Cooperative Caching for GPUs

Saumay Dublish, Vijay Nagarajan, Nigel Topham
The University of Edinburgh

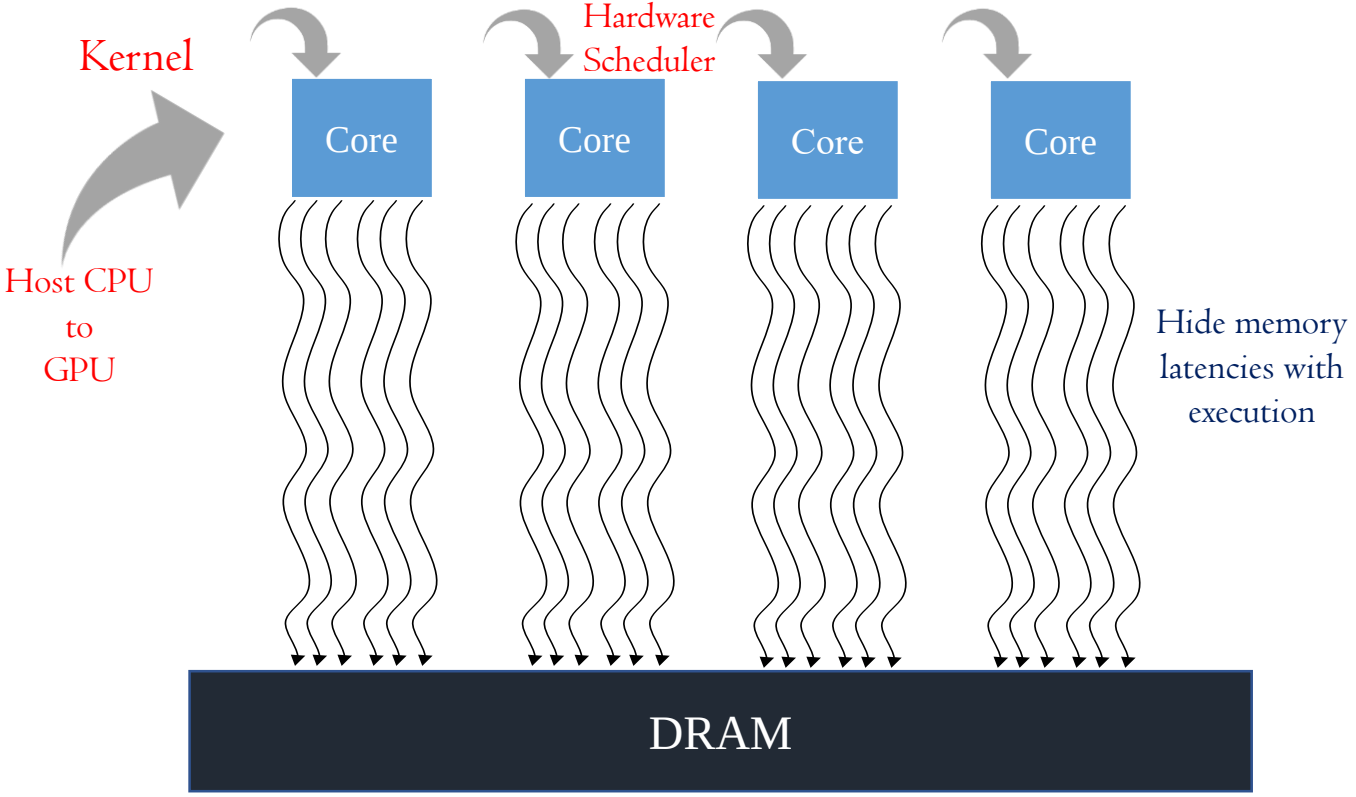
at

HiPEAC
Stockholm, Sweden
24th January, 2017

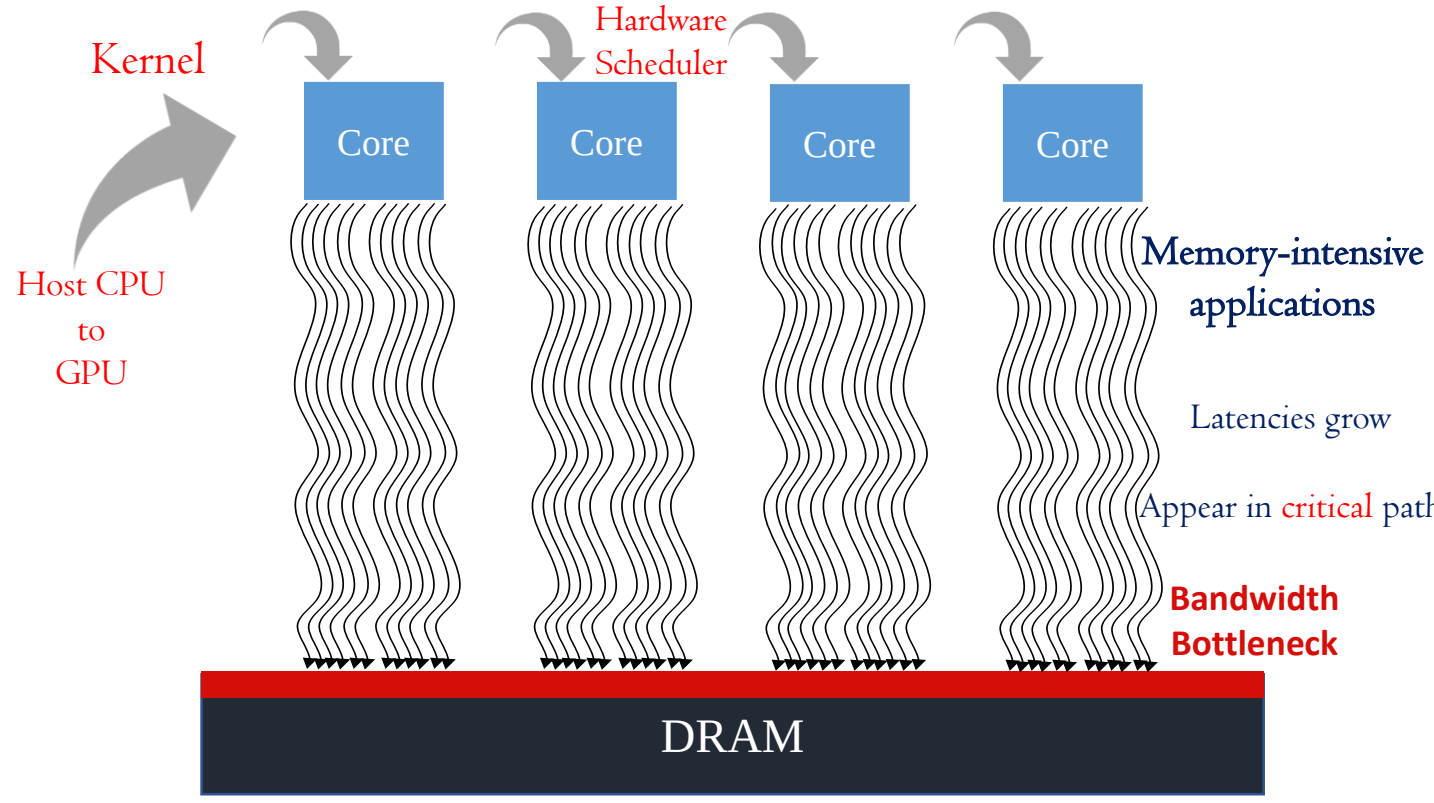
icsa | Institute for Computing Systems Architecture

THE UNIVERSITY of EDINBURGH

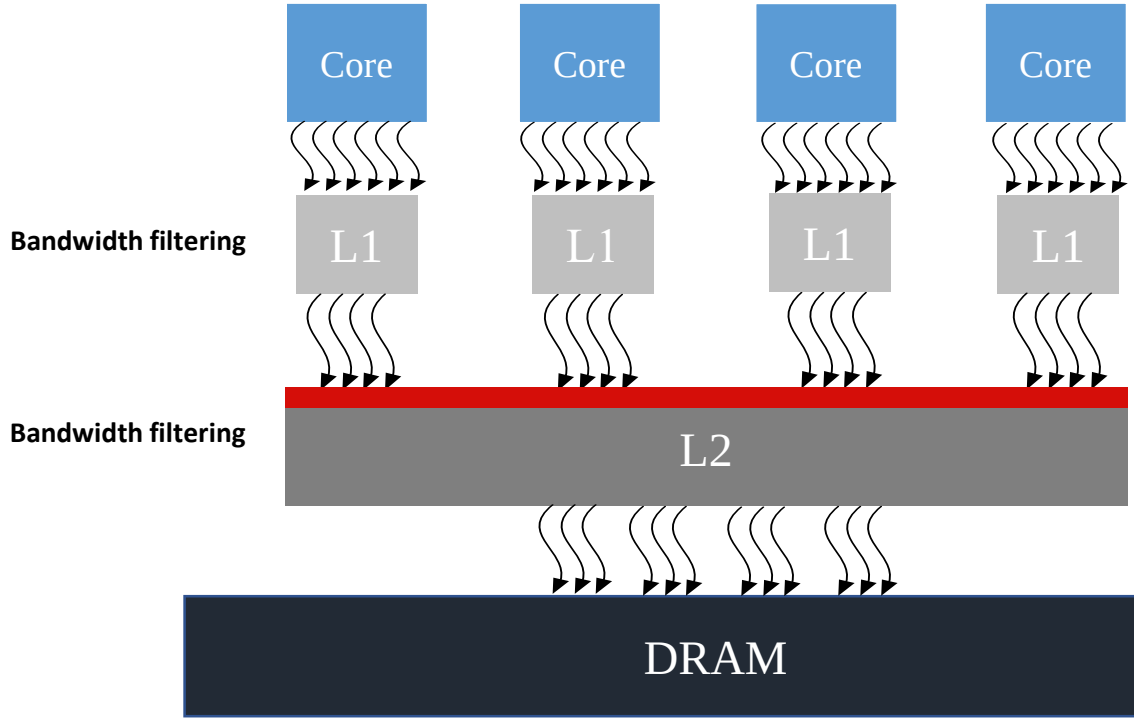
Multithreading on GPUs



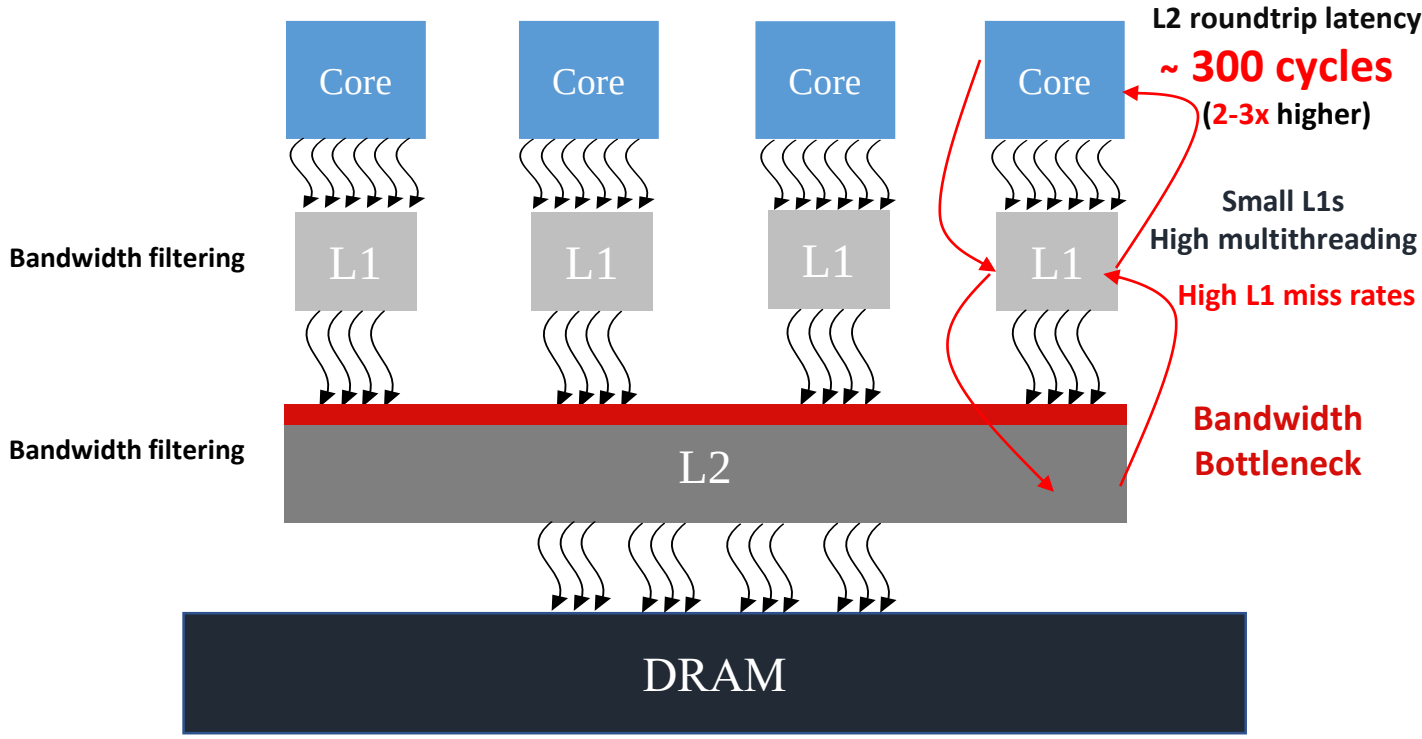
Multithreading on GPUs



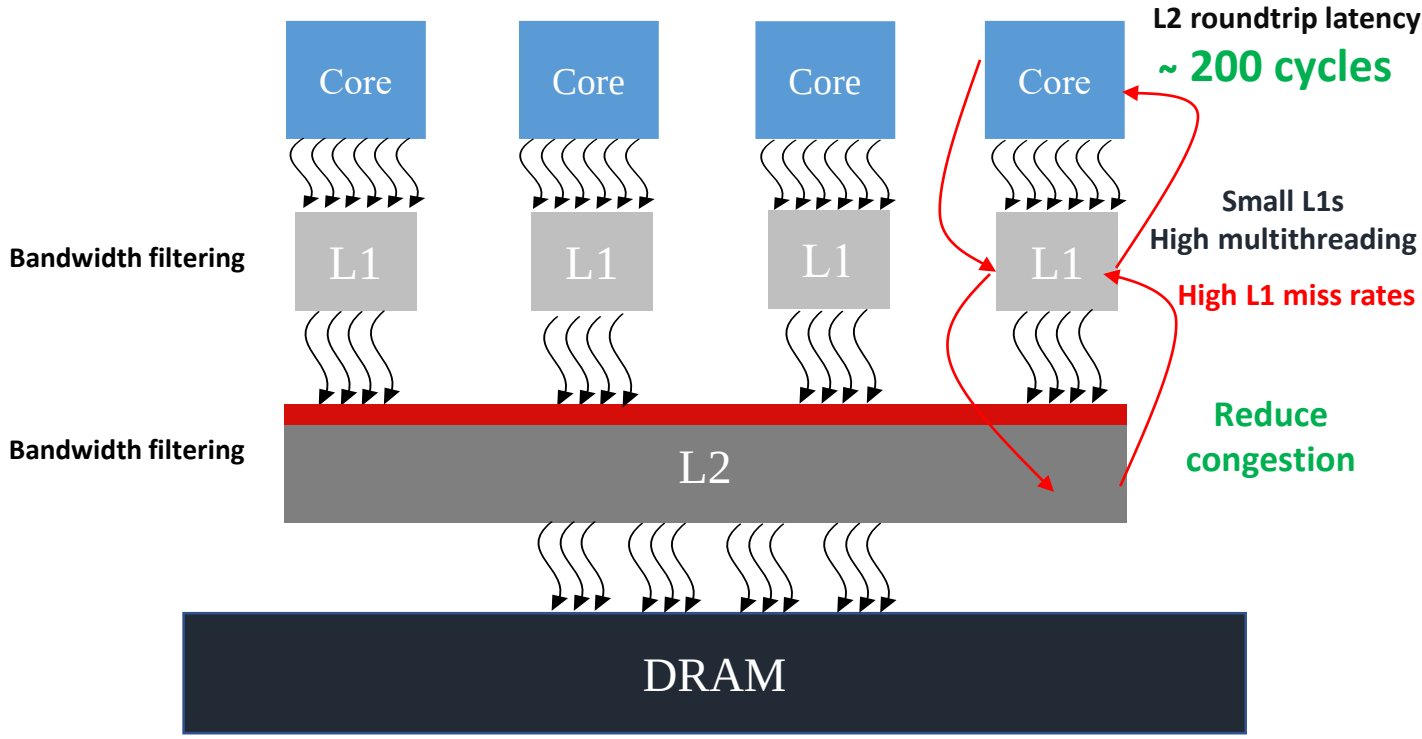
Deeper Memory Hierarchy



Deeper Memory Hierarchy



Deeper Memory Hierarchy



Outline

- Observations : Inter-core reuse
- Proposed Architecture : Cooperative Caching Network
- Results : Reduced L2 Congestion, Lower Latencies, Speedup
- Comparative study : Increasing L2 banks, Clustered sharing
- Conclusion

Observations

Graphics applications

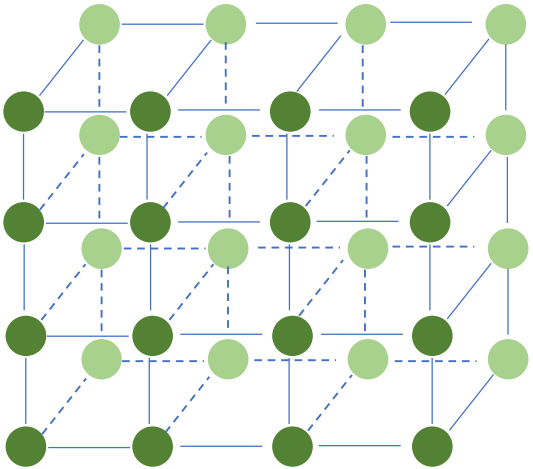
- Kernels work on independent data
- Thread-blocks execute in considerable isolation

General-purpose applications

- Thread-blocks tend to share data
- Thread-blocks scheduled on different cores lead to inter-core reuse

Inter-core Reuse

Benchmark: Coulombic Potential *(cutcp)*



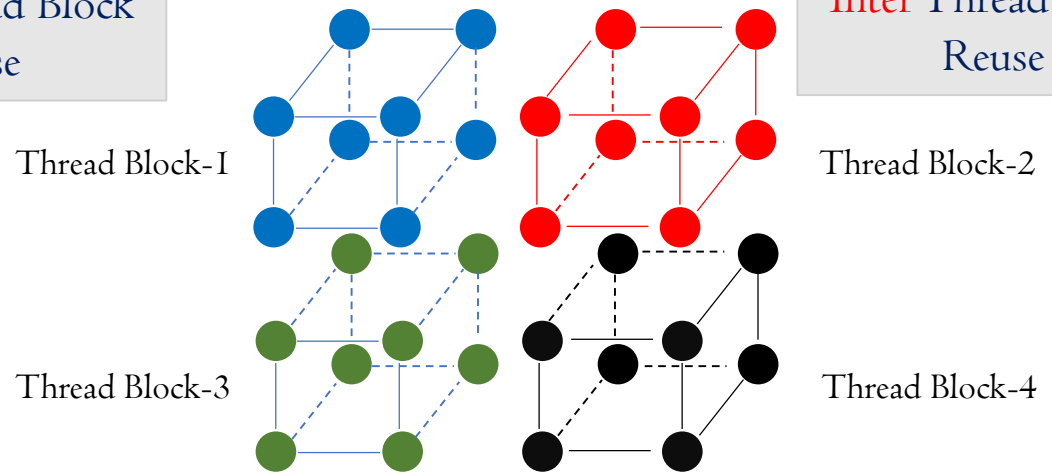
Compute electrostatic potential from neighbours

Inter-core Reuse

Benchmark: Coulombic Potential *(cutcp)*

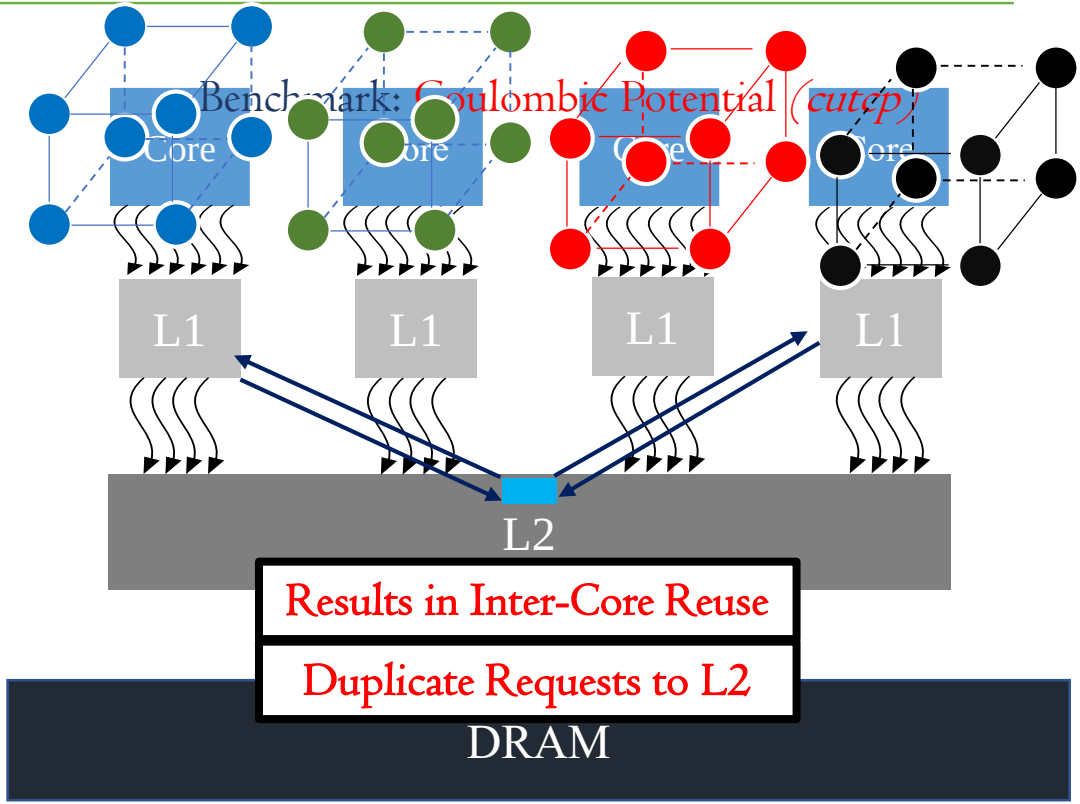
Intra Thread Block Reuse

Inter Thread Block Reuse

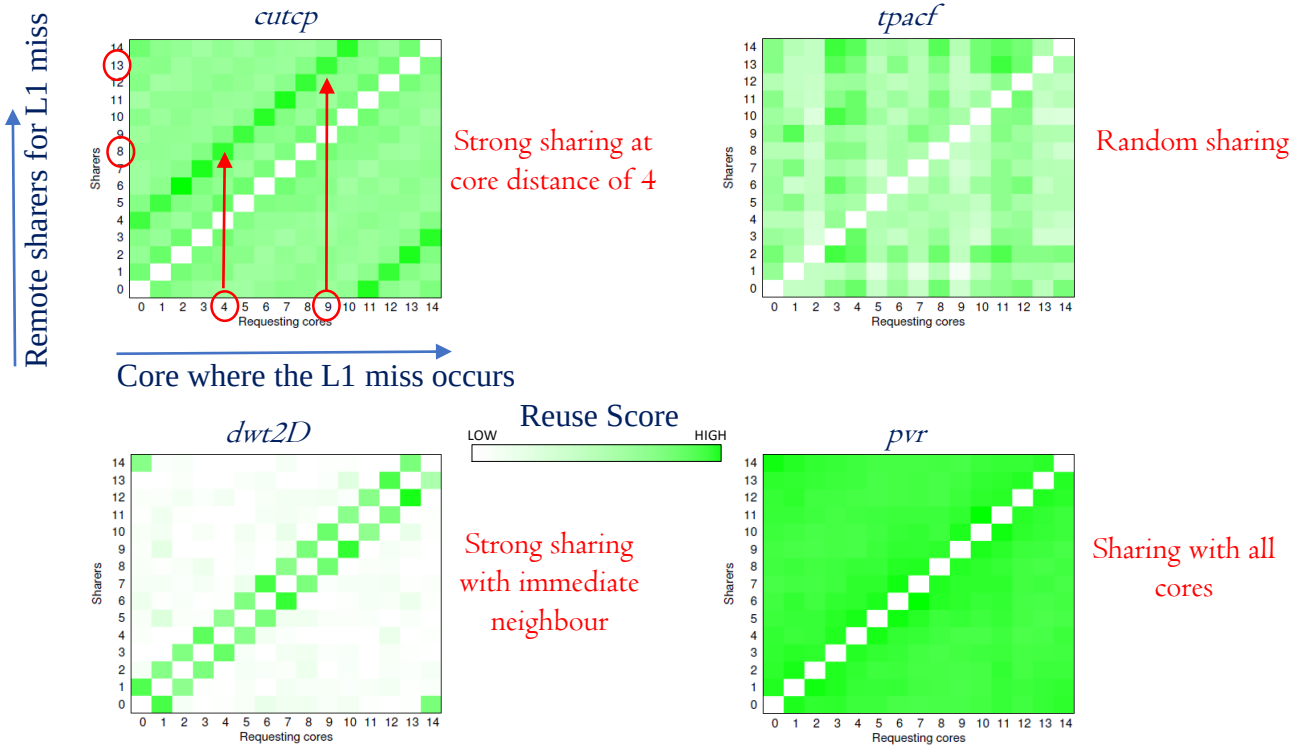


Compute electrostatic potential from neighbours

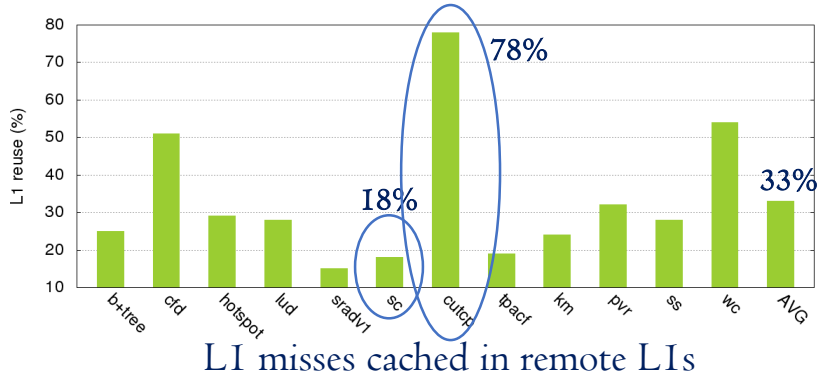
Inter-core Reuse



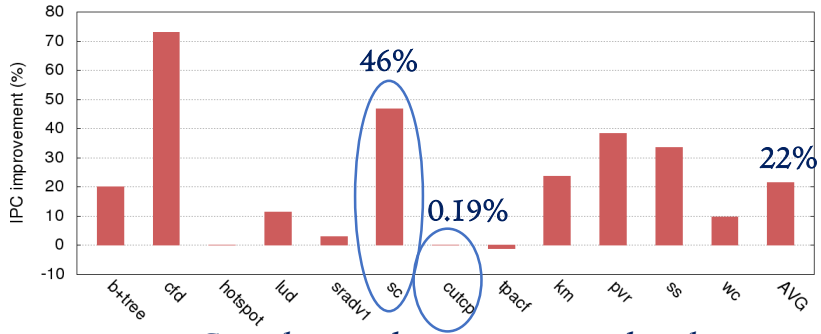
Reuse patterns



Efficacy of Reuse



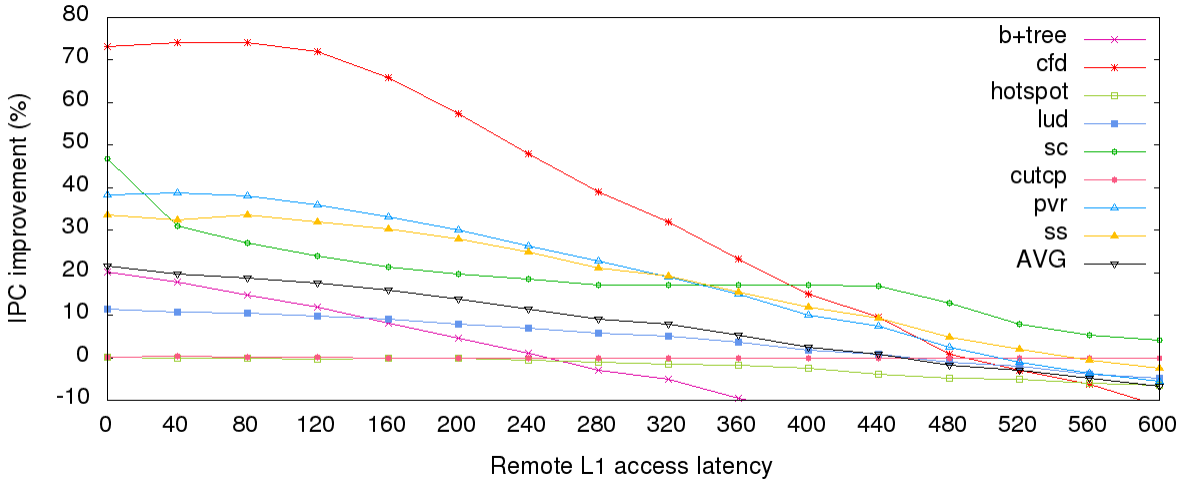
LI misses cached in remote LIs



Speedup with no reuse overhead

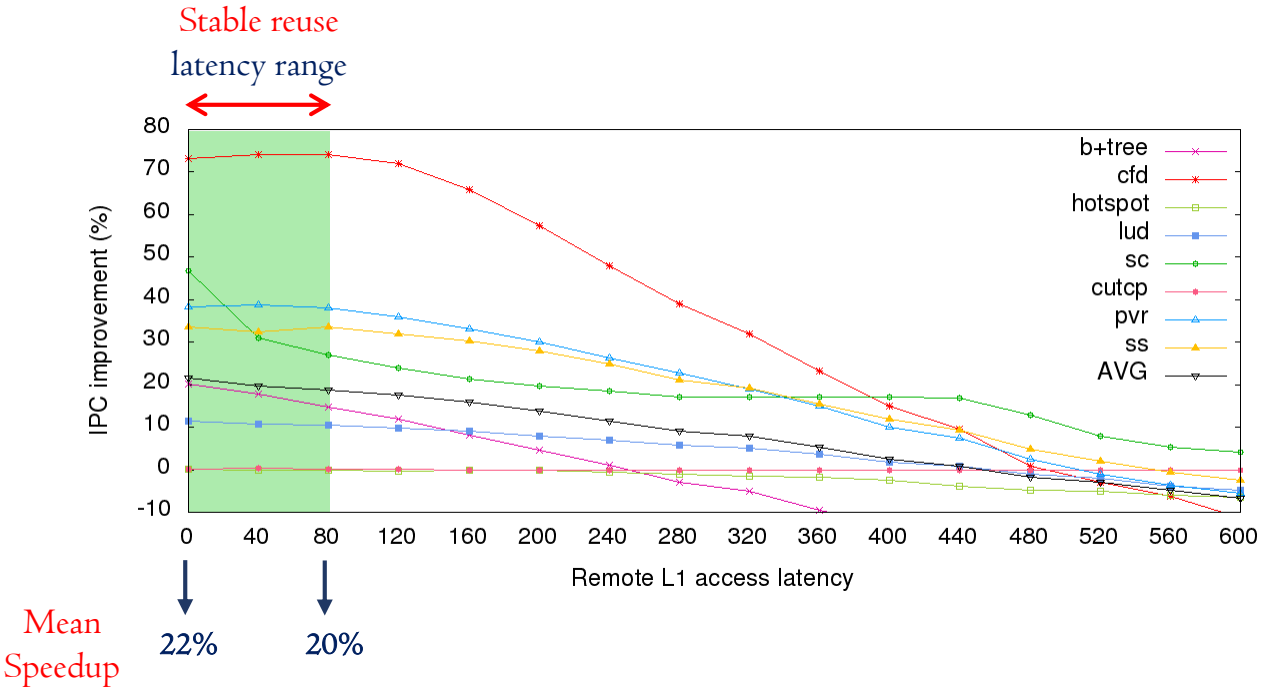
Sensitivity to Reuse Overhead

(Performance as a function of remote L1 access latencies)



Sensitivity to Reuse Overhead

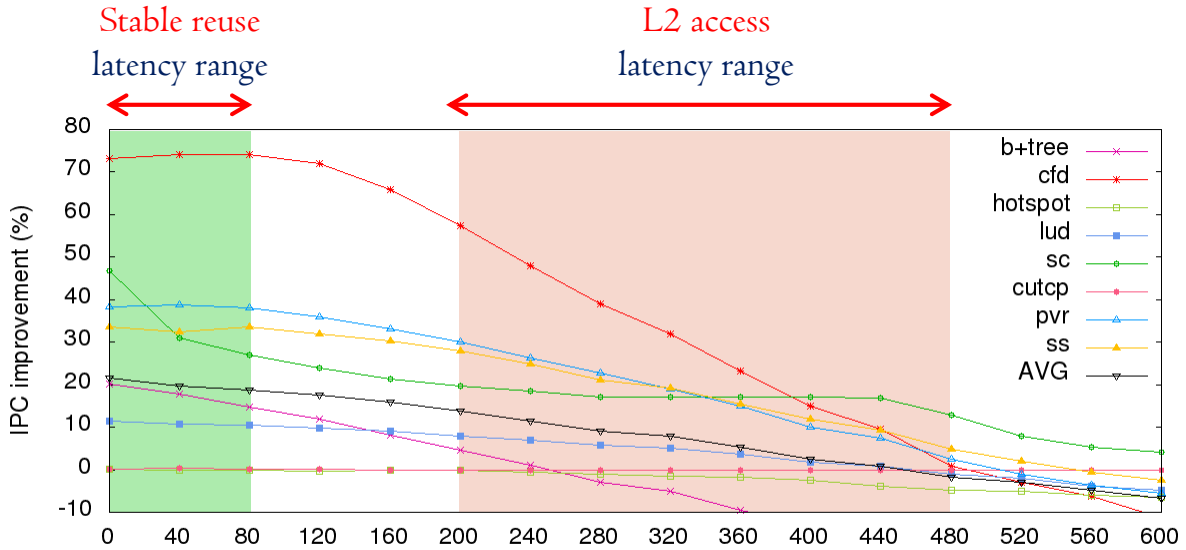
(Performance as a function of remote L1 access latencies)



Permissible range for each remote L1 access: 0-80 cycles

Sensitivity to Reuse Overhead

(Performance as a function of remote L1 access latencies)



Permissible range for each remote L1 access: 0-80 cycles

Proposal

Cooperative Caching Network

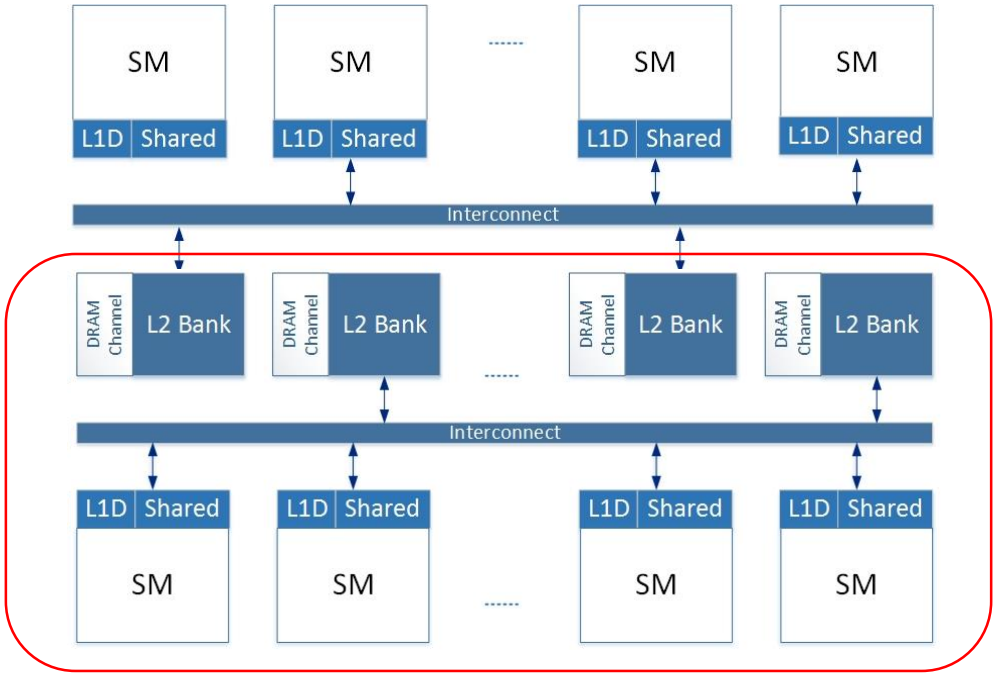
- Lightweight **ring-based** network for inter-core communication
- All core-to-core connections are **near-neighbour**
- Fewest number of inter-core connections compared to other topologies
- Routers are simple **multiplexers**
- Leverages latency tolerance of up to **80 cycles**

Key design point

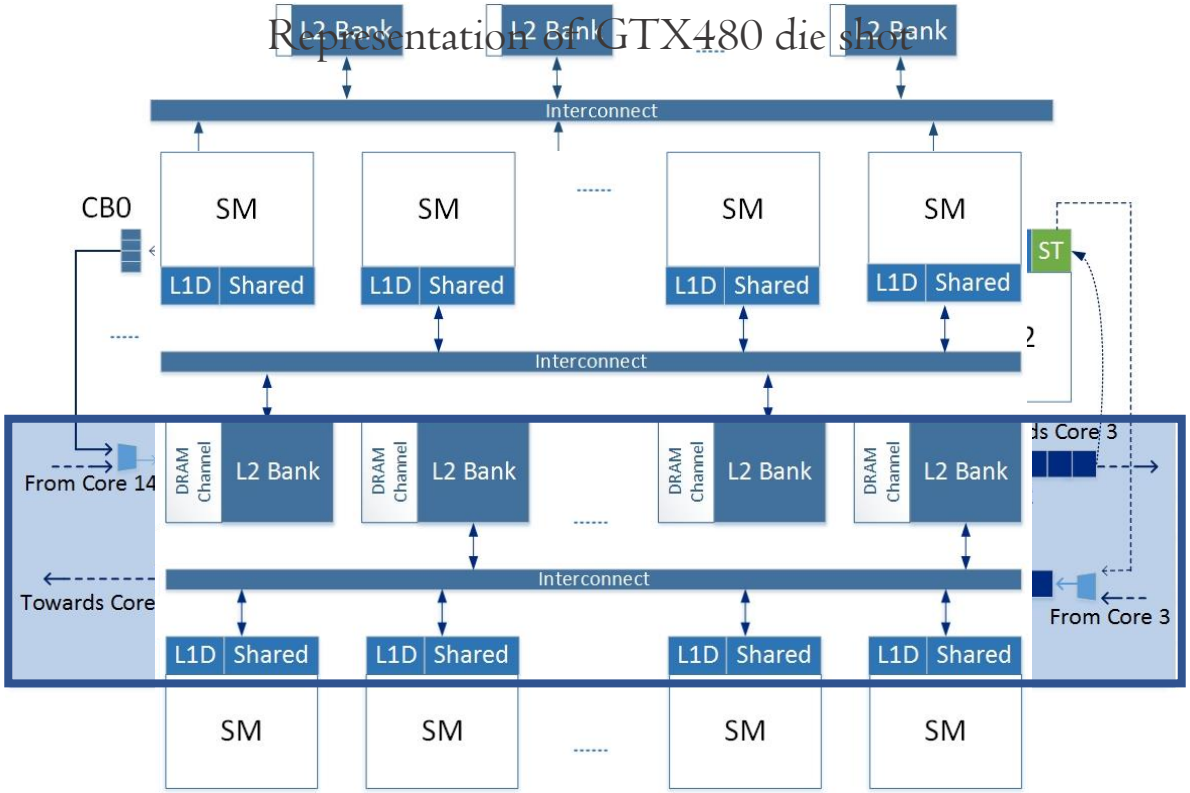
Trade-off higher latencies for simplicity and short wires

Baseline GPU

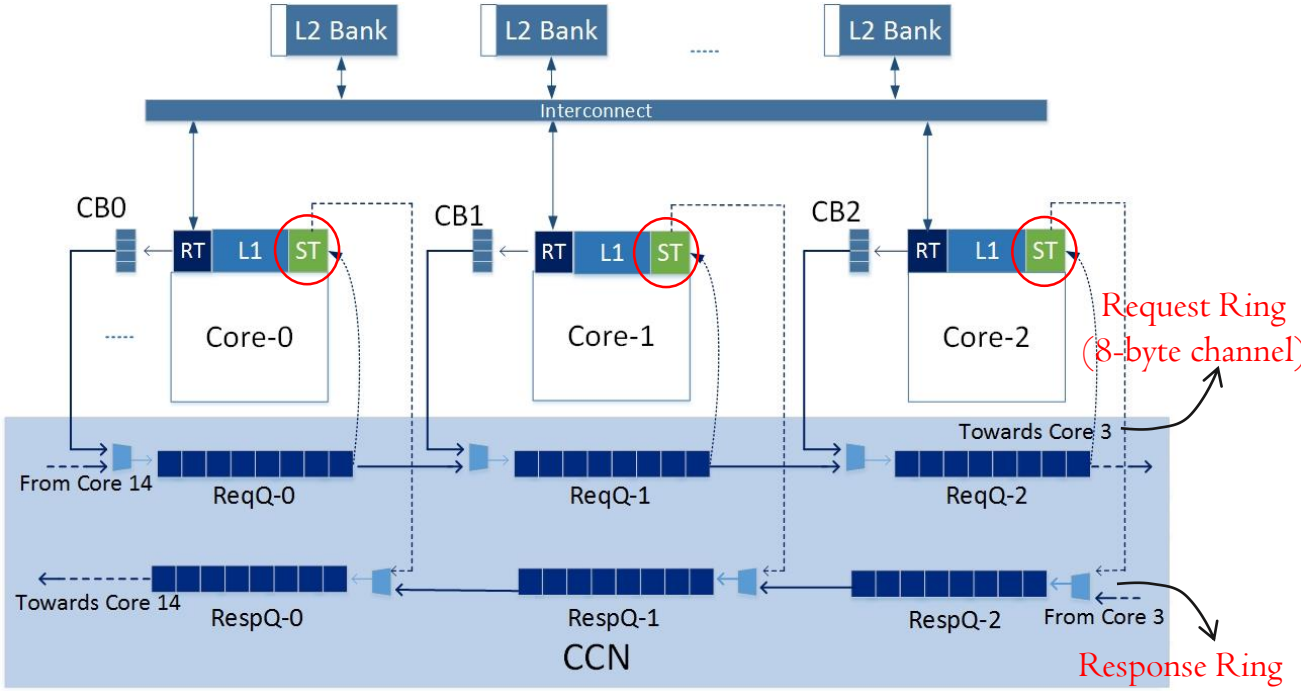
Representation of GTX480 die shot



Cooperative Caching Network (CCN)

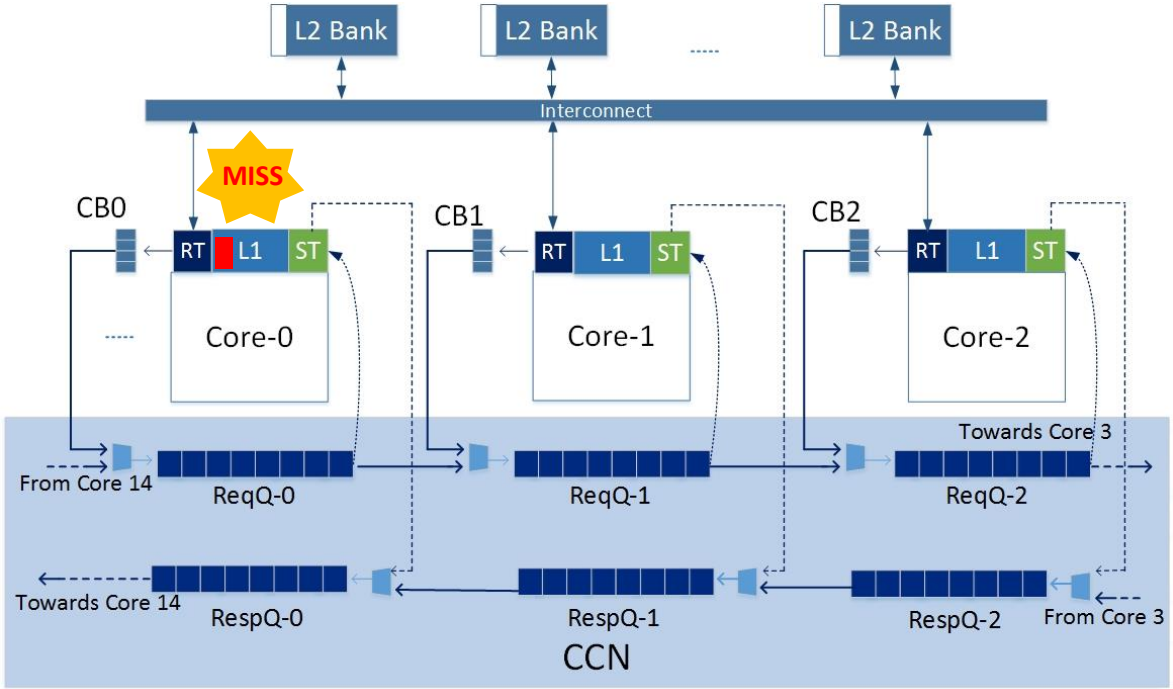


Cooperative Caching Network (CCN)

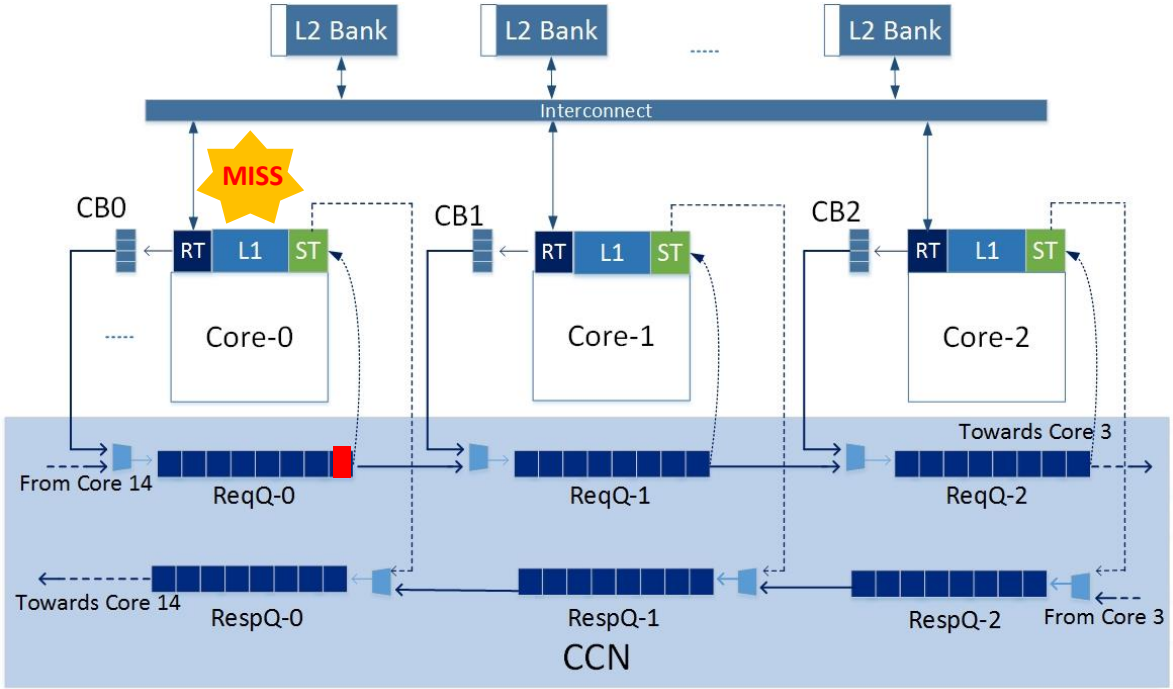


- Two unidirectional rings : Request and Response
- Shadow Tags at each L1 cache

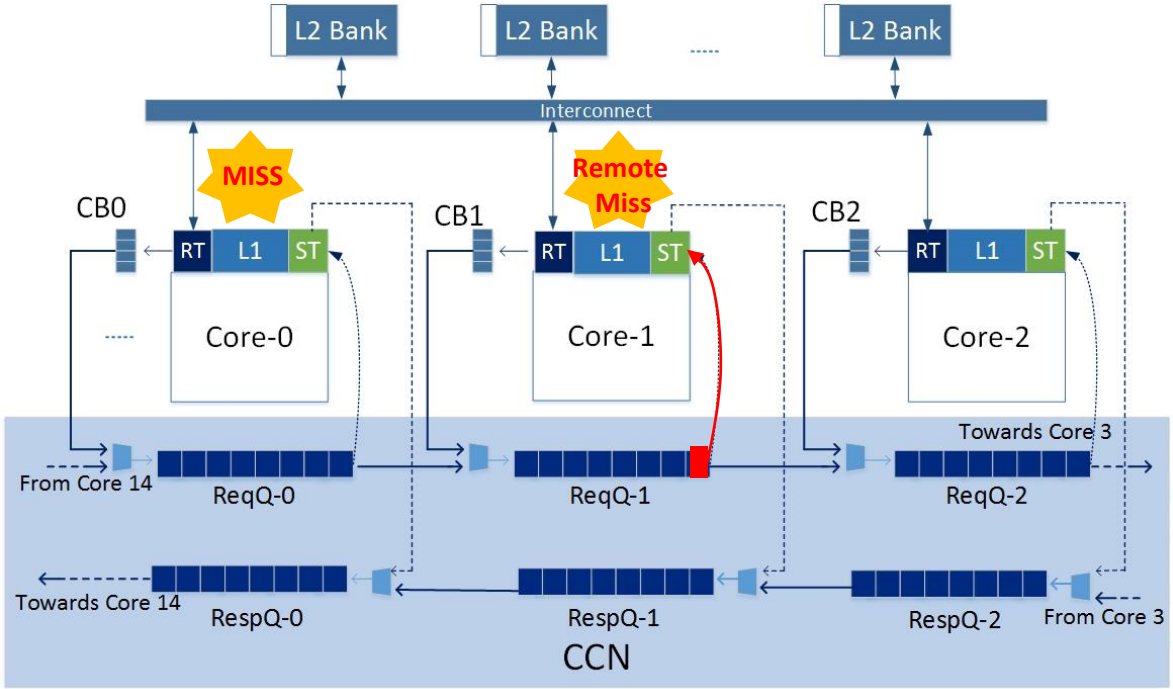
Cooperative Caching Network (CCN)



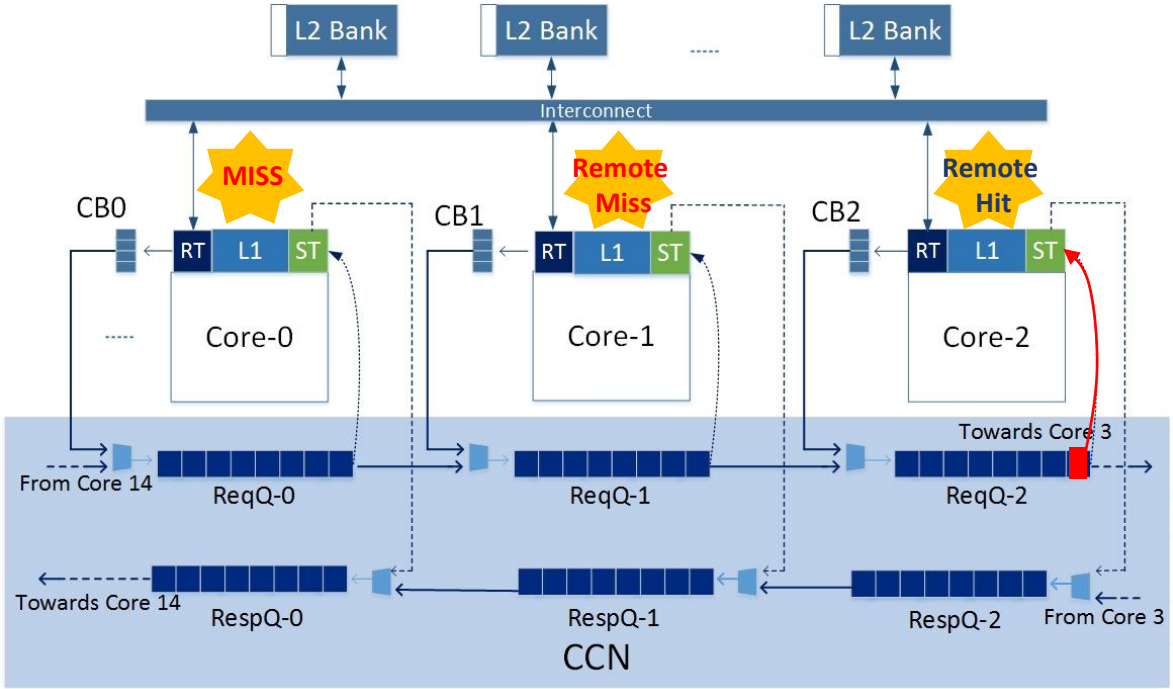
Cooperative Caching Network (CCN)



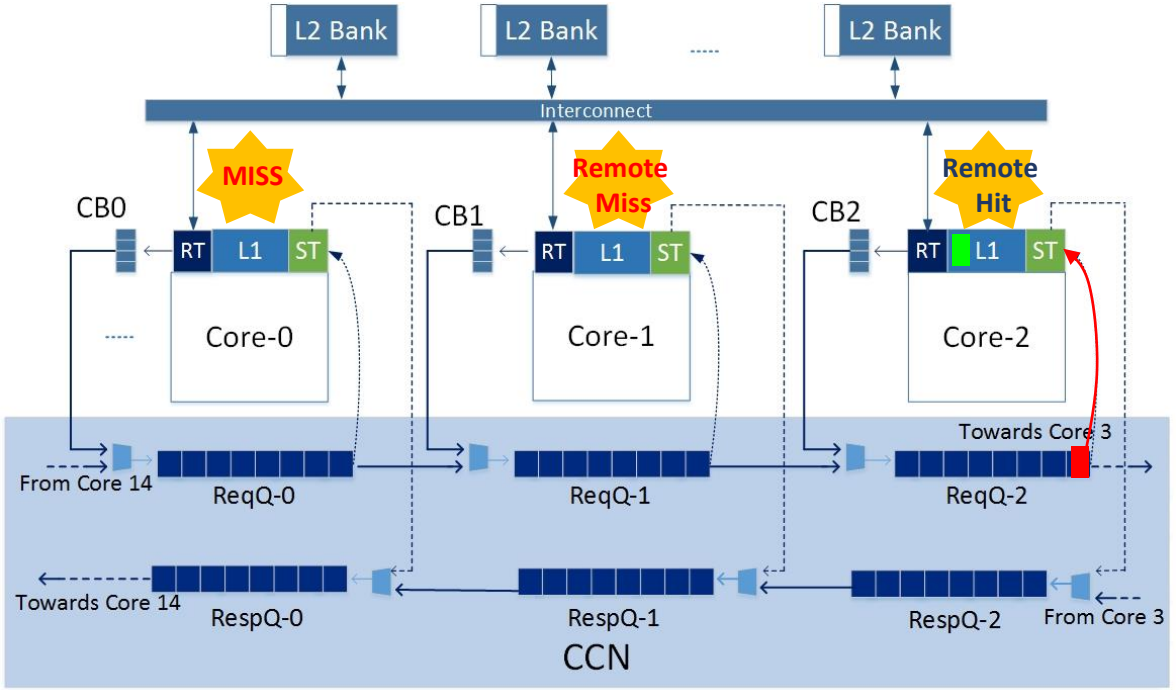
Cooperative Caching Network (CCN)



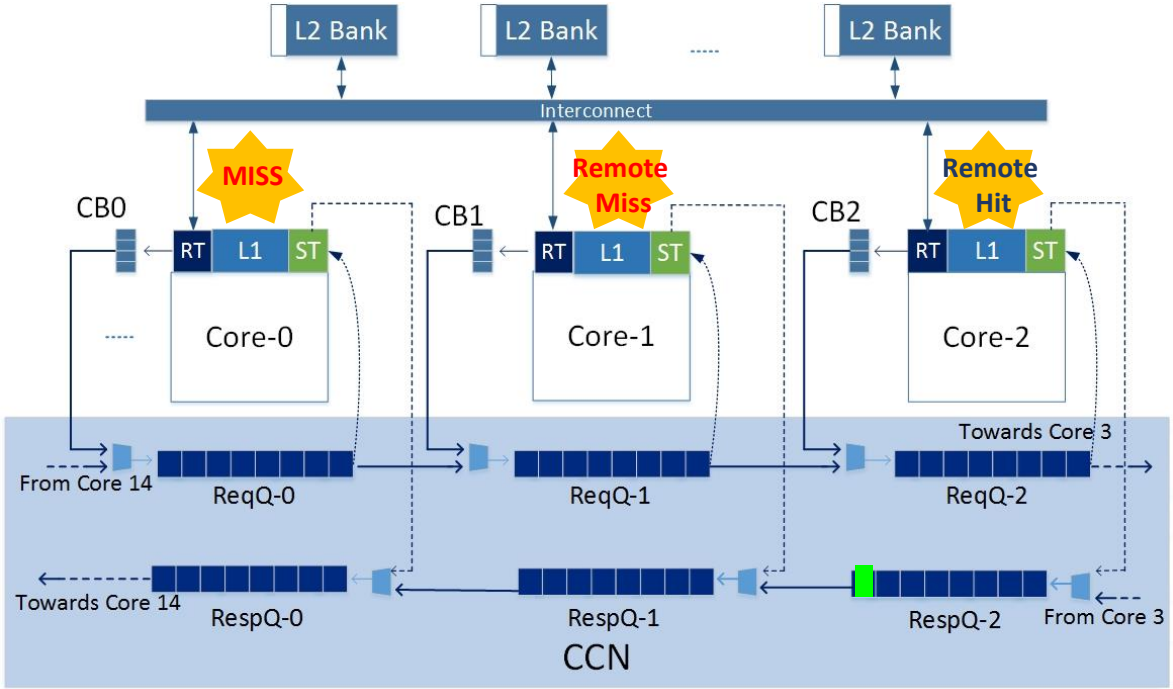
Cooperative Caching Network (CCN)



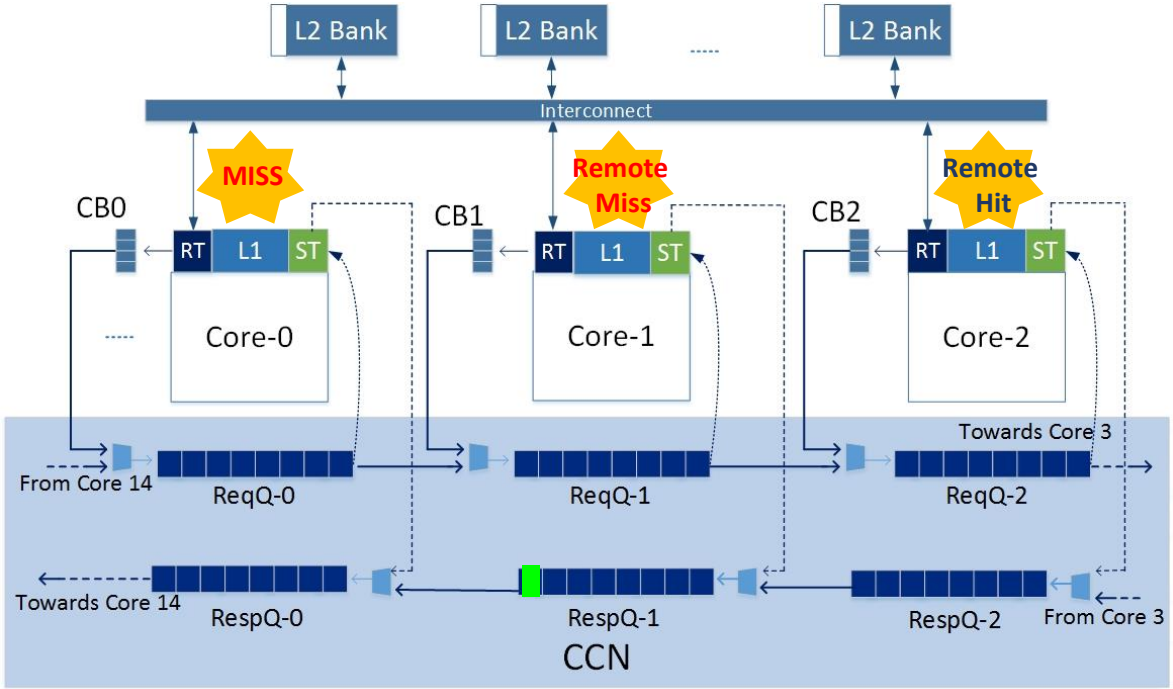
Cooperative Caching Network (CCN)



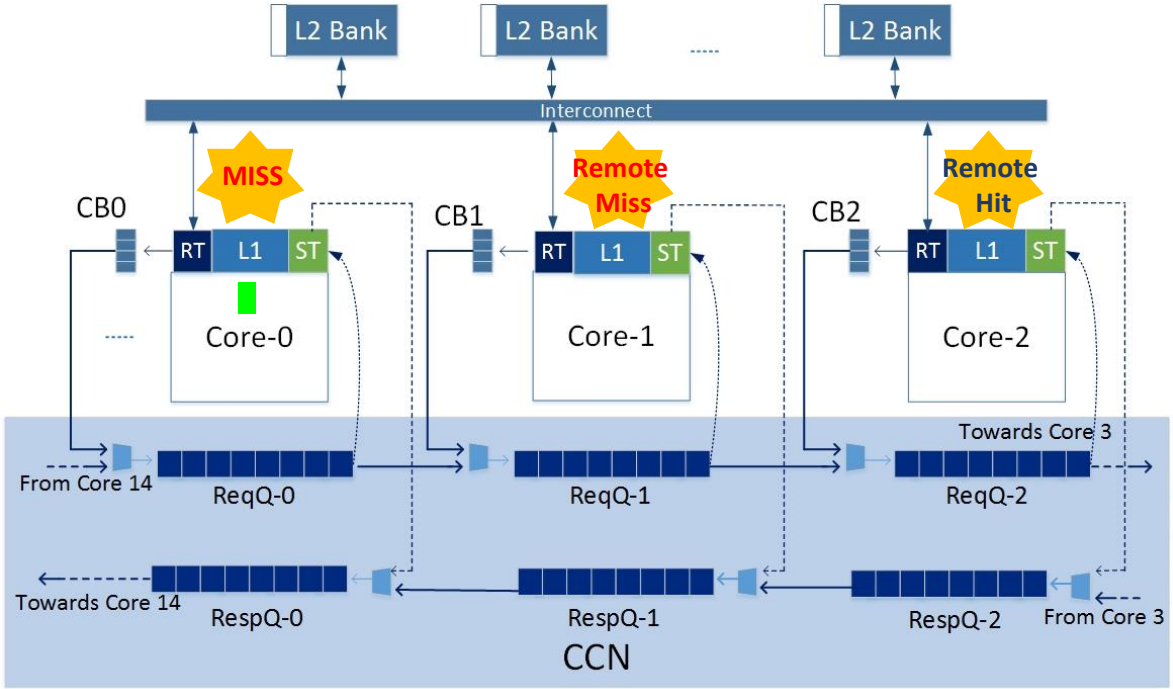
Cooperative Caching Network (CCN)



Cooperative Caching Network (CCN)



Cooperative Caching Network (CCN)



Source of Speedup?

- Reduced latency due to less congestion in L2 cache outstrips the overhead of traversing the ring
- Average memory latencies, therefore, are lower with CCN
- Detailed analytical model in the paper

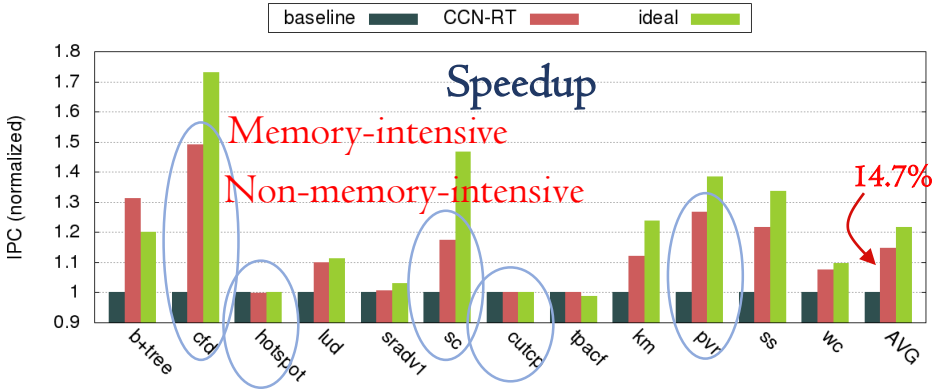
In the paper …

- **Request Throttling** (CCN-RT)
  - Record statistics about the presence of inter-core reuse
  - When no reuse is detected, throttles requests directly to L2
  - Maintains low congestion in the ring and minimizes ring overhead
- **Memory Consistency**
  - GPUs employ weak memory consistency model
  - CCN does not further weaken the existing memory model

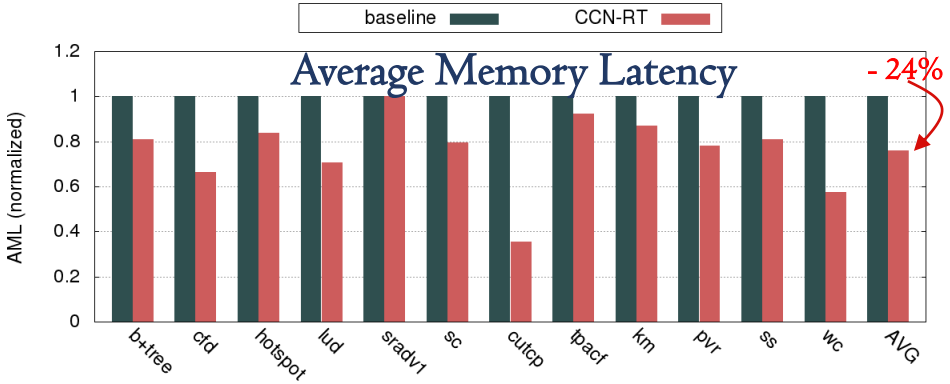
Evaluation

- **Platform**
  - GPGPU-Sim (v3.2.2)
  - GPUWattch (McPAT)
- **Benchmarks**
  - Rodinia
  - Parboil
  - MapReduce

Results

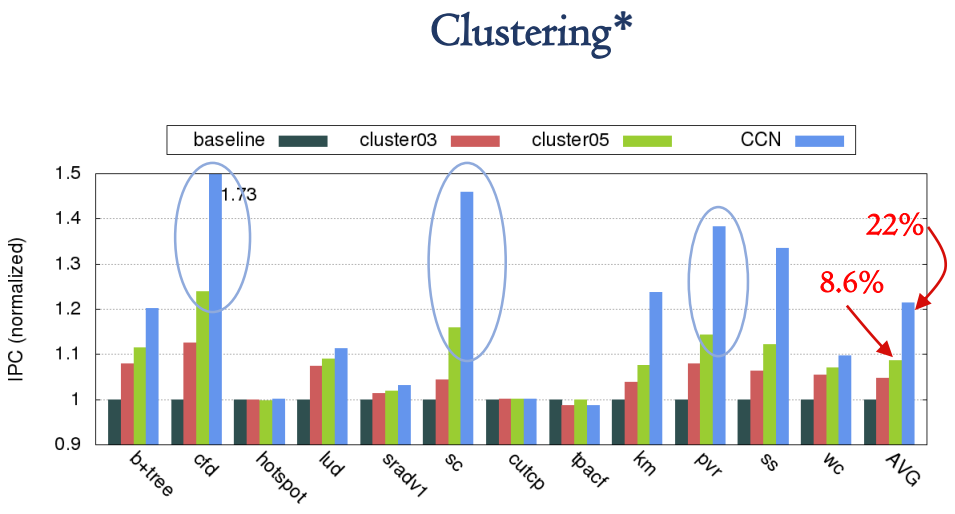

baseline
CCN-RT
ideal
Speedup
Memory-intensive
Non-memory-intensive
14.7%
IPC (normalized)
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1
0.9
b+tree
cfd
hotspot
lud
sradv1
sc
cutcp
tpacf
km
pvr
ss
wc
AVG

Results



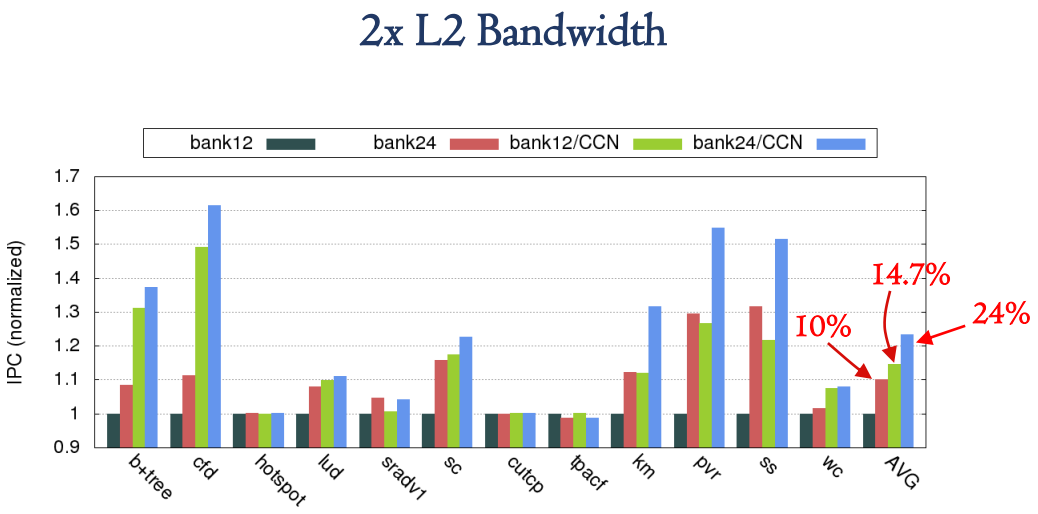
- Overheads
  - Area: 1.3%
  - Energy: 2.5%

Comparative Study



*Keshtegar *et. al.* Cluster-based approach for improving graphics processing unit performance by inter streaming multiprocessors locality, IET-CDT 2015

Comparative Study



Conclusion

- Problem :
  - High congestion between L1 and L2
  - Congestion leads to high memory latencies
  - High latencies appear in the critical path for memory-intensive applications
- Observation :
  - Considerable inter-core reuse in GPGPU applications
  - GPUs can tolerate reuse latencies gracefully up to 80 cycles
  - An aggressive policy/network to fetch sharers is an overkill
- Proposal :
  - Propose a ring-based Cooperative Caching Network
  - Trades-off higher latencies for simplicity and cost
  - Reduces congestion in the L1-L2 access path
  - Lower average memory latencies

Questions?

Saumay Dublish
saumay.dublish@ed.ac.uk
http://homepages.inf.ed.ac.uk/s1433370/