

Multicore Computer Architecture - Storage and Interconnects

Lecture 17

QoS of NoC and Caches in TCMP Systems



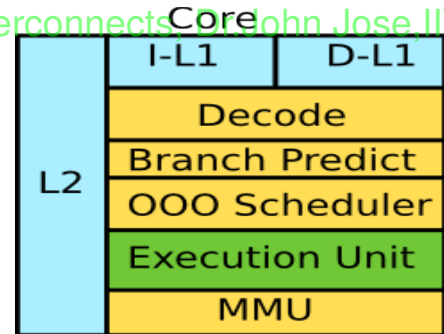
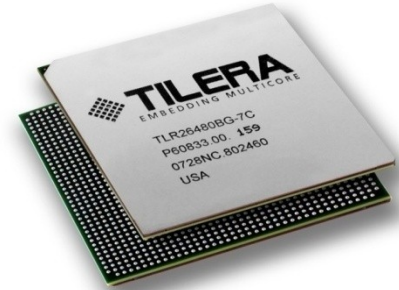
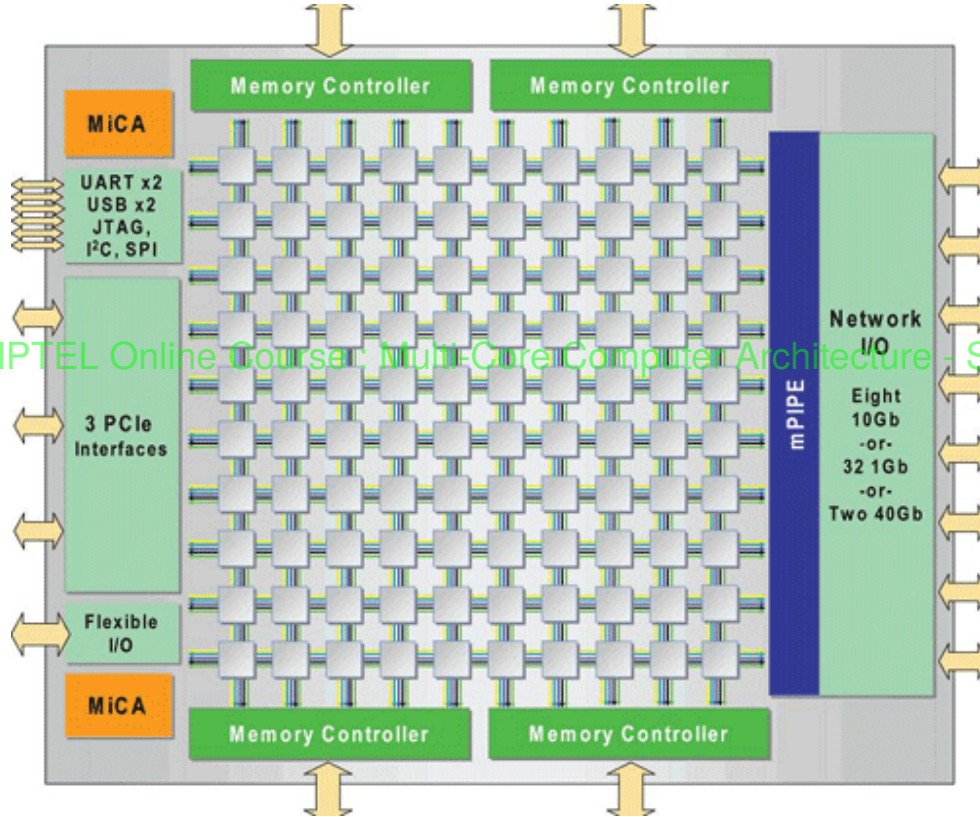
NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

Dr. John Jose

Assistant Professor

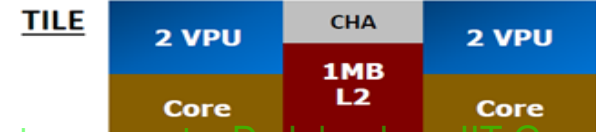
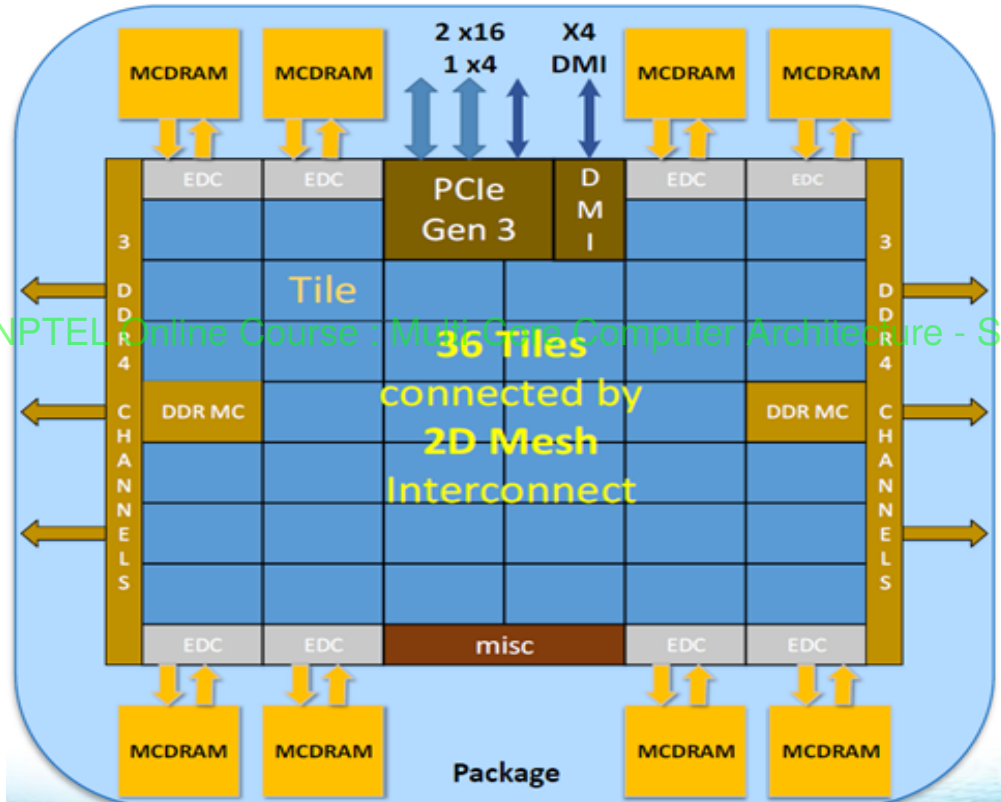
**Department of Computer Science & Engineering
Indian Institute of Technology Guwahati, Assam.**

Tiled Chip Multi Processors (TCMP)



NPTEL Online Course: Multi-Core Computer Architecture - Storage and Interconnects - Dr. John Jose IIT Guwahati

Intel KNL architecture - Cloud On Chip



- ❖ 36 tiles in 2D mesh
- ❖ 2 cores, 2 VPU/core
- ❖ 1 MB L2 cache /core
- ❖ 8 channel DDR4 MCs

On-Chip Communication

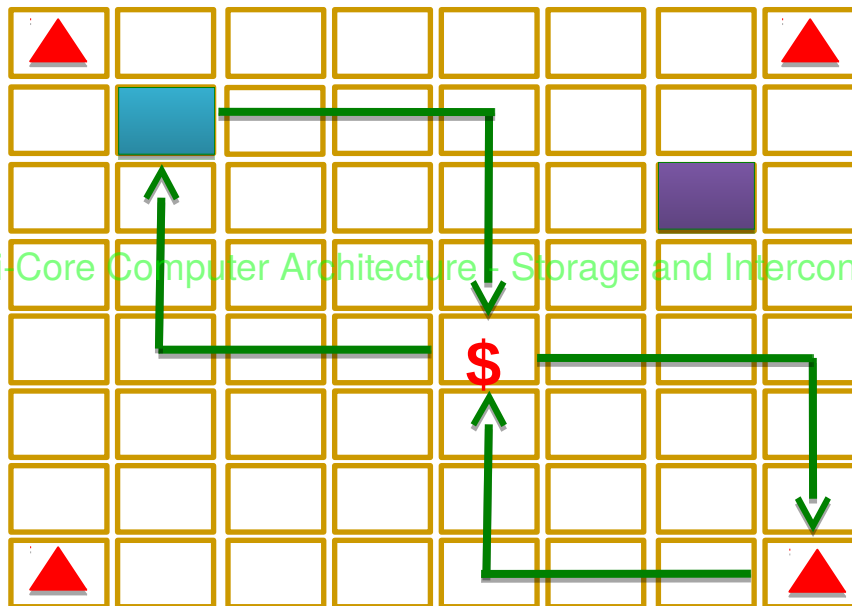
Applications



Light



Heavy

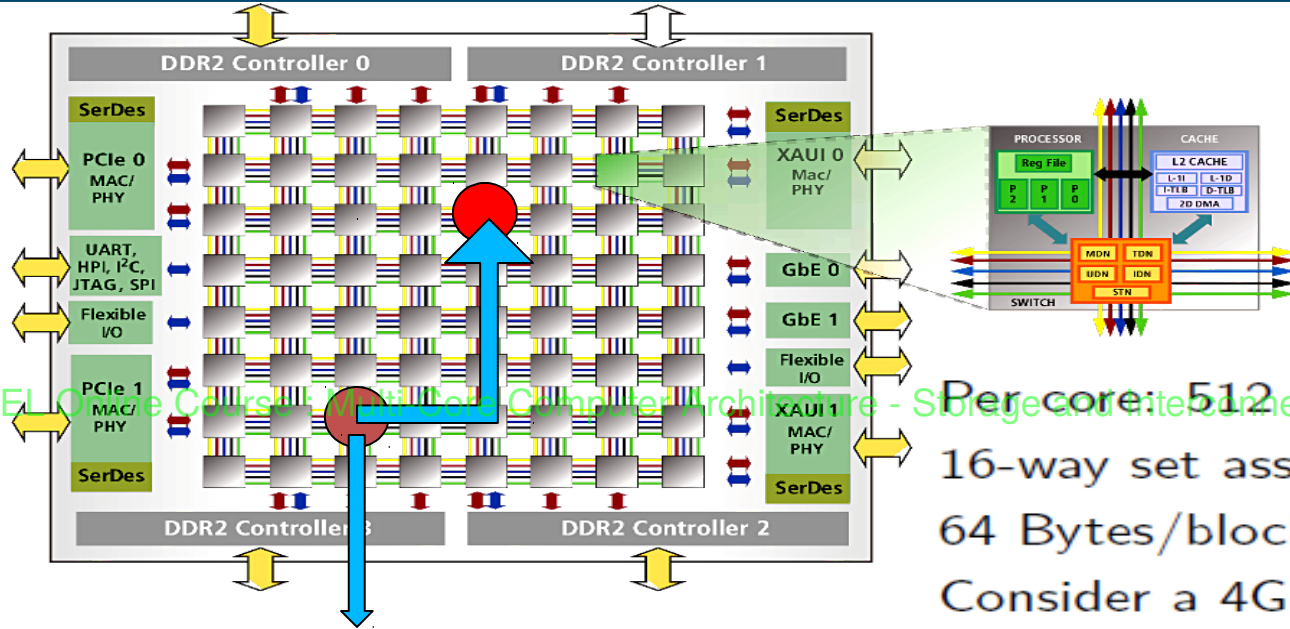


Memory
Controller



Shared
Cache Bank

On-Chip Cache Address Mapping



Per core: 512 KB

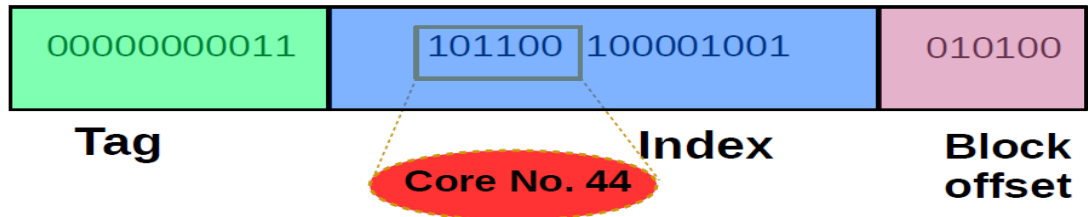
16-way set associative mapping

64 Bytes/block

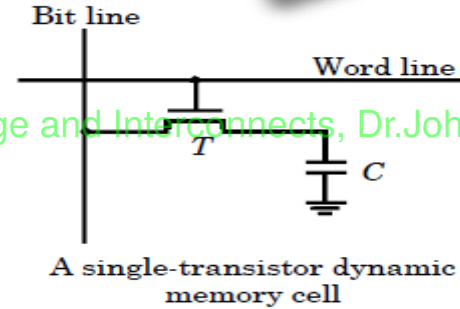
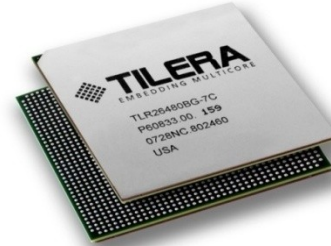
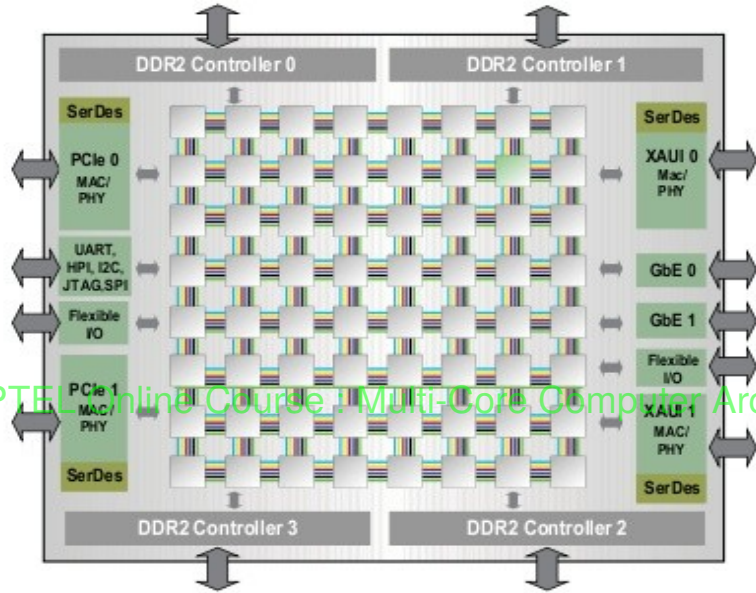
Consider a 4GB RAM (2^{32} bits)

Hex Address : 0x764254

11 bits	15 bits	6 bits
Tag	Index	Block offset



What is the role of memory controllers?

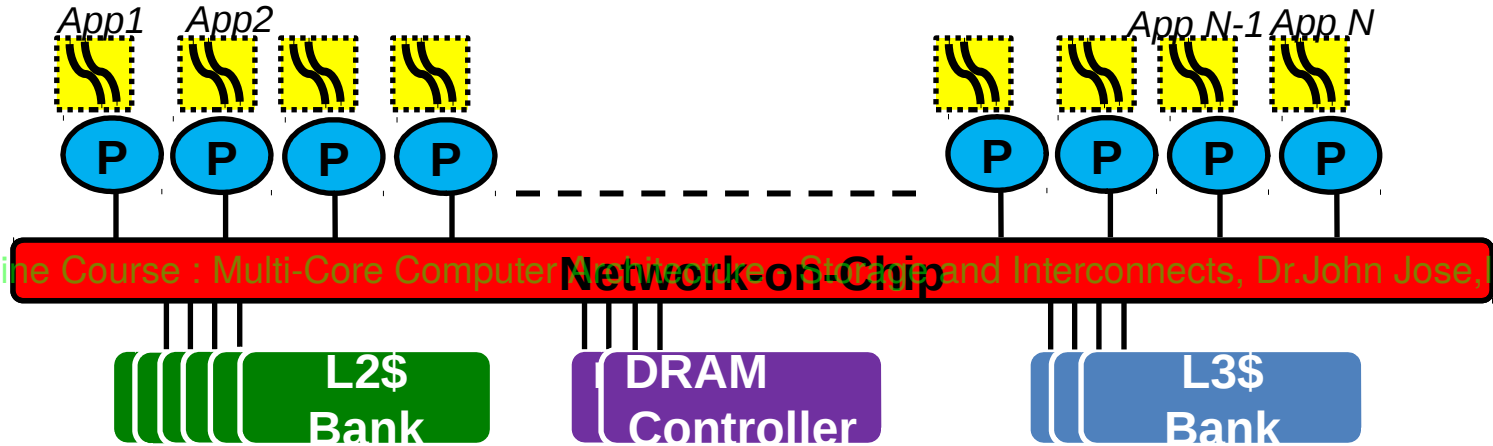


QoS Optimization Techniques in CMP

- ❖ **Congestion Management in On-Chip Interconnects**
 - ❖ **Slack Aware NoC Routing**
 - ❖ **Packet Throttling**

NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

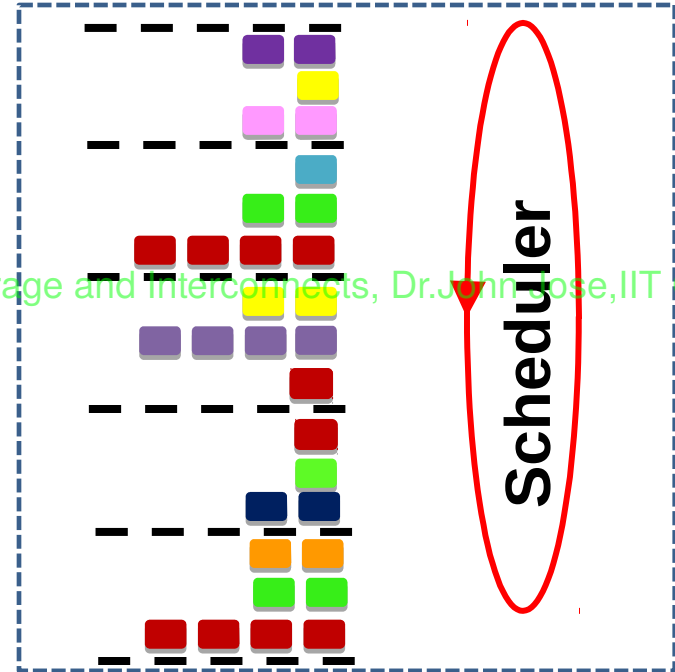
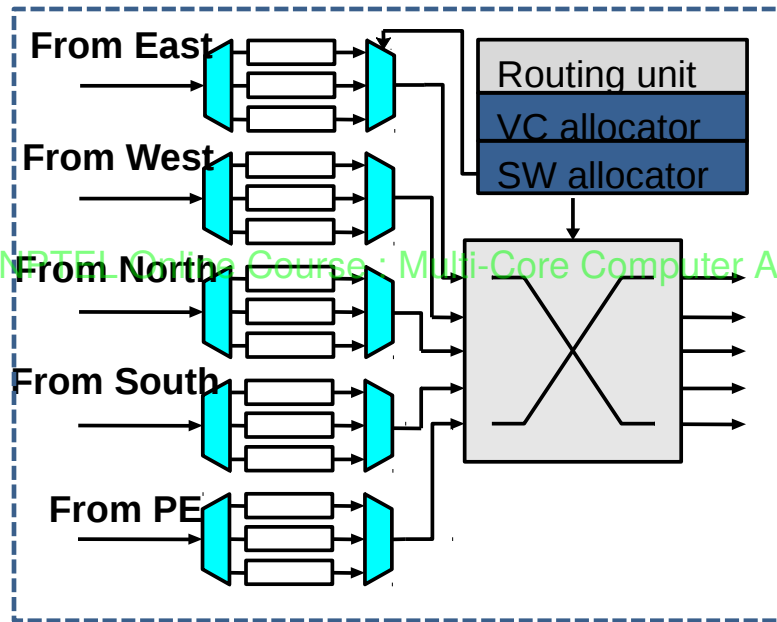
How critical is NoC ?



Network-on-Chip is a **critical** resource
shared by **multiple applications**

Input / Output Channel Selection

Which packet to choose?



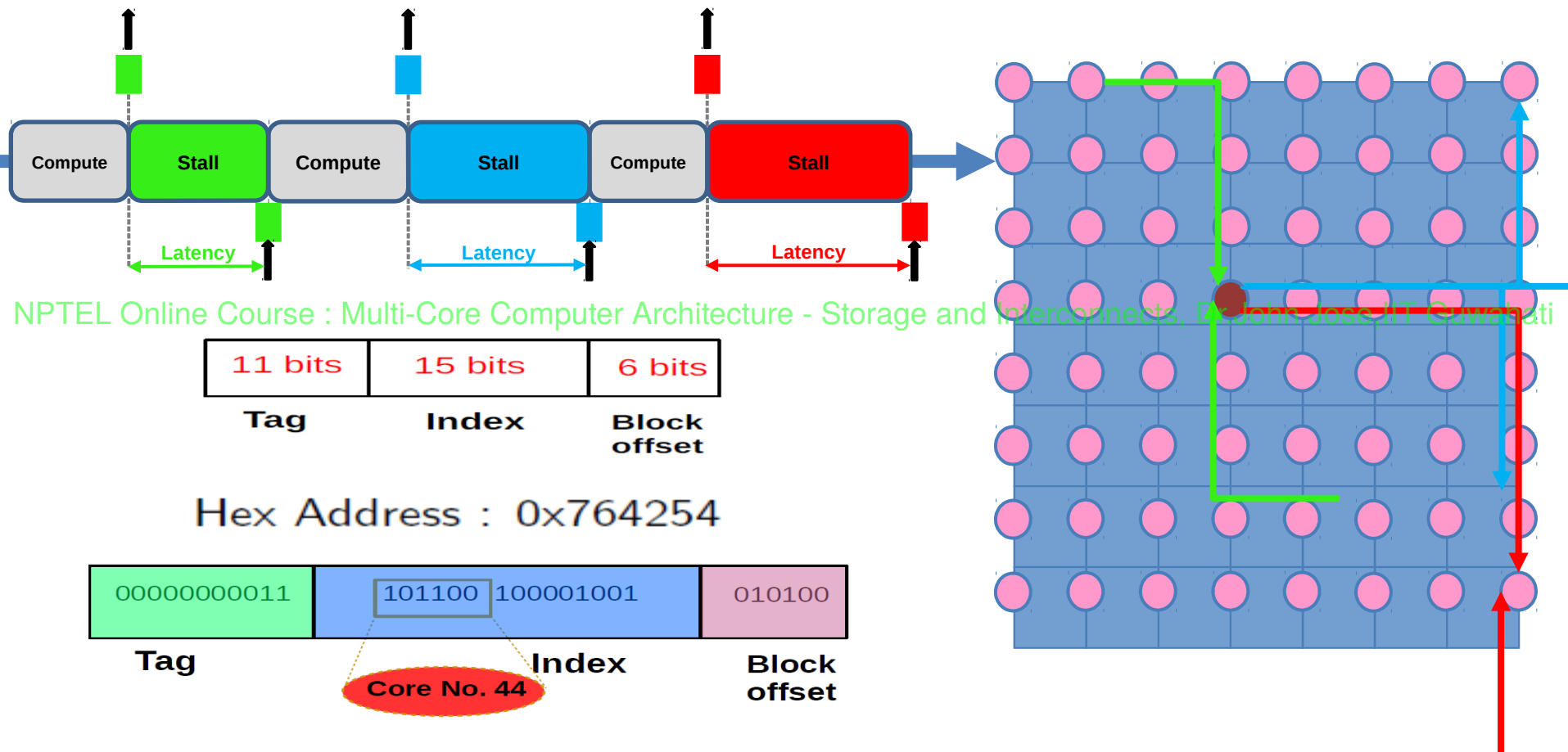
Legend for Applications:

App1	App2	App3	App4
App5	App6	App7	App8

Switch Level Scheduling Policies

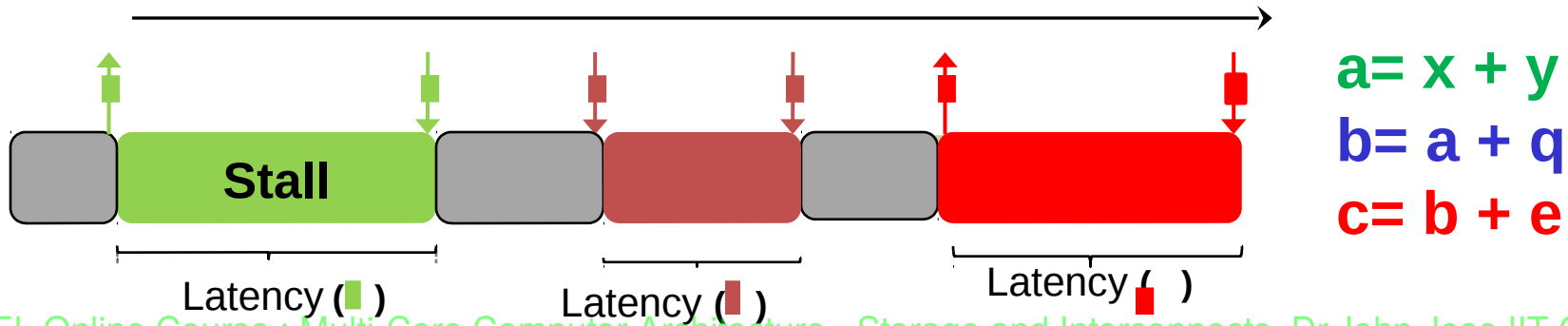
- ❖ Conventional Scheduling policies
 - ❖ Round robin
 - ❖ Age
- ❖ Problem
 - ❖ Treat all packets equally
 - ❖ Application-oblivious
- ❖ Packets have **different criticality**
 - ❖ Packet is critical if latency of a packet affects application's performance
 - ❖ Different criticality due to memory level parallelism (MLP)

Memory Level Parallelism (MLP)



NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

Memory Level Parallelism

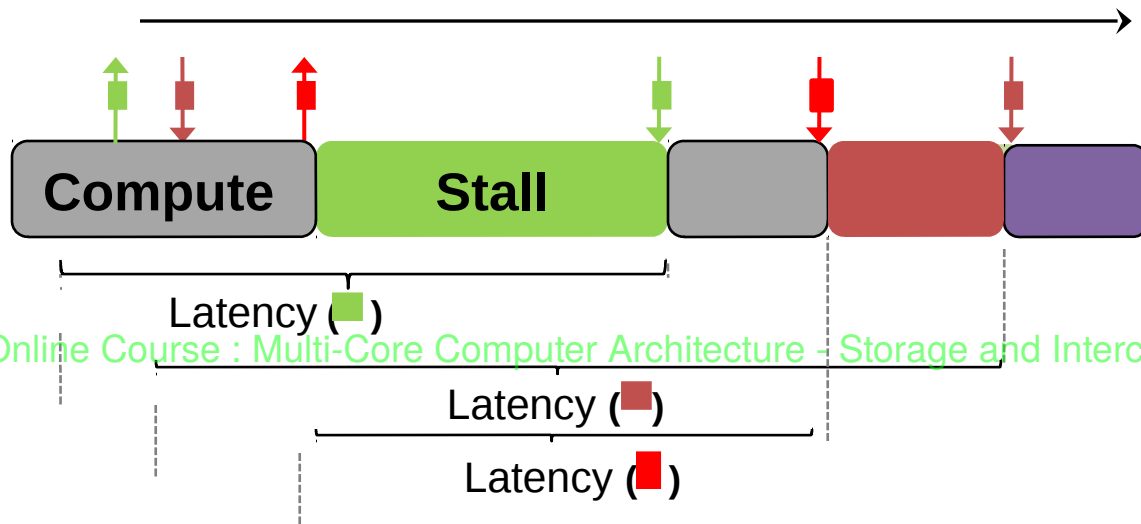


NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

- ❖ Packet Latency = Network Stall Time
- ❖ No miss overlap, so all misses stall the application

$$\text{Criticality}(\text{green square}) \cup \text{Criticality}(\text{red square}) = \text{Criticality}(\text{red square})$$

What is Memory Level Parallelism ?



$$a = x + y$$

$$b = p + q$$

$$c = m + n$$

$$s = b + c$$

$$\text{Stall}(\blacksquare) = 0$$

❖ Packet Latency $\neq$ Network Stall Time

❖ Different Packets have different criticality due to MLP

$$\text{Criticality}(\blacksquare) < \text{Criticality}(\blacksquare) > \text{Criticality}(\blacksquare)$$

Slack of Packets

- ❖ **What is slack of a packet?**

- ❖ Slack of a packet is number of cycles it can be delayed in a router without reducing application's performance

- ❖ **Source of slack: Memory-Level Parallelism (MLP)**

NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

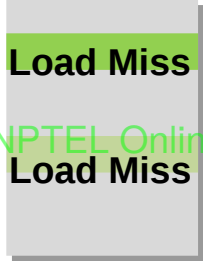
- ❖ Latency of an application's packet hidden from application due to **overlap** with latency of pending cache miss requests

- ❖ **Prioritize packets with **lower slack****

What is slack of a packet ?

$$a = x + y$$

Instruction Window



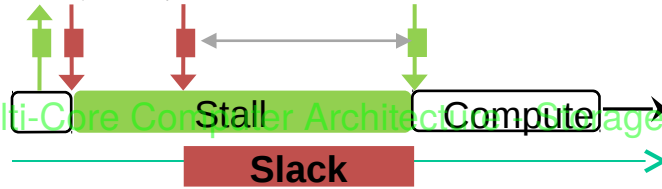
Causes

Causes

Execution Time

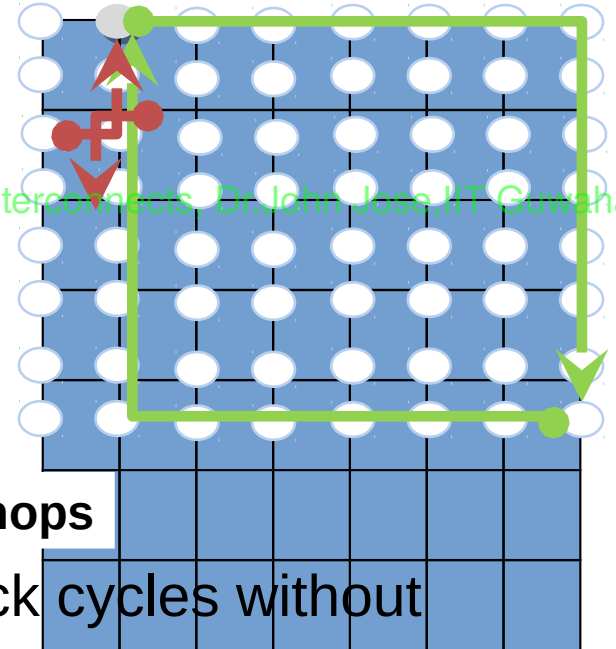
Latency (■)

Latency (■)



■ returns earlier than necessary

Network-on-Chip

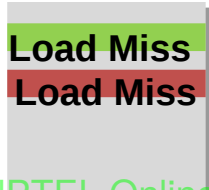


Slack (■) = Latency (■) - Latency (■) = 26 - 6 = 20 hops

❖ Packet() ■ can be delayed for available slack cycles without reducing performance.

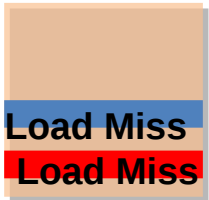
How to exploit slack ?

Core A

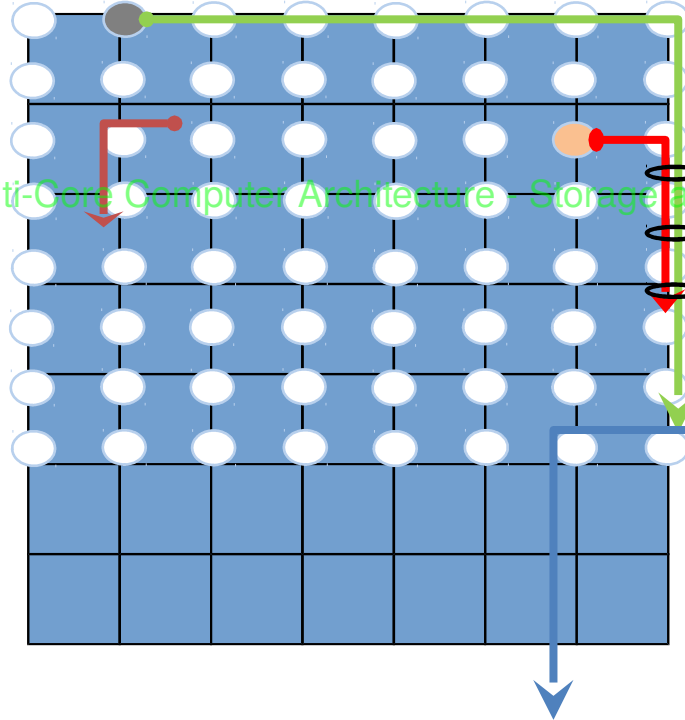


Causes
Causes

Core B



Causes
Causes



Packet	Latency (hops)	Slack
	13	0
	3	10
	10	0
	4	6

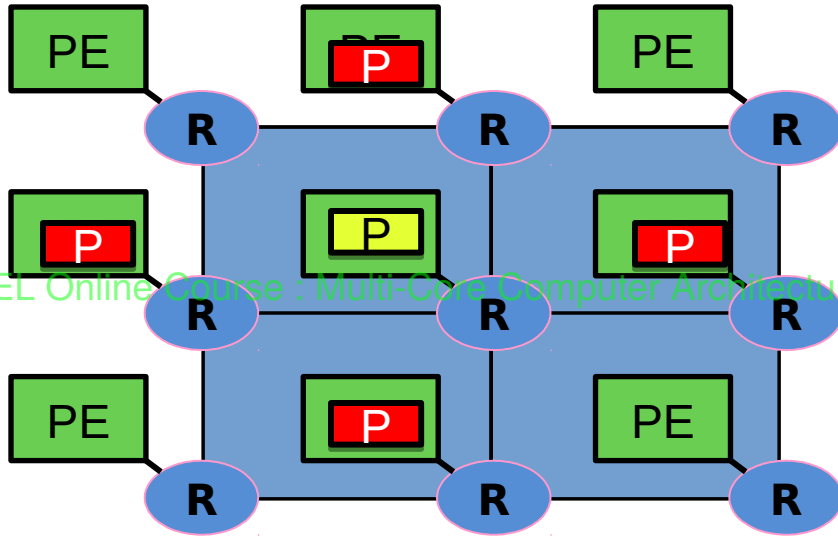
Interference at 3 hops

Slack() > Slack ()

Prioritize !!!!

Network Congestion

❖ Network congestion degrades system performance

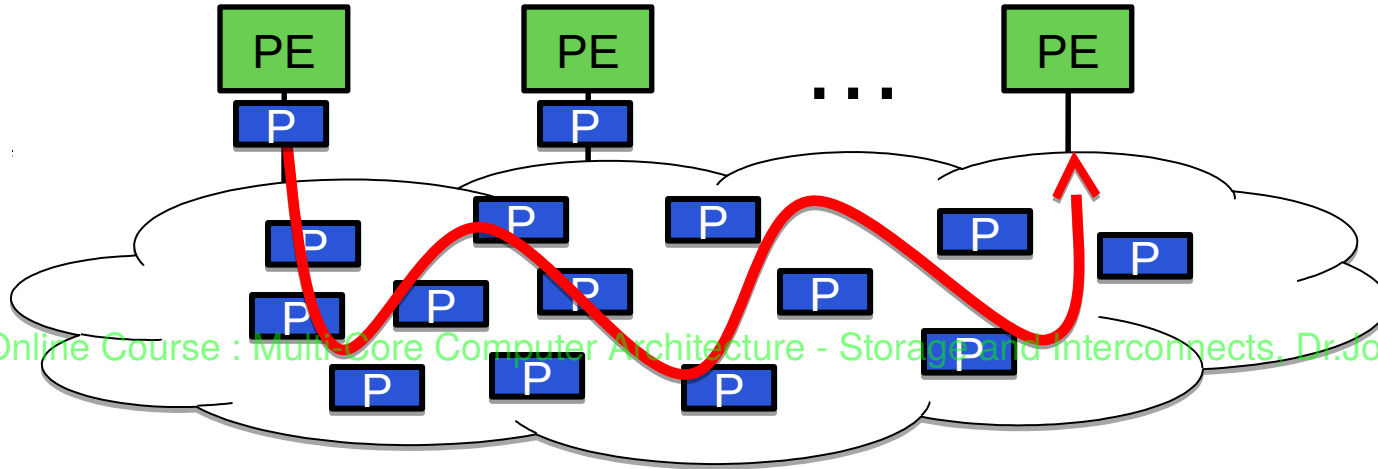


R Router **P** Packet
PE Processing Element
(Cores, L2 Banks, Memory Controllers, etc)

Network Congestion Management

- ❖ Improve system performance in a highly congested network
- ❖ Reducing network load (number of packets in the network) decreases network congestion, hence improves system performance
- ❖ Source throttling (temporarily delaying new traffic injection) to reduce network load

Source Throttling

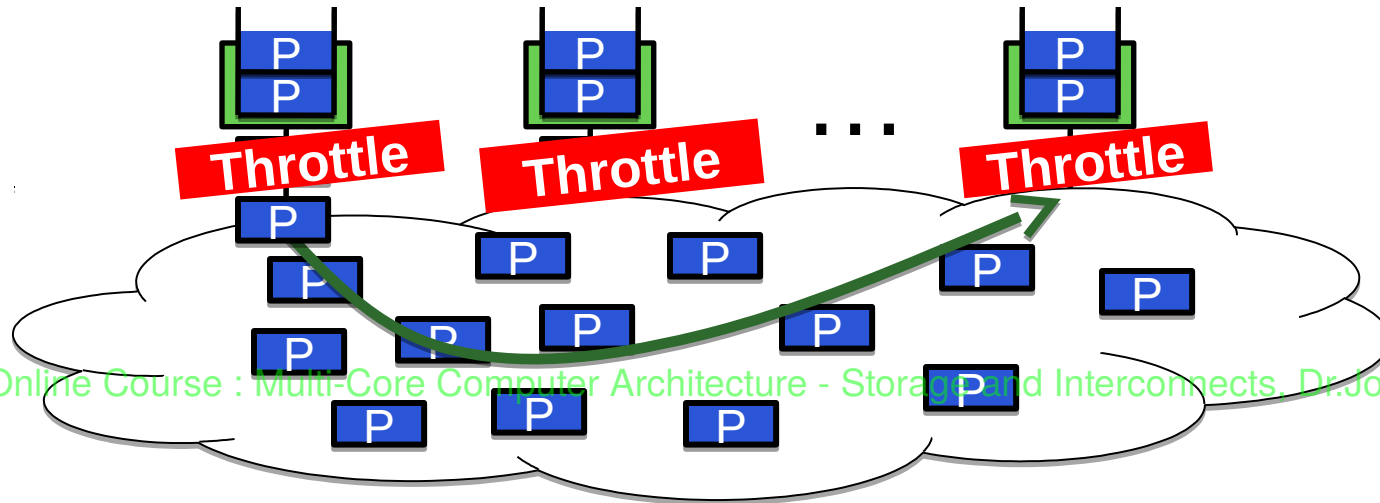


NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

❖ Long network latency when the network is congested

 Network  Packet
 Processing Element
(Cores, L2 Banks, Memory Controllers, etc)

Source Throttling



NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

- ❖ Throttling makes some packets wait longer to inject
- ❖ Average network throughput increases, hence higher system performance



Network



Packet

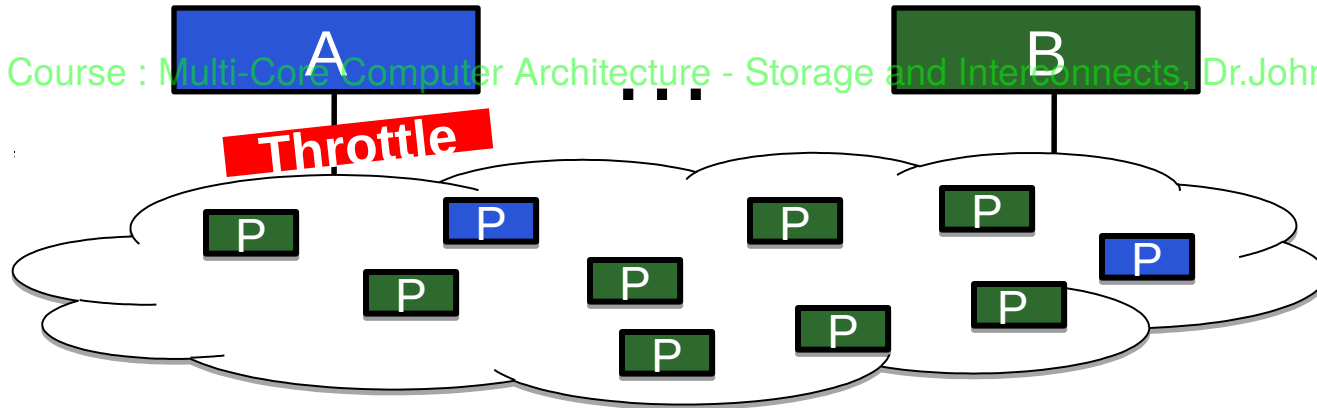


Processing Element

(Cores, L2 Banks, Memory Controllers, etc)

Source Throttling

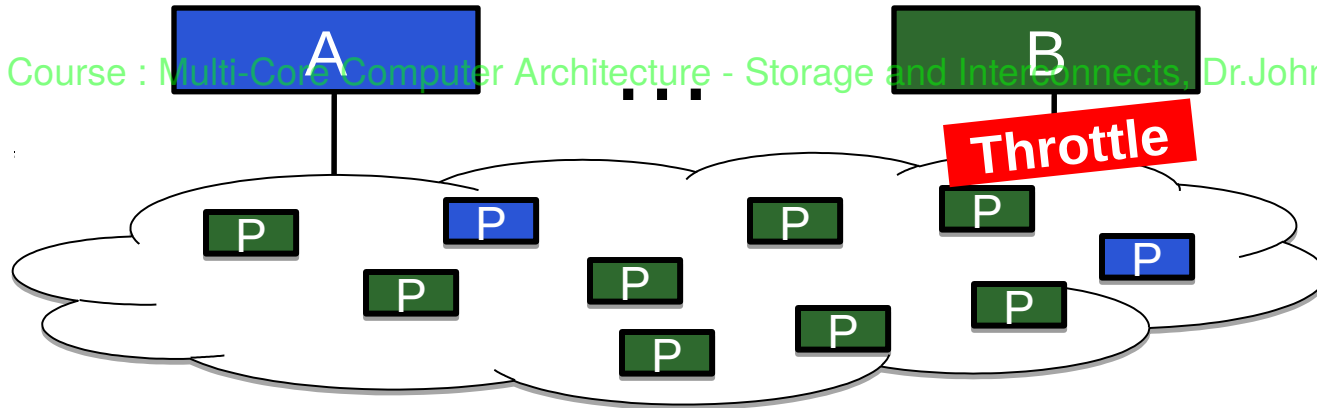
Configuration: 16-node system, 4x4 mesh network,
8 A (network-non-intensive), and 8 B (network-intensive)



❖ Throttling A decreases system performance due to minimal network load reduction

Source Throttling

Configuration: 16-node system, 4x4 mesh network,
8 A (network-non-intensive), and 8 B (network-intensive)

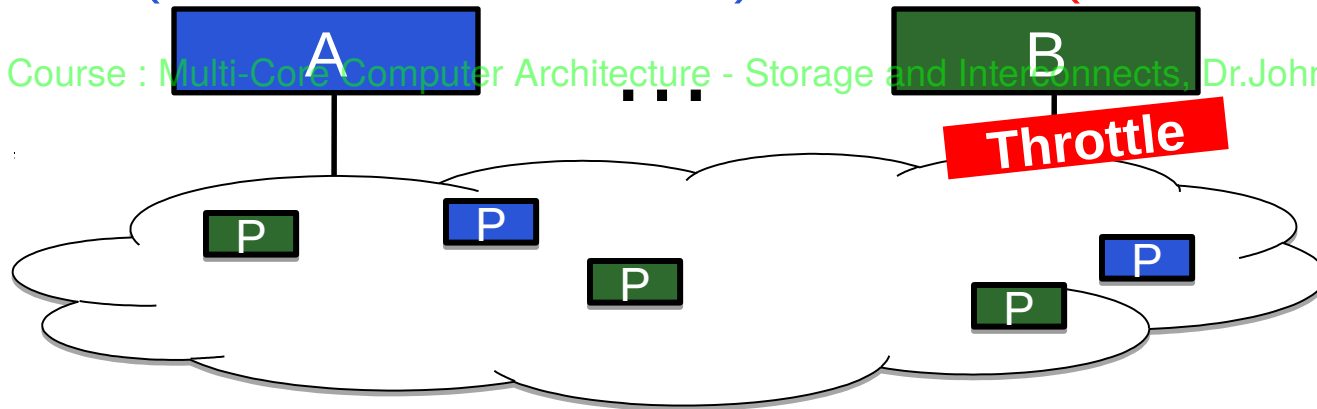


- ❖ Throttling B increases system performance due to reduced congestion

Source Throttling

- ❖ Throttling **network-intensive** applications leads to higher system performance

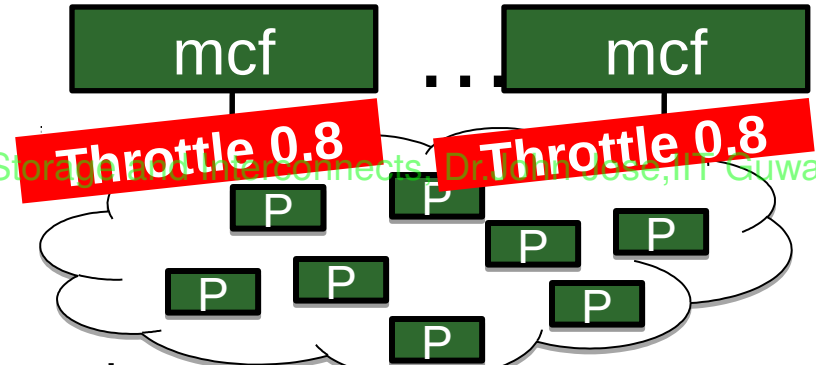
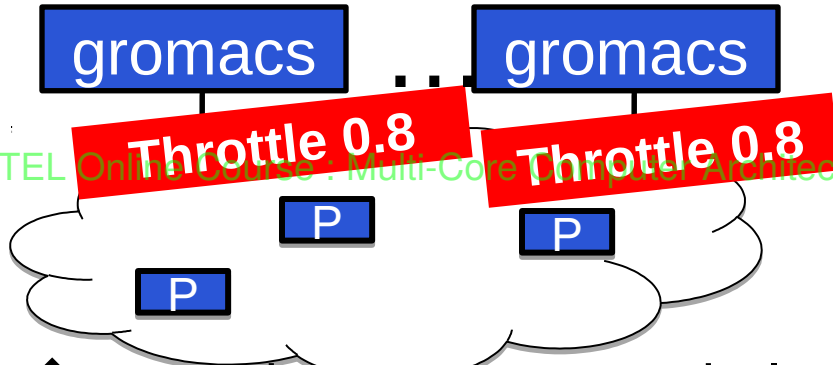
Configuration: 16-node system, 4x4 mesh network,
8 gromacs (network-non-intensive), and 8 mcf (network-intensive)



- ❖ Throttling **B** reduces congestion
- ❖ **A** benefits more from less network latency

Source Throttling

- ❖ There is no single **throttling rate** that works well for every application workload



- ❖ Network runs best at or below a certain network load
- ❖ Adjust throttling rate to avoid overload and under-utilization

Application-Aware Throttling

- ❖ Measure applications' network intensity
- ❖ Use **L1 MPKI** (misses per thousand instructions) to estimate network intensity
- ❖ Throttle network-intensive applications

NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

Network-non-intensive
(Unthrottled)



$\Sigma \text{ MPKI} < \text{Threshold}$

Network-intensive
(Throttled)



Higher L1 MPKI



Application-Aware Throttling

- ❖ Application classification and throttling rate adjustment are expensive if done every cycle
- ❖ Solution: recompute at fixed time interval granularity

During epoch:

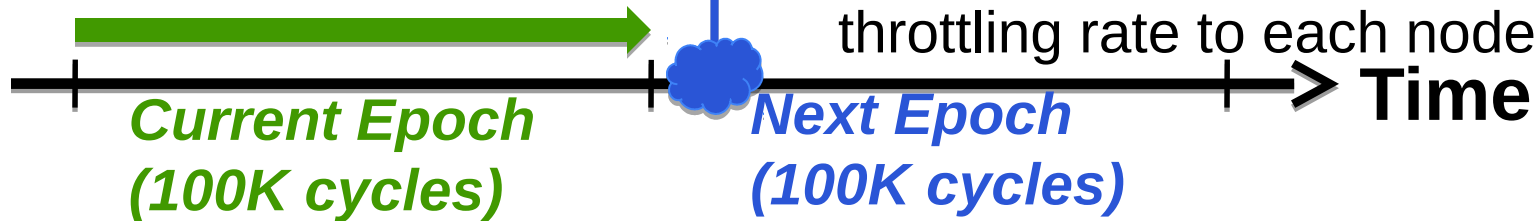
Every node:

- 1) Measure **L1 MPKI**
- 2) Measure **network load**

Beginning of epoch:

All nodes send measured info to a **central controller**, which:

- 1) Classifies applications
- 2) Adjusts throttling rate
- 3) Sends new classification and throttling rate to each node

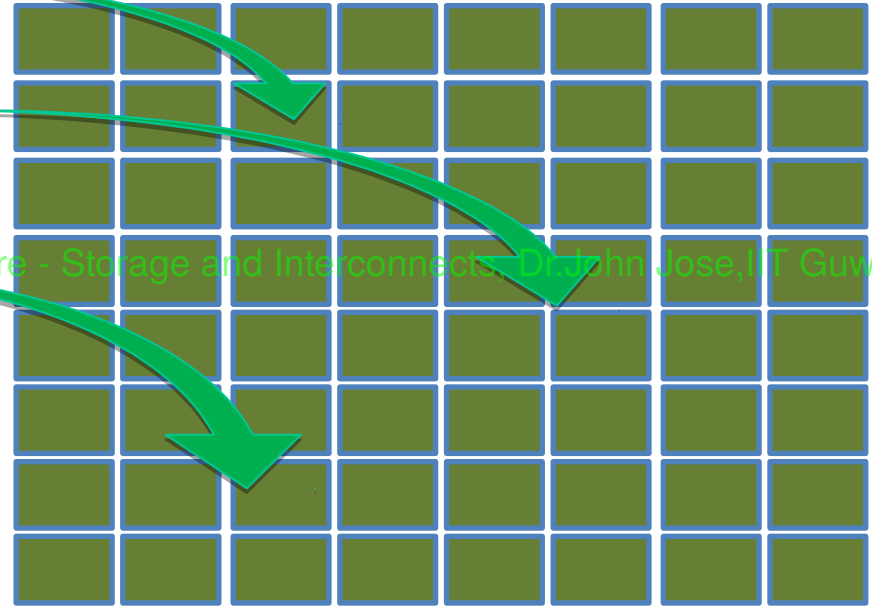


Application-to-Core Mapping Policies

Applications



Cores



How to map applications to cores?

Application-to-Core Mapping Policies

❖ Application To Core Mapping

❖ Clustering

❖ Balancing

❖ Isolation

❖ Radial mapping



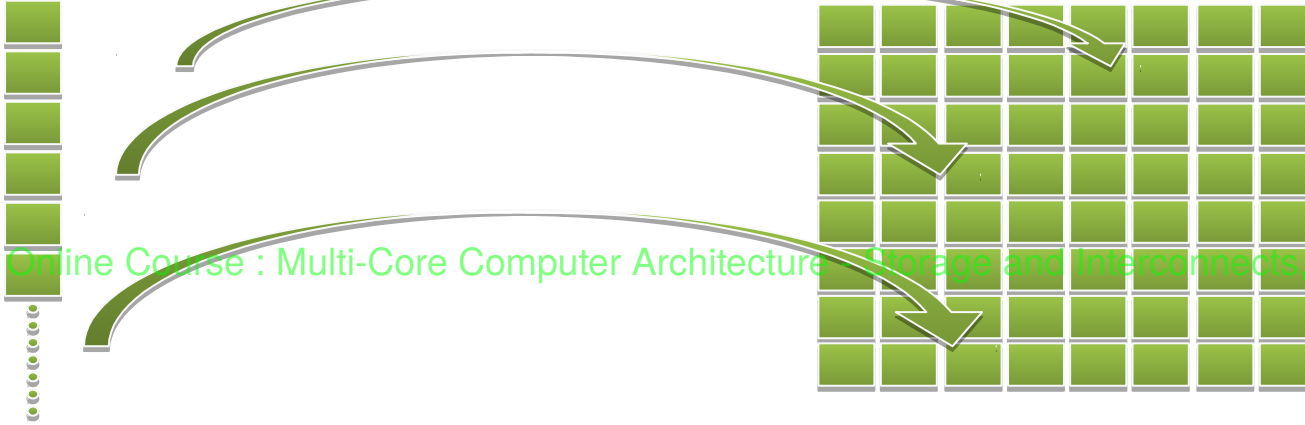
Task Scheduling

- ❖ Traditional
 - ❖ When to schedule a task? – Temporal
- ❖ Many-Core
 - ❖ When to schedule a task? – Temporal
 - ❖ **Where to schedule a task? – Spatial**
- ❖ Spatial scheduling impacts performance of memory hierarchy
- ❖ Latency is impacted by interference in NoC, memory, and caches

Challenges in Spatial Task Scheduling

Applications

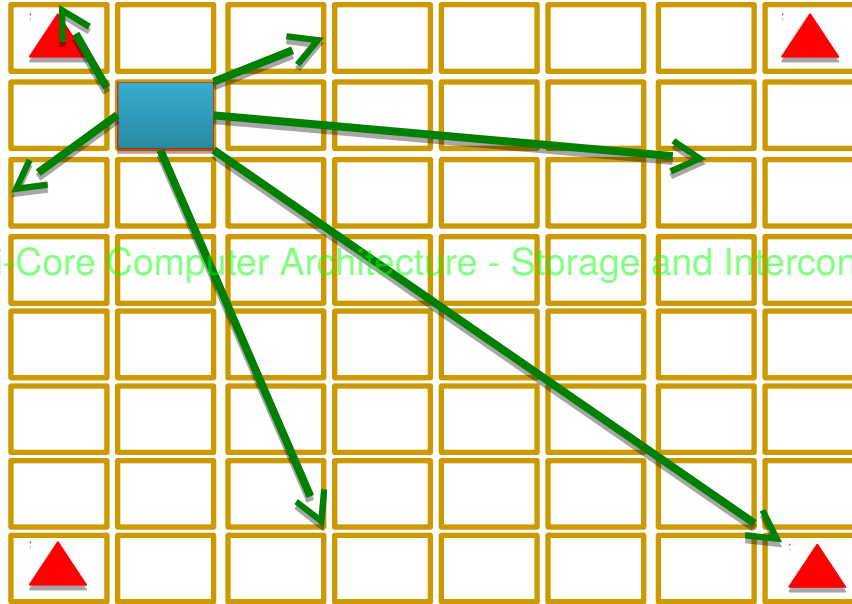
Cores



NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr.John Jose,IIT Guwahati

- ❖ How to **reduce communication distance**?
- ❖ How to **reduce destructive interference** between applications?
- ❖ How to **prioritize applications** to improve throughput?

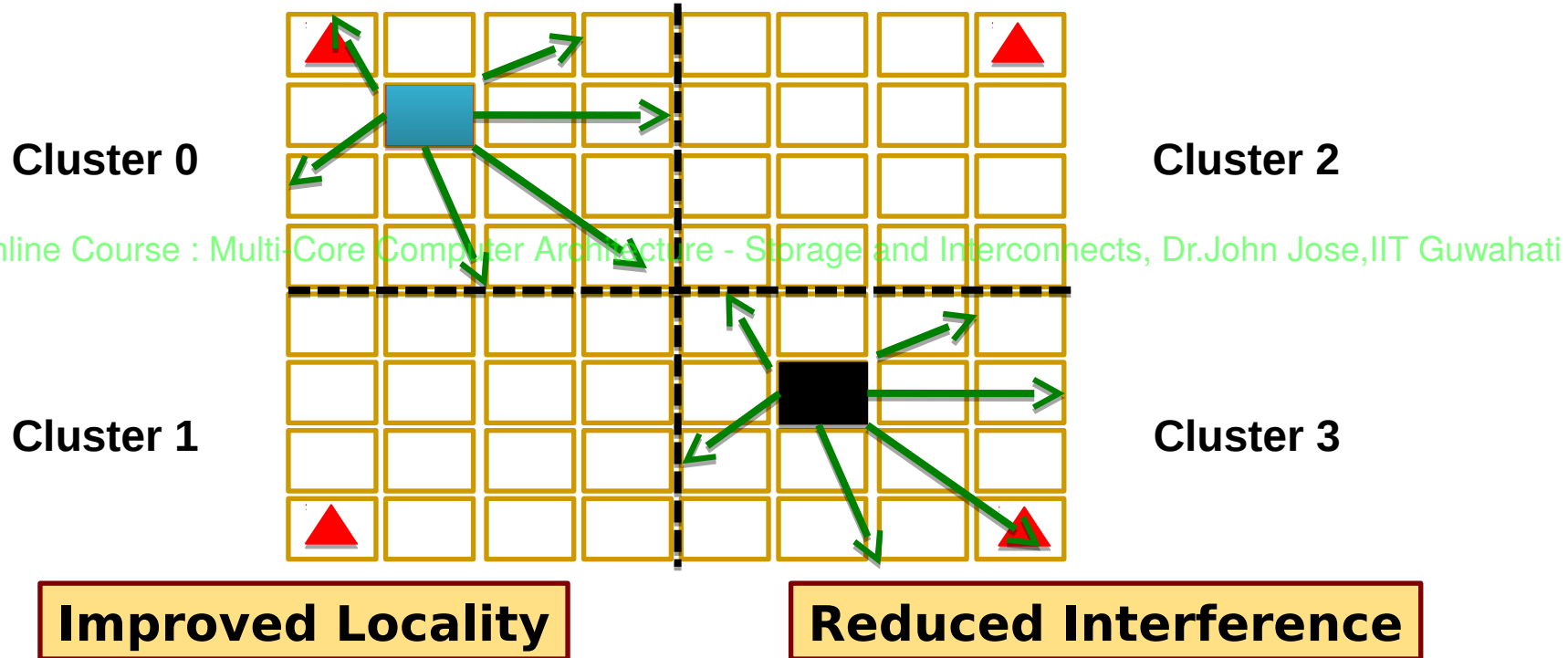
Clustering



▲
**Memory
Controller**

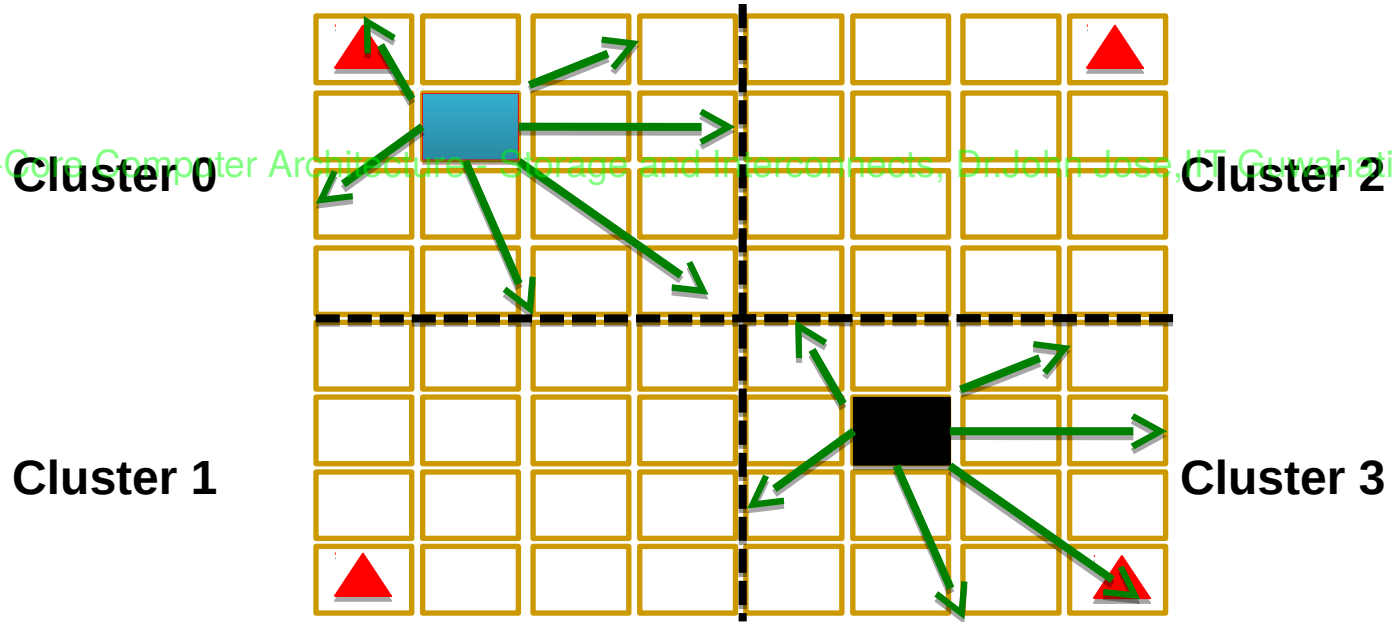
Inefficient data mapping to memory and caches

Clustering



Clustering

- ❖ Locality aware page replacement policy
- ❖ When allocating free page, give preference to pages belonging to the cluster's memory controllers (MCs)



Balancing

Applications

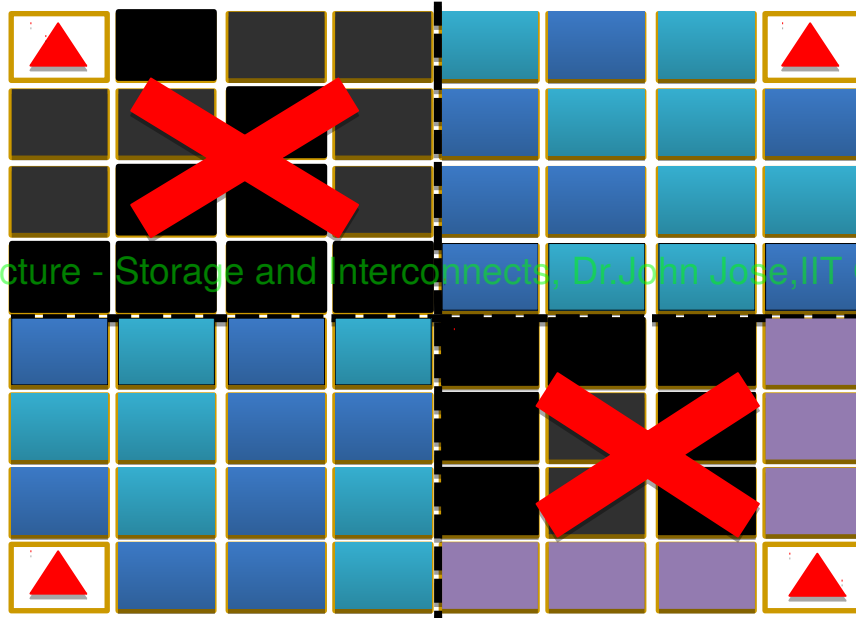


Heavy



Light

Cores



❖ Too much load in clusters with heavy applications

Balancing

Applications

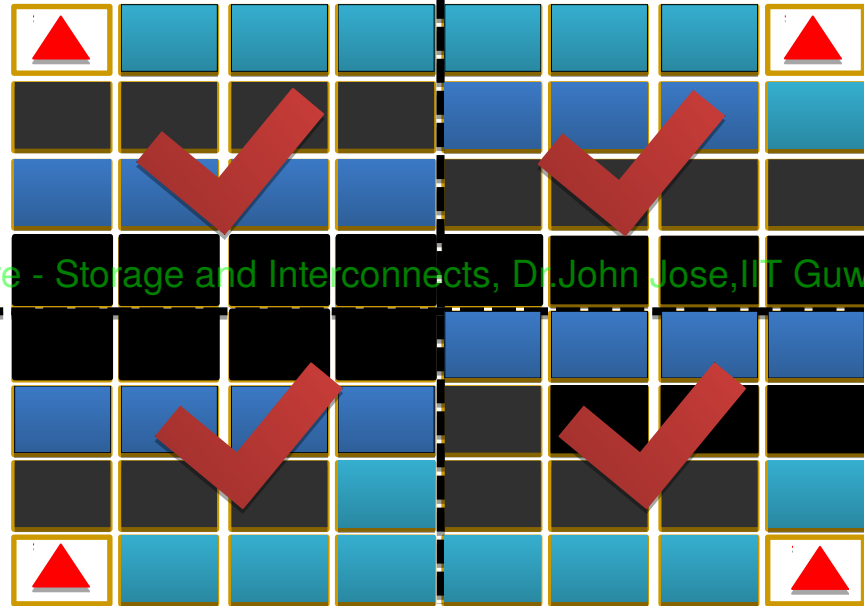


Heavy



Light

Cores



Better bandwidth utilization

Is this the best we can do? Let's take a look at **application characteristics**

Isolation

Applications



Sensitive



Light

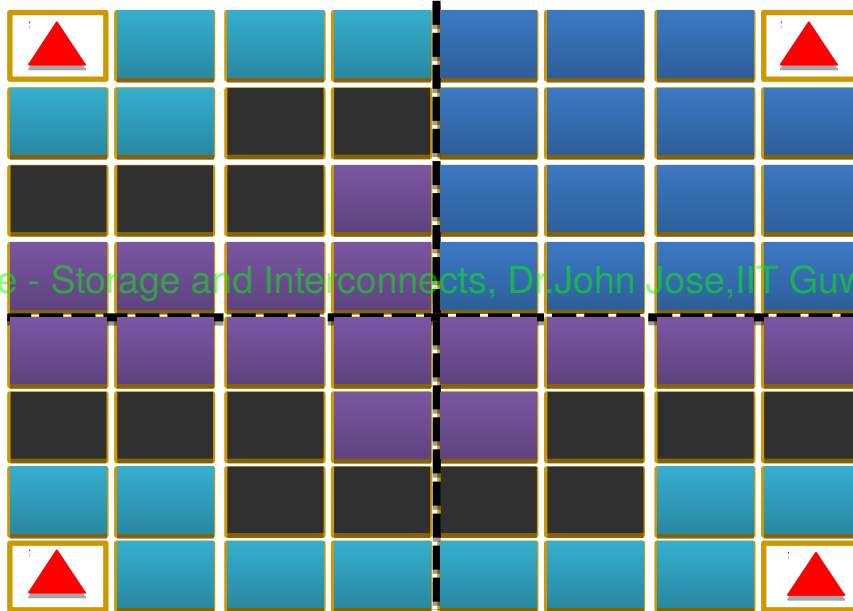


Medium



Heavy

Cores



Isolate **sensitive** applications to a cluster

Balance load for remaining applications across clusters

Isolation

- ❖ How to estimate sensitivity?
 - ❖ High Miss— high misses per kilo instruction (MPKI)
 - ❖ Low MLP— high relative stall cycles per miss (STPM)
 - ❖ Sensitive if $MPKI > \text{Threshold}$ & relative STPM is high
- ❖ Whether to or not to allocate cluster to sensitive applications?

NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects, Dr. John Jose, IIT Guwahati

Radial Mapping

Applications



Sensitive



Light

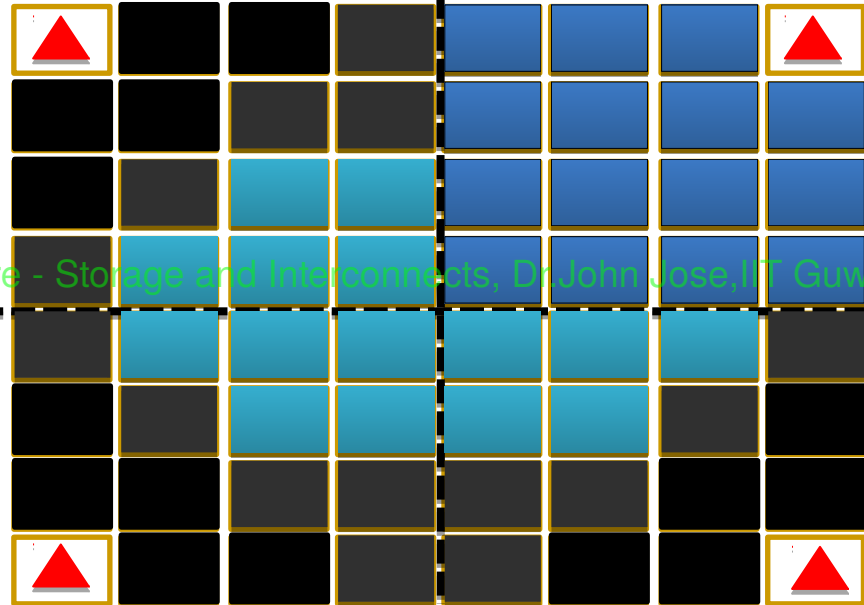


Medium



Heavy

Cores

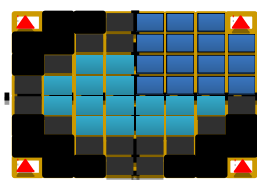
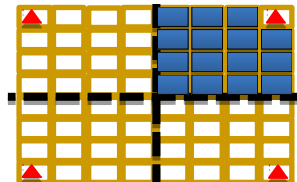
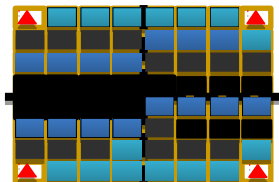
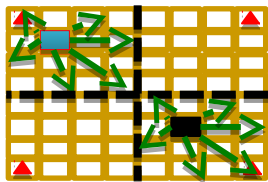


- ❖ Map applications **that benefit most from** being close to **memory controllers** close to these resources

Put It All Together for Performance

Inter-Cluster Mapping

Intra-Cluster Mapping



Clustering

Balancing

Isolation

Radial Mapping

Improve Locality

Reduce Interference

Improve Shared Resource Utilization

**THANK
YOU!**

FOR

**YOUR
PRESENCE**

johnjose@iitg.ac.in

<http://www.iitg.ac.in/johnjose/>