中科院计算所
INSTITUTE OF COMPUTING TECHNOLOGY ,CAS

MSST 2020
2020.10.29~30

NUMA-Aware Thread Migration for High Performance NVMM File Systems

Ying Wang, Dejun Jiang, Jin Xiong

Institute of Computing Technology, CAS

University of Chinese Academy of Sciences

Outline

- Background & Motivation
- NThread design
  - Reduce remote access
  - Reduce resource contention
  - Increase CPU cache sharing
- Evaluation
- Summary

Background

- Non-Volatile Main Memories(NVMMs) provide **low latency**, **high bandwidth**, **byte-addressable** and **persistent storage**
  - PCM, MRAM, RRAM, 3D Xpoint[1]
    - Intel releases Optane DC Persistent Memory (Optane PMM)

[2]	R lat.	W lat.	R BW.	W BW.
DRAM	60ns	69ns	20 GB/s	~15 GB/s
Optane PMM	305ns	81ns	~6GB/s	~2GB/s
NVMe SSD	120us	30us	2GB/s	500MB/s
HDD	10ms	10ms	0.1GB/s	0.1GB/s

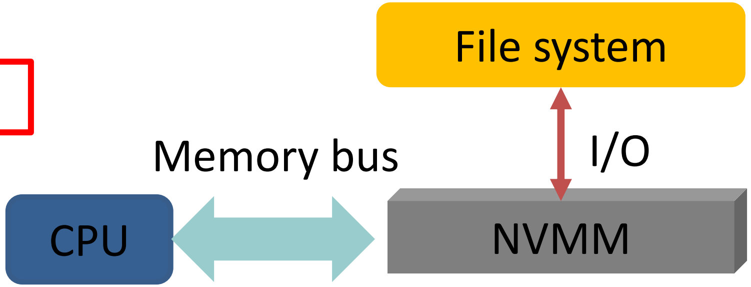
[1] What is Intel Optane DC Persistent Memory. Intel.

[2] The data from our evaluation and the paper of *"Basic Performance Measurements of the Intel Optane DC Persistent Memory Module"*

Background

- Non-Volatile Main Memories(NVMMs) provide **low latency**, **high bandwidth**, **byte-addressable** and **persistent storage**
  - PCM, MRAM, RRAM, 3D Xpoint[1]
    - Intel releases Optane DC Persistent Memory (Optane PMM)
- File system can be directly built on memory
  - Improve file system I/O performance

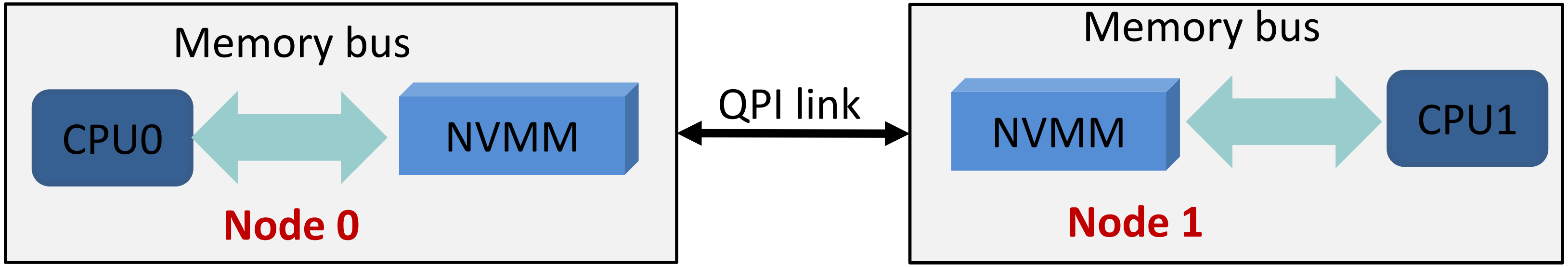
[2]	R lat.	W lat.	R BW.	W BW.
DRAM	60ns	69ns	20 GB/s	~15 GB/s
Optane PMM	305ns	81ns	~6GB/s	~2GB/s
NVMe SSD	120us	30us	2GB/s	500MB/s
HDD	10ms	10ms	0.1GB/s	0.1GB/s



[1] What is Intel Optane DC Persistent Memory. Intel.
[2] The data from our evaluation and the paper of *"Basic Performance Measurements of the Intel Optane DC Persistent Memory Module"*

Background

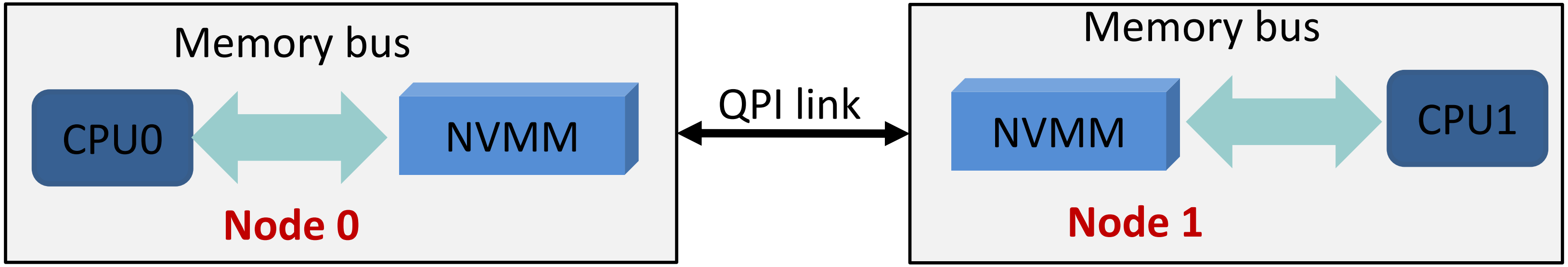
- Non-Uniform Memory Access architecture(NUMA) is widely used in data center [1,2,3,4,5,6,7]



[1] Lepers, ATC'2015 [2] Dashti, ASPLOS'2013 [3] Blagodurov, ATC'2011 [4] Tam, EuroSys'2007
[5] Yu, CS'2017 [6] Calciu, ASPLOS'2017 [7] Blagodurov, ACM Trans'2010

Background

- Non-Uniform Memory Access architecture(NUMA) is widely used in data center [1,2,3,4,5,6,7]
    - Multiple NUMA (memory) nodes

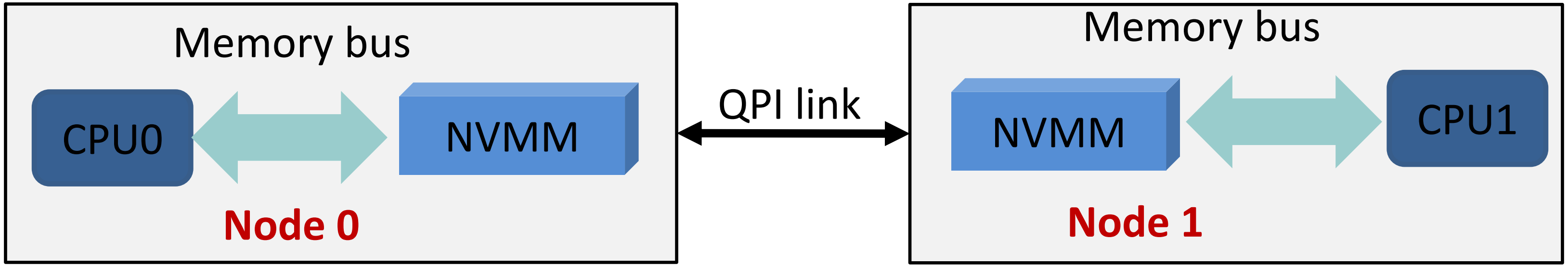


[1] Lepers, ATC'2015 [2] Dashti, ASPLOS'2013 [3] Blagodurov, ATC'2011 [4] Tam, EuroSys'2007

[5] Yu, CS'2017 [6] Calciu, ASPLOS'2017 [7] Blagodurov, ACM Trans'2010

Background

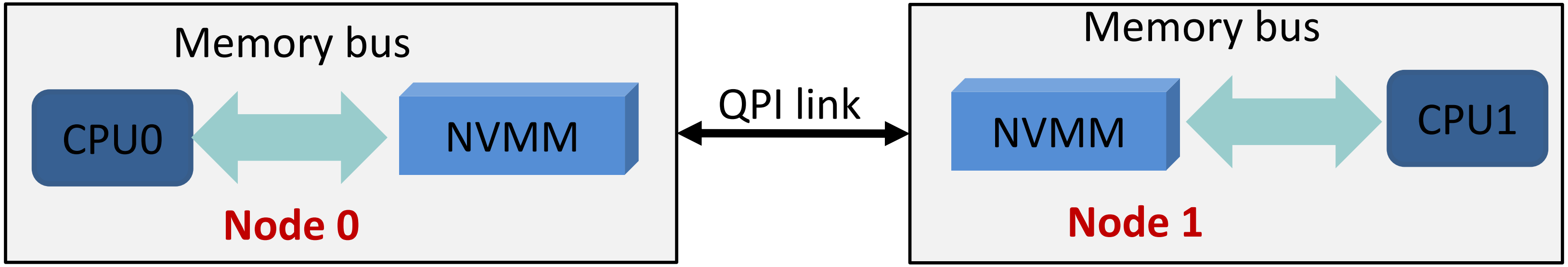
- Non-Uniform Memory Access architecture(NUMA) is widely used in data center [1,2,3,4,5,6,7]
  - Multiple NUMA (memory) nodes
    - Each memory node contains independent CPU and memory



[1] Lepers, ATC'2015 [2] Dashti, ASPLOS'2013 [3] Blagodurov, ATC'2011 [4] Tam, EuroSys'2007
[5] Yu, CS'2017 [6] Calciu, ASPLOS'2017 [7] Blagodurov, ACM Trans'2010

Background

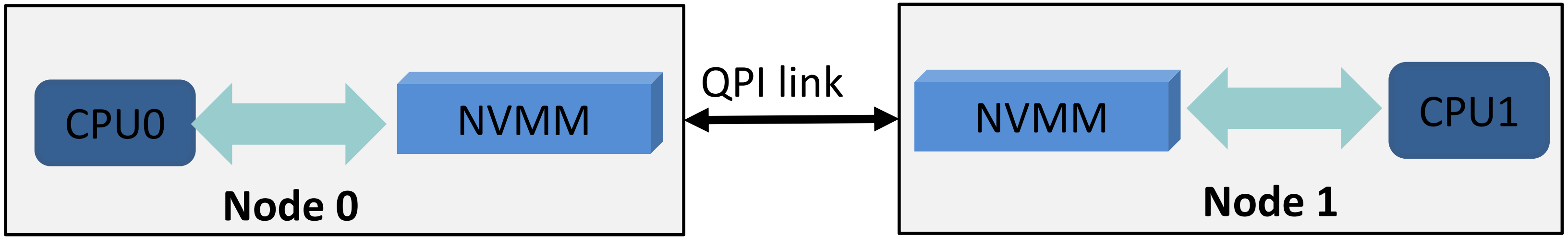
- Non-Uniform Memory Access architecture(NUMA) is widely used in data center [1,2,3,4,5,6,7]
  - Multiple NUMA (memory) nodes
    - Each memory node contains independent CPU and memory
    - Each node can run in parallel without interference



[1] Lepers, ATC'2015 [2] Dashti, ASPLOS'2013 [3] Blagodurov, ATC'2011 [4] Tam, EuroSys'2007
[5] Yu, CS'2017 [6] Calciu, ASPLOS'2017 [7] Blagodurov, ACM Trans'2010

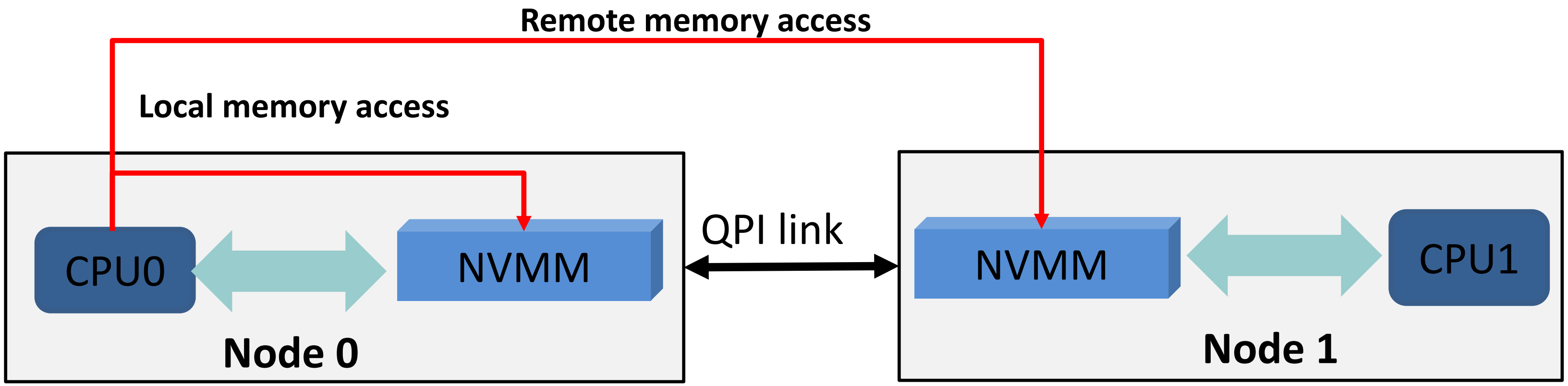
Background

- NUMA architecture



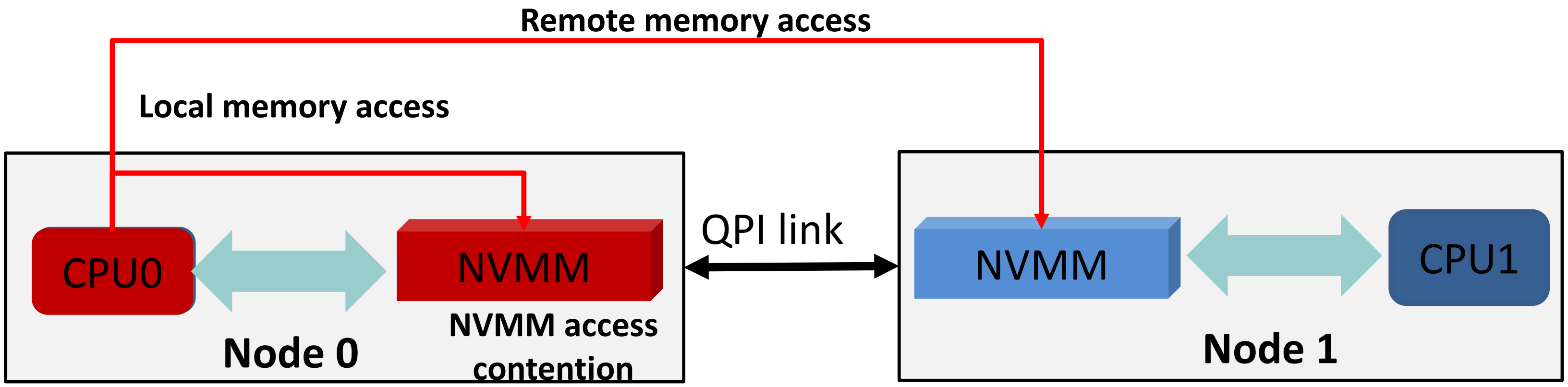
Background

- NUMA architecture
  - Remote nodes is accessed in more time than local ones



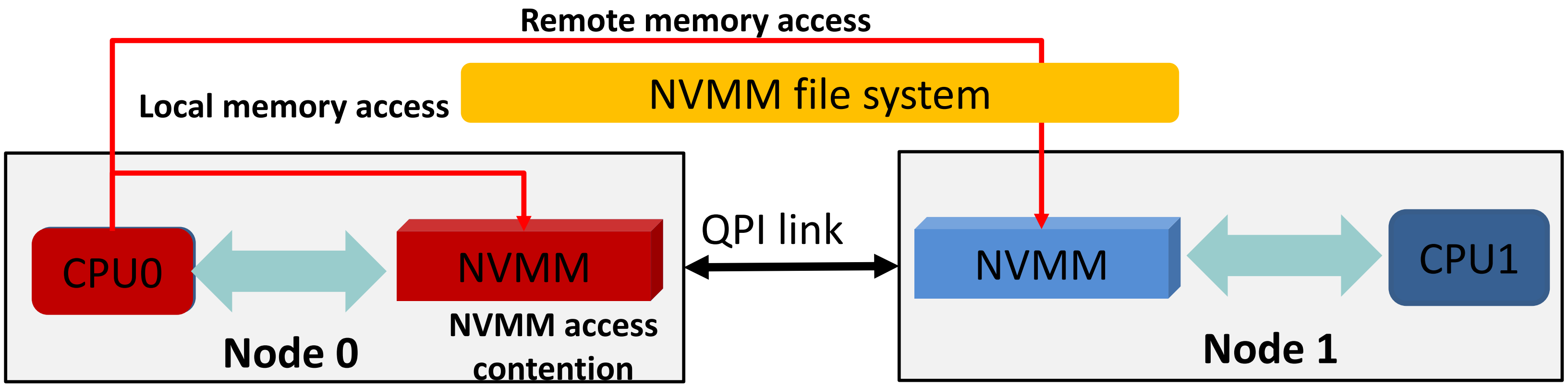
Background

- NUMA architecture
  - Remote nodes is accessed in more time than local ones
  - Unbalanced resource usage among nodes causes resource contention, reducing system performance
    - Memory (DRAM, NVMM), CPU



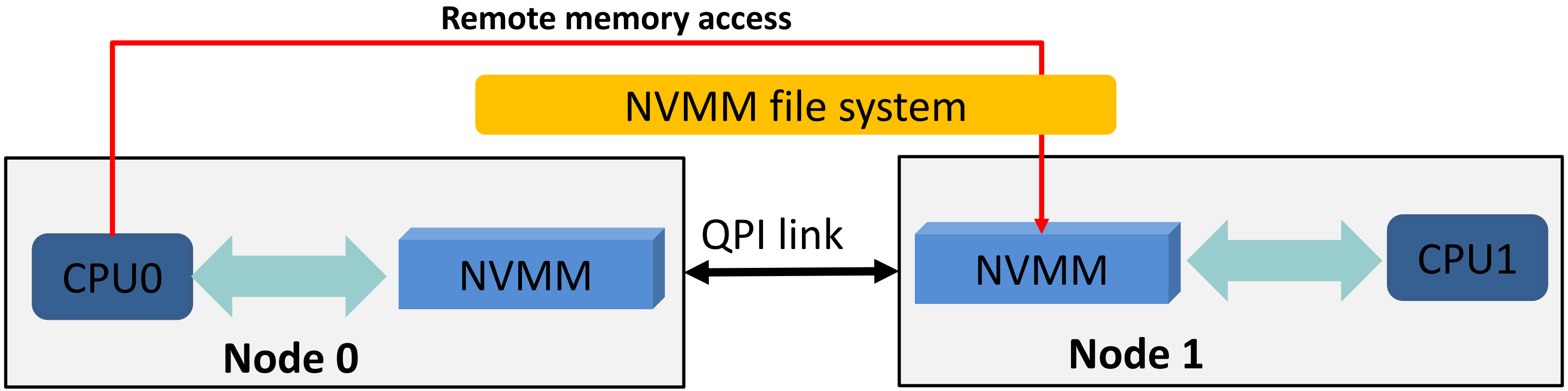
Background

- NUMA architecture
  - Remote nodes is accessed in more time than local ones
  - Unbalanced resource usage among nodes causes resource contention, reducing system performance
    - Memory (DRAM, NVMM), CPU
- The I/O performance of NVMM file system is affected by the these factors



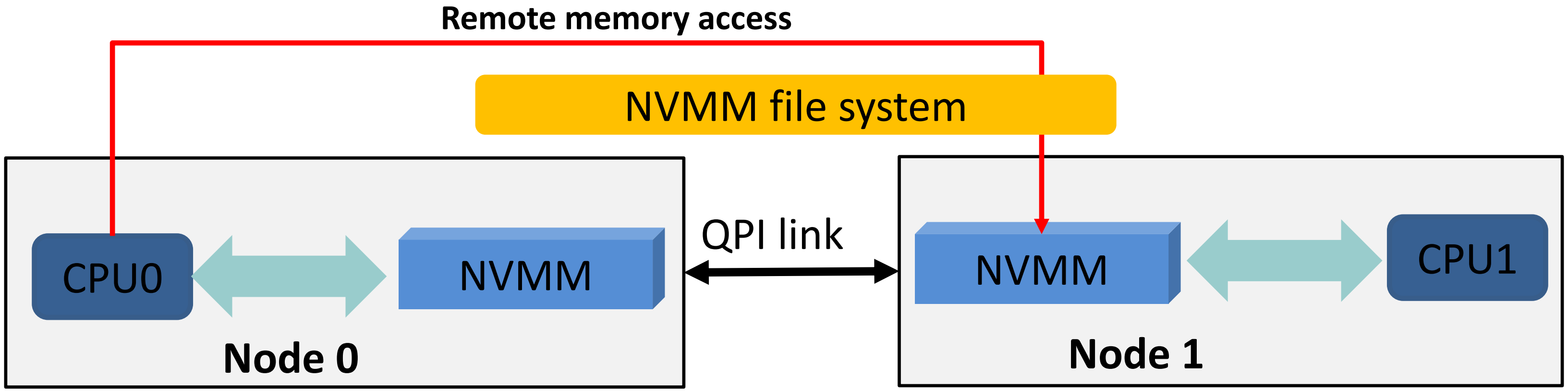
Motivation

- Existing NVMM file systems are not aware of NUMA
  - Remote memory access



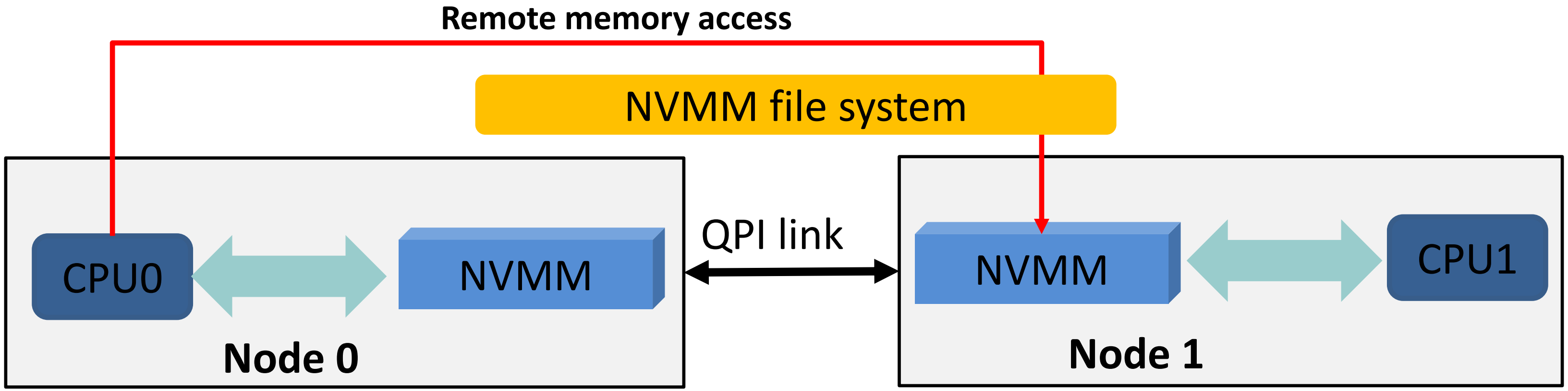
Motivation

- Existing NVMM file systems are not aware of NUMA
  - Remote memory access
    - File location is transparent to thread



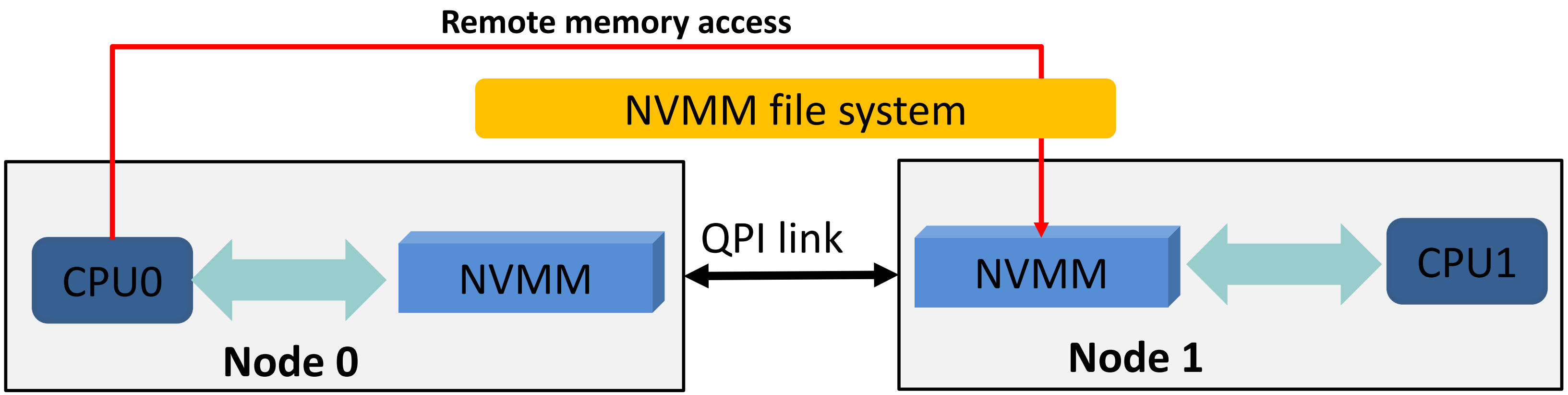
Motivation

- Existing NVMM file systems are not aware of NUMA
  - Remote memory access
    - File location is transparent to thread
    - Thread is randomly scheduled by OS



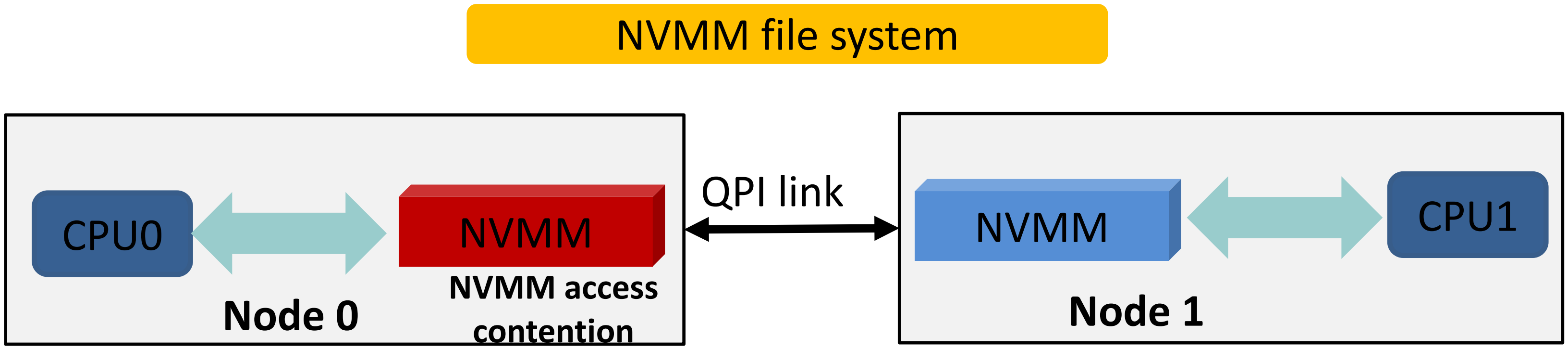
Motivation

- Existing NVMM file systems are not aware of NUMA
  - Remote memory access
    - File location is transparent to thread
    - Thread is randomly scheduled by OS
    - Remote NVMM accesses increase the read latency of NVMM file system by 65.6%



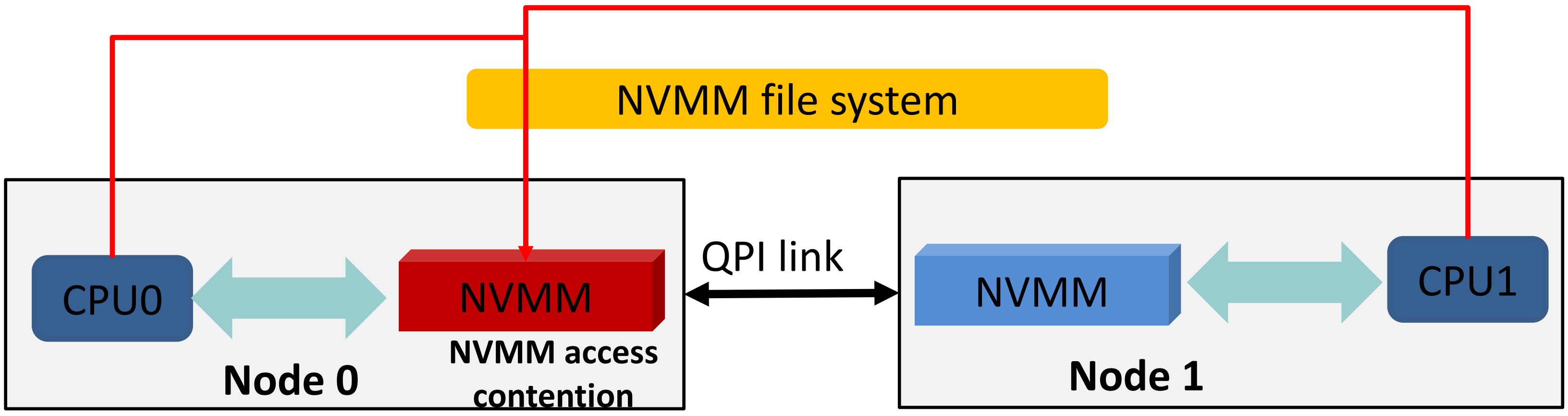
Motivation

- Existing NVMM file system is not aware of NUMA
  - Resource contention



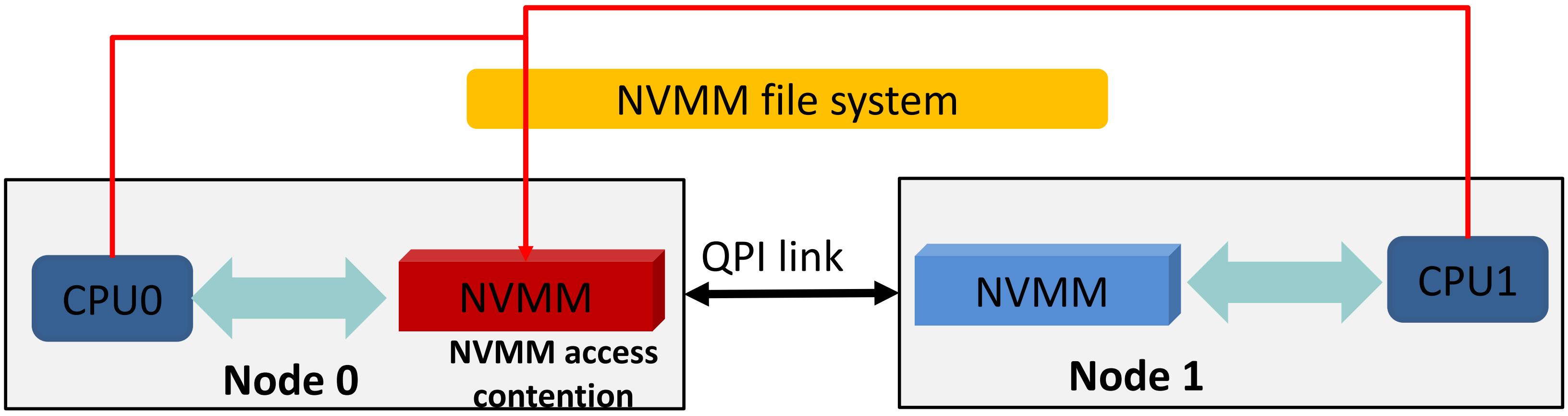
Motivation

- Existing NVMM file system is not aware of NUMA
  - Resource contention
    - Random placement of data leads to unbalanced data access among NUMA nodes



Motivation

- Existing NVMM file system is not aware of NUMA
  - Resource contention
    - Random placement of data leads to unbalanced data access among NUMA nodes
    - NVMM access contention can increases file access latency by 120.5%



Existing works

- For memory applications

[1]Matthias, SBAC-PAD’14 [2] Lachaize, ATC’12 [3] Wu, Cluster’19
[4] Xu, ASPLOS’19

Existing works

- For memory applications
  - Allocating memory on the memory node where the thread runs
    - Cannot solve the problem of NVMM contention

[1]Matthias, SBAC-PAD'14 [2] Lachaize, ATC'12 [3] Wu, Cluster'19
[4] Xu, ASPLOS'19

Existing works

- For memory applications
  - Allocating memory on the memory node where the thread runs
    - Cannot solve the problem of NVMM contention
  - Migrating thread and thread data (such as stack, heap) [1,2,34]
    - Reduce remote access
    - Reduce resource contention by unbalanced use of resources
    - A lot of data migration overhead

[1]Matthias, SBAC-PAD'14 [2] Lachaize, ATC'12 [3] Wu, Cluster'19
[4] Xu, ASPLOS'19

High data migration overhead on NVMM FS

High data migration overhead on NVMM FS

- NVMM has long latency and low bandwidth than DRAM
  - The migrating latency of 16 KB data in NVMM is 2.8X of DRAM

High data migration overhead on NVMM FS

- NVMM has long latency and low bandwidth than DRAM
  - The migrating latency of 16 KB data in NVMM is 2.8X of DRAM
- File system needs consistency
  - Additional overhead, such as log or journal

High data migration overhead on NVMM FS

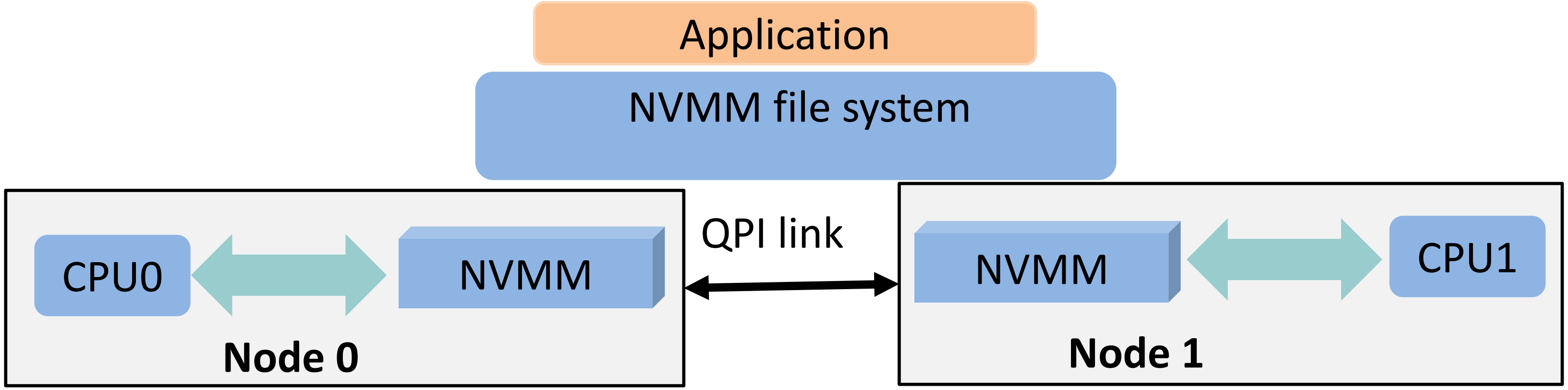
- NVMM has long latency and low bandwidth than DRAM
  - The migrating latency of 16 KB data in NVMM is 2.8X of DRAM
- File system needs consistency
  - Additional overhead, such as log or journal
- File data is shared between threads
  - Difficult to decide the node to migrate data

High data migration overhead on NVMM FS

- NVMM has long latency and low bandwidth than DRAM
  - The migrating latency of 16 KB data in NVMM is 2.8X of DRAM
- File system needs consistency
  - Additional overhead, such as log or journal
- File data is shared between threads
  - Difficult to decide the node to migrate data
- NVMM has low write endurance
  - Reduce the lifetime of NVMM

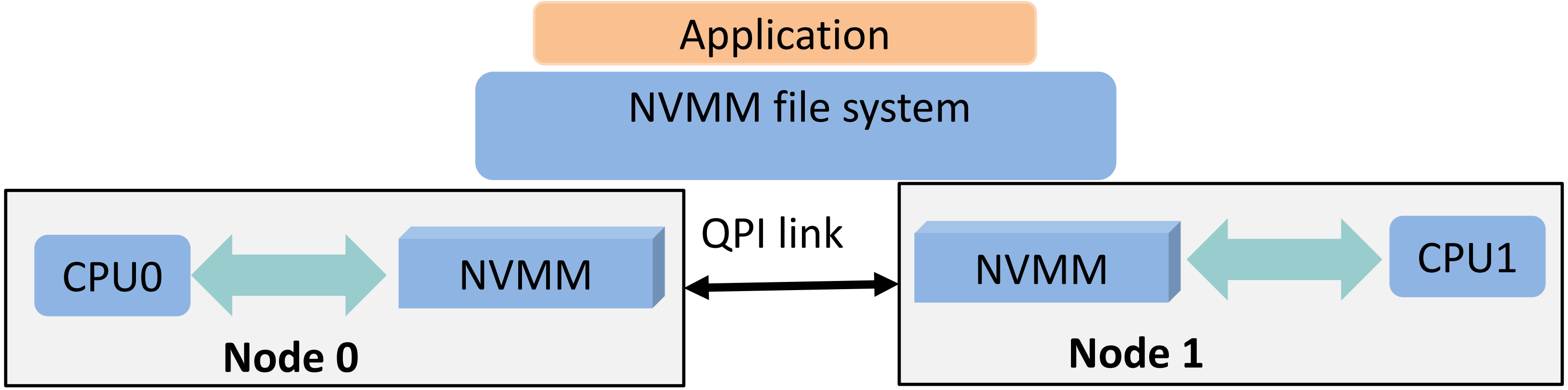
Contribution

- A NUMA-Aware thread migration for NVMM FS



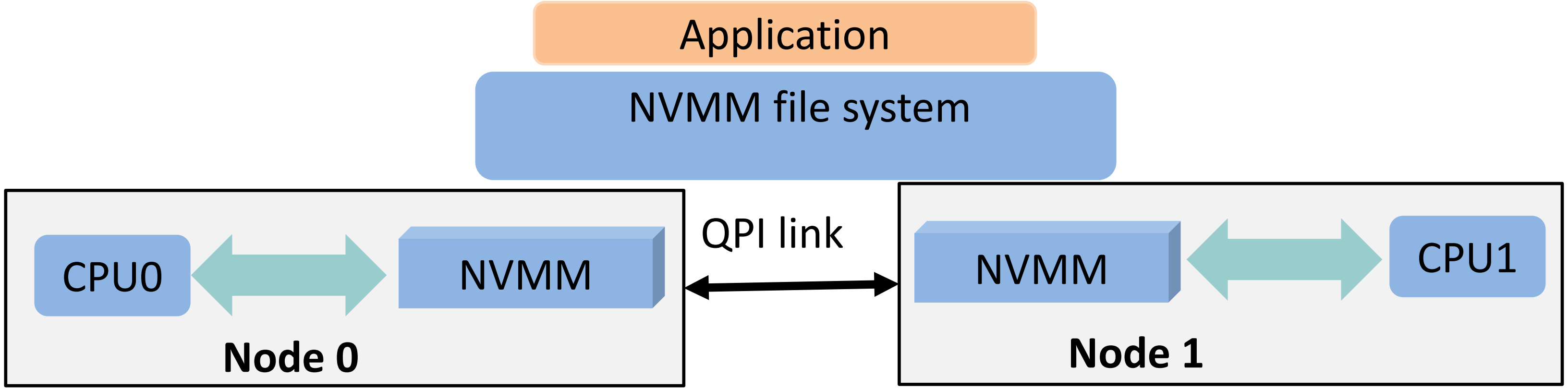
Contribution

- A NUMA-Aware thread migration for NVMM FS
  - Reduce remote access



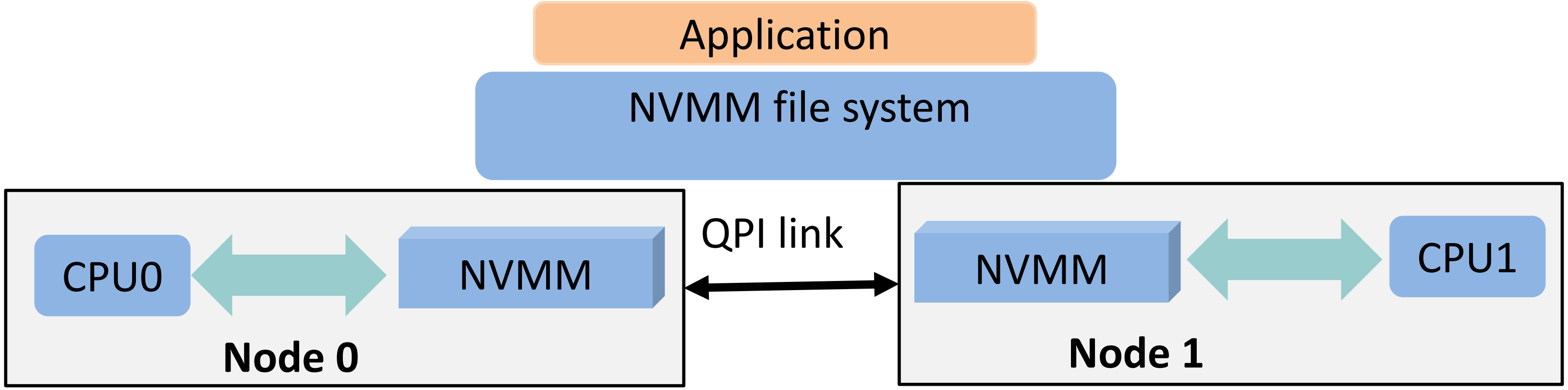
Contribution

- A NUMA-Aware thread migration for NVMM FS
  - Reduce remote access
  - Reduce resource contention
    - CPU
    - NVMM



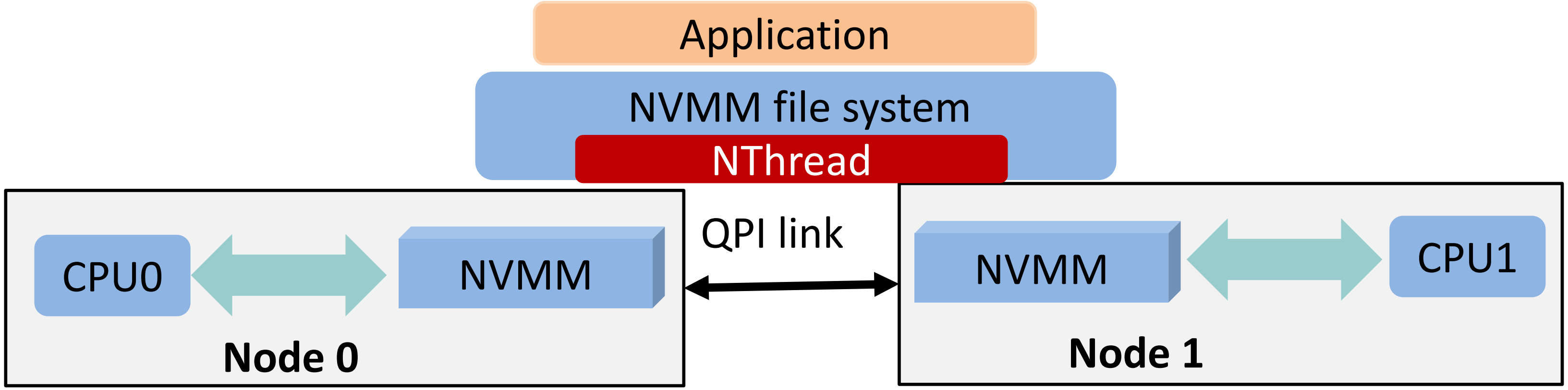
Contribution

- A NUMA-Aware thread migration for NVMM FS
  - Reduce remote access
  - Reduce resource contention
    - CPU
    - NVMM
  - Increase CPU cache sharing between threads



Contribution

- A NUMA-Aware thread migration for NVMM FS
  - Reduce remote access
  - Reduce resource contention
    - CPU
    - NVMM
  - Increase CPU cache sharing between threads
  - Transparent to application



Outline

- Background & Motivation
- NThread design
  - Reduce remote access
  - Reduce resource contention
  - Increase CPU cache sharing
- Evaluation
- Summary

Reduce remote access

- How to reduce remote access

- How to avoid ping-pong migration

Reduce remote access

- How to reduce remote access
  - Write
    - allocate new space to perform write operations
    - Write data on the node where the thread running

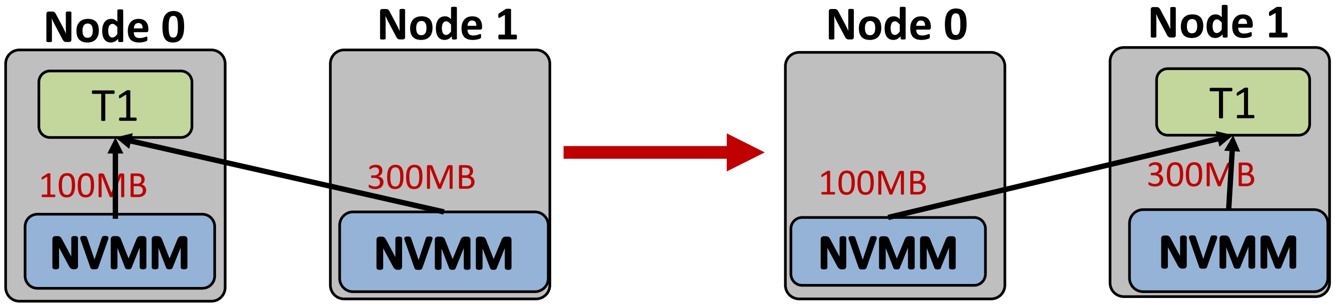
- How to avoid ping-pong migration

Reduce remote access

- How to reduce remote access
  - Write
    - allocate new space to perform write operations
    - Write data on the node where the thread running
  - Read
    - Count the read amount of each node for each thread
    - Migrate threads to the node with the most data read
- How to avoid ping-pong migration

Reduce remote access

- How to reduce remote access
  - Write
    - allocate new space to perform write operations
    - Write data on the node where the thread running
  - Read
    - Count the read amount of each node for each thread
    - Migrate threads to the node with the most data read
- How to avoid ping-pong migration
    - When the read size of a thread on one node is higher than all other nodes by a value per period (such as 200 MB per second)

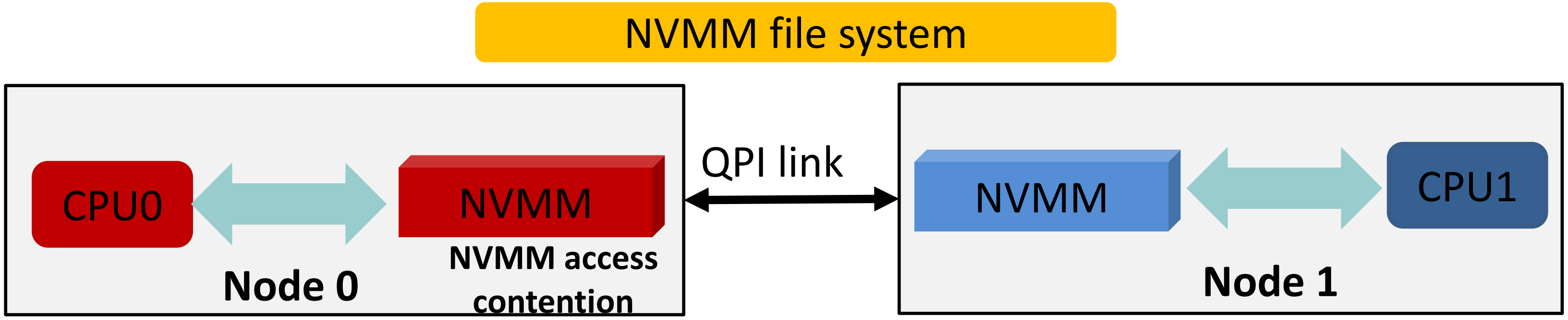


Outline

- Background & Motivation
- NThread design
  - Reduce remote access
  - Reduce resource contention
  - Increase CPU cache sharing
- Evaluation
- Summary

Reduce resource contention

- Problems
  - How to find contention
  - How to reduce contention
  - How to avoid new contention



Reduce NVMM contention

- How to find contention

Reduce NVMM contention

- How to find contention
  - The access amount of NVMM in one node exceeds a threshold that the use of other nodes is less than ½ of the node

Reduce NVMM contention

- How to find contention
  - The access amount of NVMM in one node exceeds a threshold that the use of other nodes is less than ½ of the node
  - How to define access amount

Reduce NVMM contention

- How to find contention
  - The access amount of NVMM in one node exceeds a threshold that the use of other nodes is less than ½ of the node
  - How to define access amount
    - Bandwidth !!!!
      - Considering the theoretical bandwidth with running bandwidth of NVMM
      - Bandwidth = read bandwidth + write bandwidth

Reduce NVMM contention

- How to find contention
  - The access amount of NVMM in one node exceeds a threshold that the use of other nodes is less than ½ of the node
  - How to define access amount
    - Bandwidth !!!!
      - Considering the theoretical bandwidth with running bandwidth of NVMM
      - Bandwidth = read bandwidth + write bandwidth
    - However
      - The write bandwidth of NVMM is about 1/3 of the read bandwidth

Reduce NVMM contention

- How to find contention
  - Bandwidth

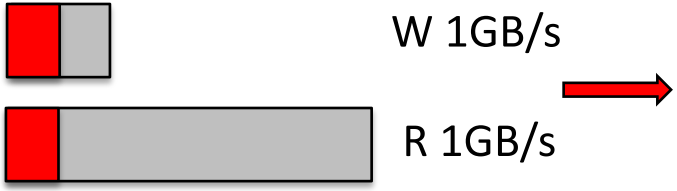
Reduce NVMM contention

- How to find contention
  - Bandwidth
    - It is inaccurate to calculate NVMM access by using the sum of read and write bandwidth

Reduce NVMM contention

- How to find contention
  - Bandwidth
    - It is inaccurate to calculate NVMM access by using the sum of read and write bandwidth
      - Read 1 GB/s + Write 1 GB/s = 2GB/s → low contention

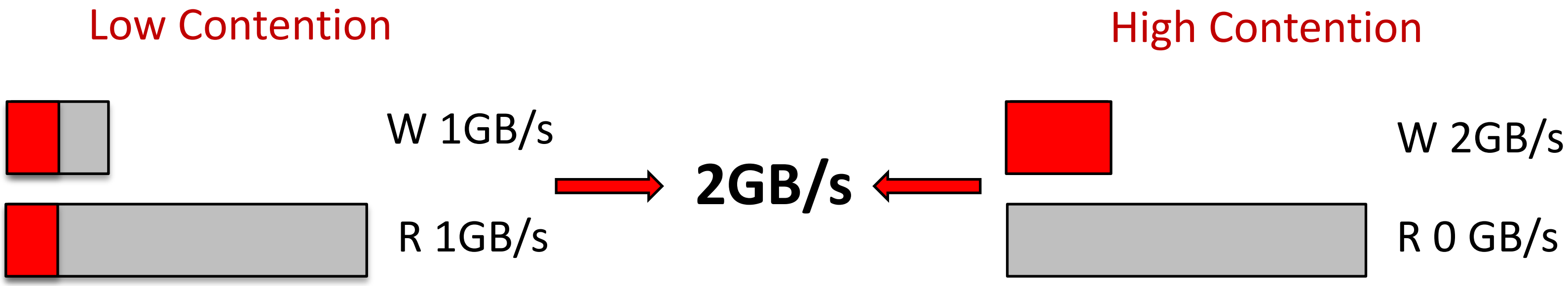
Low Contention



2GB/s

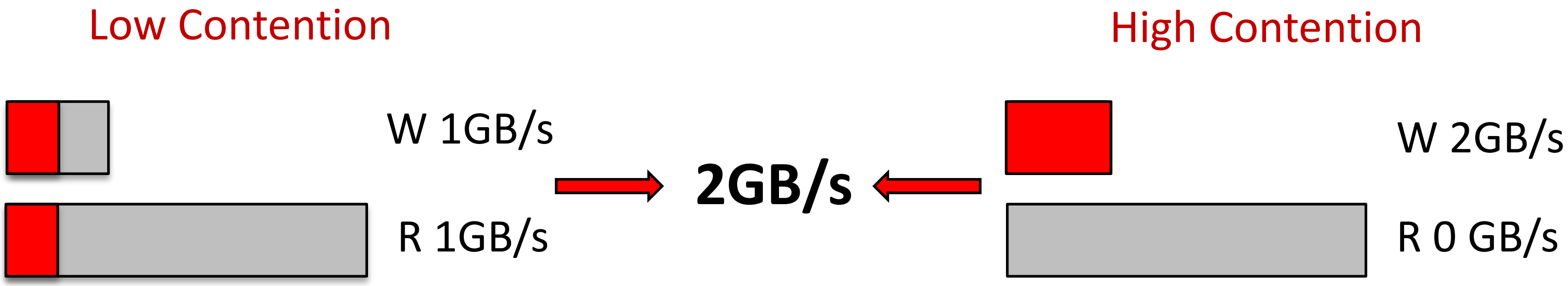
Reduce NVMM contention

- How to find contention
  - Bandwidth
    - It is inaccurate to calculate NVMM access by using the sum of read and write bandwidth
      - Read 1 GB/s + Write 1 GB/s = 2GB/s → low contention
      - Read 0 GB/s + write 2 GB/s = 2GB/s → high contention



Reduce NVMM contention

- How to find contention
  - Bandwidth
    - It is inaccurate to calculate NVMM access by using the sum of read and write bandwidth
      - Read 1 GB/s + Write 1 GB/s = 2GB/s → low contention
      - Read 0 GB/s + write 2 GB/s = 2GB/s → high contention
    - Solution
      - Change the read and write weight of bandwidth
        - » $BW_N = NWr_N * 1/3 + BWw_N$ (Refer to paper)



Low Contention

High Contention

W 1GB/s

R 1GB/s

2GB/s

W 2GB/s

R 0 GB/s

Reduce NVMM contention

- How to reduce contention

Reduce NVMM contention

- How to reduce contention
  - The access contention come from read and write

Reduce NVMM contention

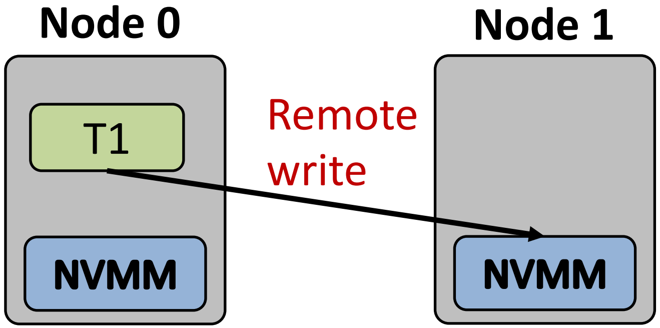
- How to reduce contention
  - The access contention come from read and write
    - Read
      - data location is fixed

Reduce NVMM contention

- How to reduce contention
  - The access contention come from read and write
    - Read
      - data location is fixed
    - Write
      - Specify the node where data is written

Reduce NVMM contention

- How to reduce contention
  - The access contention come from read and write
    - Read
      - data location is fixed
    - Write
      - Specify the node where data is written
      - Long remote write latency: reduce performance by 65.5%



Reduce NVMM contention

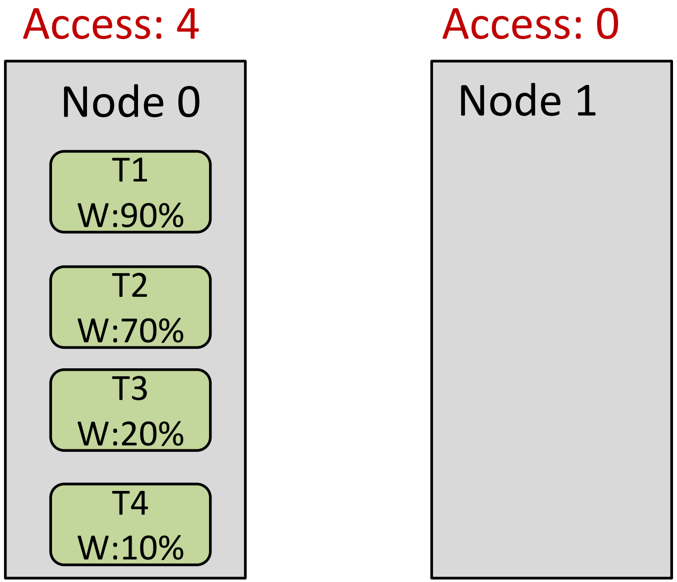
- How to reduce contention

Reduce NVMM contention

- How to reduce contention
  - Migrating threads with high write rate to the nodes with low access pressure
    - Reduce remote write
    - Reduce NVMM contention

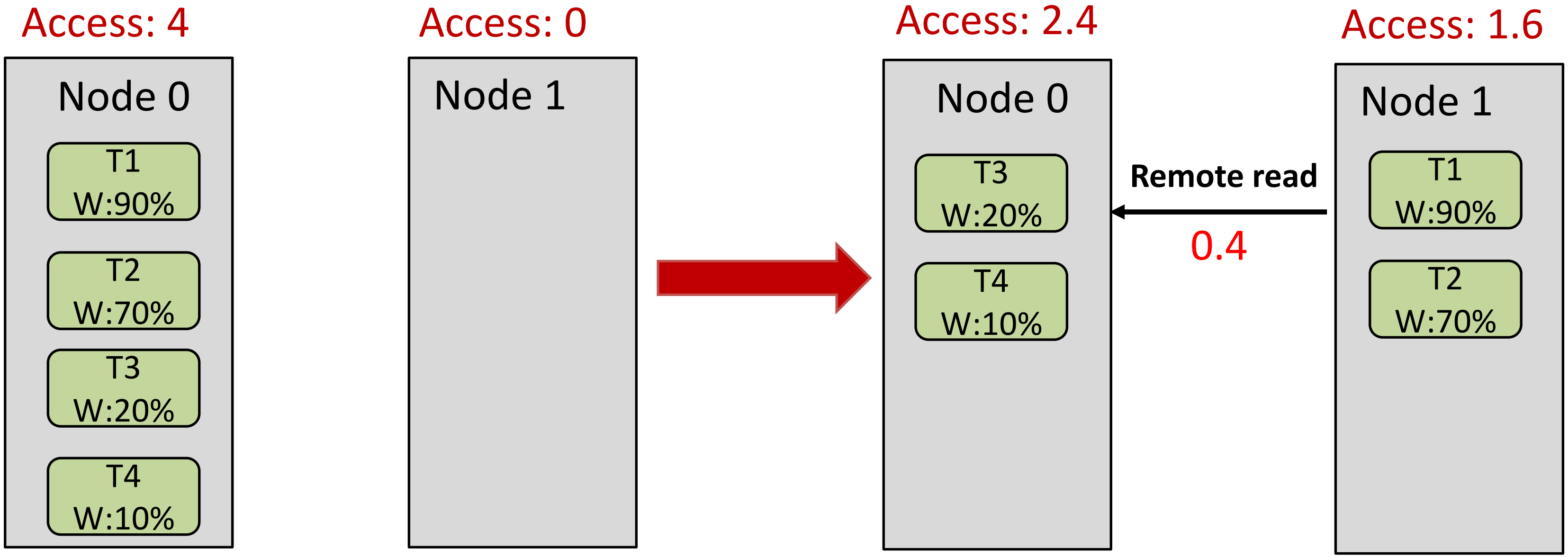
Reduce NVMM contention

- How to reduce contention
  - Migrating threads with high write rate to the nodes with low access pressure
    - Reduce remote write
    - Reduce NVMM contention



Reduce NVMM contention

- How to reduce contention
  - Migrating threads with high write rate to the nodes with low access pressure
    - Reduce remote write
    - Reduce NVMM contention



Reduce NVMM contention

- How to avoid new contention

Reduce NVMM contention

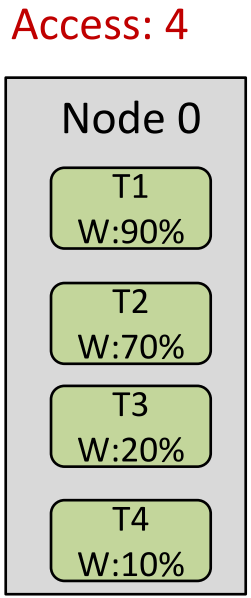
- How to avoid new contention
  - Migrate too much threads to low contention nodes

Reduce NVMM contention

- How to avoid new contention
  - Migrate too much threads to low contention nodes
  - Determine the number of threads to migrate according to the current bandwidth of each node

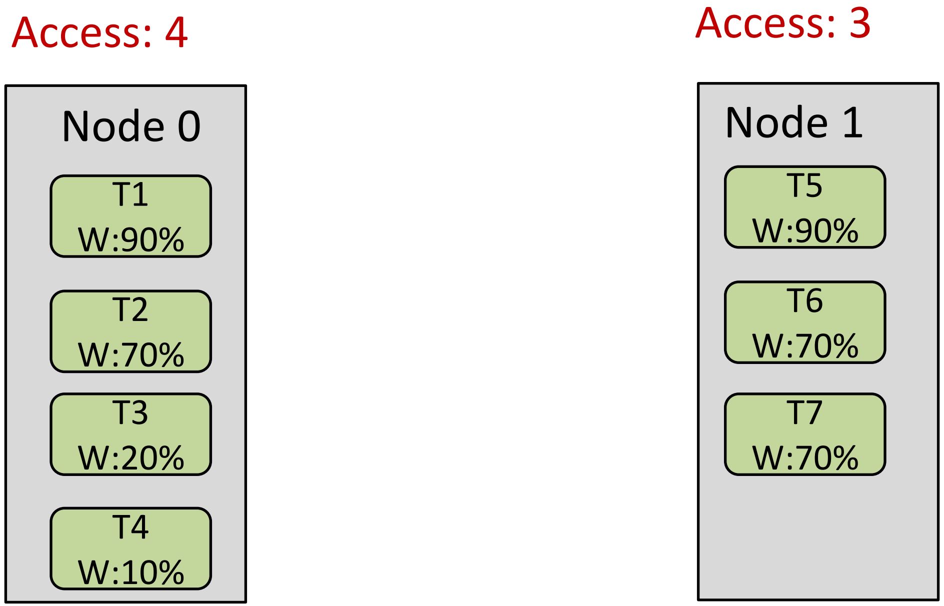
Reduce NVMM contention

- How to avoid new contention
  - Migrate too much threads to low contention nodes
  - Determine the number of threads to migrate according to the current bandwidth of each node



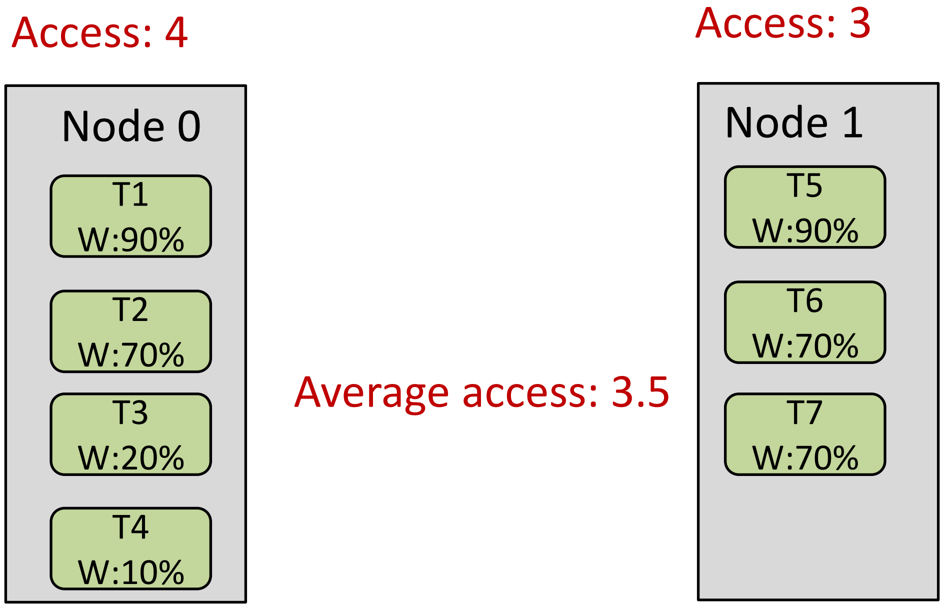
Reduce NVMM contention

- How to avoid new contention
  - Migrate too much threads to low contention nodes
  - Determine the number of threads to migrate according to the current bandwidth of each node



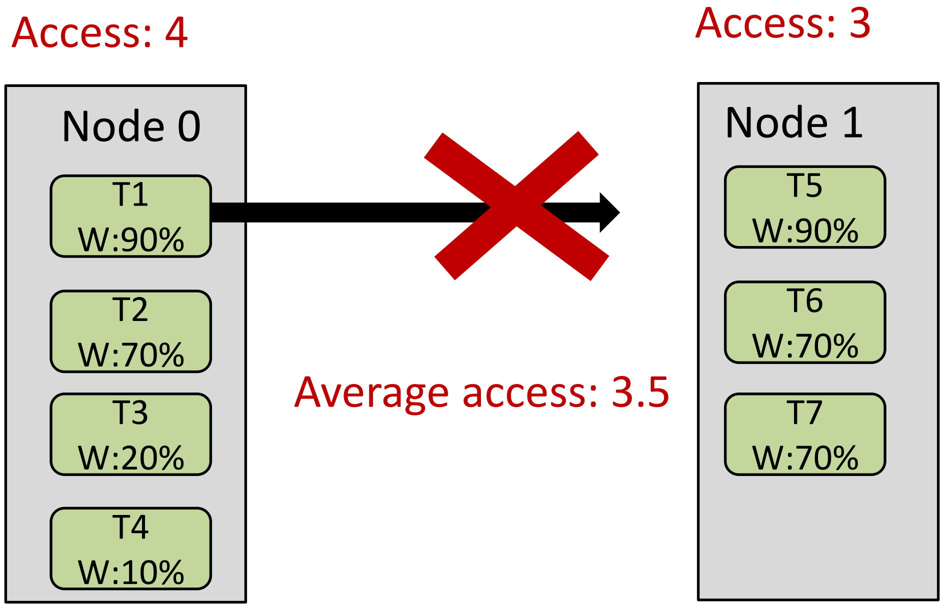
Reduce NVMM contention

- How to avoid new contention
  - Migrate too much threads to low contention nodes
  - Determine the number of threads to migrate according to the current bandwidth of each node



Reduce NVMM contention

- How to avoid new contention
  - Migrate too much threads to low contention nodes
  - Determine the number of threads to migrate according to the current bandwidth of each node



Reduce CPU contention

- How to find contention

Reduce CPU contention

- How to find contention
  - When the CPU utilization of a node exceeds 90% and is 2x of other nodes

Reduce CPU contention

- How to find contention
  - When the CPU utilization of a node exceeds 90% and is 2x of other nodes
- How to reduce contention
  - Migrating threads from NUMA node with high CPU utilization to other low CPU utilization node

Reduce CPU contention

- How to find contention
  - When the CPU utilization of a node exceeds 90% and is 2x of other nodes
- How to reduce contention
  - Migrating threads from NUMA node with high CPU utilization to other low CPU utilization node
- How to avoid new contention
  - If the CPU utilization of migrate thread and target NUMA node does not exceed 90%, migrating thread

Outline

- Background & Motivation
- NThread design
  - Reduce remote access
  - Reduce resource contention
  - Increase CPU cache sharing
- Evaluation
- Summary

Increase CPU cache sharing

Increase CPU cache sharing

- How to find threads that share data

Increase CPU cache sharing

- How to find threads that share data
  - Once a file accessed by multiple threads, all threads accessing the file share data

Increase CPU cache sharing

- How to find threads that share data
  - Once a file accessed by multiple threads, all threads accessing the file share data
- How to increase CPU cache sharing

Increase CPU cache sharing

- How to find threads that share data
  - Once a file accessed by multiple threads, all threads accessing the file share data
- How to increase CPU cache sharing
  - Reducing remote memory access

Composing Optimizations together

- Remote access, resource contention and CPU cache sharing

Composing Optimizations together

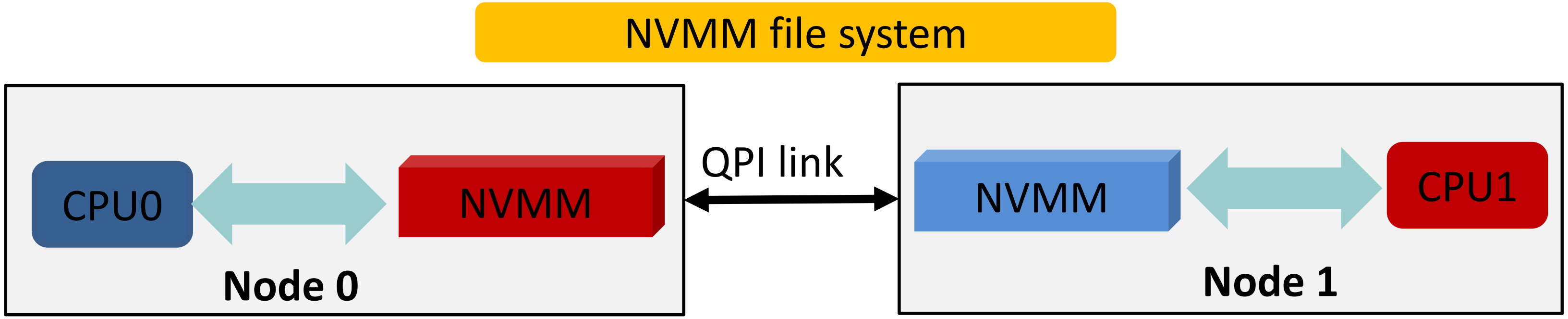
- Remote access, resource contention and CPU cache sharing
  - Reduce remote access can increase CPU cache sharing
    - Threads accessing the same data run in the same node, sharing CPU cache

Composing Optimizations together

- Remote access, resource contention and CPU cache sharing
  - Reduce remote access can increase CPU cache sharing
    - Threads accessing the same data run in the same node, sharing CPU cache
  - Reduce resource contention may increase remote memory access and destroy CPU cache sharing

Composing Optimizations together

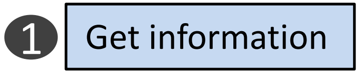
- Remote access, resource contention and CPU cache sharing
  - Reduce remote access can increase CPU cache sharing
    - Threads accessing the same data run in the same node, sharing CPU cache
  - Reduce resource contention may increase remote memory access and destroy CPU cache sharing
  - Reduce NVMM contention may increase CPU contention



Composing Optimizations together

- What-if analysis

Composing Optimizations together

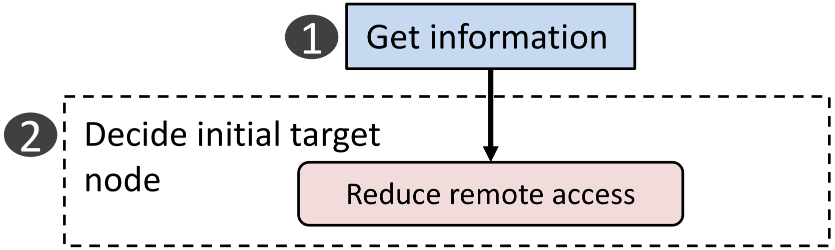


❶ Get information

- What-if analysis
  - Get information each second
    - Data access size, NVMM bandwidth, CPU utilization and data sharing

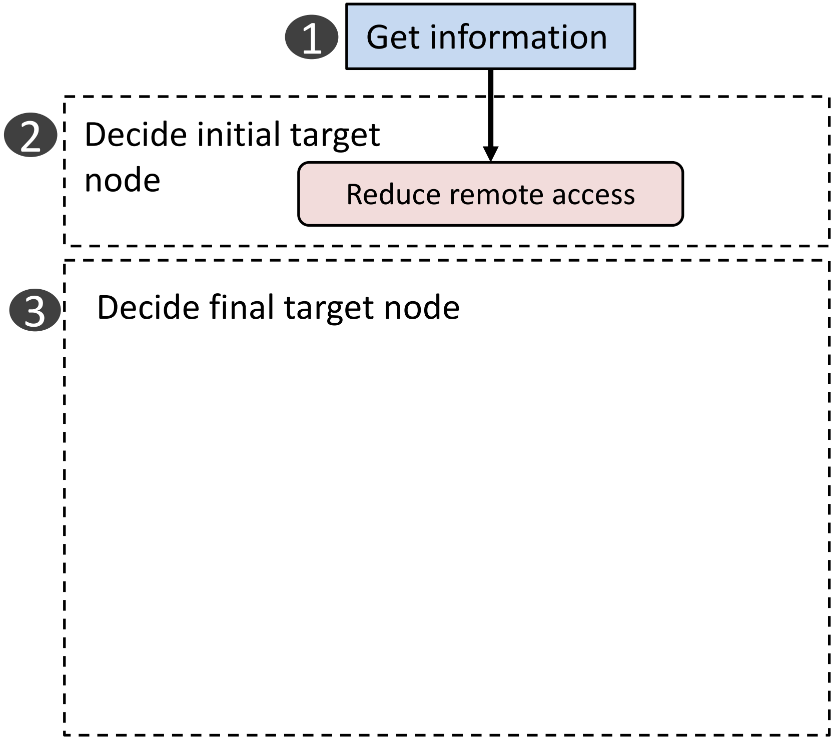
Composing Optimizations together

- What-if analysis
  - Get information each second
    - Data access size, NVMM bandwidth, CPU utilization and data sharing
  - Decide initial target node
    - Reduce remote memory access



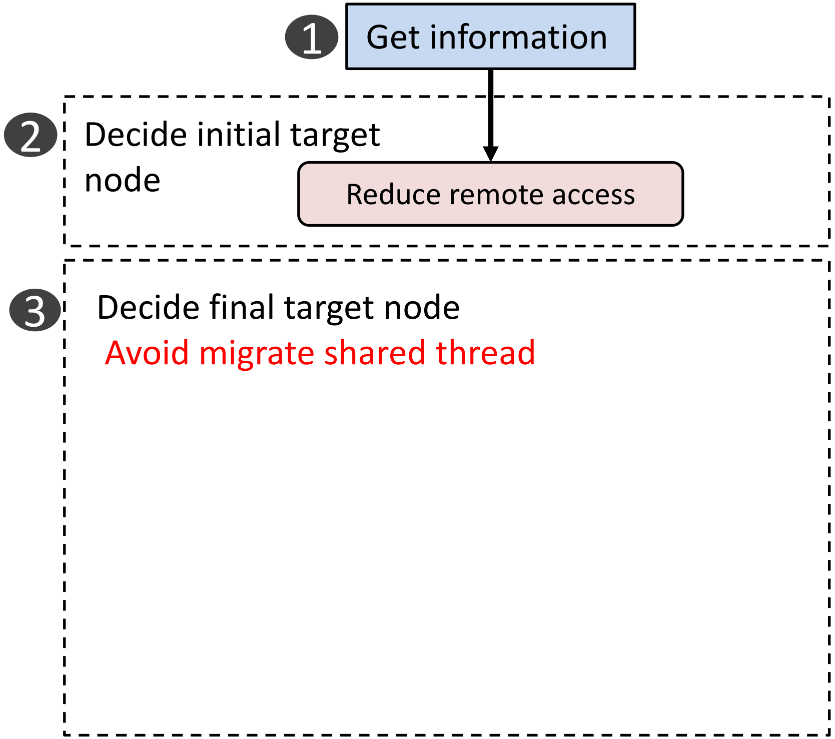
Composing Optimizations together

- What-if analysis
  - Get information each second
    - Data access size, NVMM bandwidth, CPU utilization and data sharing
  - Decide initial target node
    - Reduce remote memory access
  - Decide final target node
    - Reduce NVMM and CPU contention



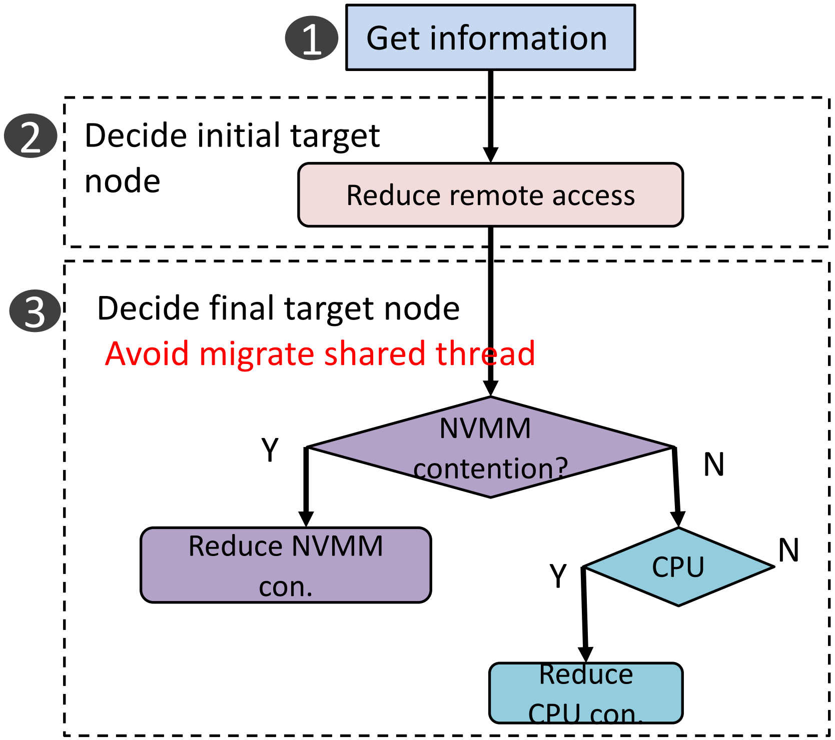
Composing Optimizations together

- What-if analysis
  - Get information each second
    - Data access size, NVMM bandwidth, CPU utilization and data sharing
  - Decide initial target node
    - Reduce remote memory access
  - Decide final target node
    - Reduce NVMM and CPU contention
      - Avoid migrate shared thread



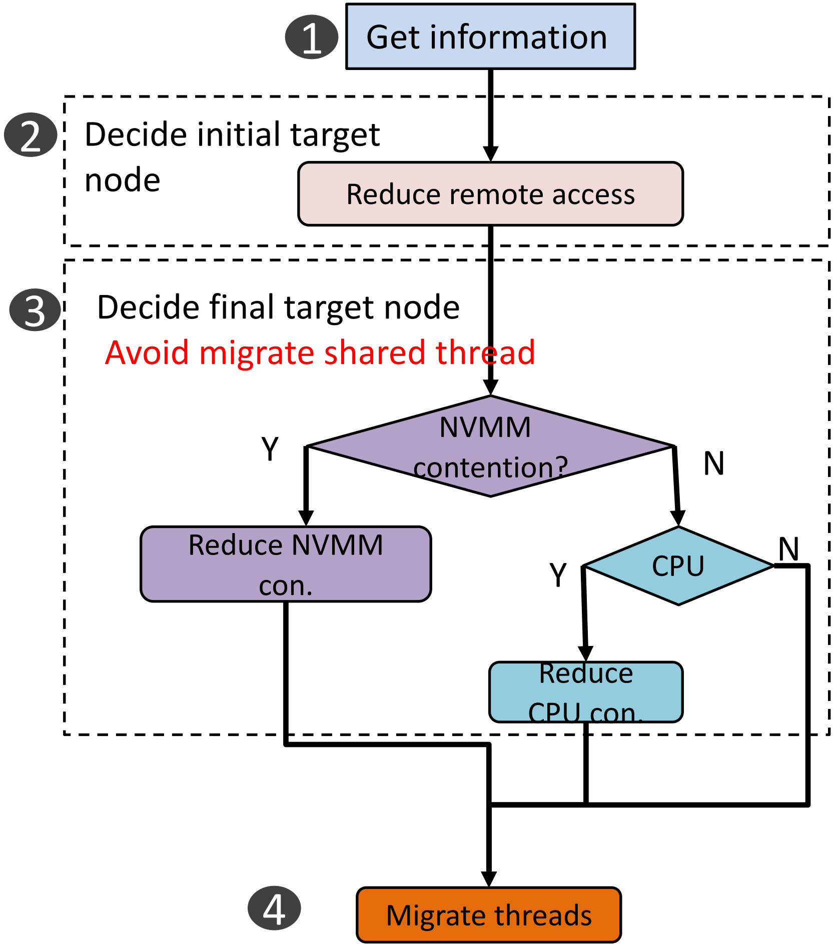
Composing Optimizations together

- What-if analysis
  - Get information each second
    - Data access size, NVMM bandwidth, CPU utilization and data sharing
  - Decide initial target node
    - Reduce remote memory access
  - Decide final target node
    - Reduce NVMM and CPU contention
      - Avoid migrate shared thread
      - NVMM > CPU (Refer to paper)



Composing Optimizations together

- What-if analysis
  - Get information each second
    - Data access size, NVMM bandwidth, CPU utilization and data sharing
  - Decide initial target node
    - Reduce remote memory access
  - Decide final target node
    - Reduce NVMM and CPU contention
      - Avoid migrate shared thread
      - NVMM > CPU (Refer to paper)
  - Migrate threads



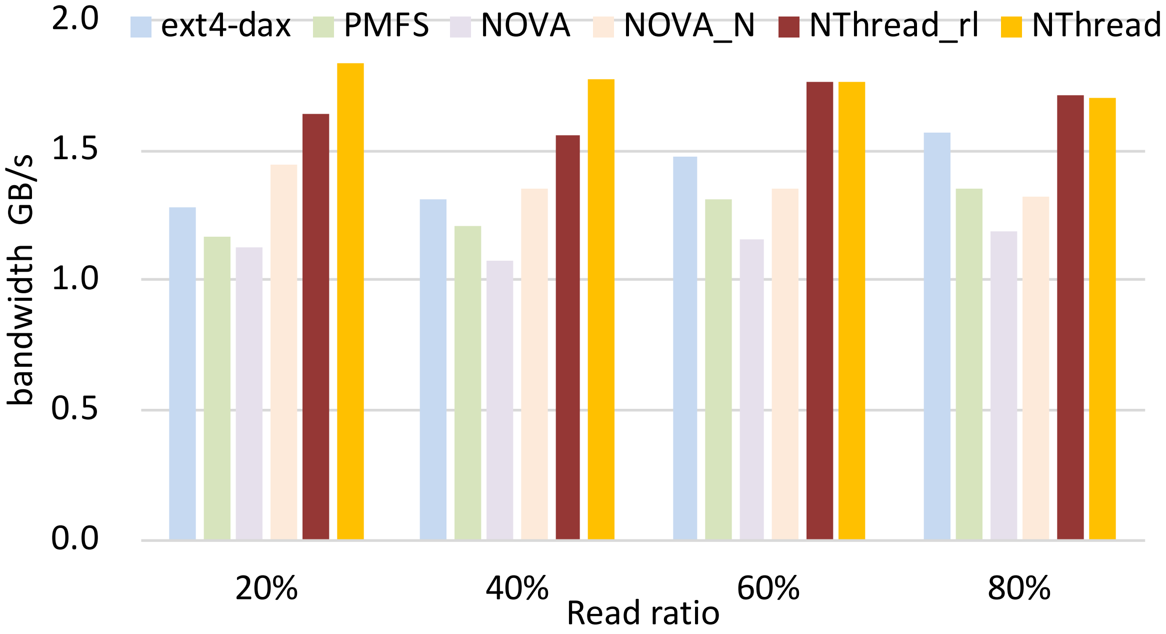
Outline

- Background & Motivation
- NThread design
  - Reduce remote access
  - Reduce resource contention
  - Increase CPU cache sharing
- Evaluation
- Summary

Evaluation

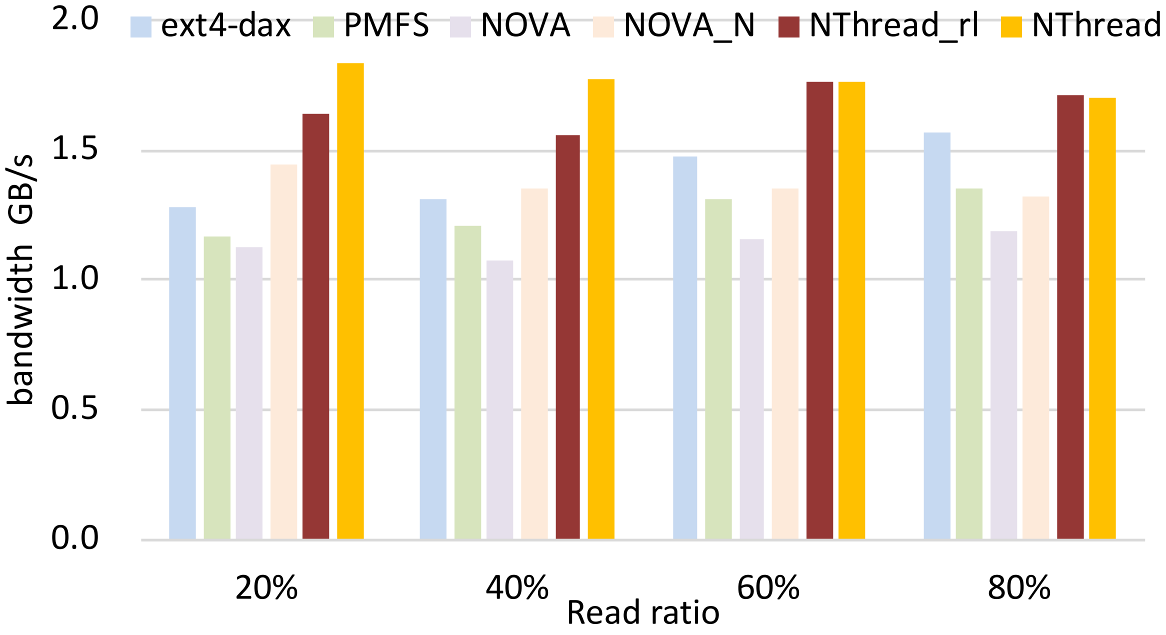
- Platform
  - Two NUMA nodes
    - Intel Xeon 5214 CPU，10 CPU core
    - 64G DRAM, 128G Optane PMM
  - Four NUMA nodes
    - Intel Xeon 5214 CPU，10 CPU core
    - 4GB DRAM, 12GB Emulated PMM
- Compared system
  - Existing FS: Ext4-dax, PMFS, NOVA, NOVA_n
  - Modified FS: NOVA_n (A NOVA-based multi-node support FS)

Micro-benchmark: fio



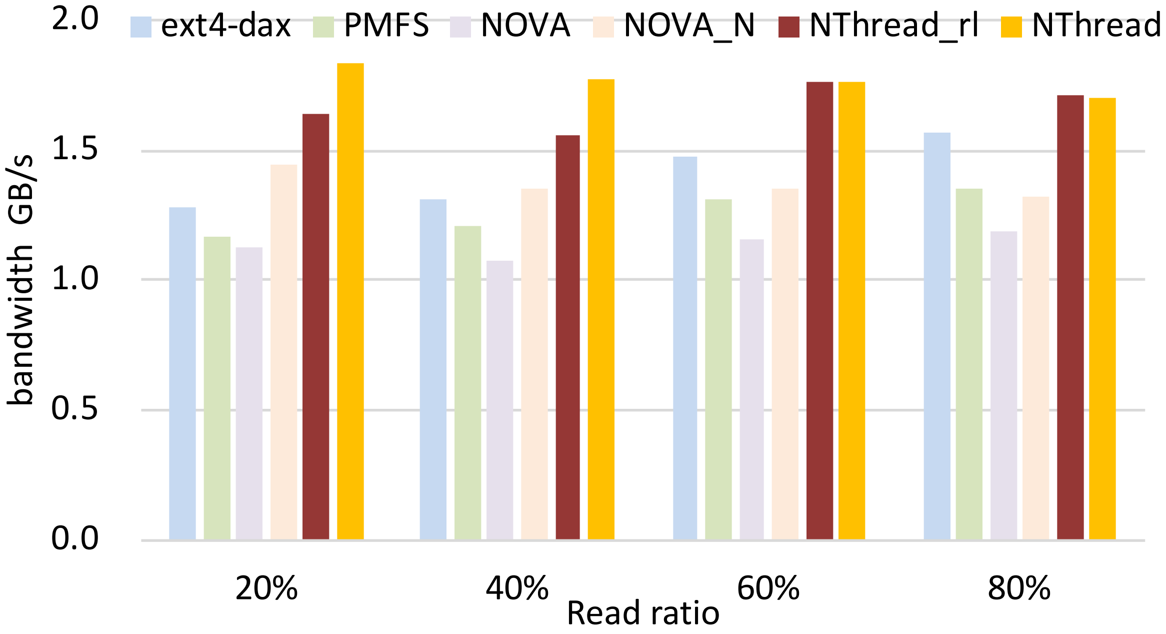
Micro-benchmark: fio

- NThread_rl: reduce remote access
  - The bandwidth is increased by 26.9% when the read ratio is 40%



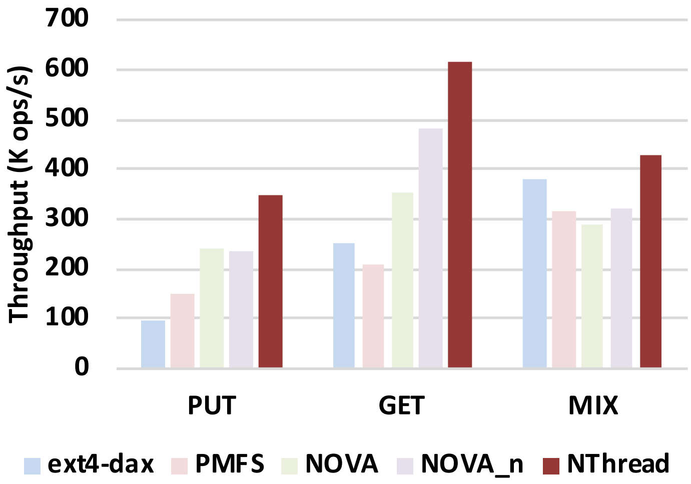
Micro-benchmark: fio

- NThread_rl: reduce remote access
  - The bandwidth is increased by 26.9% when the read ratio is 40%
- NThread: reduce remote access, avoid contention and increase CPU sharing
  - Bandwidth increased by an average of 43.8%

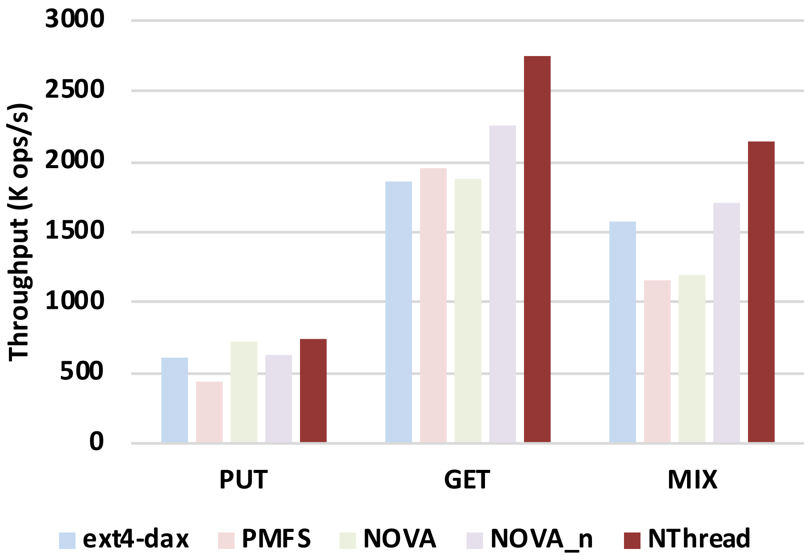


Application: RocksDB

- NThread increases the throughput by 88.6% on average when RocksDB runs in the NVMM file system



Two NUMA nodes



Four NUMA nodes

Outline

- Background & Motivation
- NThread design
  - Reduce remote access
  - Reduce resource contention
  - Increase CPU cache sharing
- Evaluation
- Summary

Summary

Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS

Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS
- NUMA brings remote access and resource contention to NVMM FS

Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS
- NUMA brings remote access and resource contention to NVMM FS
- NThread is a NUMA-aware thread migration

Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS
- NUMA brings remote access and resource contention to NVMM FS
- NThread is a NUMA-aware thread migration
  - Migrate threads according to data amount to reduce remote access

Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS
- NUMA brings remote access and resource contention to NVMM FS
- NThread is a NUMA-aware thread migration
  - Migrate threads according to data amount to reduce remote access
  - Reduce resource contention and avoid introducing new contention

Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS
- NUMA brings remote access and resource contention to NVMM FS
- NThread is a NUMA-aware thread migration
  - Migrate threads according to data amount to reduce remote access
  - Reduce resource contention and avoid introducing new contention
  - Avoid migrating data-sharing threads to increase CPU cache sharing

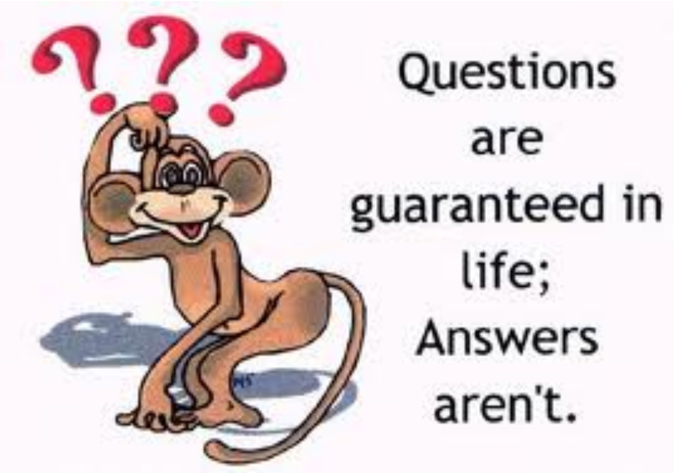
Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS
- NUMA brings remote access and resource contention to NVMM FS
- NThread is a NUMA-aware thread migration
  - Migrate threads according to data amount to reduce remote access
  - Reduce resource contention and avoid introducing new contention
  - Avoid migrating data-sharing threads to increase CPU cache sharing
  - Apply what-if analysis to decide the execution orders of these optimizations

Summary

- The features of NVMM enables FS to be built on the memory bus, improving the performance of FS
- NUMA brings remote access and resource contention to NVMM FS
- NThread is a NUMA-aware thread migration
  - Migrate threads according to data amount to reduce remote access
  - Reduce resource contention and avoid introducing new contention
  - Avoid migrating data-sharing threads to increase CPU cache sharing
  - Apply what-if analysis to decide the execution orders of these optimizations
  - Increase application throughput by 88.6% on average

Thanks！



Author email: wangying01@ict.ac.cn