NUMA Implication for Storage I/O Throughput in Modern Servers

Shoaib Akram, Manolis Marazakis, and Angelos Bilas
Computer Architecture and VLSI Laboratory
FORTH-ICS
Greece

Outline

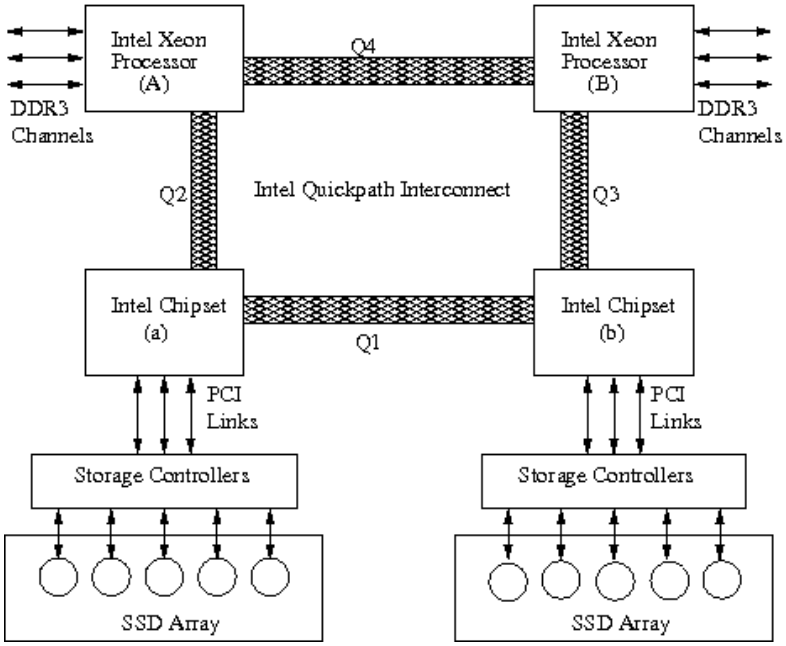
- Introduction
- Motivation
- Previous Work
- The “Affinity” Space Exploration
- Evaluation Methodology
- Results
- Conclusions

Introduction

- The number of processors on a single motherboard is increasing.
- Each processor has faster access to local memory compared to remote memory leading to the well-known NUMA problem.
- Much effort in the past is devoted to NUMA-aware memory management and scheduling.
- There is a similar **"non-uniform latency of access"** relation between processors and I/O devices.
- In this work, we quantify the combined impact of non-uniform latency in accessing memory and I/O devices.

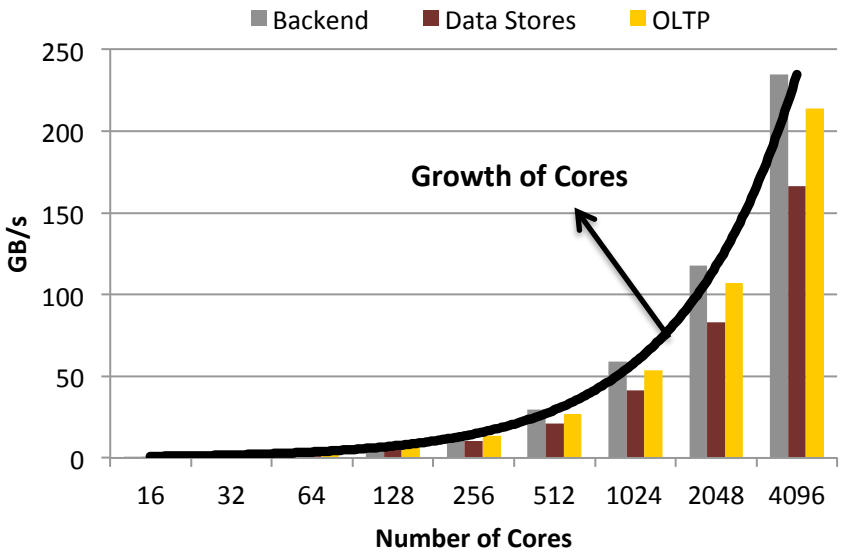
Motivation-Trends in I/O Subsystems

- Storage-related features:
  - Fast point-to-point interconnects.
  - Multiple storage controllers.
  - Arrays of storage devices with high bandwwidth.
- Result:
  - Storage bandwidth is catching up with memory bandwidth.
  - Today, NUMA affinity is a problem for the entire system.



- **The anatomy of a modern server machine with 2 NUMA domains.**

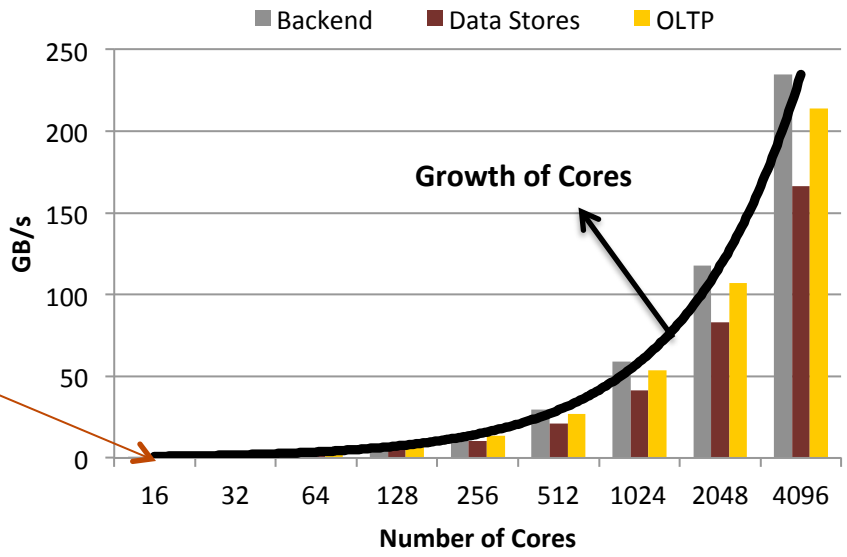
Motivation-Trends in Processor Technology and Applications



- Throughput of typical workloads in today's data-centres measured for 16 cores and projected to many cores.

Motivation-Trends in Processor Technology and Applications

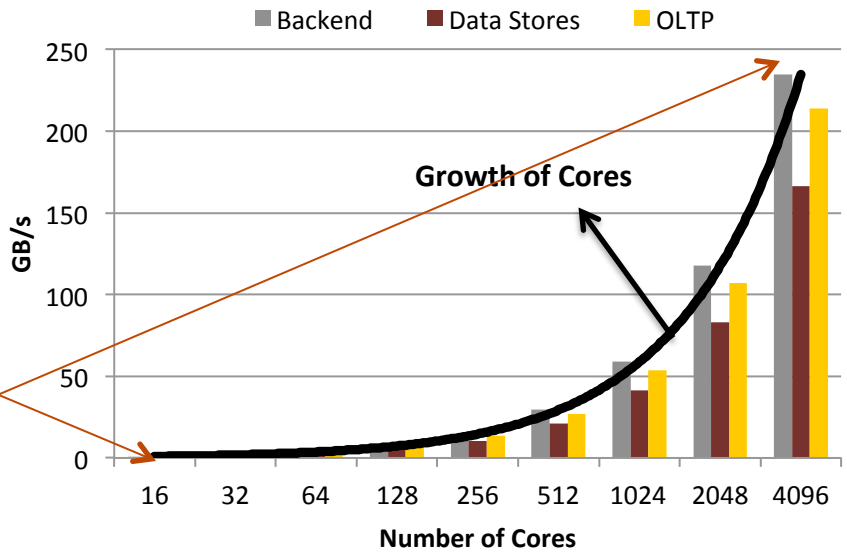
- Today:
  - Low device throughput.
  - High per I/O cycle overhead.
  - Low server utilization in data-centres.



- Throughput of typical workloads in today's data-centres measured for 16 cores and projected to many cores.

Motivation-Trends in Processor Technology and Applications

- Today:
  - Low device throughput.
  - High per I/O cycle overhead.
  - Low server utilization in data-centres.
- In Future:
  - Many GB/s of device throughput.
  - Low per I/O cycle overhead as system stacks improve.
  - High server utilization in data-centres through server consolidation etc.



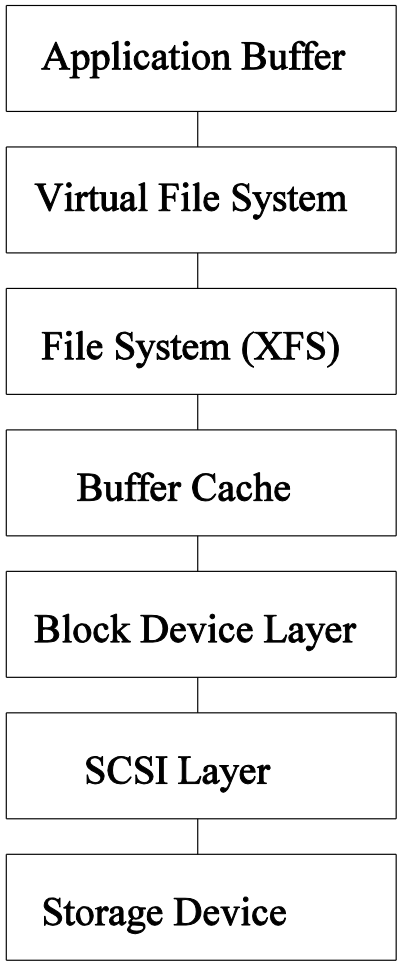
- Throughput of typical workloads in today's data-centres measured for 16 cores and projected to many cores.

Previous Work

- Multiprocessors
- Multicore Processors

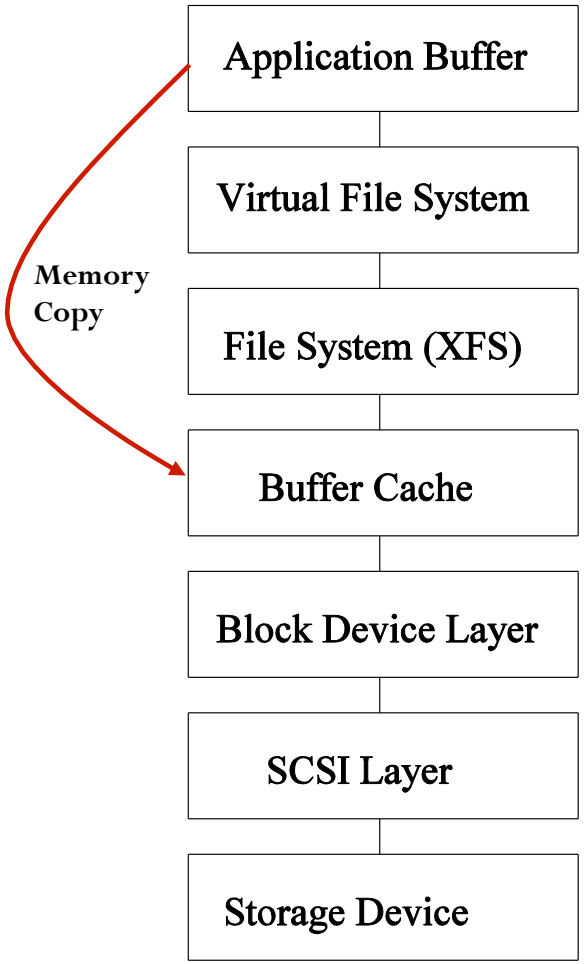
The “Affinity” Space Exploration

- The movement of data on any I/O access:



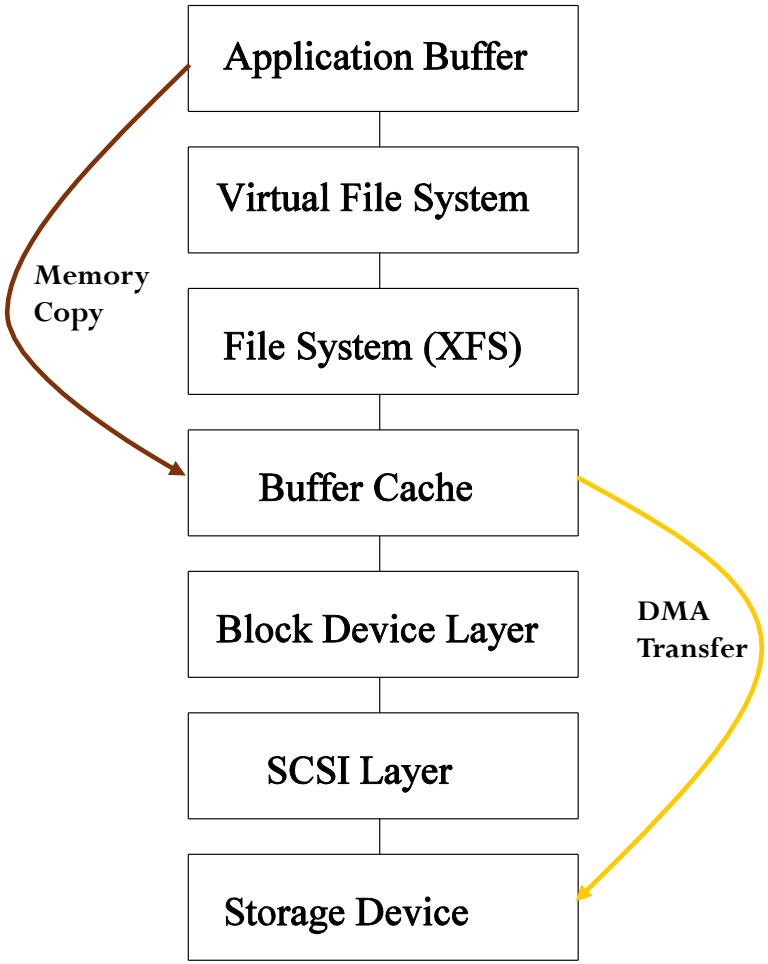
The “Affinity” Space Exploration

- The movement of data on any I/O access:
  - to/from the application buffer from/to the kernel buffer.



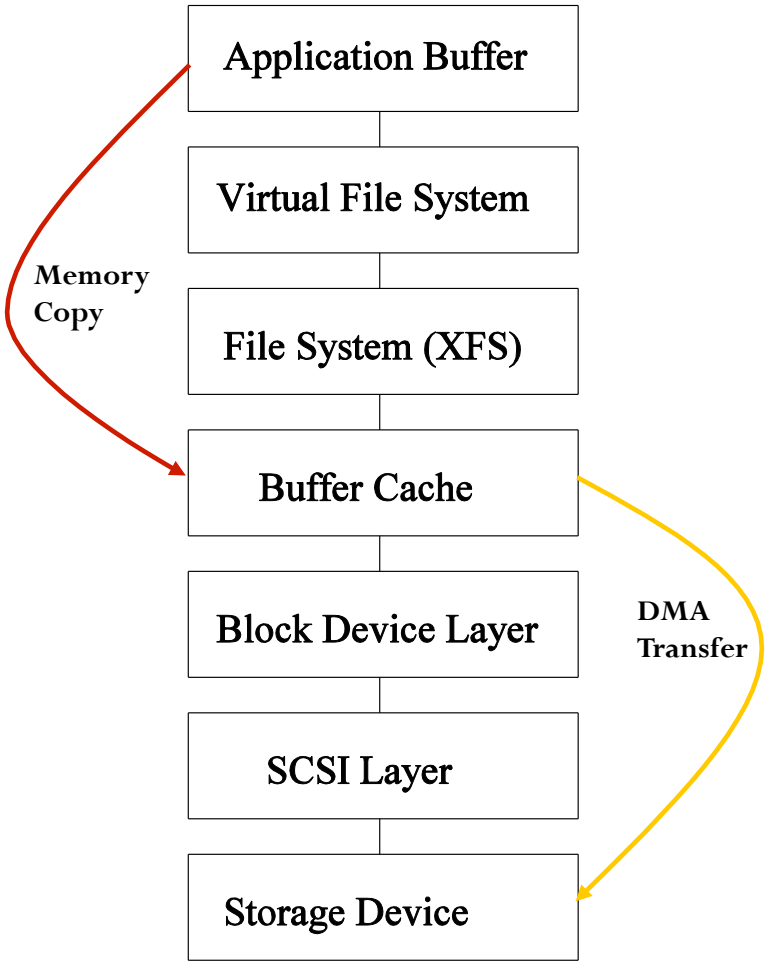
The “Affinity” Space Exploration

- The movement of data on any I/O access:
  - to/from the application buffer from/to the kernel buffer.
  - to/from the kernel buffer from/to the I/O device.



The “Affinity” Space Exploration

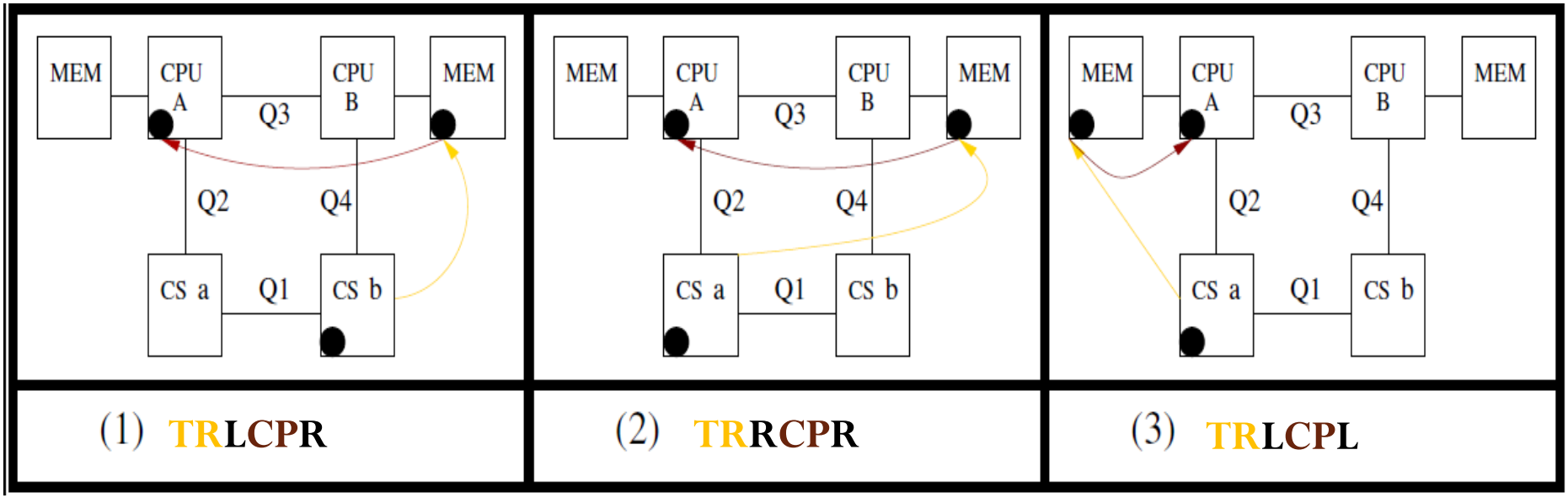
- The movement of data on any I/O access:
  - to/from the application buffer from/to the kernel buffer.
  - to/from the kernel buffer from/to the I/O device.
- Application and kernel buffers could be located in different NUMA domains.
- Kernel buffers and I/O devices could be located in different NUMA domains.



The "Affinity" Space Exploration

- The axis of the space are:
  - Transfer between I/O devices and kernel buffers.
  - Copy from kernel buffer to the application buffer.

Transfer (TR)	Copy (CP)	Configuration
Local (L)	Local (L)	TRLCPL
Local (L)	Remote (R)	TRLCPR
Remote (R)	Local (L)	TRRCPL
Remote (R)	Remote (R)	TRRCPR



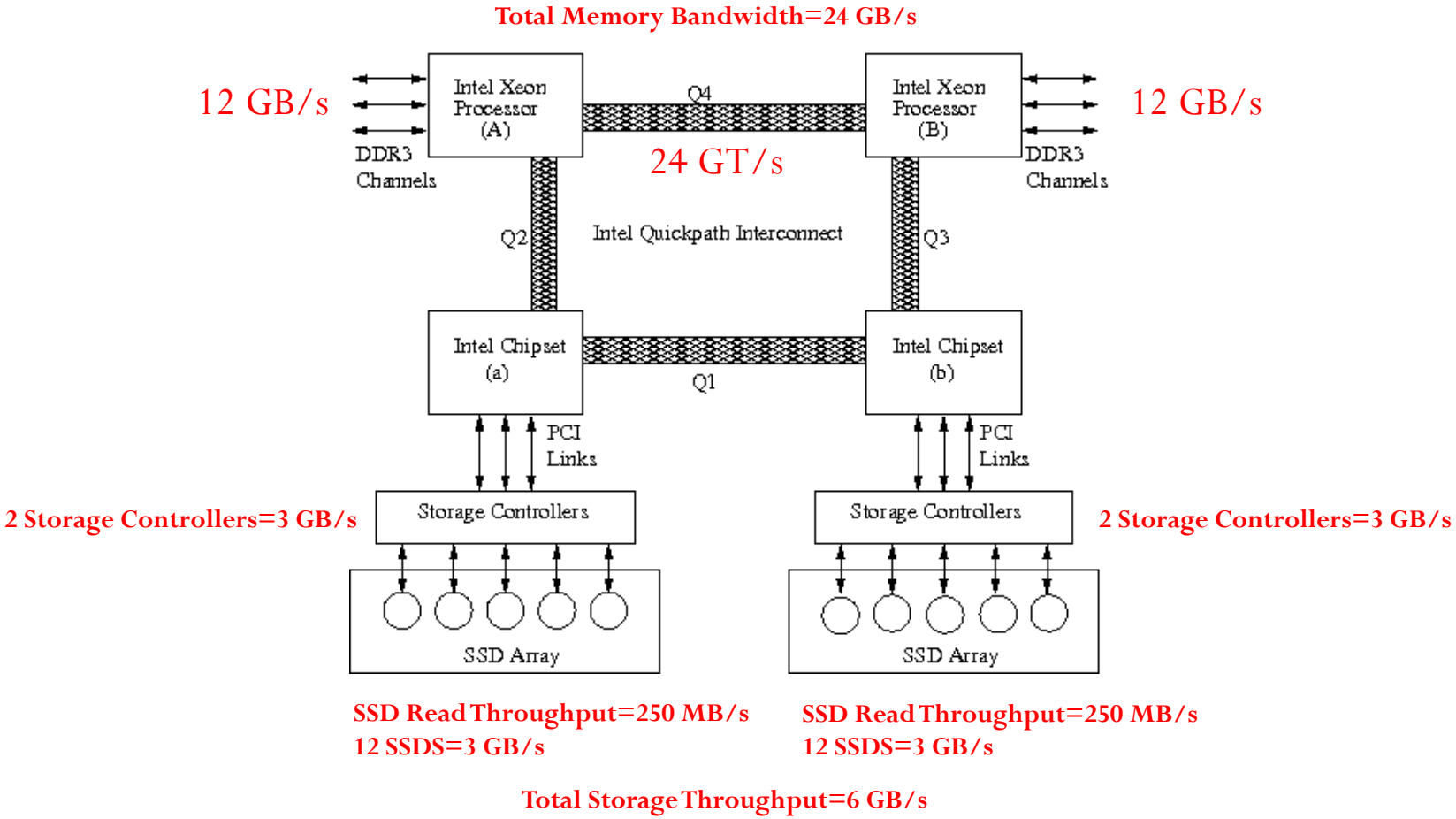
Example Scenarios

NUMA Memory Allocation

- How application buffers are allocated?
  - The domain in which the process is executing.
  - The *numactl* utility allows pinning application processes and the memory buffers on a particular domain.
- How kernel buffers are allocated?
  - Kernel buffer allocation can not be controlled using *namactl*.
  - I/O buffers in the kernel are shared by all contexts performing I/O in the kernel.
- How to "maximize possibility of keeping kernel buffers in a particular socket?"
  - Start each experiment with a clean buffer cache.
  - Large DRAM/Socket compared to application datasets.

Evaluation Methodology

- Bandwidth characterization of evaluation testbed.



Benchmarks, Applications and Datasets

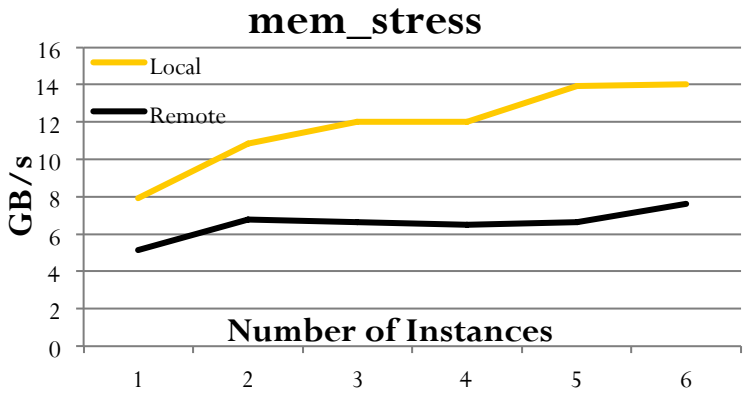
- Benchmarks and Applications
  - *zmIO*: in-house microbenchmark.
  - *fsmark*: a filesystem stress benchmarks.
  - *stream*: a streaming workload.
  - *psearchy*: a file indexing application (part of MOSBENCH).
  - *IOR*: application checkpointing.
- Workload Configuration and Datasets
  - Four software RAID Level 0 devices, each on top of 6 SSDS and 1 storage controller.
  - One workload instance per RAID device.
  - Datasets consist of **large files**, parameters that result in **high concurrency**, **high read/write throughput** and **stressing of system stack**.

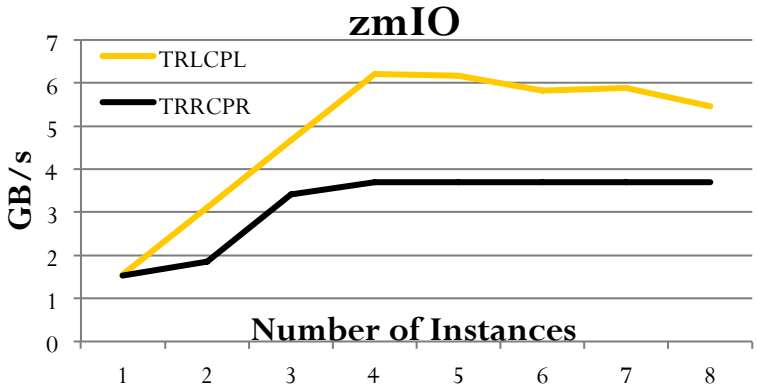
Evaluation Metrics

- Problem with high-level application metrics:
  - Not possible to map actual volume of data transmitted.
  - Not possible to look at indiviual components of complex software stacks.
- It is important to look at indiviual components of execution time(user, system, idle, and iowait).
- Cycles per I/O
  - Physical cycles consumed by the application during the execution time divided by the number of I/O operations.
  - Can be used as an efficiency metric.
  - Can be converted to energy per I/O.

Results – mem_stress and zmIO

- Remote memory accesses:
  - Memory throughput drops by one half.
  - The degradation starts from one instance of mem_stress.
- Remote transfers:
  - Device throughput drops by one half.
  - The throughput is same for one instance.
  - Contention is a possible culprit for two and more instances.

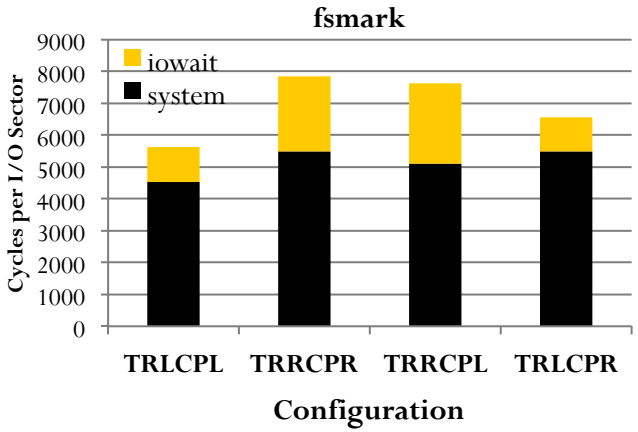


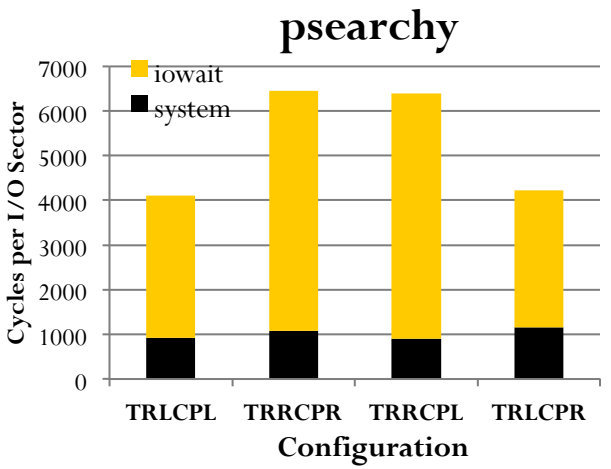


Round-robin assignments of instances to RAID devices.

Results – fsmark and psearchy

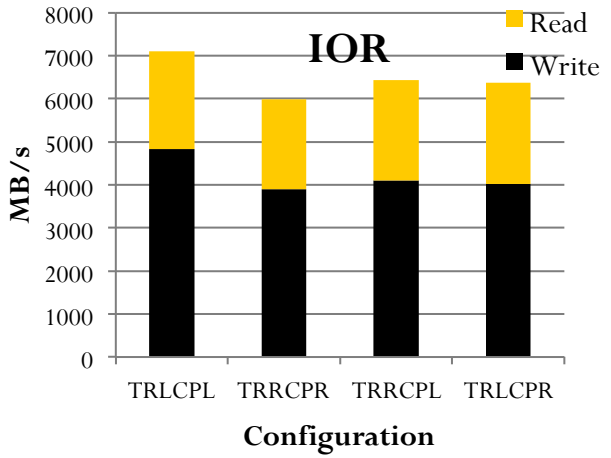
- fsmark is filesystem intensive:
  - Remote transfers result in 40% higher system time.
  - 130% increase in iowait time.
- psearchy is both filesystem and I/O intensive:
  - 57% increase in system time.
  - 70% increase in iowait time.





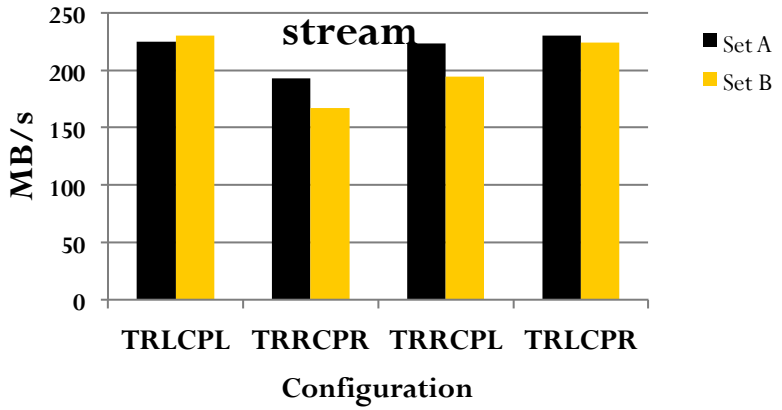
Results - IOR

- IOR is both read and write intensive benchmark.
- 15% decrease in read throughput due to remote transfers and memory copies.
- 20% decrease in write throughput due to remote transfers and copies.



Results - stream

- 24 SSDs are divided into two domains.
- Each set of 12 SSDs are connected to two controllers.
- Ideally, symmetric throughput is expected.
- Remote transfers result in a 27% drop in throughput of one of the sets.



Conclusions

- A mix of synthetic benchmarks and applications show the potential of NUMA affinity to hurt I/O throughput.
- Future systems will have increased heterogeneity, more domains and high bandwidths.
- Today, NUMA affinity is not a problem for cores within a single processor socket. Future processors with 100s of cores will have domains within a processor chip.
- The range of performance degradation is important. Different server configurations and runtime libraries result in throughput within a range.
- Partitioning of the system stacks based on sockets will become necessary.