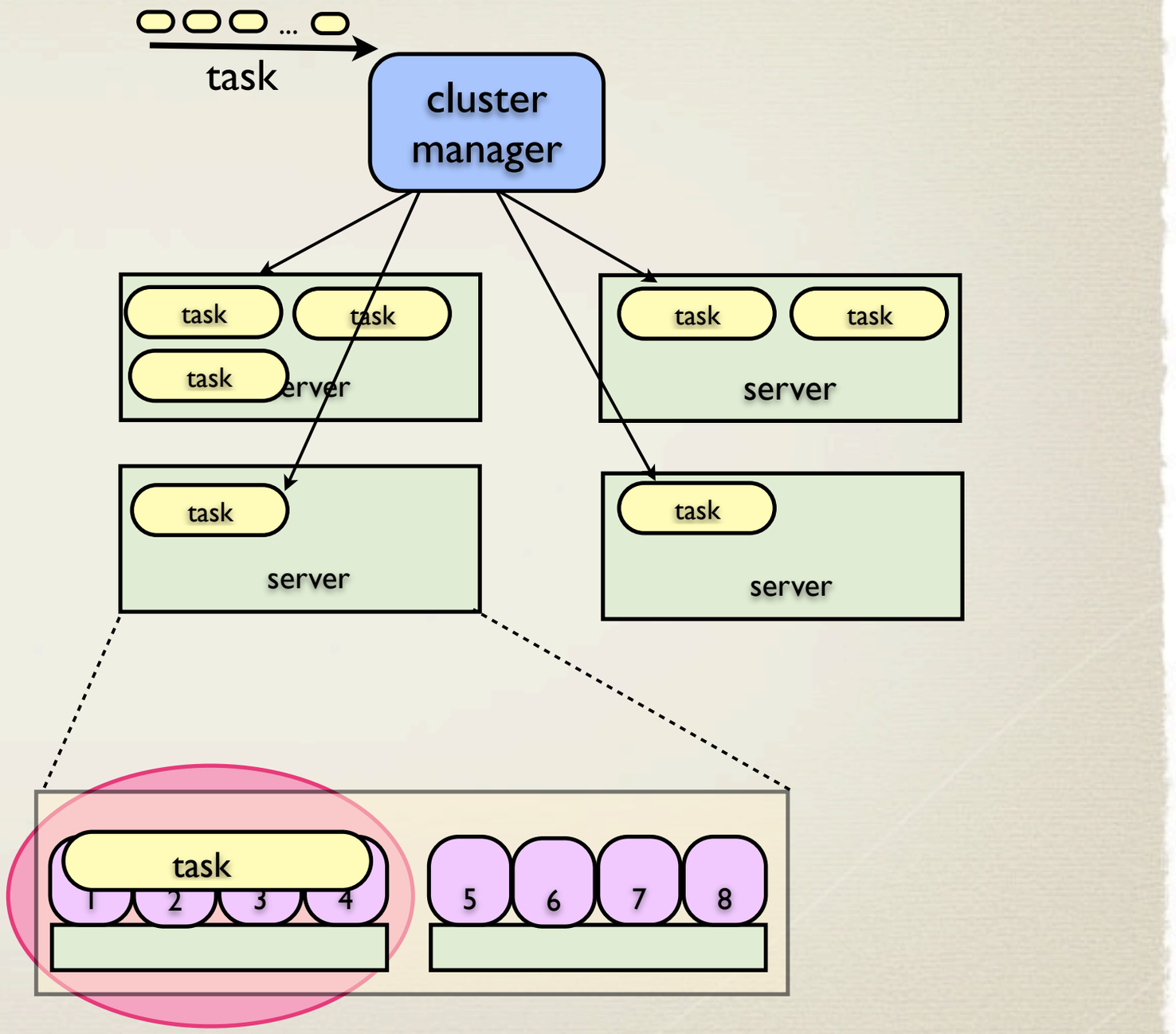
- Based on basic resource requirements.
- Applications
  - may run alone
  - may co-located with others
- Cache and bandwidth topology oblivious



Lingjia Tang: lt8f@cs.virginia.edu

Background: Datacenter scheduling

- Based on basic resource requirements.
- Applications
  - may run alone
  - may co-located with others
- Cache and bandwidth topology oblivious



Lingjia Tang: lt8f@cs.virginia.edu

Outline

- Problem, Motivation and Background
- Characterization
  - Intra-application Sharing
  - Inter-application Sharing
  - Varying number of threads and architectures
- Thread-to-Core Mapping
  - Heuristic-based mapping
  - Adaptive mapping

Lingjia Tang: lt8f@cs.virginia.edu

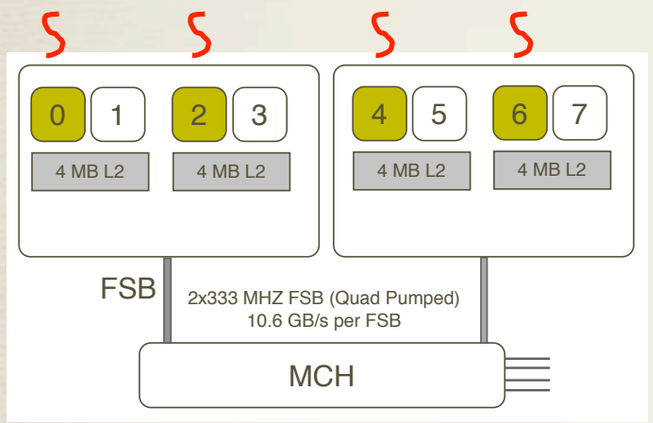
Benchmarks

application	description	metric	type
content-analyzer	content and semantic analysis, used to take key words or text documents and cluster them by their semantic meanings	throughput	latency-sensitive
bigtable	storage software for massive amount of data	average latency	latency-sensitive
websearch	industry-strength internet search engine	queries per second	latency-sensitive
stitcher	image processing and stitching, used for generating street views	N/A	batch
protobuf	protocol buffer	N/A	batch

Lingjia Tang: lt8f@cs.virginia.edu

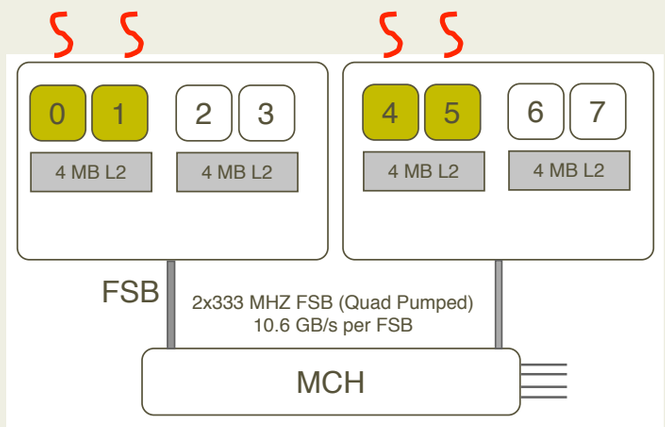
Intra-application Sharing

* Characterization Methodology
* 4 Threads, 3 Configurations



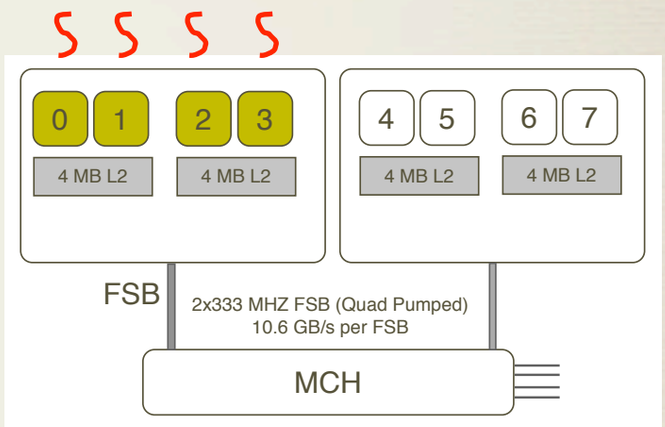
1

Separate Cache
Separate FSB
X.X.X.X.



2

Shared Cache
Separate FSB
XX..XX..



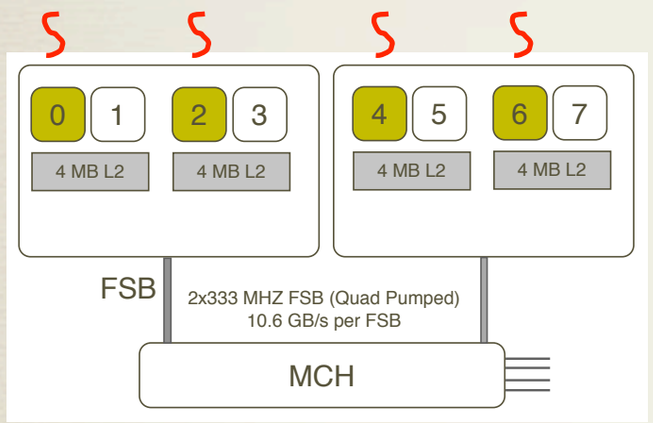
3

Shared Cache
Shared FSB
XXXX....

Lingjia Tang: lt8f@cs.virginia.edu

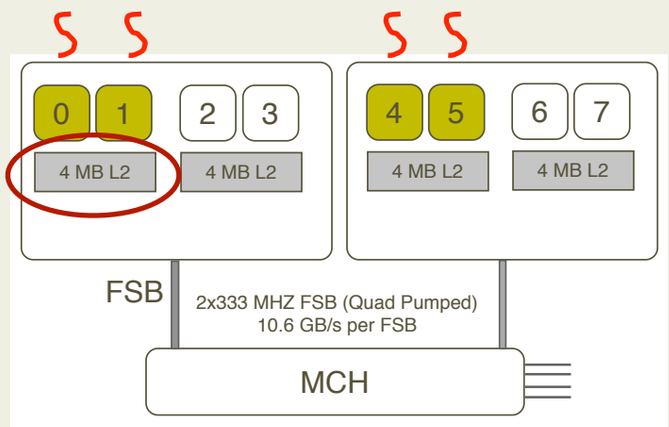
Intra-application Sharing

* Characterization Methodology
* 4 Threads, 3 Configurations



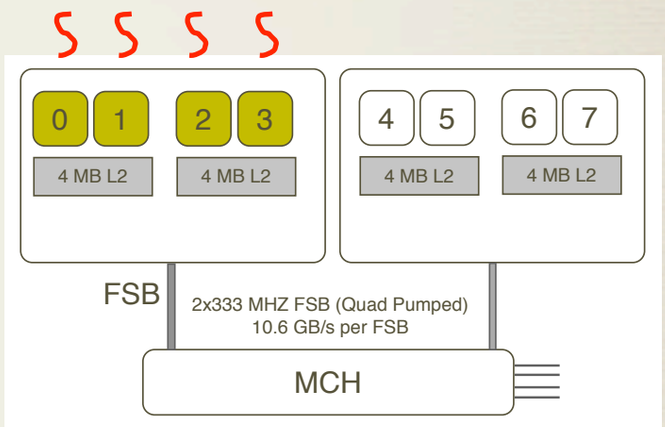
1

Separate Cache
Separate FSB
X.X.X.X.



2

Shared Cache
Separate FSB
XX..XX..



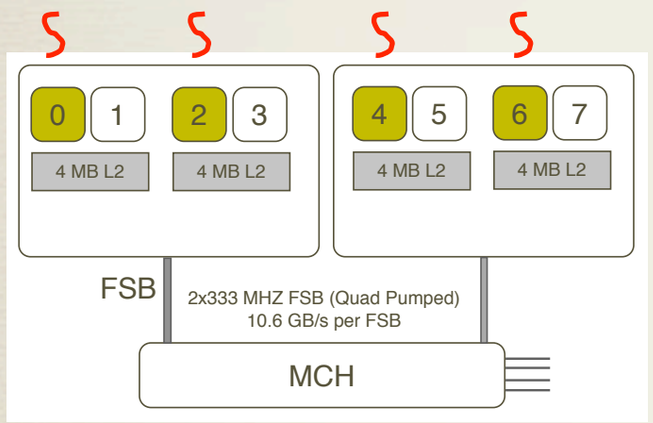
3

Shared Cache
Shared FSB
XXXX....

Lingjia Tang: lt8f@cs.virginia.edu

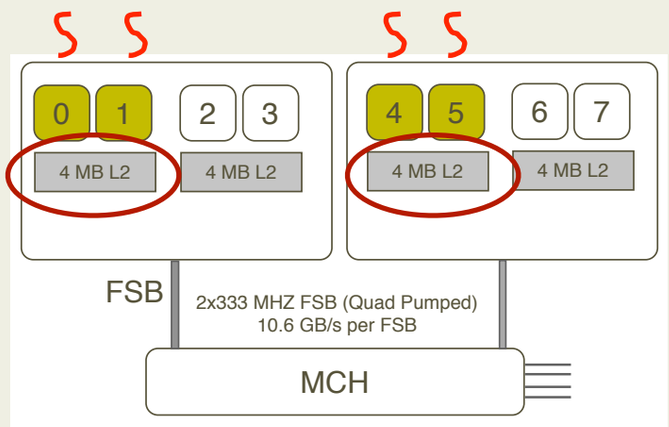
Intra-application Sharing

* Characterization Methodology
* 4 Threads, 3 Configurations



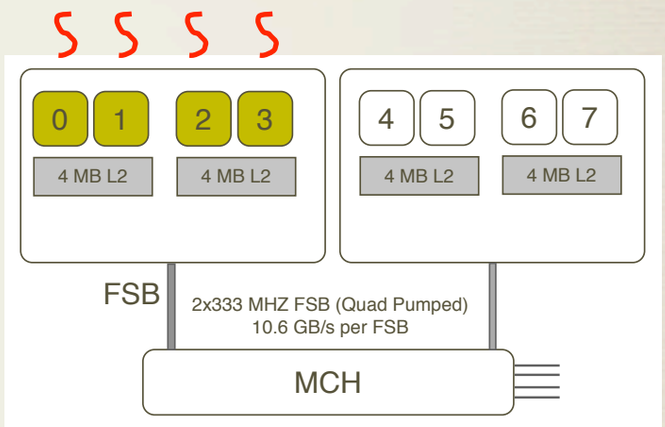
1

Separate Cache
Separate FSB
X.X.X.X.



2

Shared Cache
Separate FSB
XX..XX..



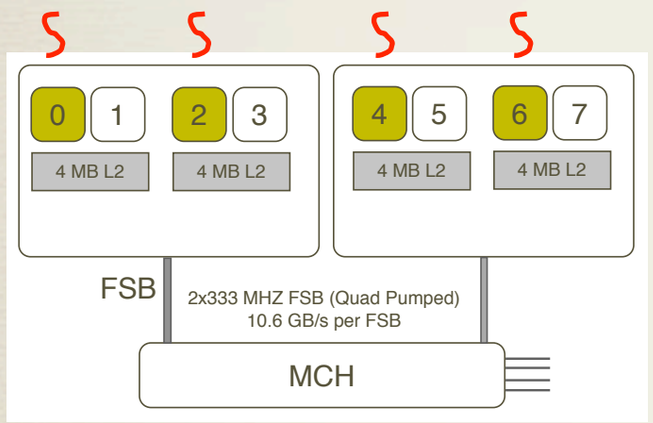
3

Shared Cache
Shared FSB
XXXX....

Lingjia Tang: lt8f@cs.virginia.edu

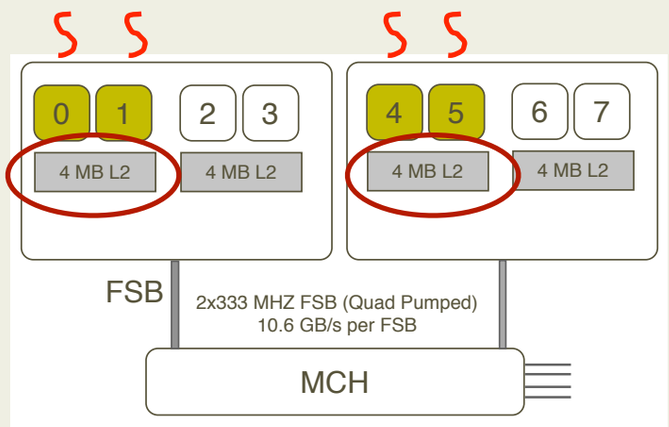
Intra-application Sharing

* Characterization Methodology
* 4 Threads, 3 Configurations



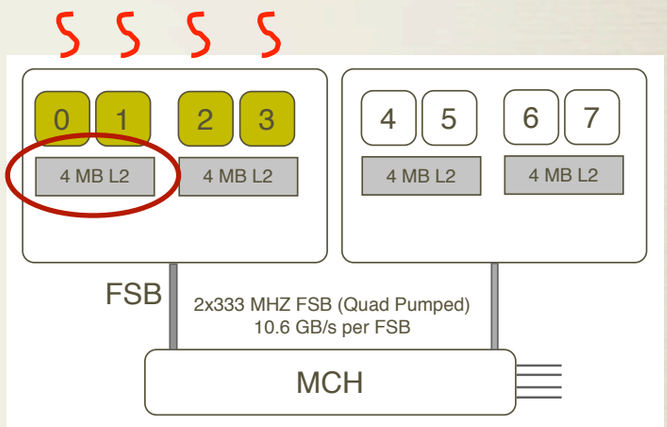
1

Separate Cache
Separate FSB
X.X.X.X.



2

Shared Cache
Separate FSB
XX..XX..



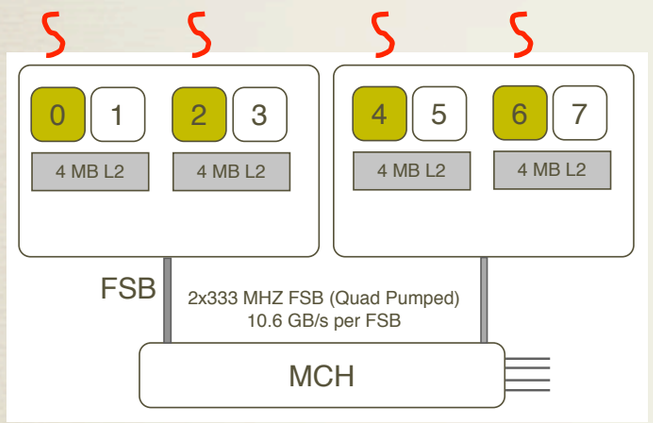
3

Shared Cache
Shared FSB
XXXX....

Lingjia Tang: lt8f@cs.virginia.edu

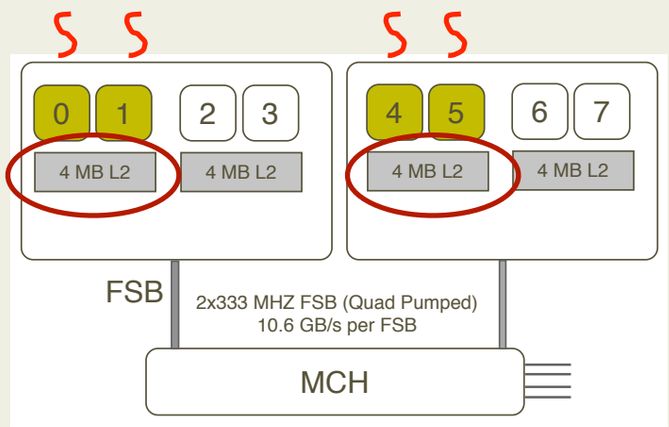
Intra-application Sharing

* Characterization Methodology
* 4 Threads, 3 Configurations



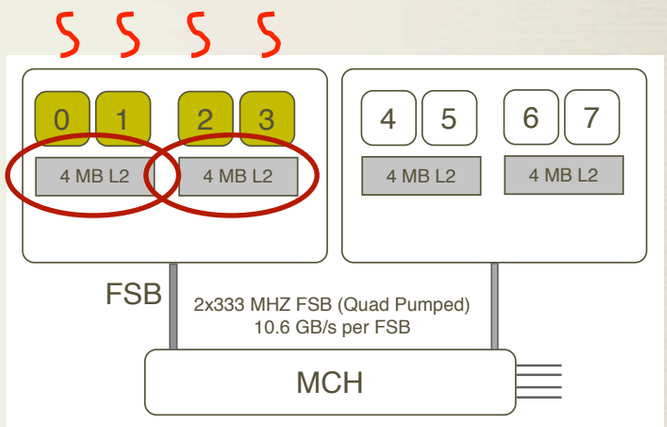
1

Separate Cache
Separate FSB
X.X.X.X.



2

Shared Cache
Separate FSB
XX..XX..



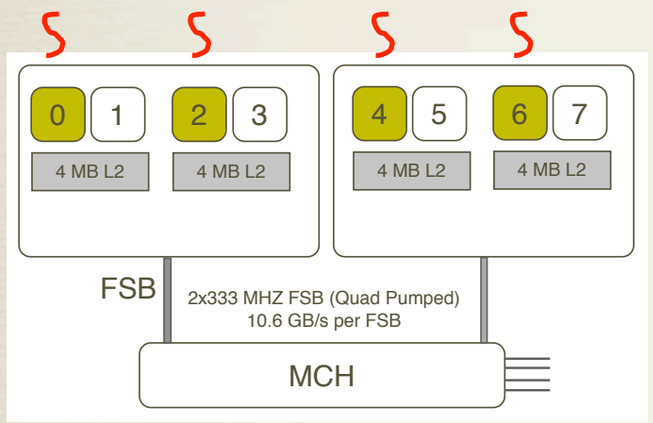
3

Shared Cache
Shared FSB
XXXX....

Lingjia Tang: lt8f@cs.virginia.edu

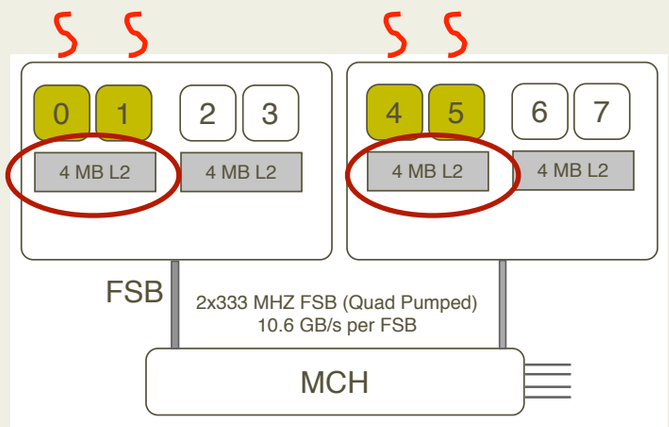
Intra-application Sharing

* Characterization Methodology
* 4 Threads, 3 Configurations



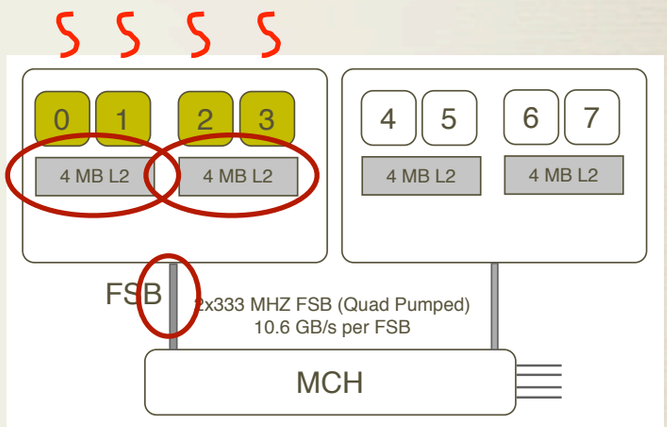
1

Separate Cache
Separate FSB
X.X.X.X.



2

Shared Cache
Separate FSB
XX..XX..

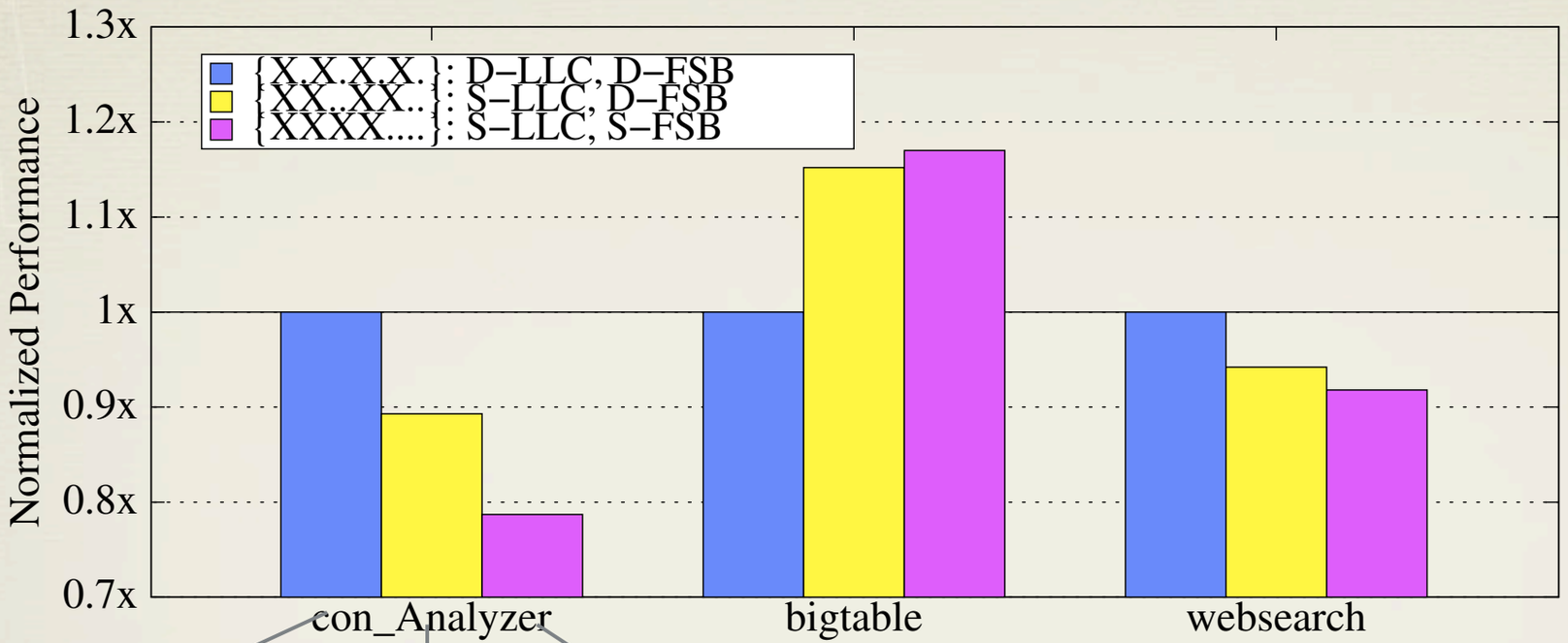


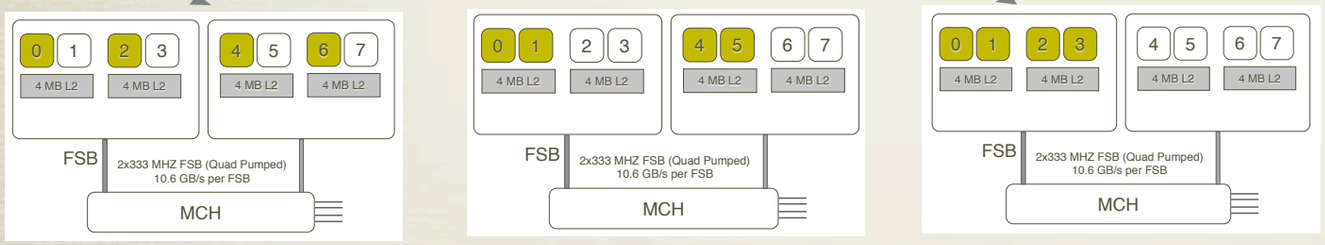
3

Shared Cache
Shared FSB
XXXX....

Lingjia Tang: lt8f@cs.virginia.edu

Intra-application Sharing

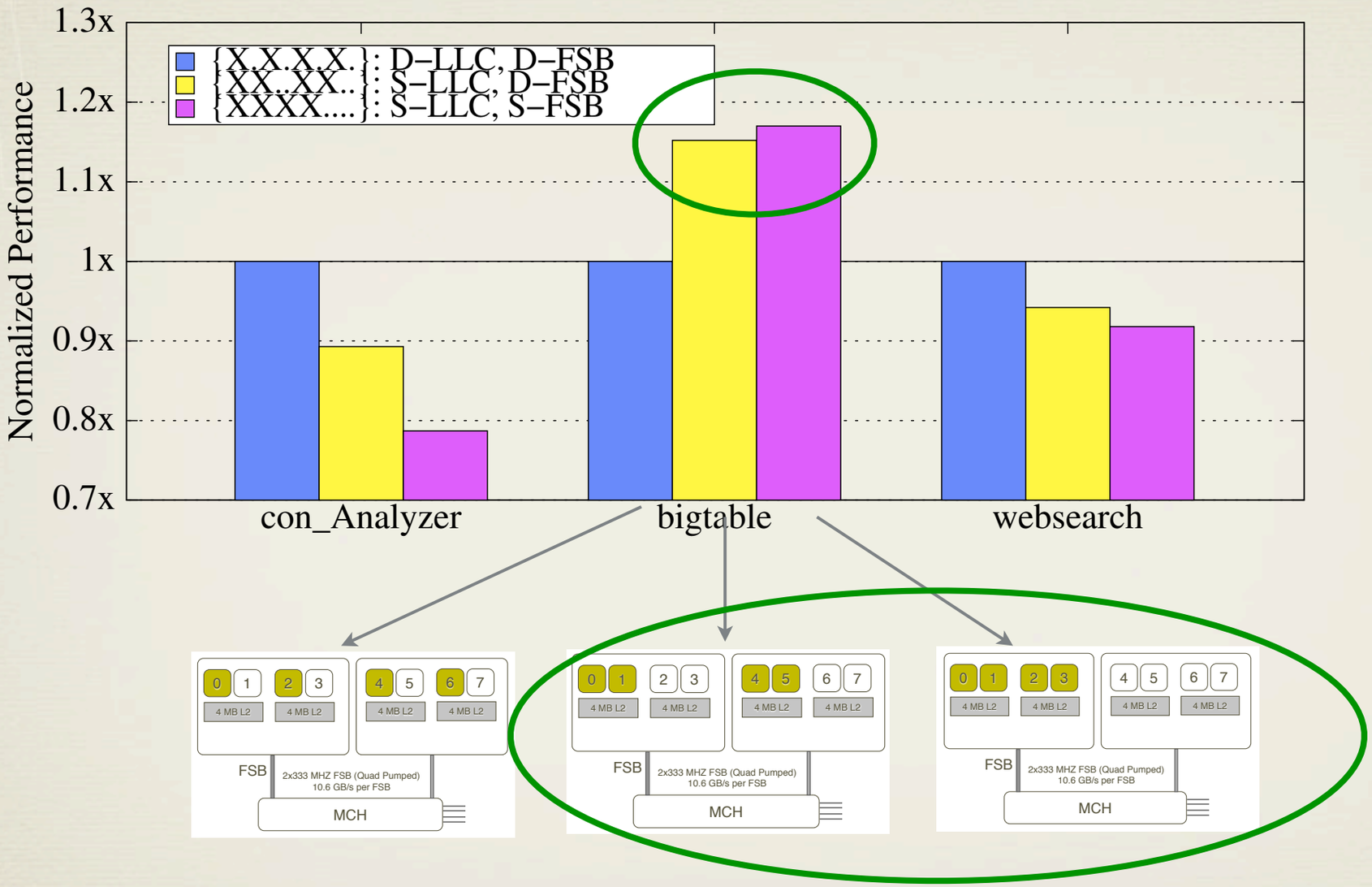




- Big performance swing (up to 22%)
- Both constructive and destructive

Lingjia Tang: lt8f@cs.virginia.edu

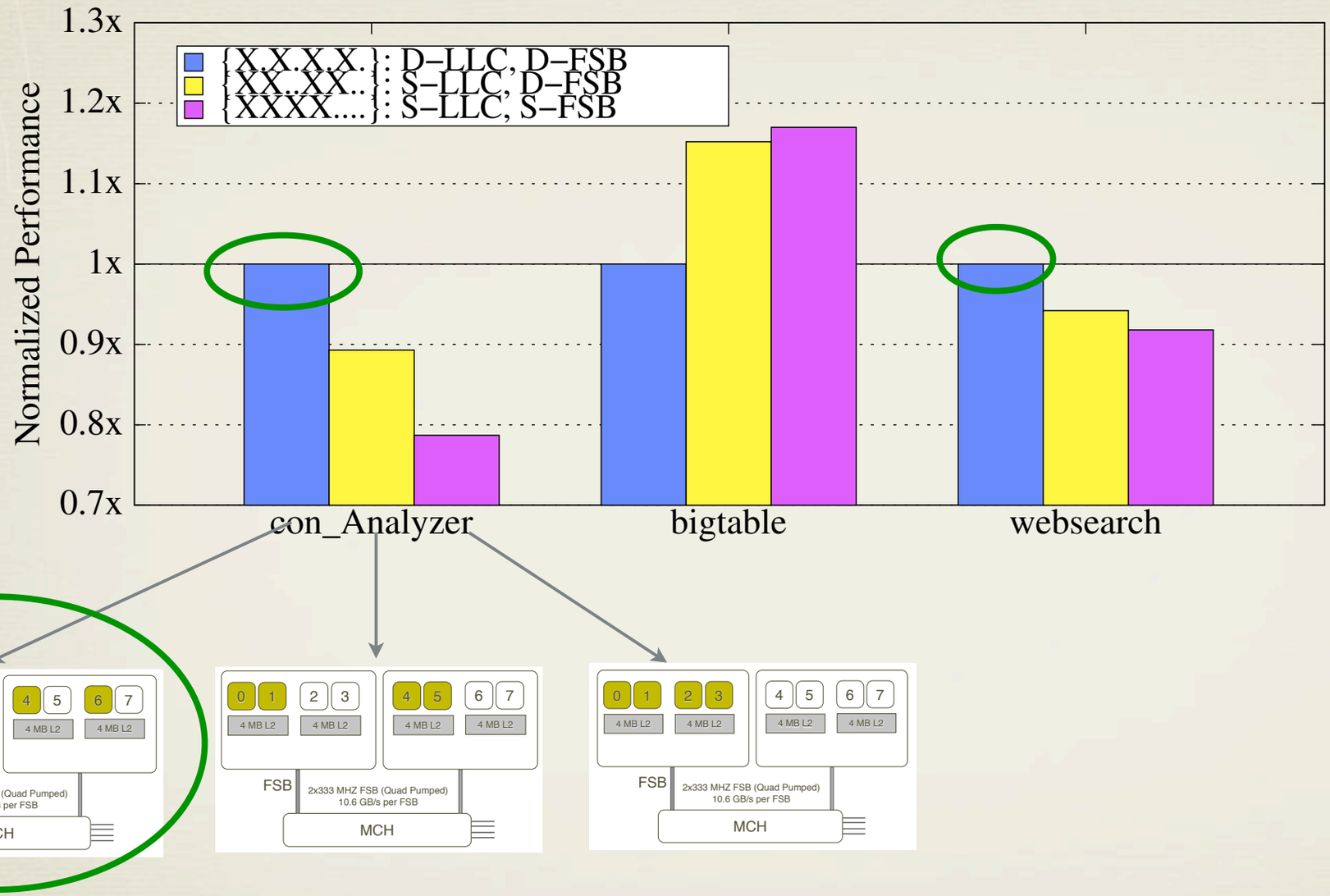
Intra-application Sharing



* *Bigtable* benefits from cache sharing

Lingjia Tang: lt8f@cs.virginia.edu

Intra-application Sharing



* *Con_Analyzer* and *websearch*: cache contention and bandwidth contention

Lingjia Tang: lt8f@cs.virginia.edu

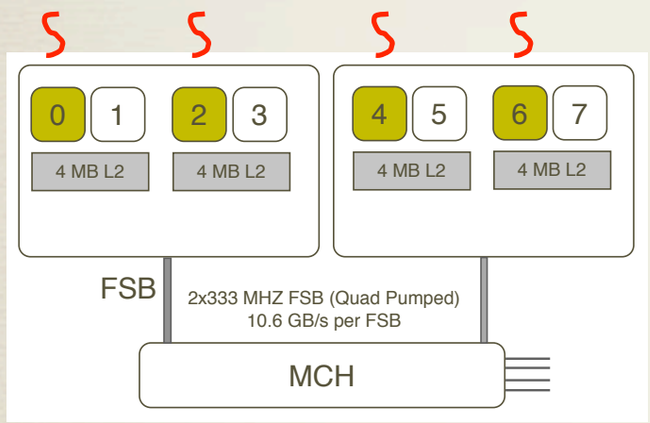
Inter-application Sharing

- Intra-application 
  - significant (22%)
  - both constructive and destructive
- Inter-application
  - Colocation of multiple applications
  - What's the impact of resource sharing between applications?

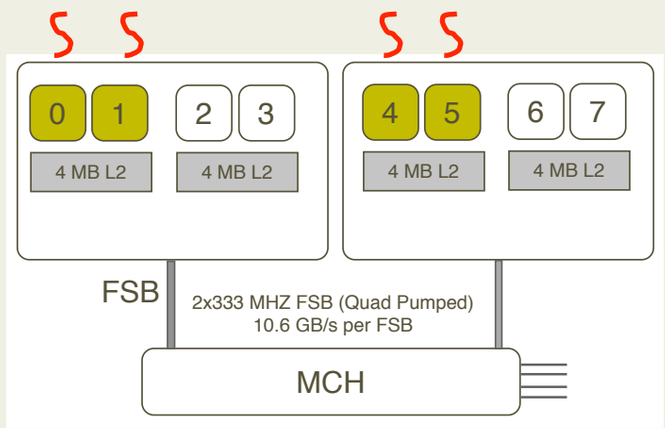
Lingjia Tang: lt8f@cs.virginia.edu

Inter-application Sharing

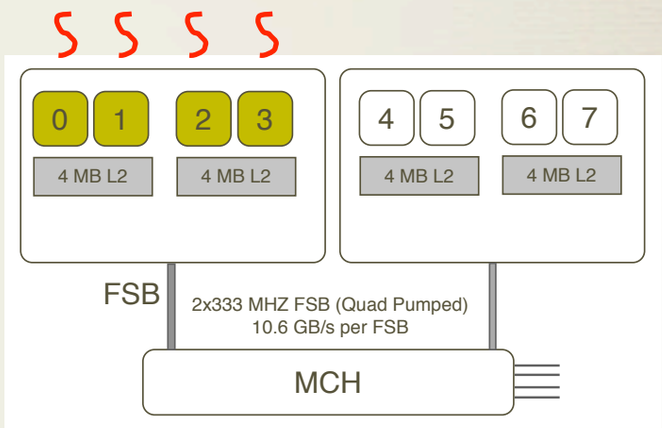
- Characterization Methodology
- 2 Applications, 8 Threads (4/4), 3 Configurations



X*X*X*X*



XX**XX**



XXXX****

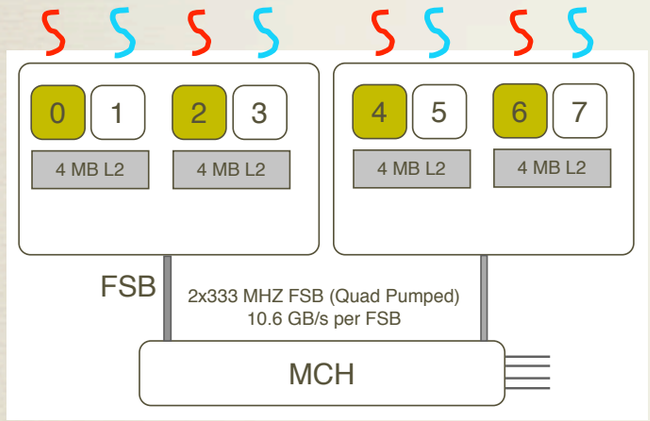
more → less

Resource sharing between applications

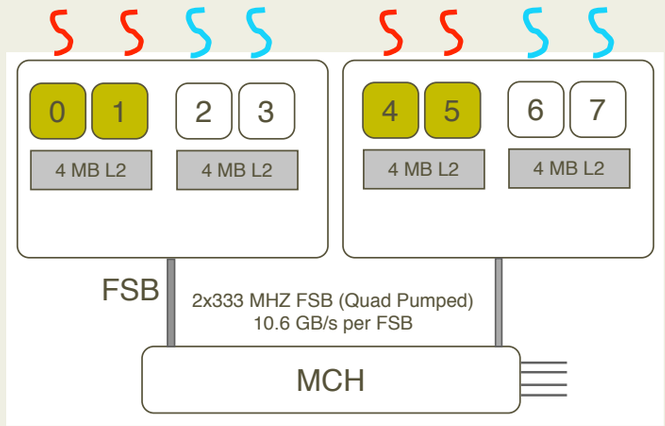
Lingjia Tang: lt8f@cs.virginia.edu

Inter-application Sharing

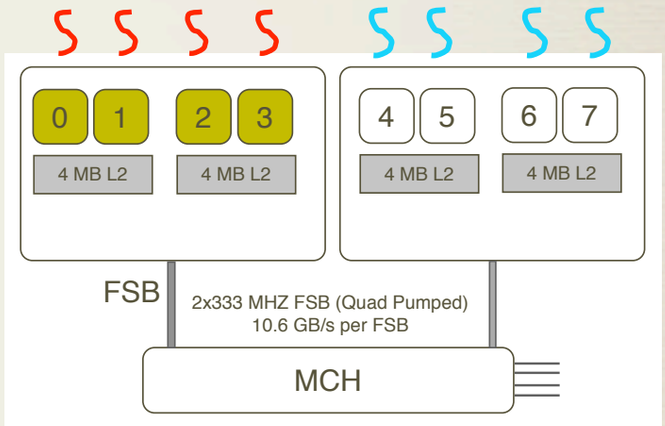
- Characterization Methodology
- 2 Applications, 8 Threads (4/4), 3 Configurations



X*X*X*X*



XX**XX**



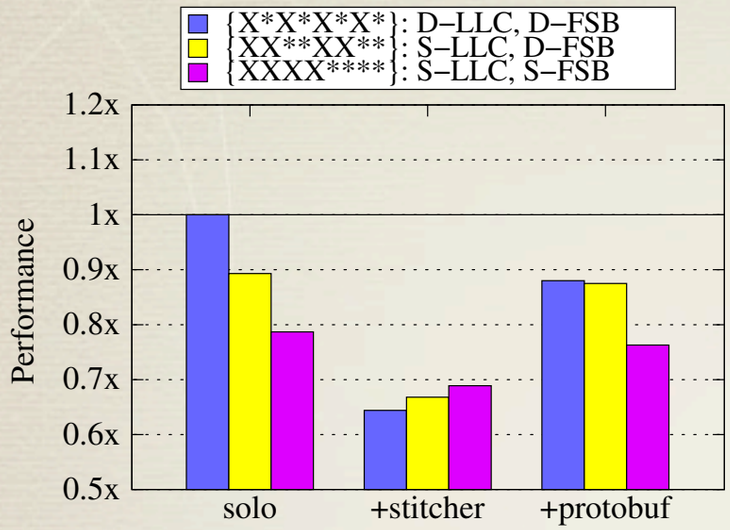
XXXX****

more → less

Resource sharing between applications

Lingjia Tang: lt8f@cs.virginia.edu

Inter-application Sharing

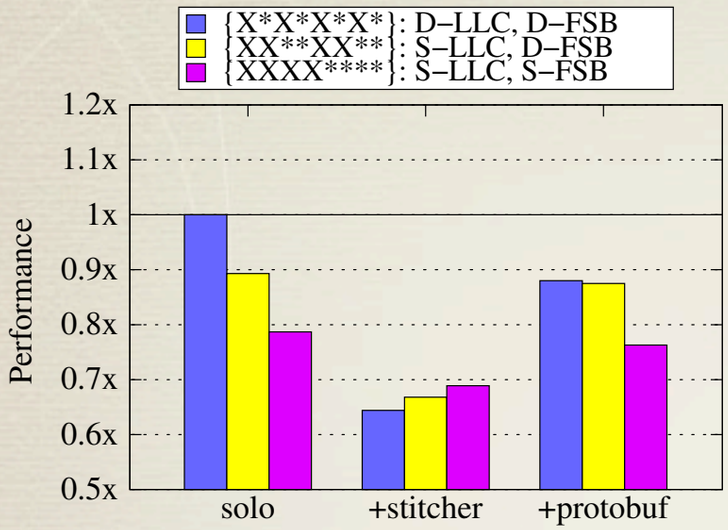


Content Analyzer

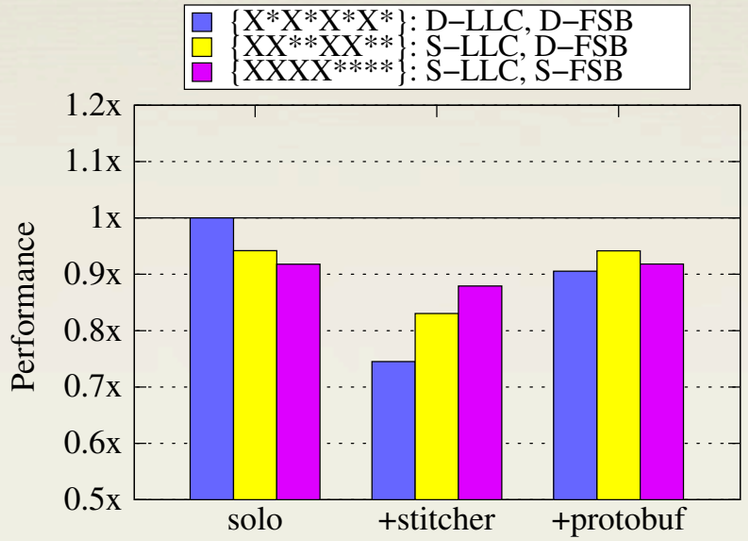
* Performance Degradation

Lingjia Tang: lt8f@cs.virginia.edu

Inter-application Sharing



Content Analyzer

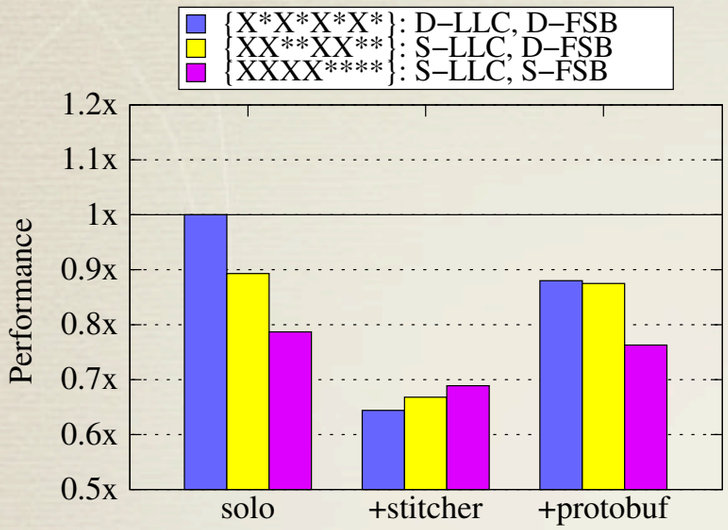


Websearch

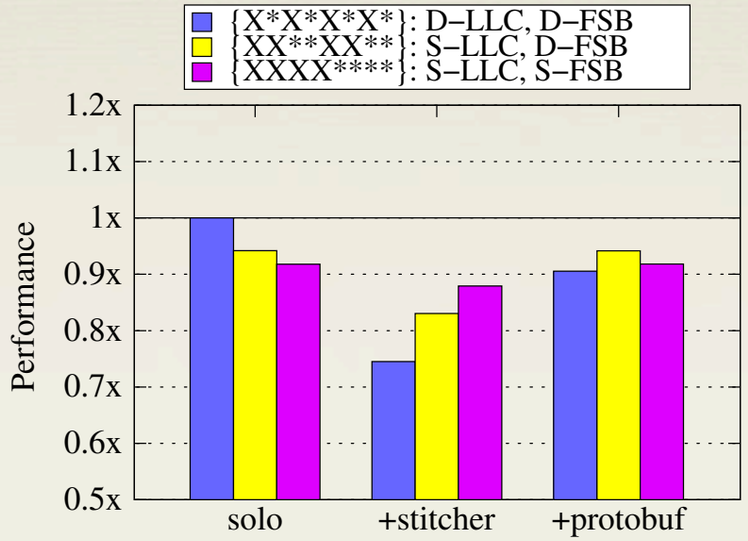
- Performance Degradation

Lingjia Tang: lt8f@cs.virginia.edu

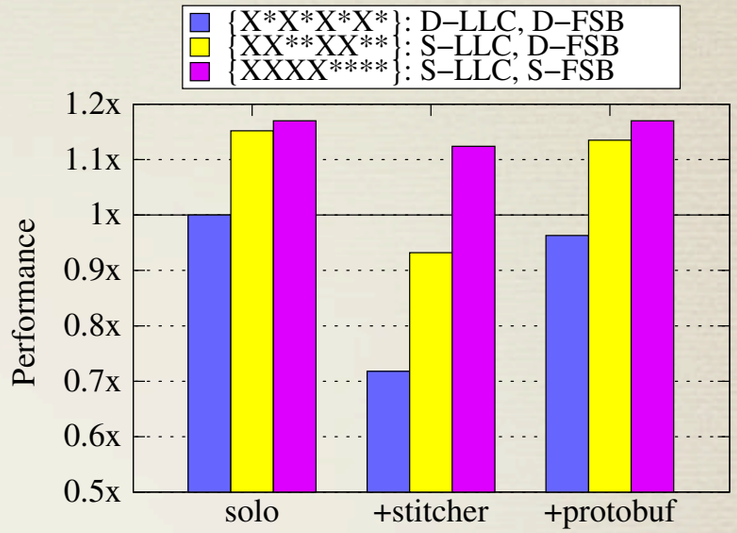
Inter-application Sharing



Content Analyzer



Websearch

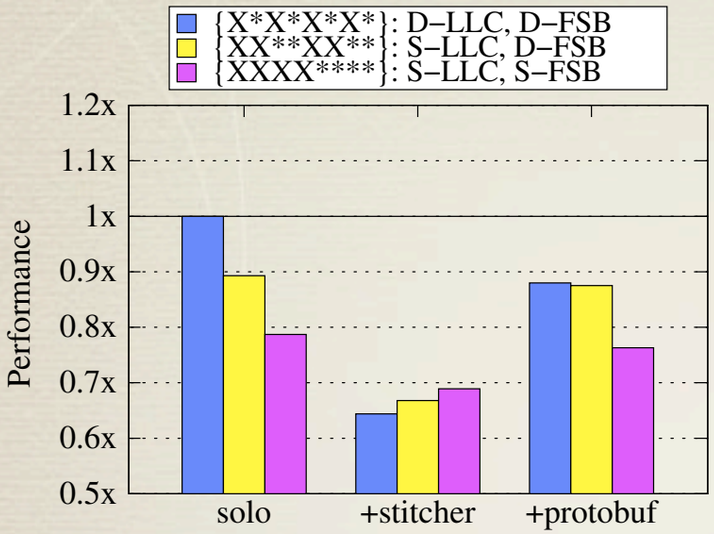


Bigtable

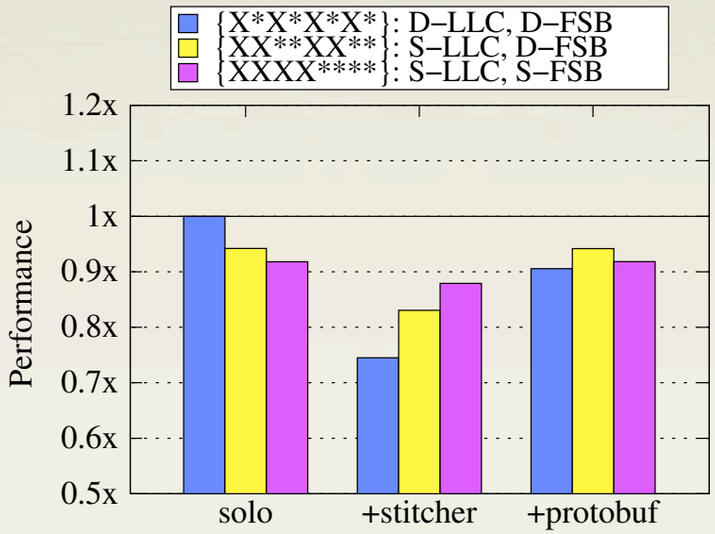
- Performance Degradation

Lingjia Tang: lt8f@cs.virginia.edu

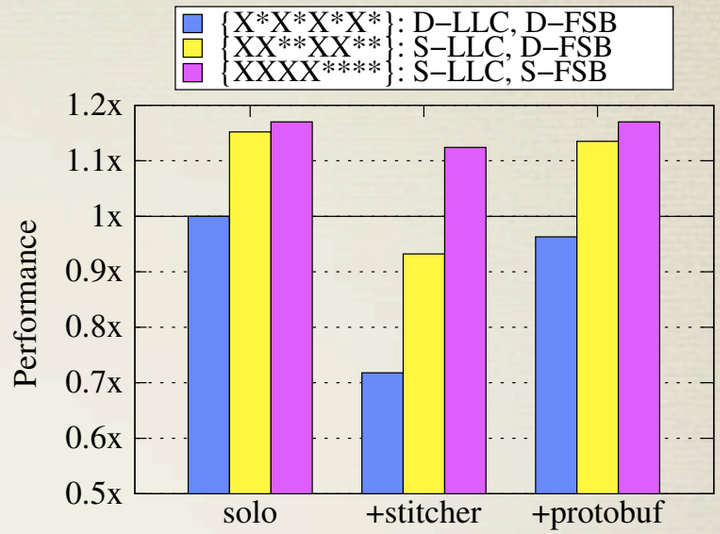
Inter-application Sharing



Content Analyzer



Websearch

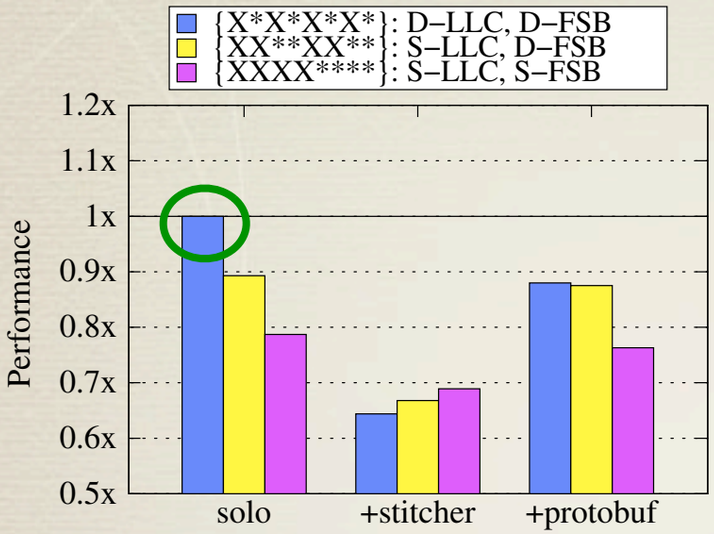


Bigtable

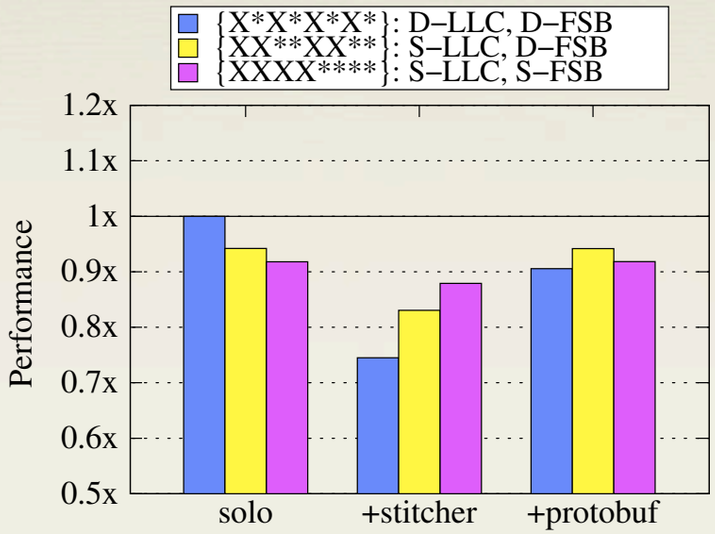
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

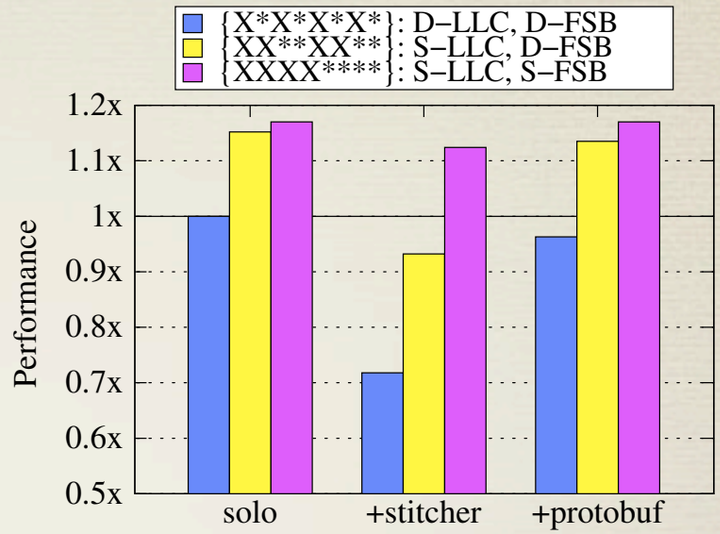
Inter-application Sharing



Content Analyzer



Websearch

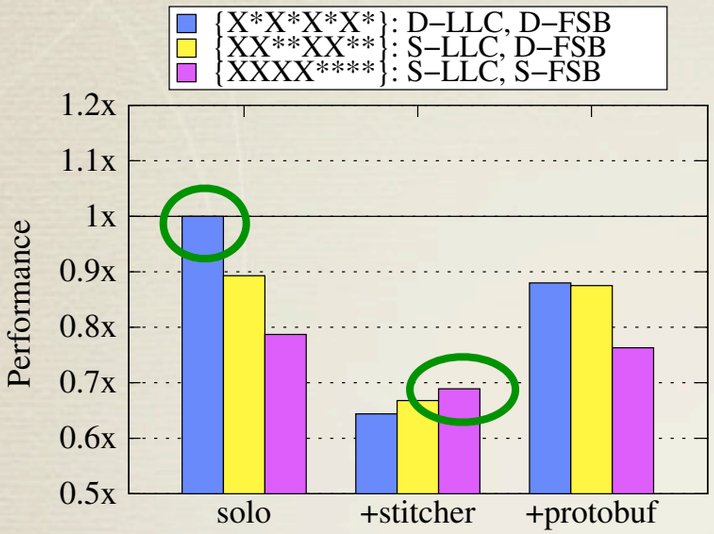


Bigtable

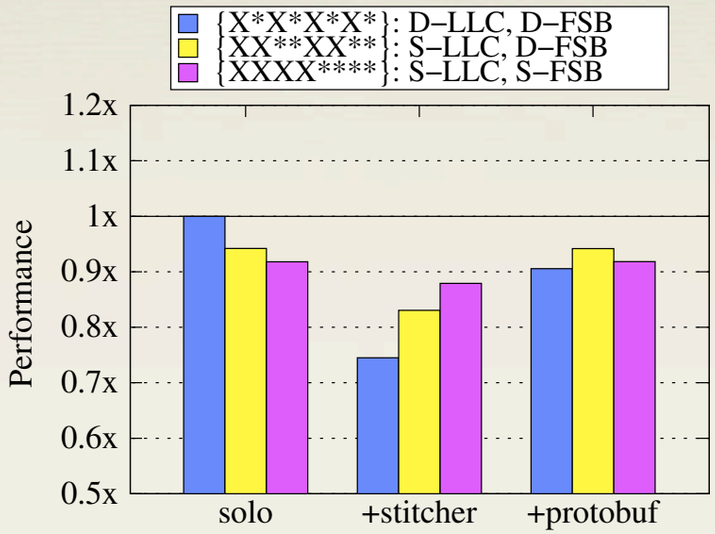
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

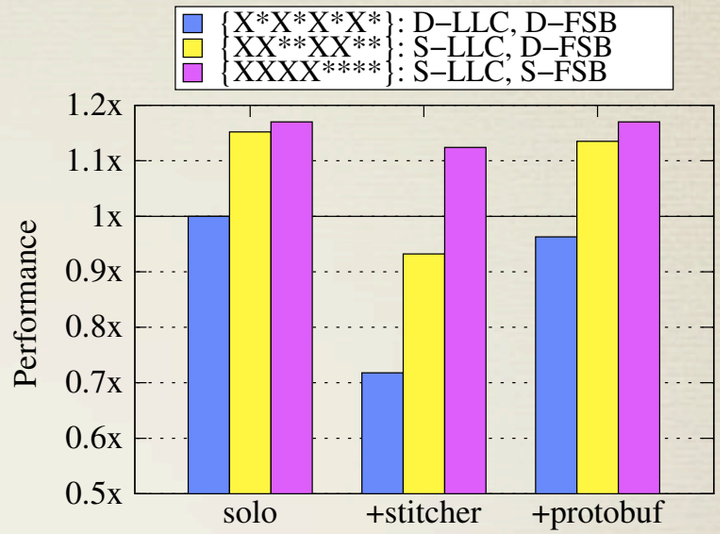
Inter-application Sharing



Content Analyzer



Websearch

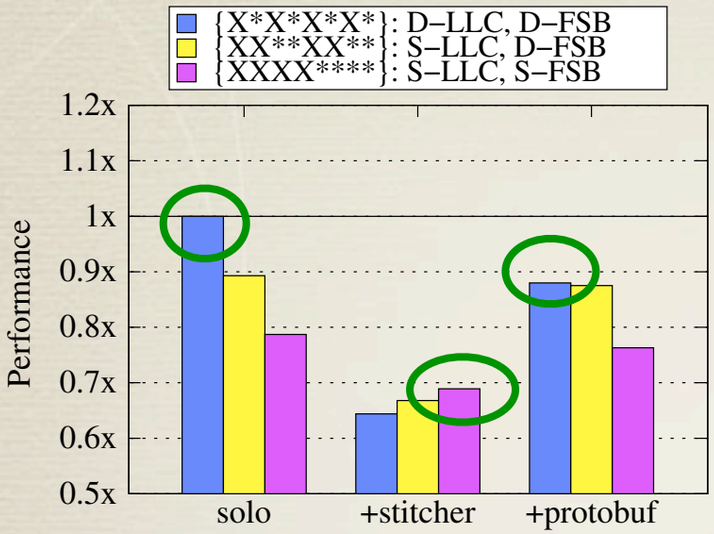


Bigtable

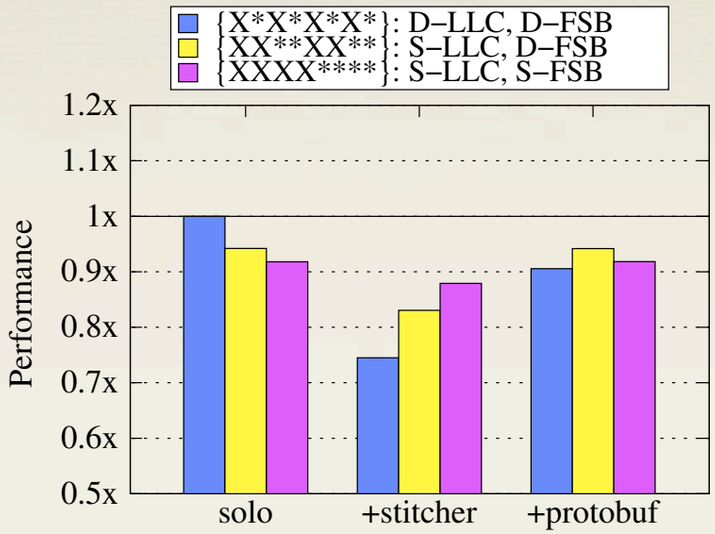
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

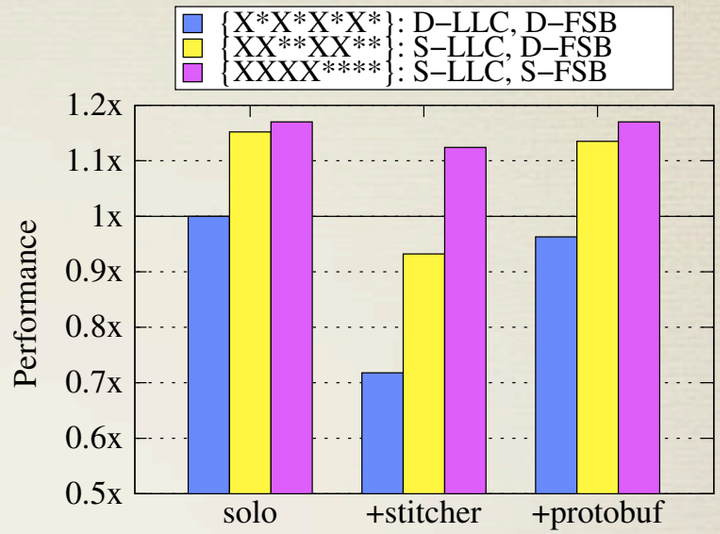
Inter-application Sharing



Content Analyzer



Websearch

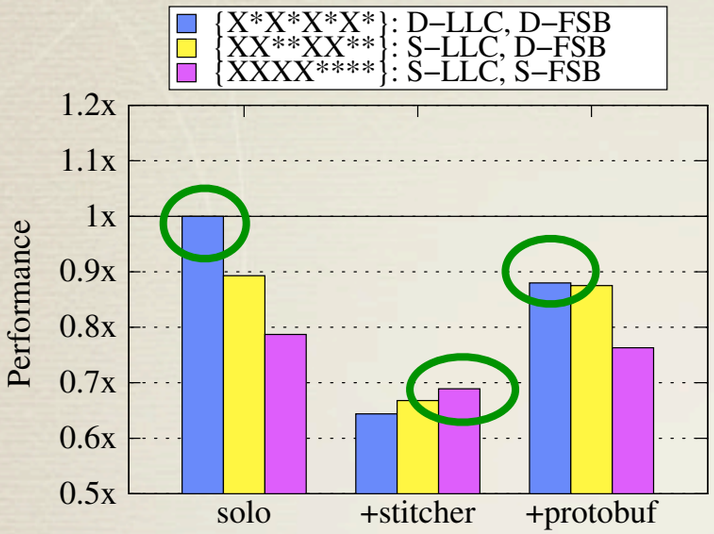


Bigtable

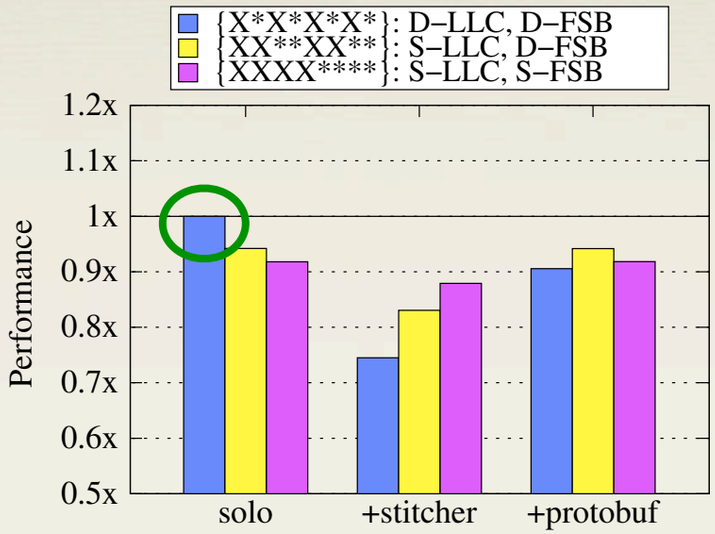
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

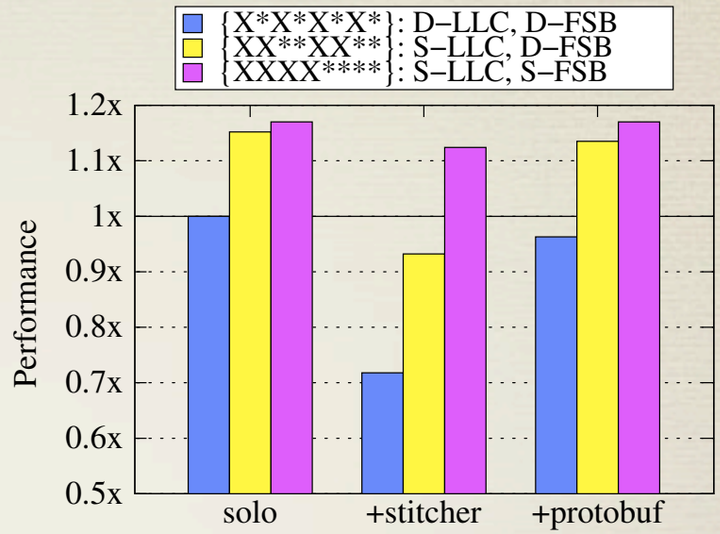
Inter-application Sharing



Content Analyzer



Websearch

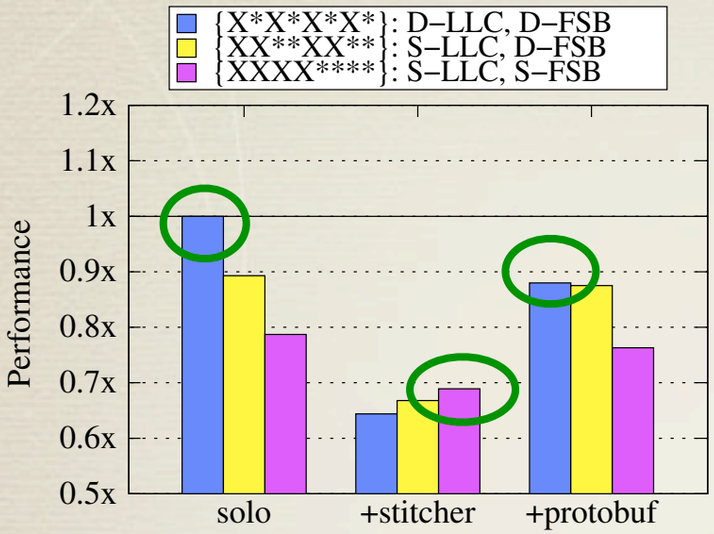


Bigtable

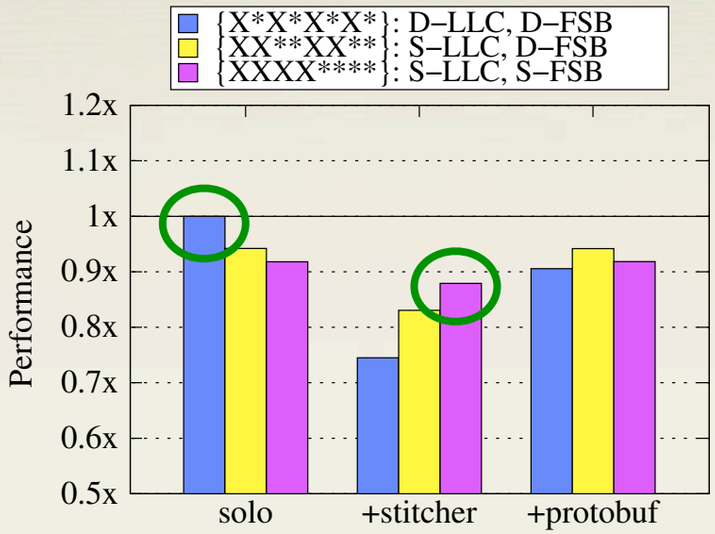
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

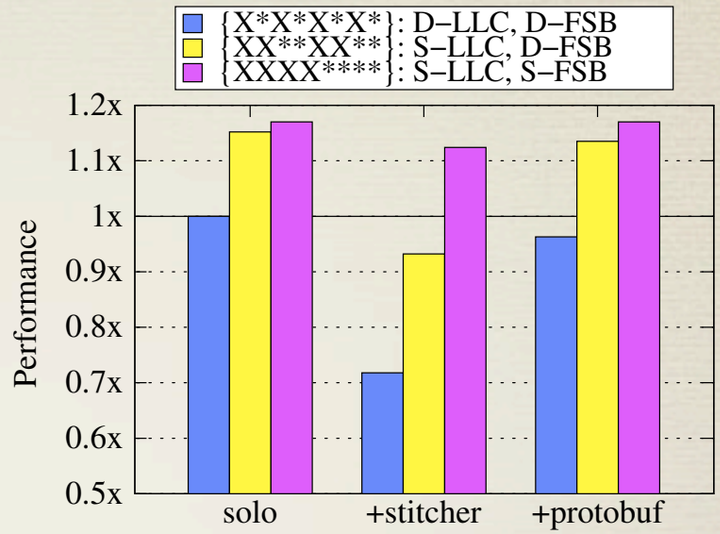
Inter-application Sharing



Content Analyzer



Websearch

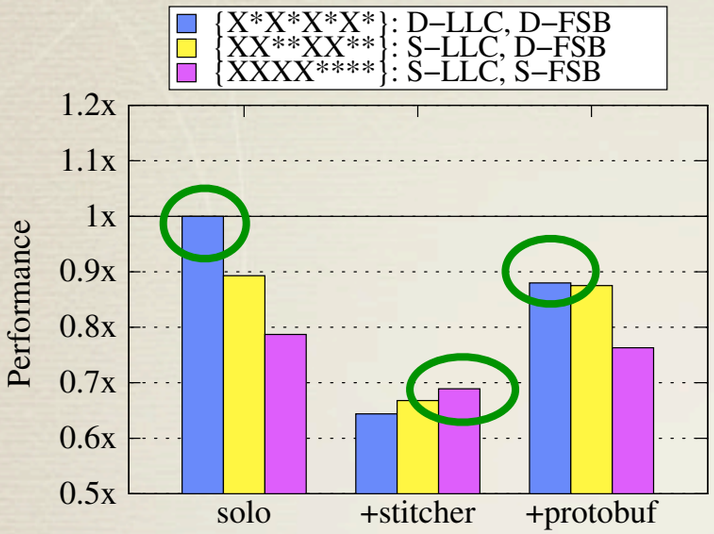


Bigtable

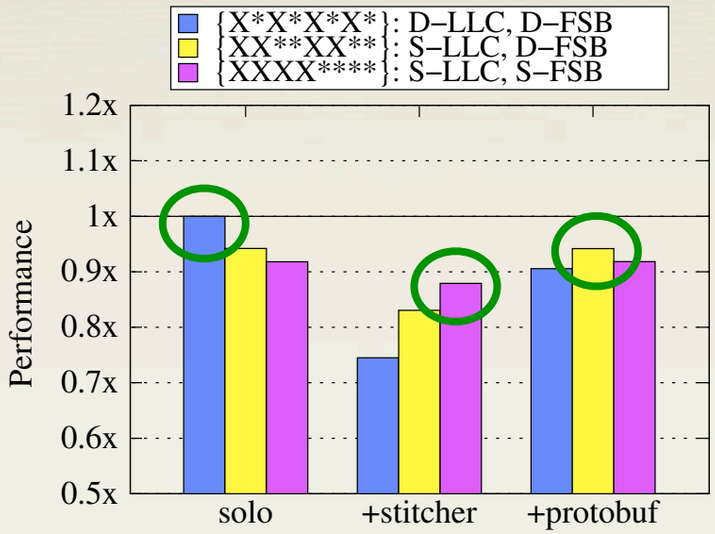
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

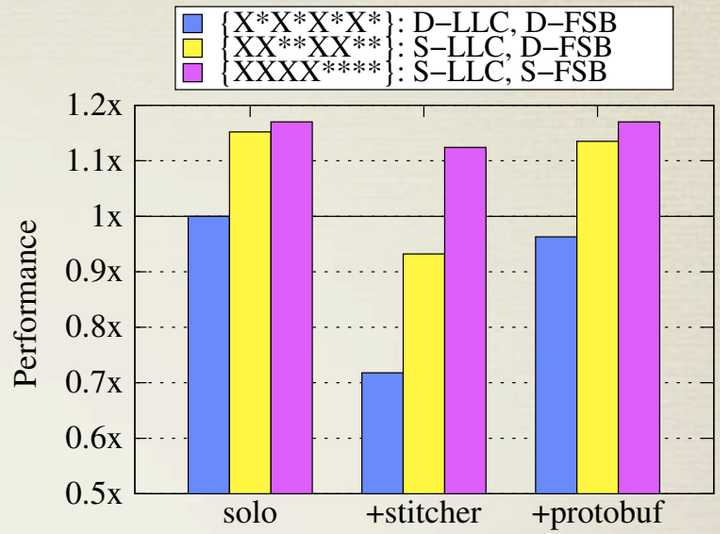
Inter-application Sharing



Content Analyzer



Websearch

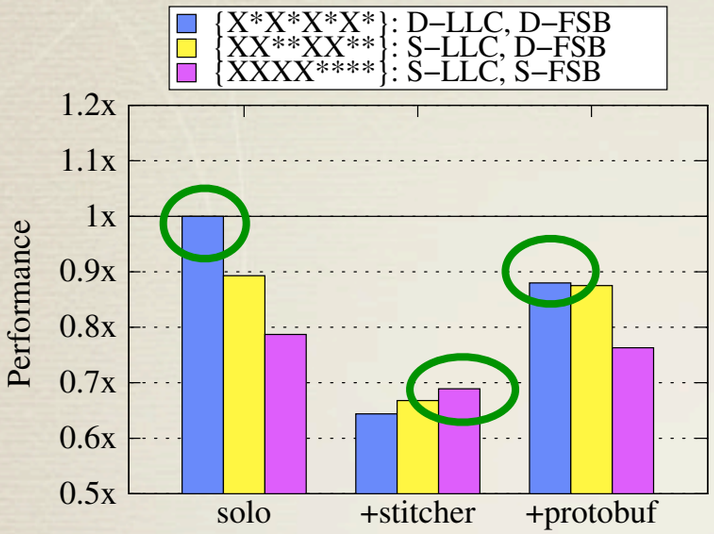


Bigtable

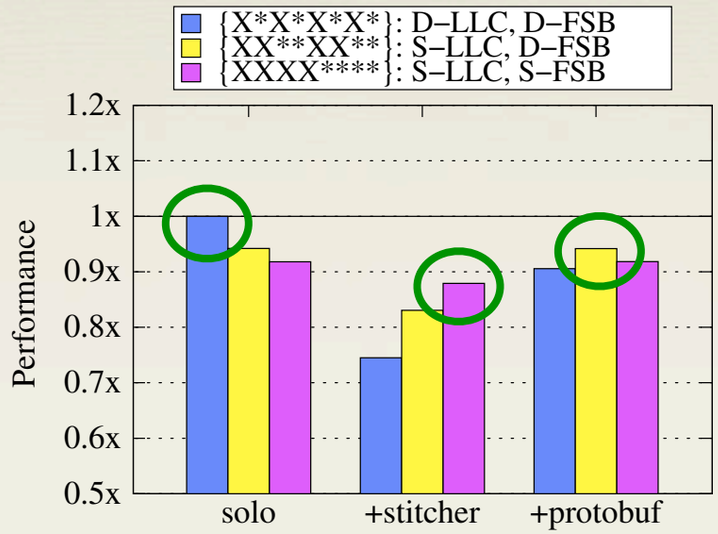
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

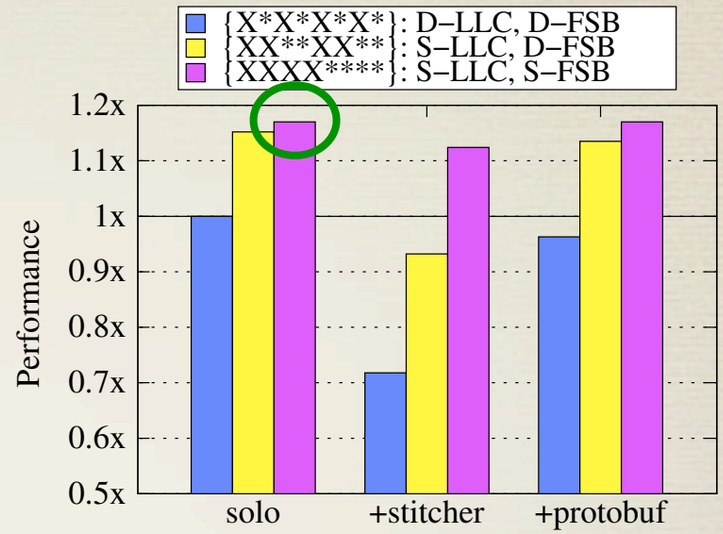
Inter-application Sharing



Content Analyzer



Websearch

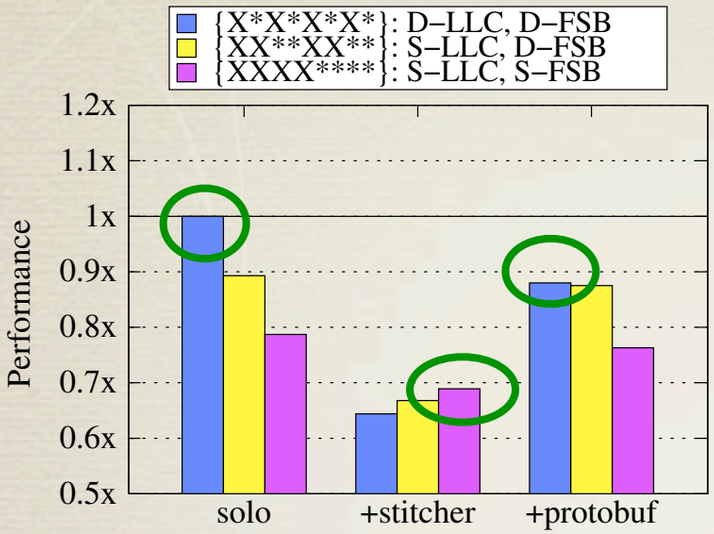


Bigtable

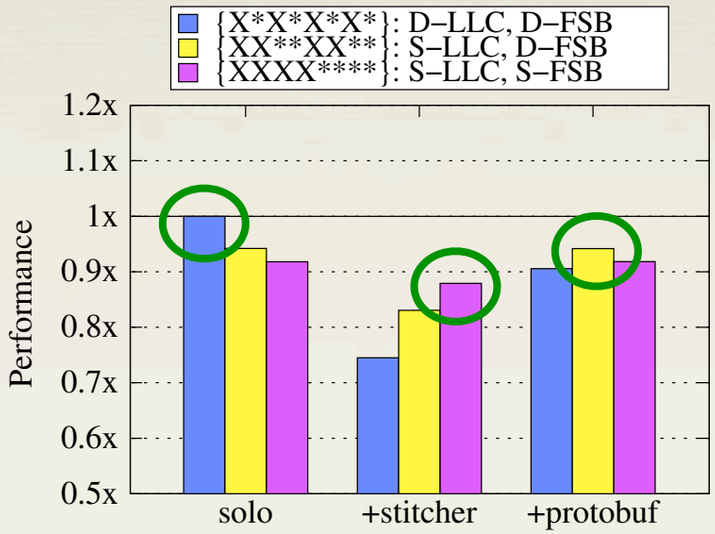
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

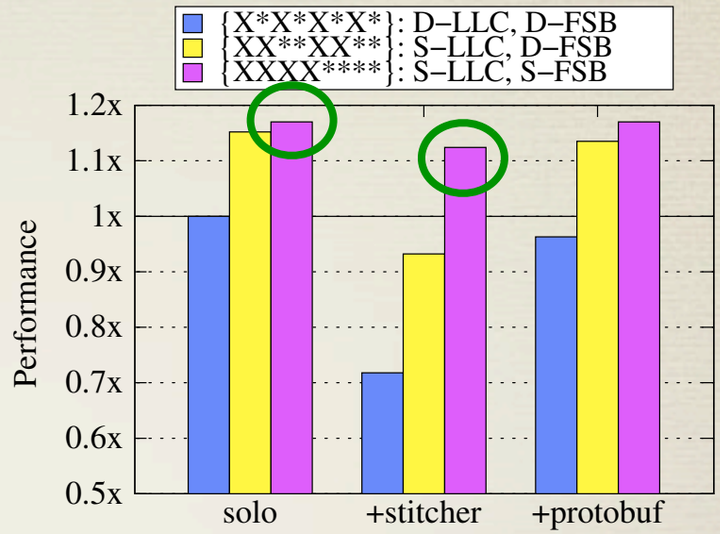
Inter-application Sharing



Content Analyzer



Websearch

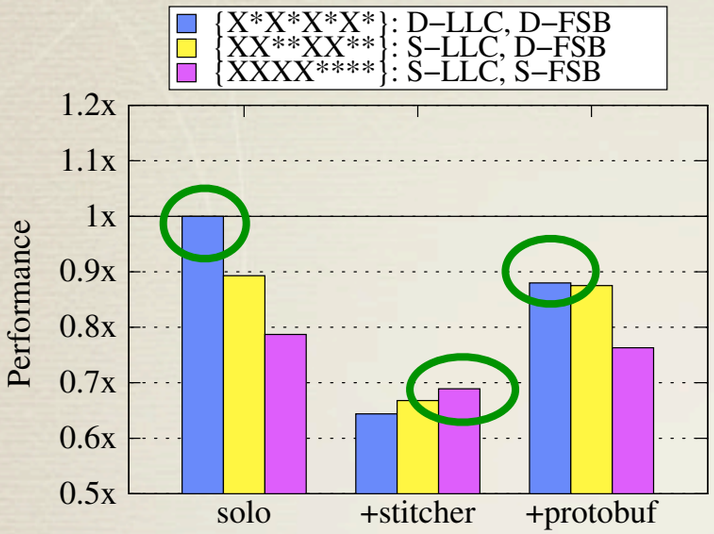


Bigtable

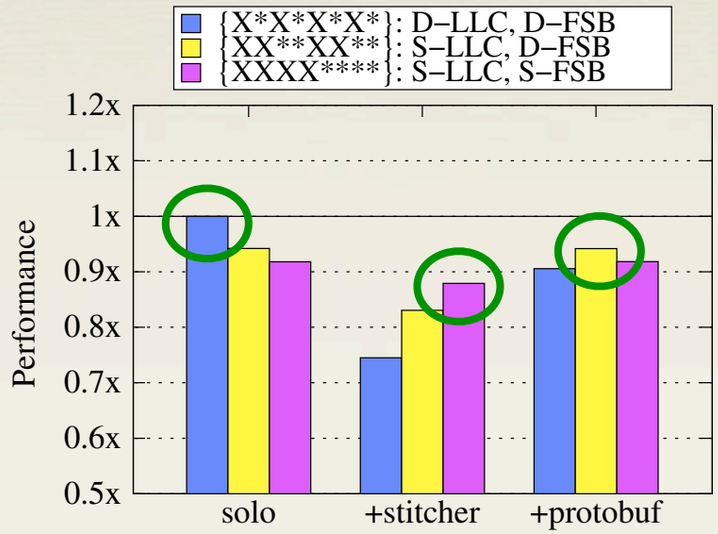
- Optimal mapping changes when co-runner changes
  - Difference can be significant

Lingjia Tang: lt8f@cs.virginia.edu

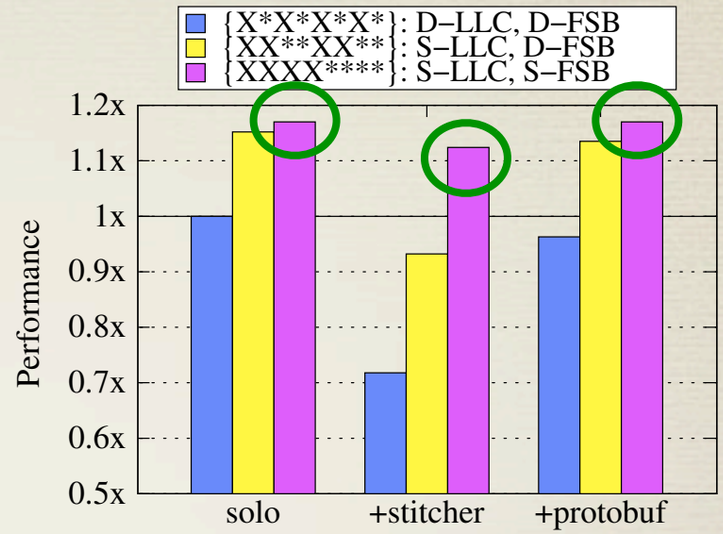
Inter-application Sharing



Content Analyzer



Websearch



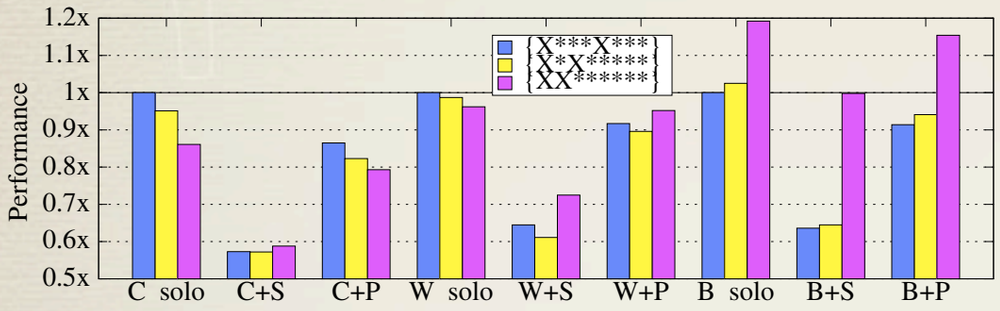
Bigtable

- Optimal mapping changes when co-runner changes
  - Difference can be significant

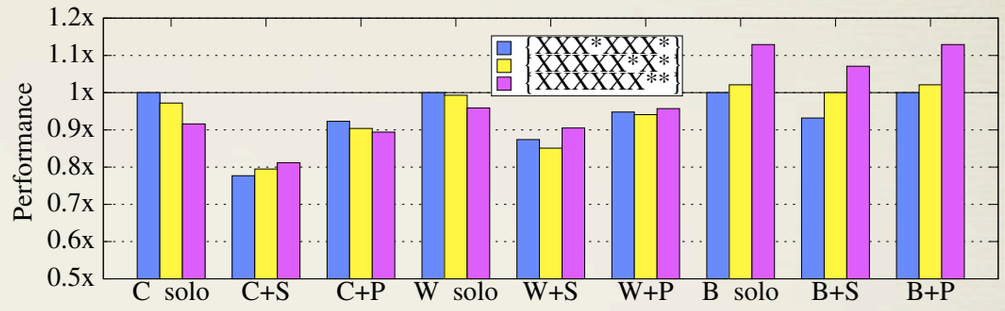
Lingjia Tang: lt8f@cs.virginia.edu

Varying Number of Threads and Architecture

Varying Thread Count

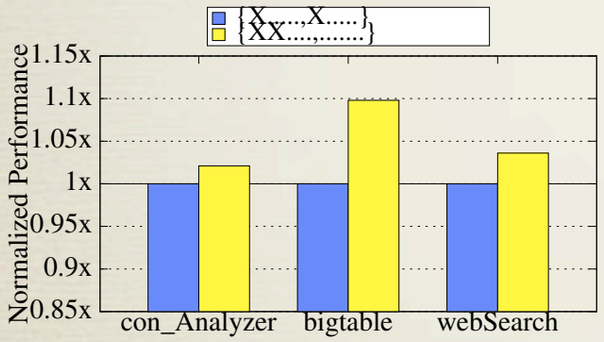


2 threads on Clovertown

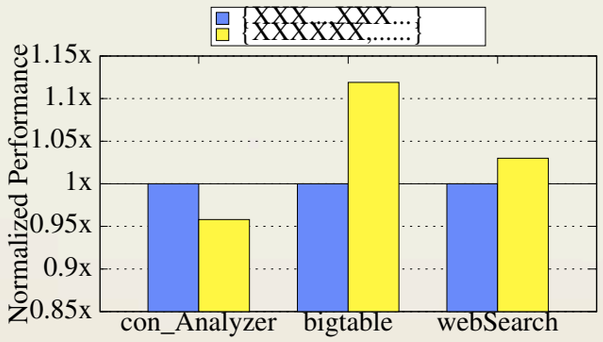


6 threads on Clovertown

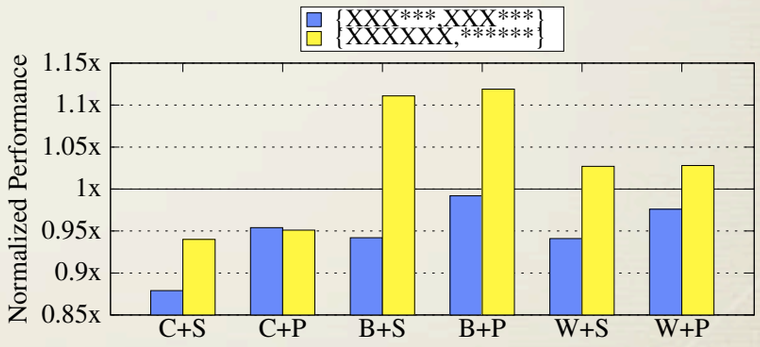
Varying Architecture



2 threads on Westmere



4 threads on Westmere



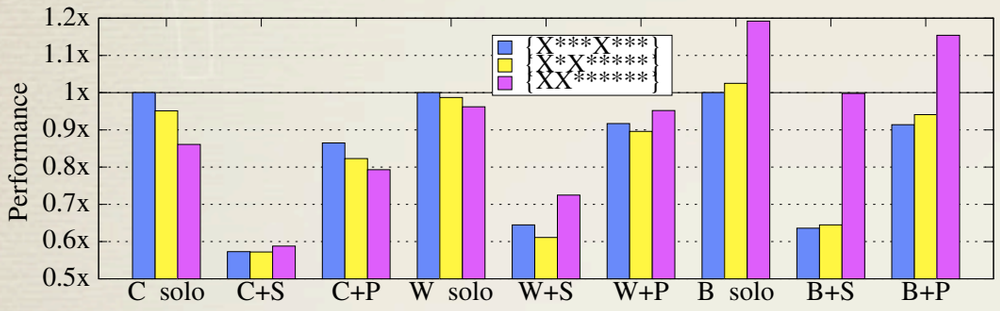
6 threads on Westmere

* Impact from sharing changes when architecture changes.

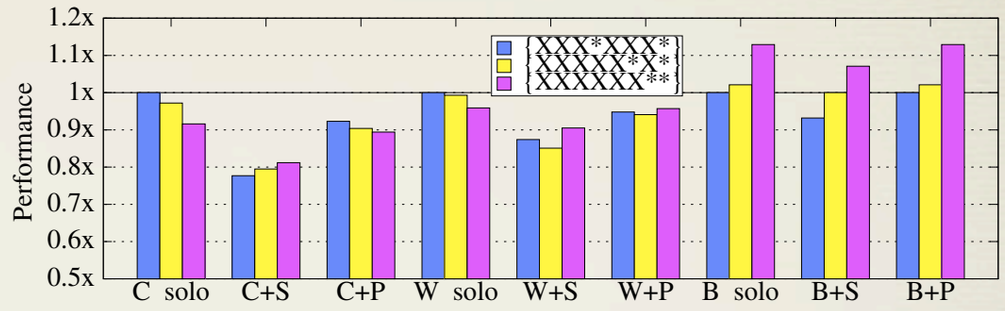
Lingjia Tang: lt8f@cs.virginia.edu

Varying Number of Threads and Architecture

Varying Thread Count

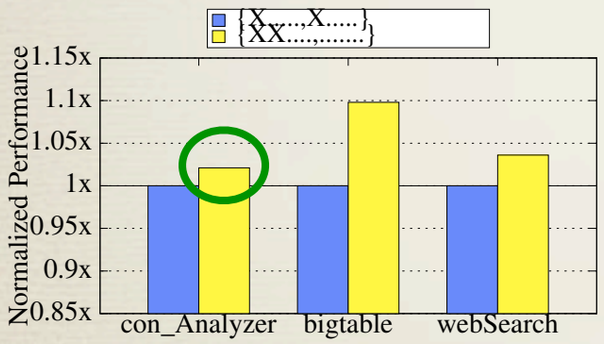


2 threads on Clovertown

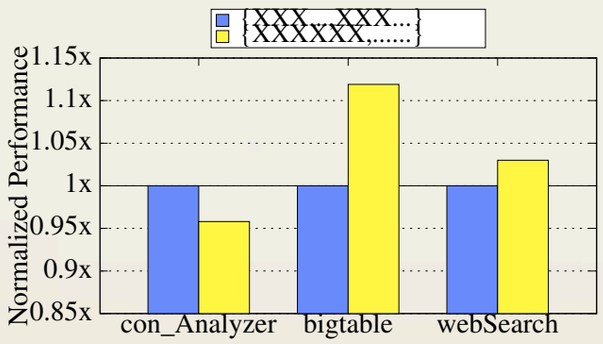


6 threads on Clovertown

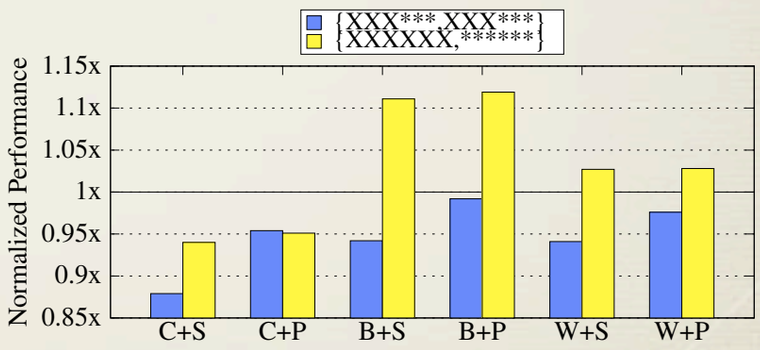
Varying Architecture



2 threads on Westmere



4 threads on Westmere



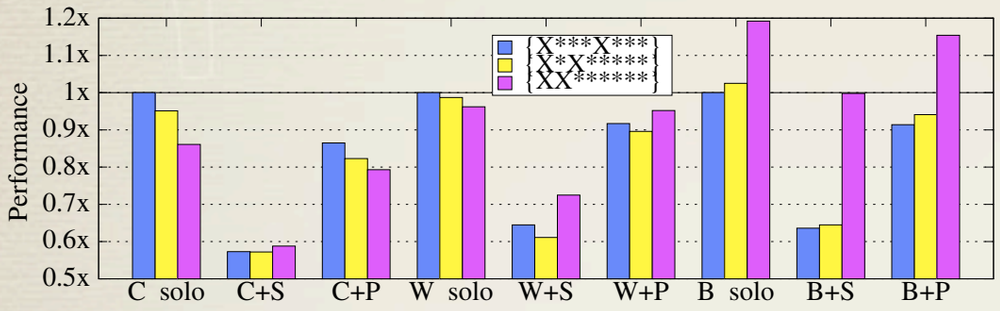
6 threads on Westmere

* Impact from sharing changes when architecture changes.

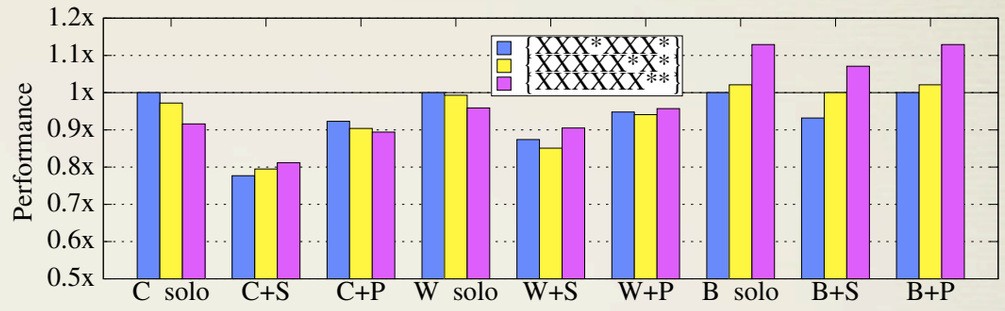
Lingjia Tang: lt8f@cs.virginia.edu

Varying Number of Threads and Architecture

Varying Thread Count

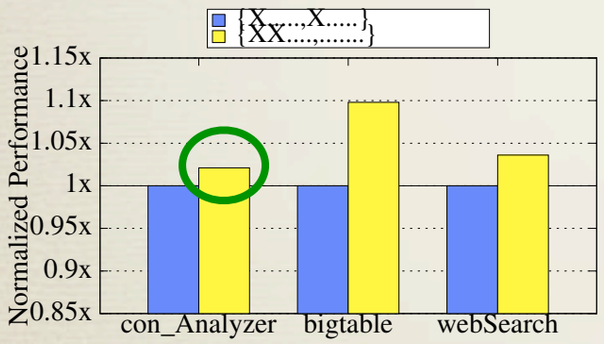


2 threads on Clovertown

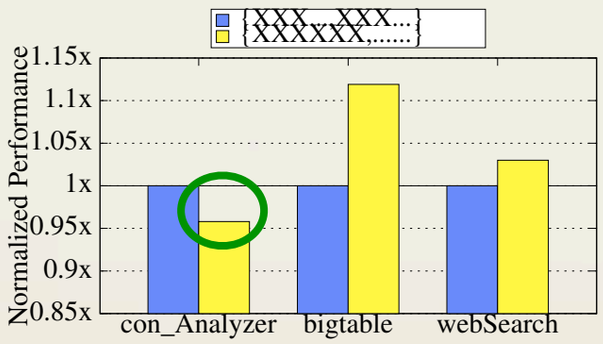


6 threads on Clovertown

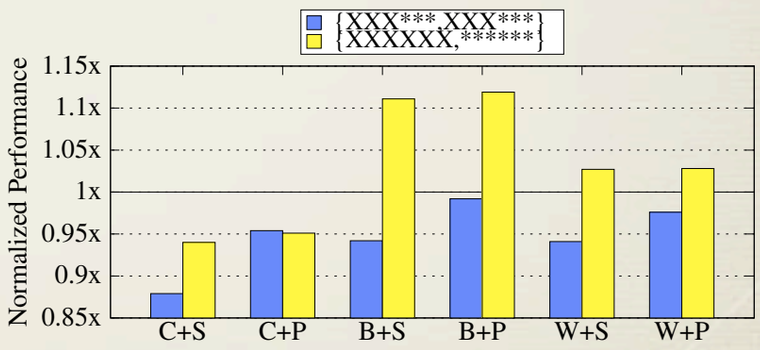
Varying Architecture



2 threads on Westmere



4 threads on Westmere



6 threads on Westmere

* Impact from sharing changes when architecture changes.

Lingjia Tang: lt8f@cs.virginia.edu

Outline

- Problem, Motivation and Background
- Characterization
  - Intra-application Sharing
  - Inter-application Sharing
  - Varying architectures
- Thread-to-Core Mapping
  - Heuristic-based mapping
  - Adaptive mapping

Lingjia Tang: lt8f@cs.virginia.edu

Thread-to-Core Mapping

- Heuristics-based Approach
  - Identify sharing and contention properties
  - Map threads accordingly
- Adaptive Approach
  - Searching for the optimal mapping online

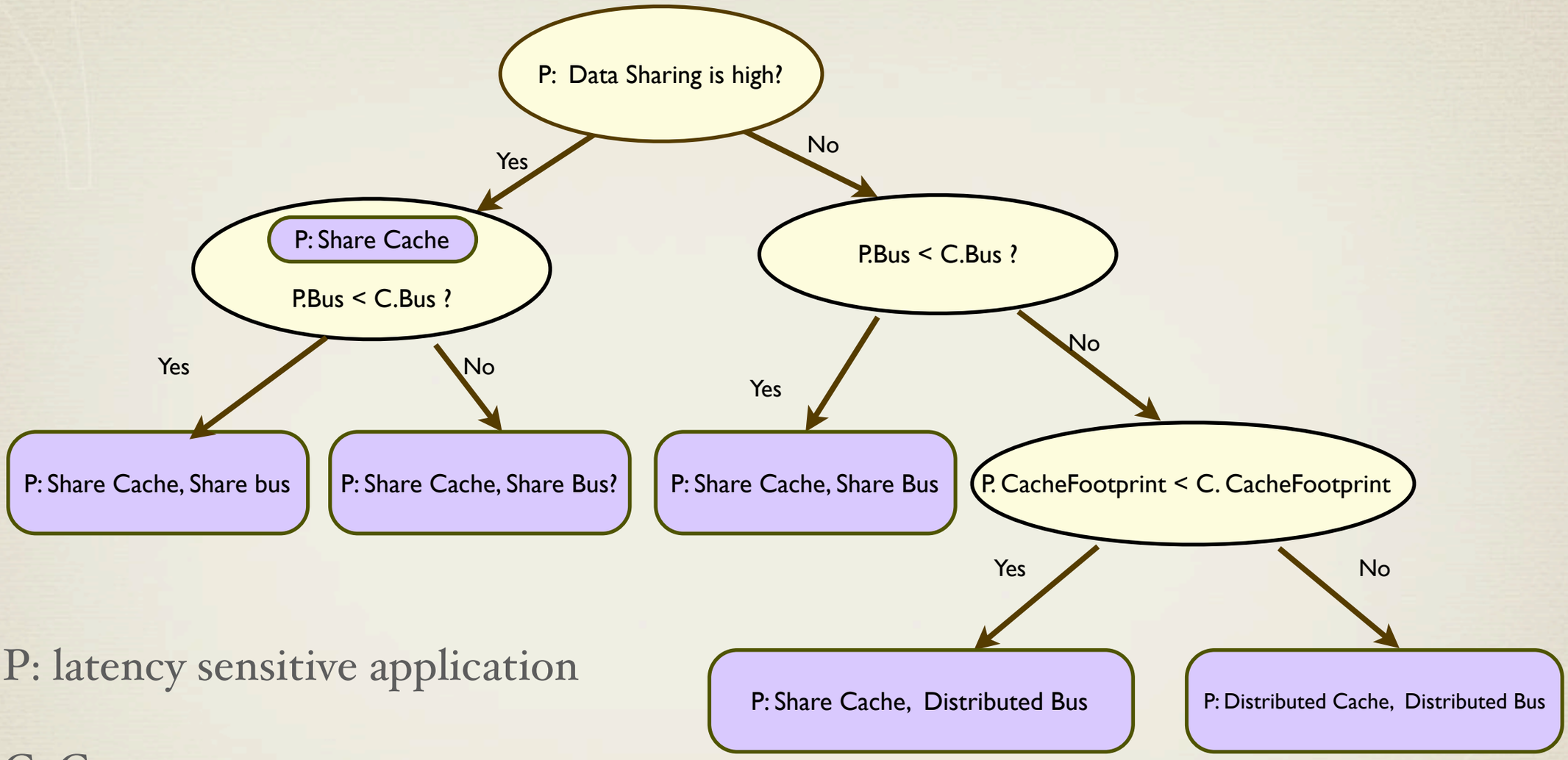
Lingjia Tang: lt8f@cs.virginia.edu

Heuristic-based TTC Mapping

- Mapping based on application's characteristics and resource topology
  - Data Sharing
  - Cache usage
  - Bandwidth usage
- Colocation: prioritize latency-sensitive applications

Lingjia Tang: lt8f@cs.virginia.edu

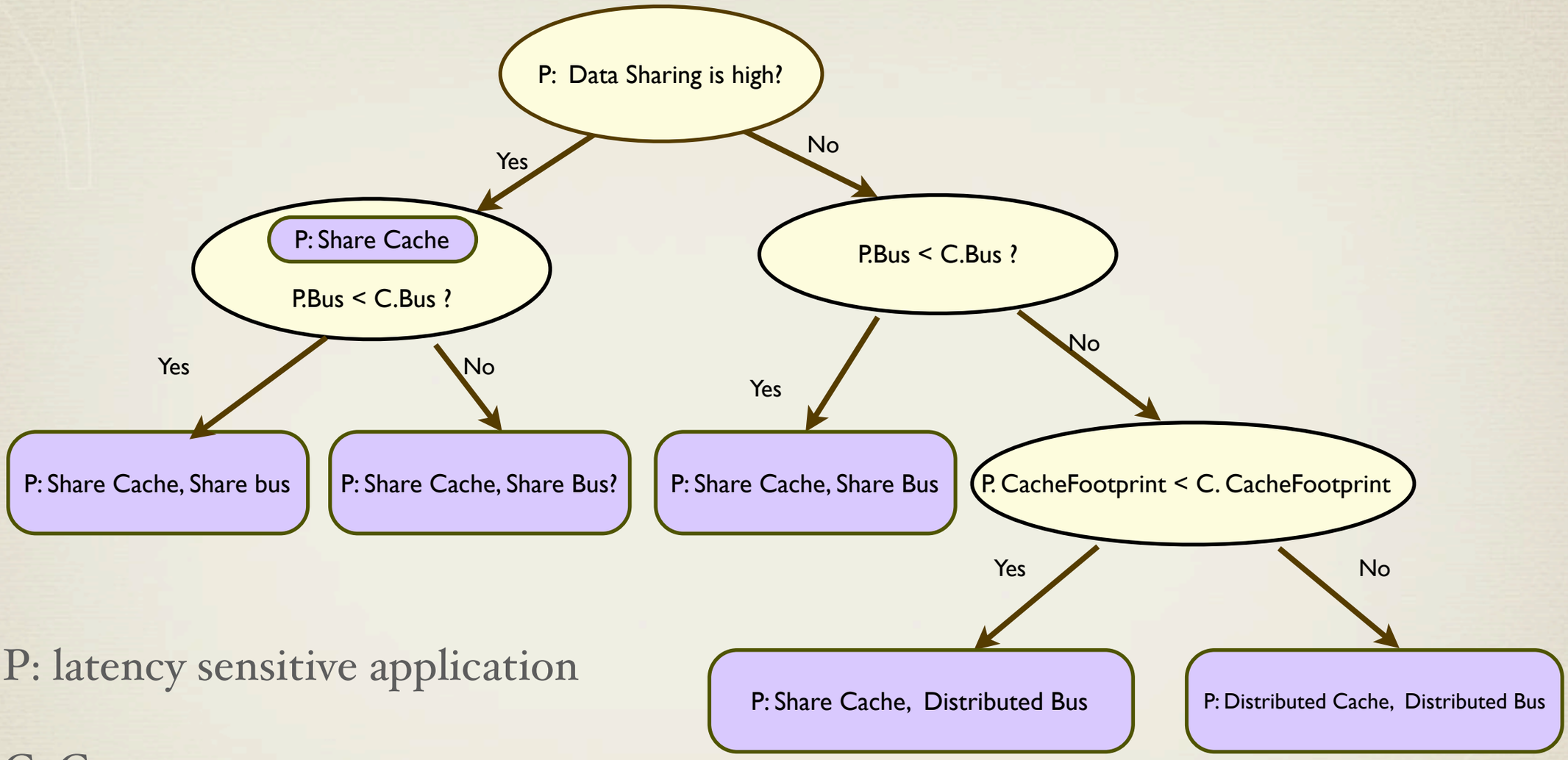
Heuristic-based TTC Mapping



- P: latency sensitive application
- C: Corunner

Lingjia Tang: lt8f@cs.virginia.edu

Heuristic-based TTC Mapping



- P: latency sensitive application
- C: Corunner

- Pros: analytical, known applications
- Cons: architecture specific; may require profiling

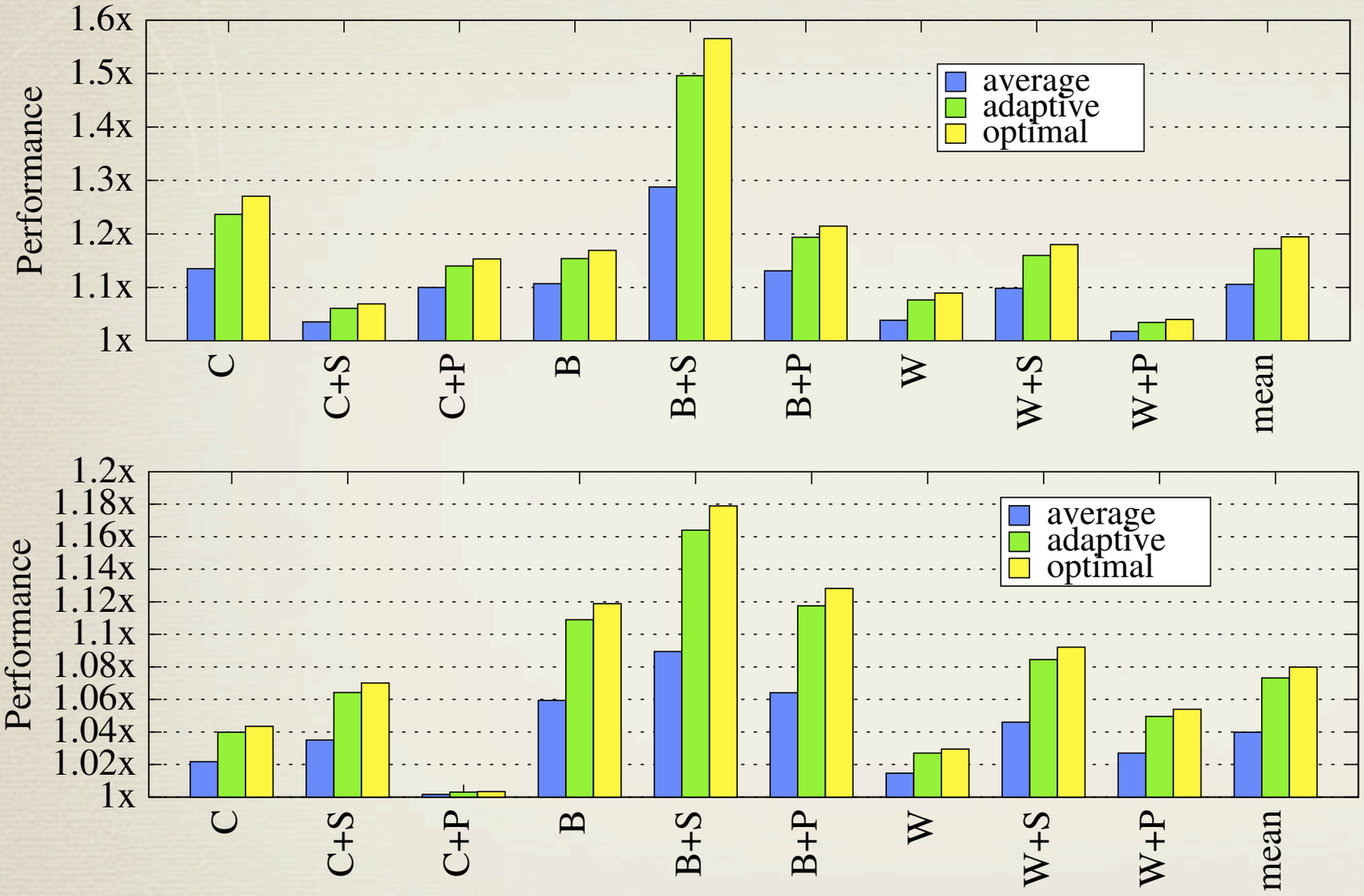
Lingjia Tang: lt8f@cs.virginia.edu

Adaptive TTC Mapping

- Insight: Optimal mapping changes when co-runner changes, when number of threads and architecture changes
- Competition Heuristics
  - Learning Phase
  - Execution Phase
- Datacenter applications:
  - Steady phases
  - Co-runner may change
  - Architecture may change

Lingjia Tang: lt8f@cs.virginia.edu

Evaluation



Clovertown Xeon

Westmere Xeon

* Outperforms the average random mapping by up to 20%

Lingjia Tang: lt8f@cs.virginia.edu

Conclusion

- Memory resource sharing has significant impact on datacenter applications (both constructive and destructive)
- Performance swing up to 40% simply based on thread-to-core mapping.
- Optimal thread-to-core mapping changes when co-runner changes.
- Importance of intelligent TTC mapping: adaptive

Lingjia Tang: lt8f@cs.virginia.edu

Lingjia Tang: www.lingjia.org

Jason Mars: www.jasonmars.com

Lingjia Tang: lt8f@cs.virginia.edu