Efficient Virtualization for HPC Applications

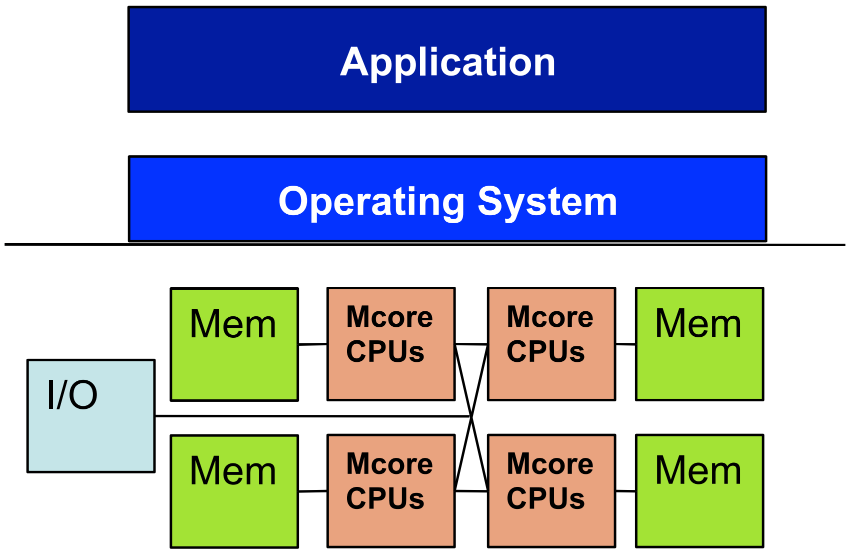
Khaled Ibrahim and Costin Iancu

Future Technologies Group

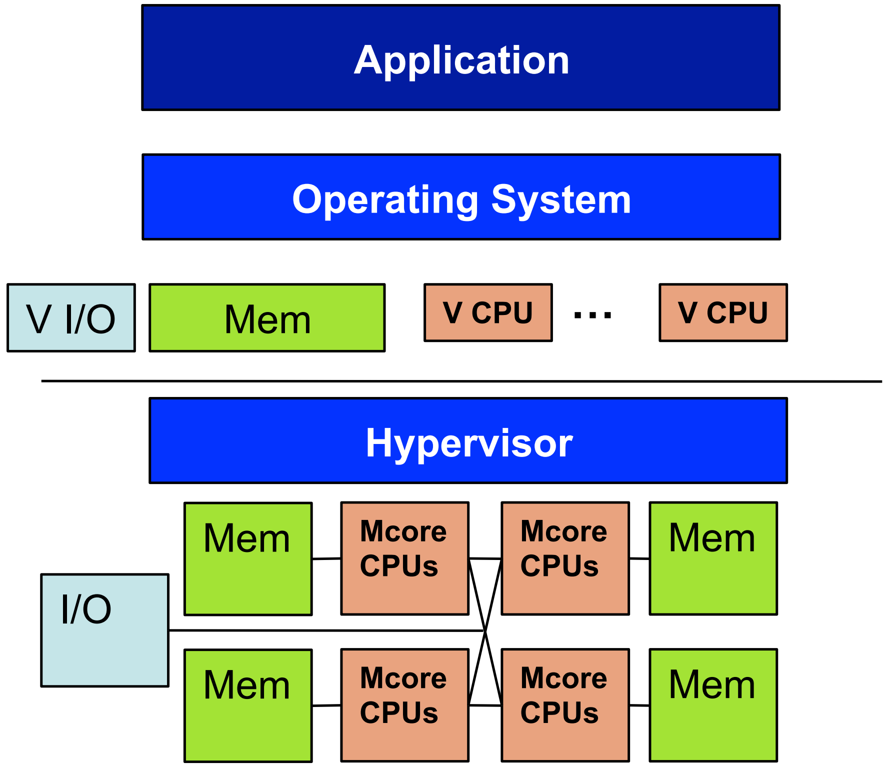
Lawrence Berkeley National Laboratory

Virtualized Computing

- **Conventional Computing Environment**



- **Virtualized Computing Environment**

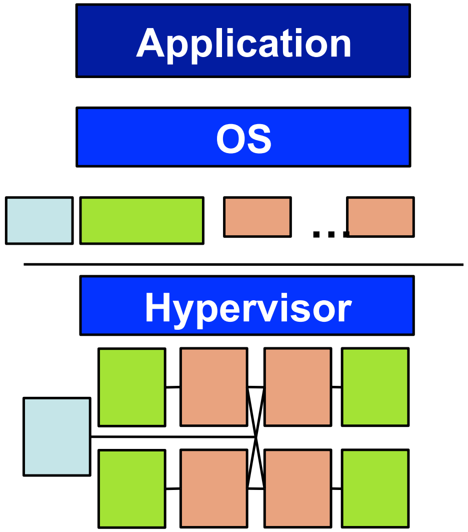


Key difference:

How to interact with H/W resources

Why Virtualization?

- Benefits
  - Resource consolidation
  - Fault isolation & tolerance (leadership HPC centers)
  - Decoupling resource management (for administrators and system users).
- Enabling technology for
  - Cloud Computing
  - Green Computing
- The question
  - What is on the price tag, especially on multicore architectures?



Talk Layout

- Is virtualization ready for the primetime?
- Performance analysis of virtualized environment.
- How to improve the performance of HPC application in virtualized environment!

Talk Layout

- **Is virtualization ready for the primetime?**
- Performance analysis of virtualized environment.
- How to improve the performance of HPC application in virtualized environment!

Performance Expectation and Reality

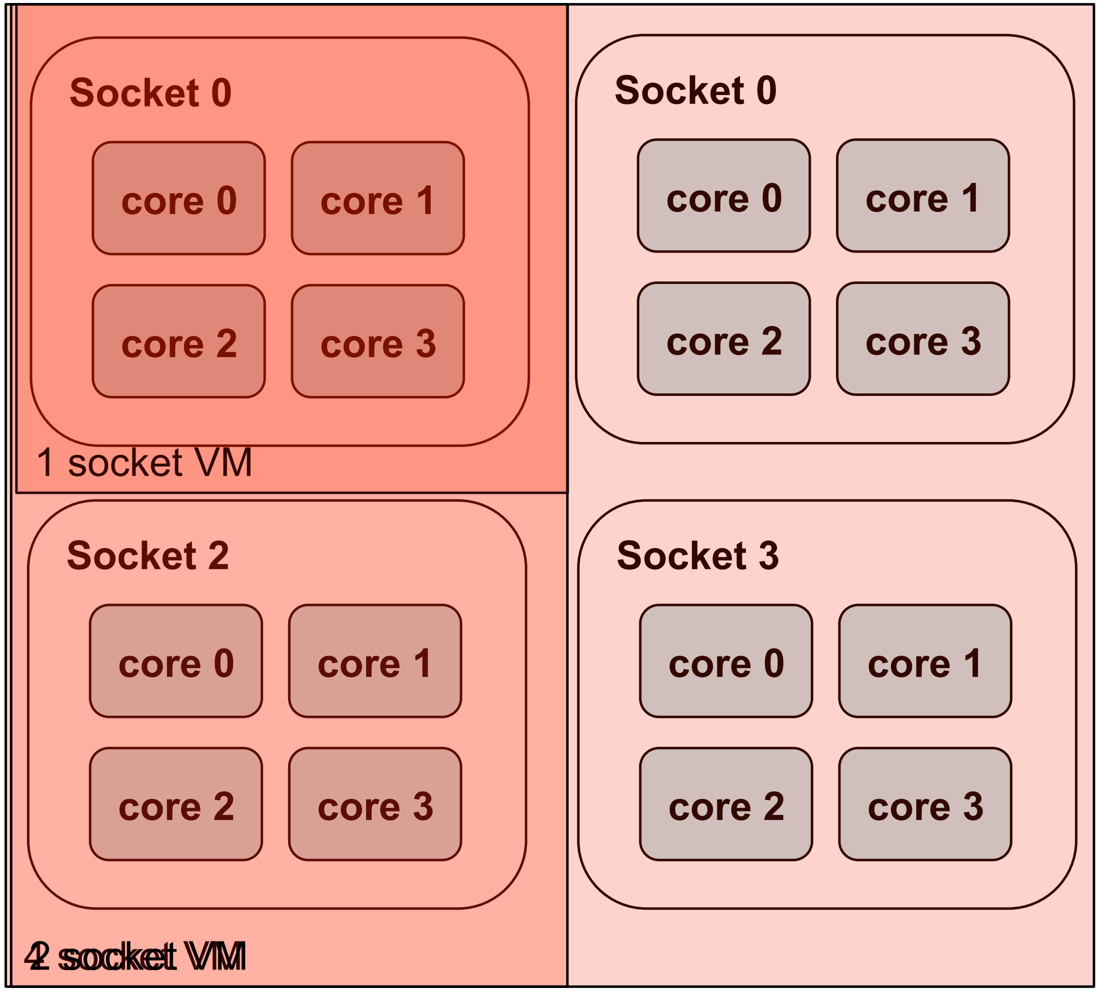
- Virtualization Performance Expectations
  - Performance overhead is low (within 3-5% of raw performance)
  - H/W support for virtualization significantly improve it!
- Studies on Performance
  - Most earlier studies are single socket on few core systems!
  - New studies seen degradation on some popular cloud computing infrastructures (Amazon EC2)!
- HPC Workloads
  - Persistently use a large fraction of the system memory
  - Data locality determines performance – NUMA support
  - Sensitive to network bandwidth and latency – I/O support
  - Use shared and/or distributed memory programming models – configuration/software support

Experimental Setup

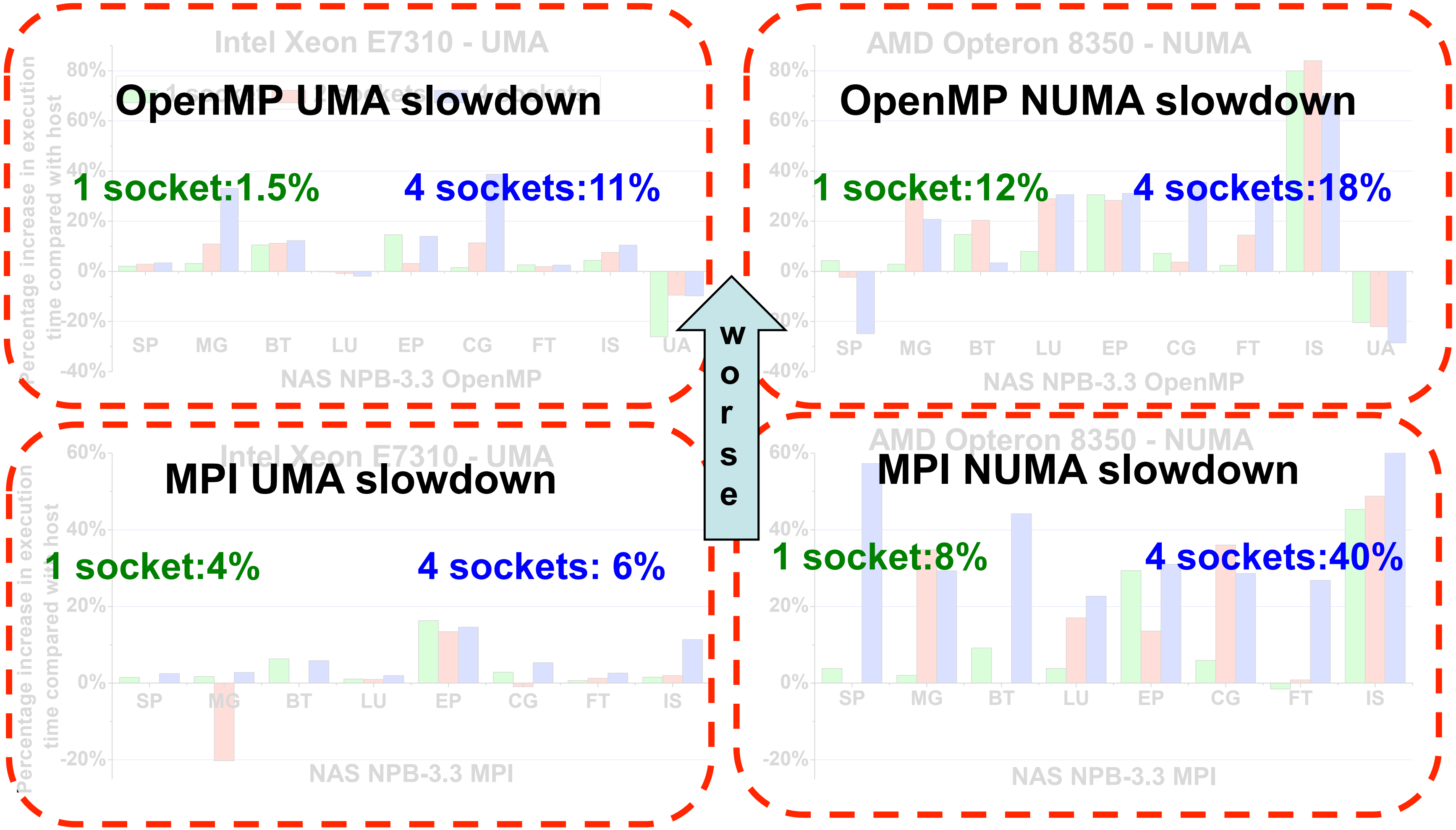
- **Virtualization technology full H/W support for memory and I/O**
  - **KVM/QEMU 0.13.0**
  - **Xen 4.0**
- **Operating Systems** Linux (Kernel 2.6.32.8)
- **Programming Models**
  - MPI
  - OpenMP
  - UPC
- **Benchmarks NAS Parallel benchmarks (3.3)**
- **Architectures**
  - 4X4 UMA : Tigerton Xeon(R) CPU E7310
  - 4X4 NUMA: AMD Opteron(tm) Processor 8350
  - 2X4 NUMA: Intel Xeon E5530 (Nehalem EP).
- **Multinode Experiments**
  - Two 4x4 UMA Tigerton connected through Giga-bit Ethernet.

Virtualization Overhead Experiment

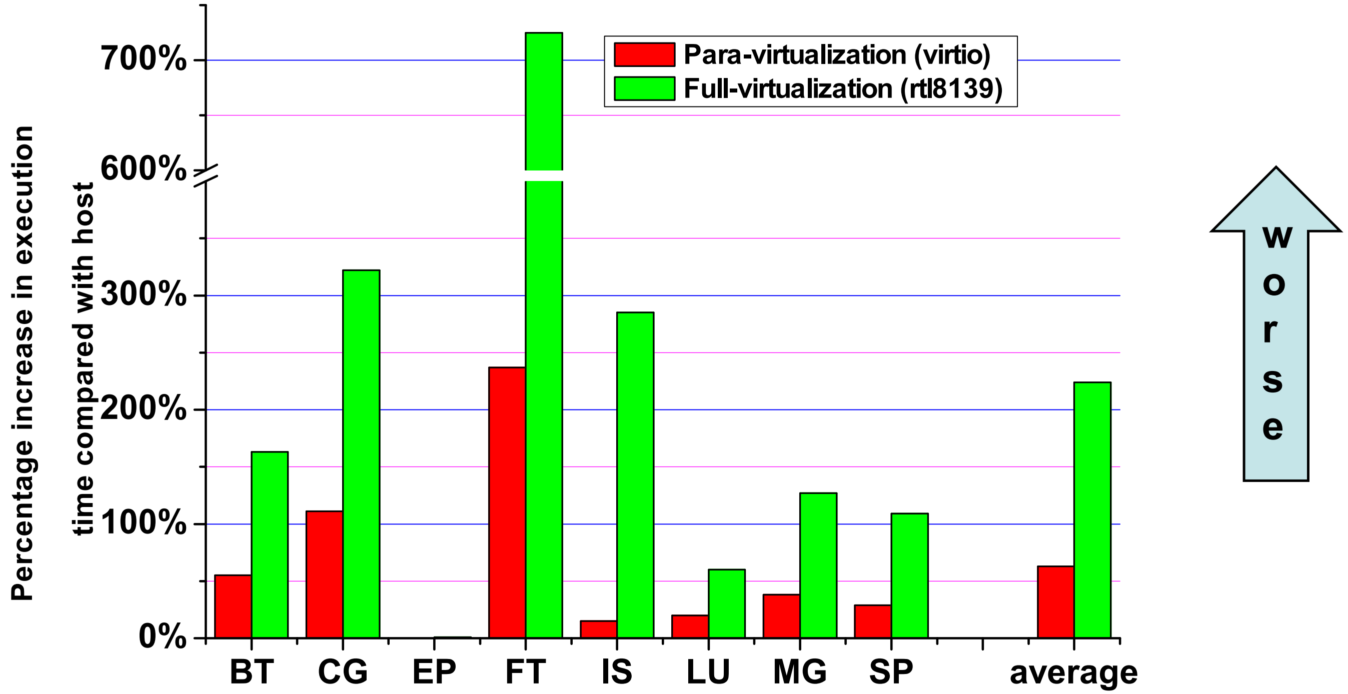
- **Three configurations**
  - 1 socket VM
  - 2 socket VM
  - 4 socket VM
- **Two architectures**
  - UMA
  - NUMA
- **Two programming models**
  - MPI
  - OpenMP



Performance of KVM on Multi-Socket System (Single Node)



Multinode Performance



Significant slowdowns with IO activity:

At least 63% slowdown with virtio on average on UMA machines. (220% for full virtualization)

Talk Layout

- **Is virtualization ready for the primetime?**

  Not out of the box

- Performance analysis of virtualized environment.

- How to improve the performance of HPC application in virtualized environment!

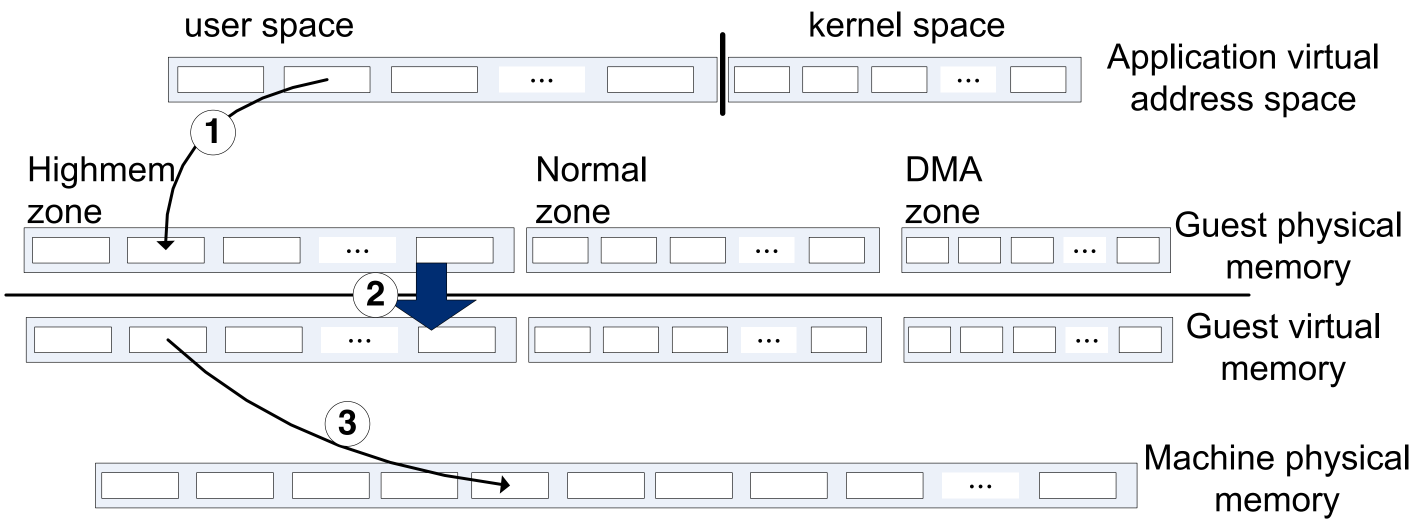
Talk Layout

- Is virtualization ready for the primetime?

  Not out of the box

- **Performance analysis of virtualized environment.**
  - Page Mapping and NUMA Locality
  - IO Performance (full vs. para-virtualization)
- How to improve the performance of HPC application in virtualized environment!

Page Translation Mechanism (KVM)

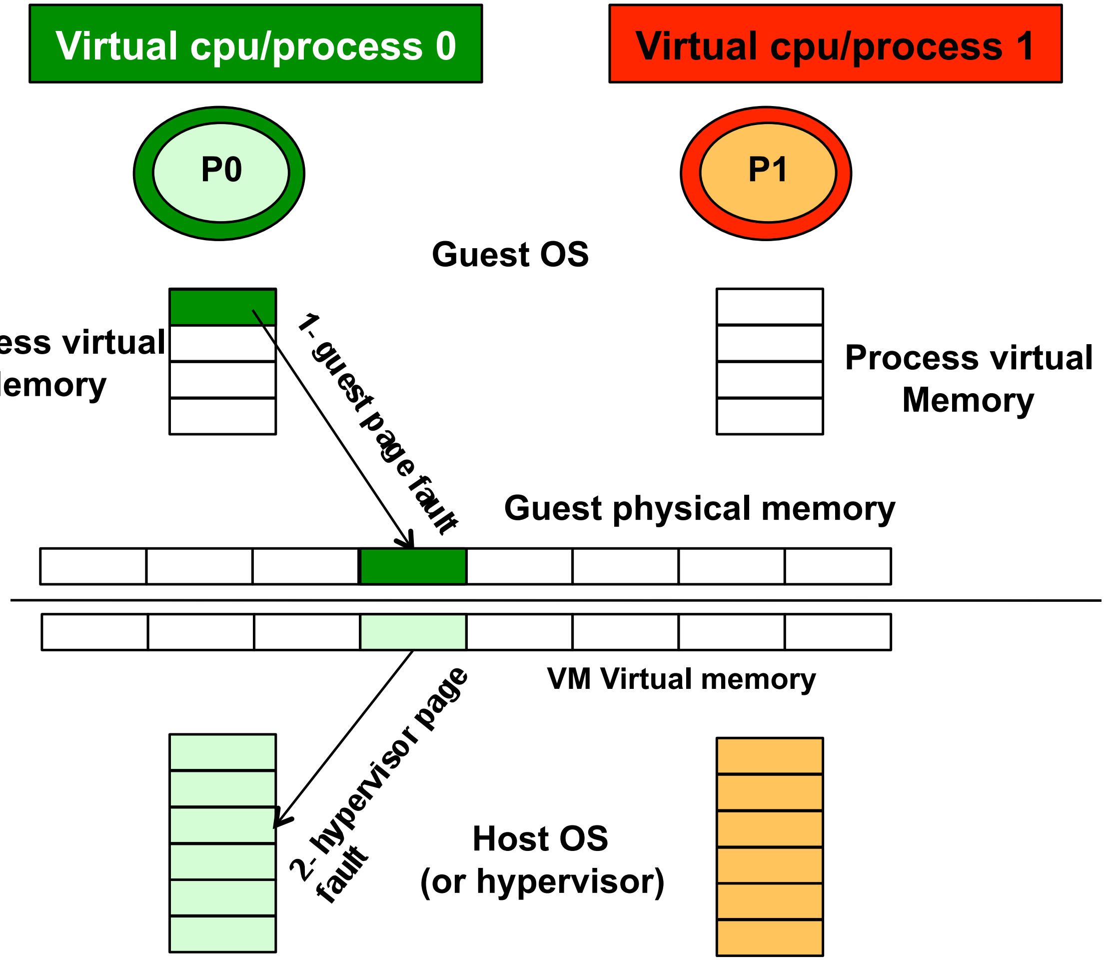


- Three stage translation
  - 2 Dynamic (runtime) and one static (launch time)
- Page translation mechanism cause locality problem.

Page Translation in Two NUMA domains

F U T U R E T E C H N O L O G I E S G R O U P

- Cold touch involves two page faults
  - Guest fault (NUMA oblivious)
  - Hypervisor fault (NUMA aware)

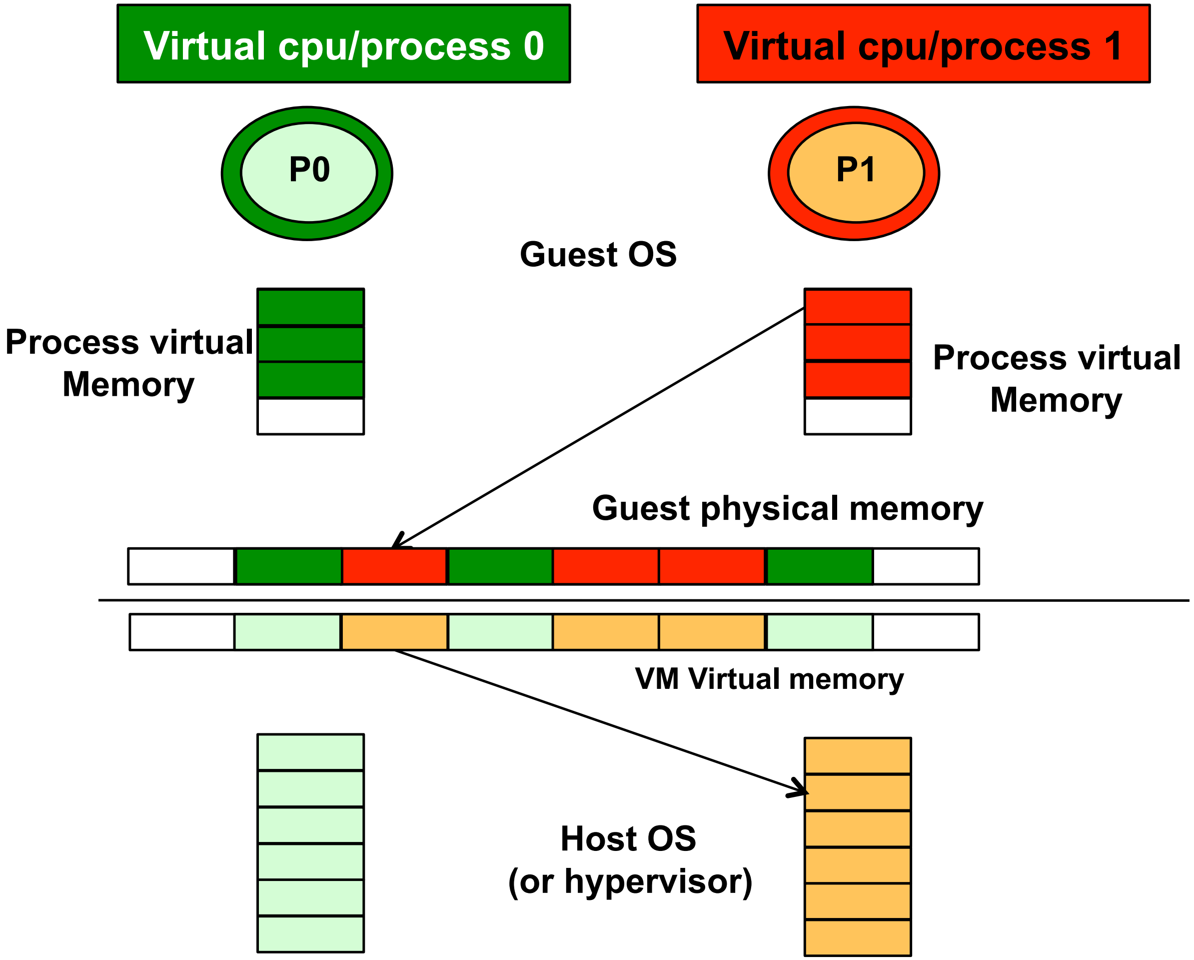


Two phase translation mechanism for application for the first touch of a guest page

Multiple Page Fault Outcome

FUTURE TECHNOLOGIES GROUP

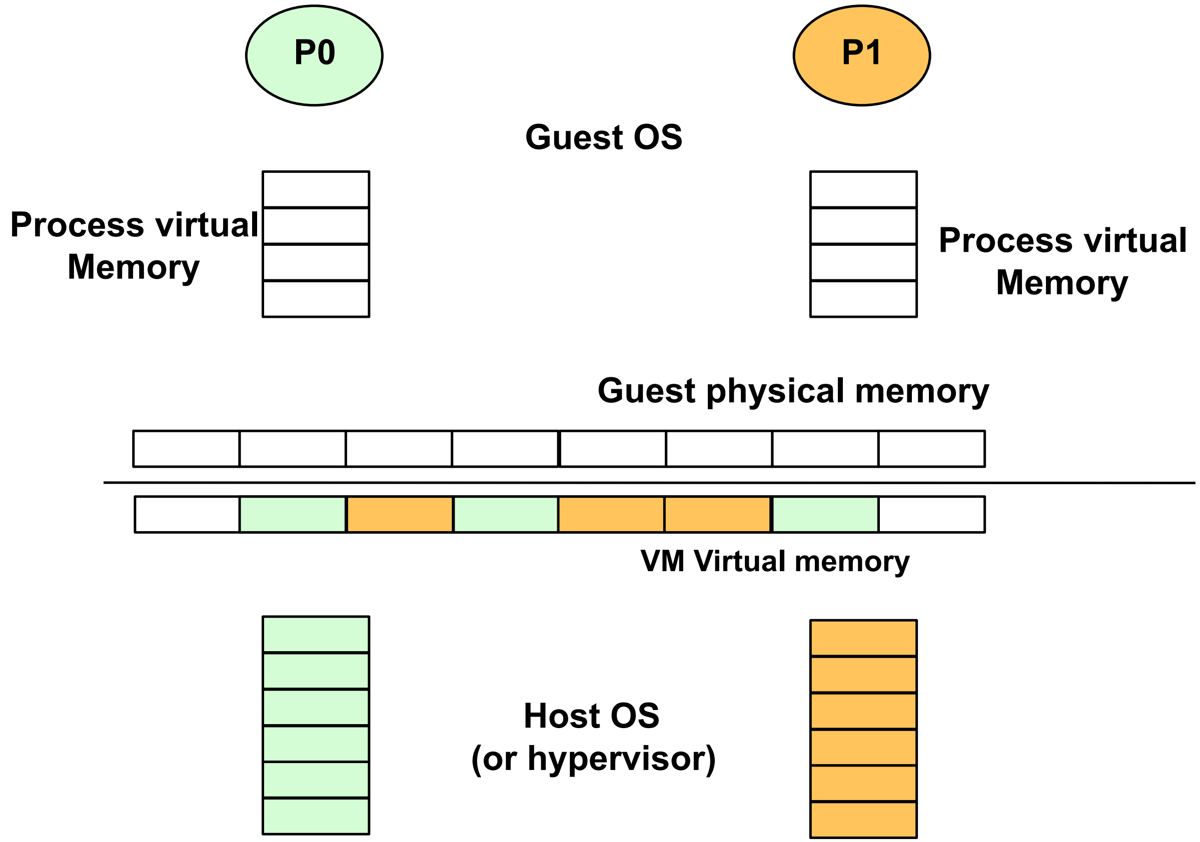
- Correct NUMA affinity is managed by hypervisor.



Two phase translation mechanism for application for the first touch of a page

Guest Application Termination/ Page Release

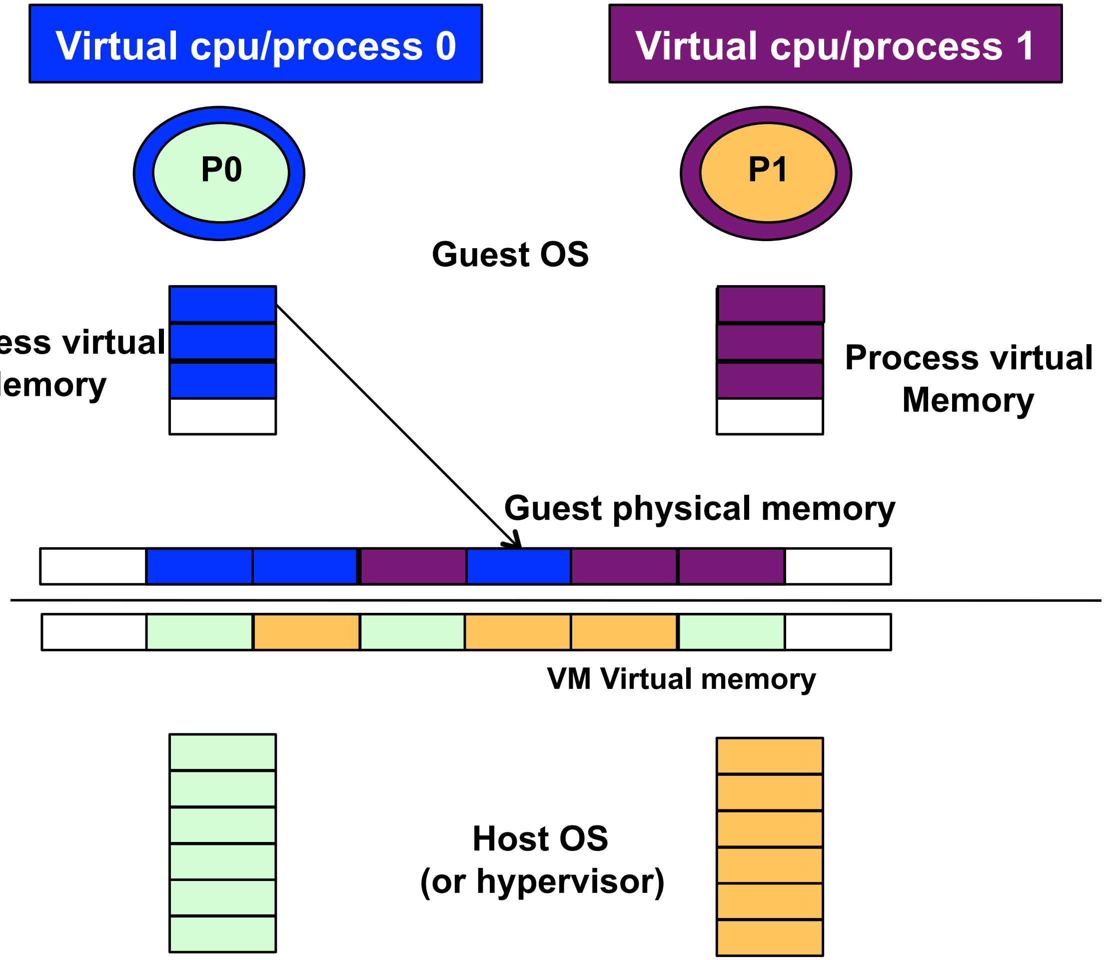
- Memory mappings in hypervisor are persistent.



System image after application termination.

New Application is launched

- Hypervisor mapping is recycled and locality is not guaranteed.



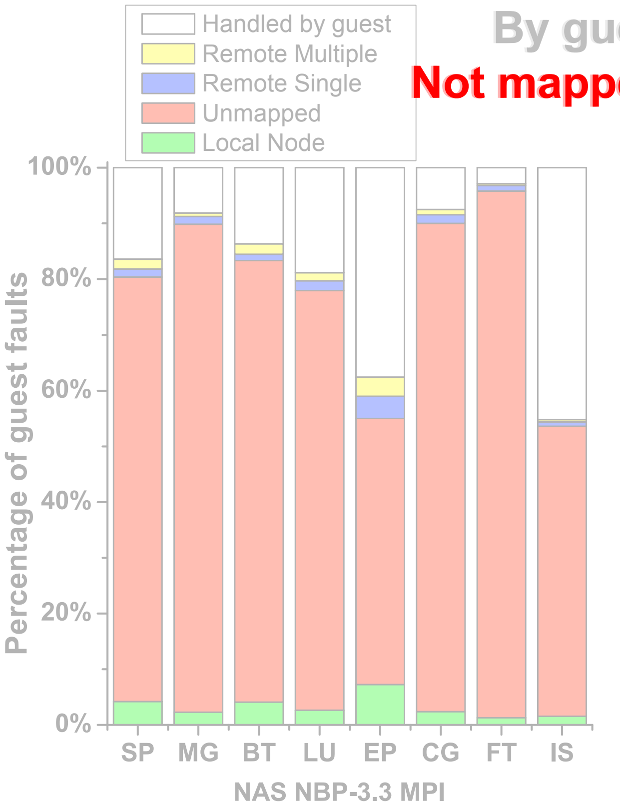
Page reuse results in host only page fault

Page Faults Propagated to Hypervisor

Cold VM

By guest 18%

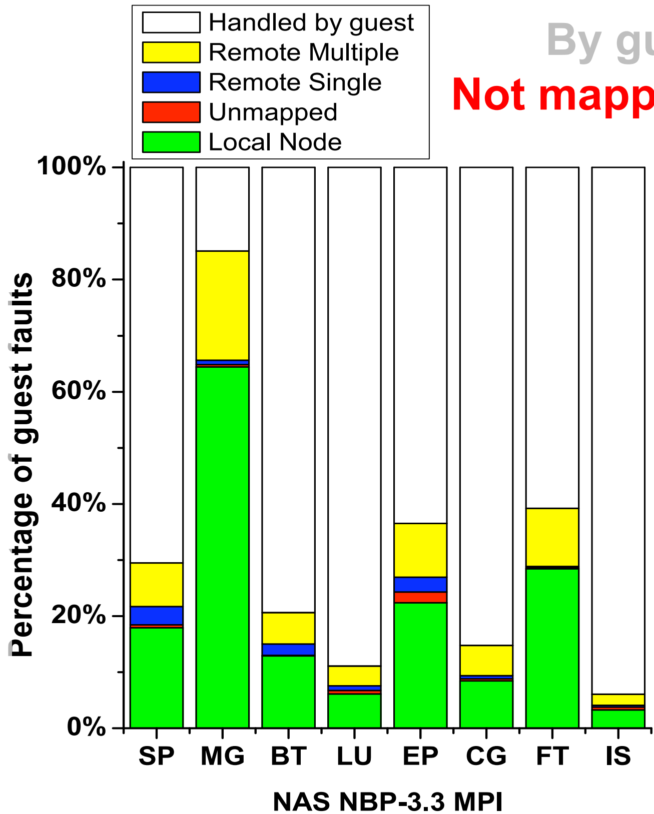
Not mapped 75%



Warm VM

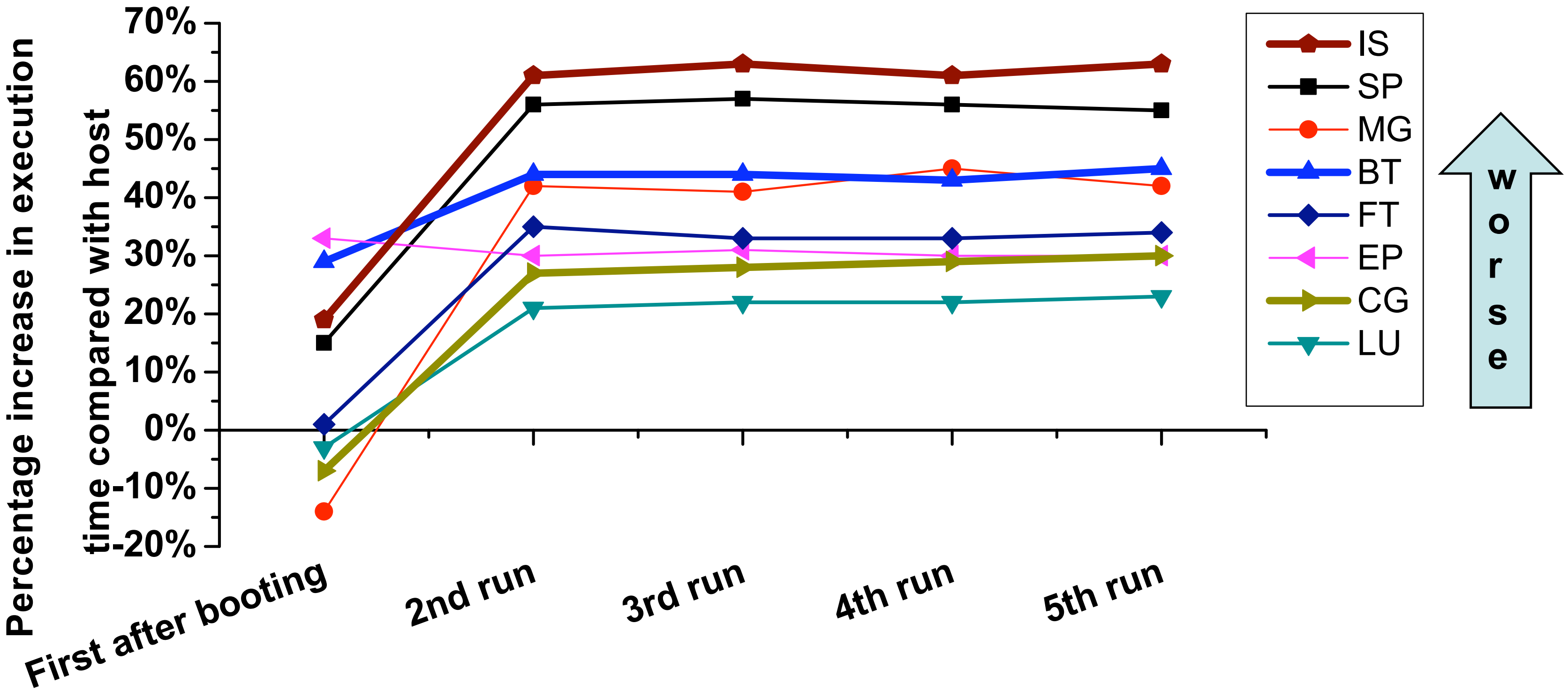
By guest 70%

Not mapped 0.6%



First Run Behavior (MPI)

Warm VMs provide lower performance!



First run avg. slowdown: 9%, second run avg. slowdown: 40%

Other Virtualization Technologies NUMA Support

- **Xen (The other open-source)**
  - Two phase page translation.
  - Pre-allocation of VM memory from first NUMA node.
  - 233% average slowdown (compared with 40% for KVM).

- **VMWare**
  - Limited vcpus
  - Guest is not NUMA aware
  - Restrictions on reporting number for VMWare

Talk Layout

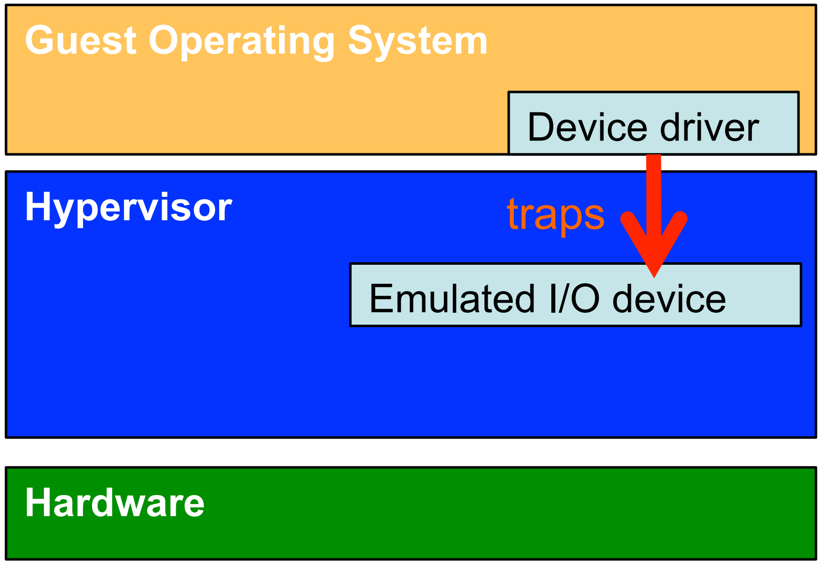
- Is virtualization ready for the primetime?

  Not out of the box

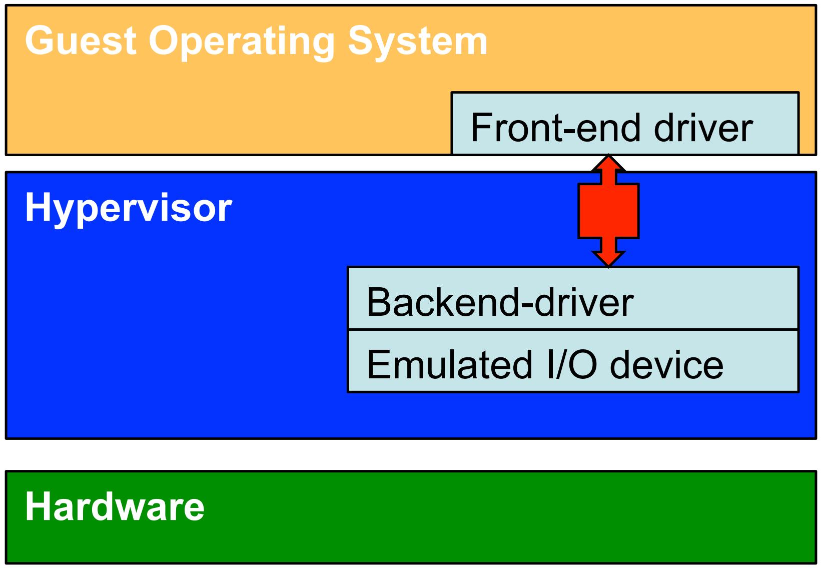
- **Performance analysis of virtualized environment.**
  - Page Mapping and NUMA Locality
  - IO Performance (full vs. para-virtualization)
- How to improve the performance of HPC application in virtualized environment!

IO Virtualization

Full Virtualization (e.g. rtl8139)

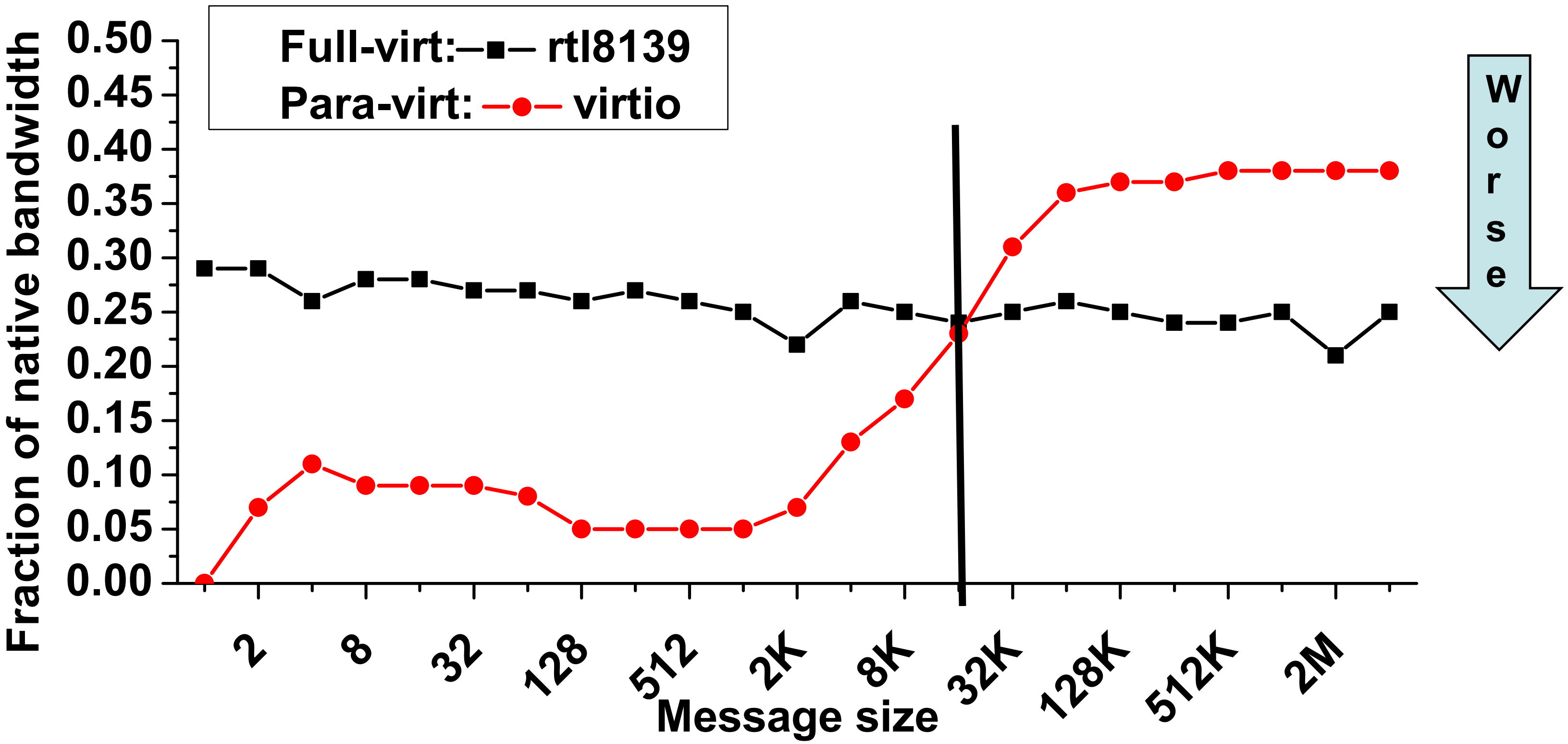


Para-Virtualization (e.g. virtio)



IO Performance VM

FUTURE TECHNOLOGIES GROUP



Para-virtualization better for large messages
full-virtualization better for small messages
Why?

Talk Layout

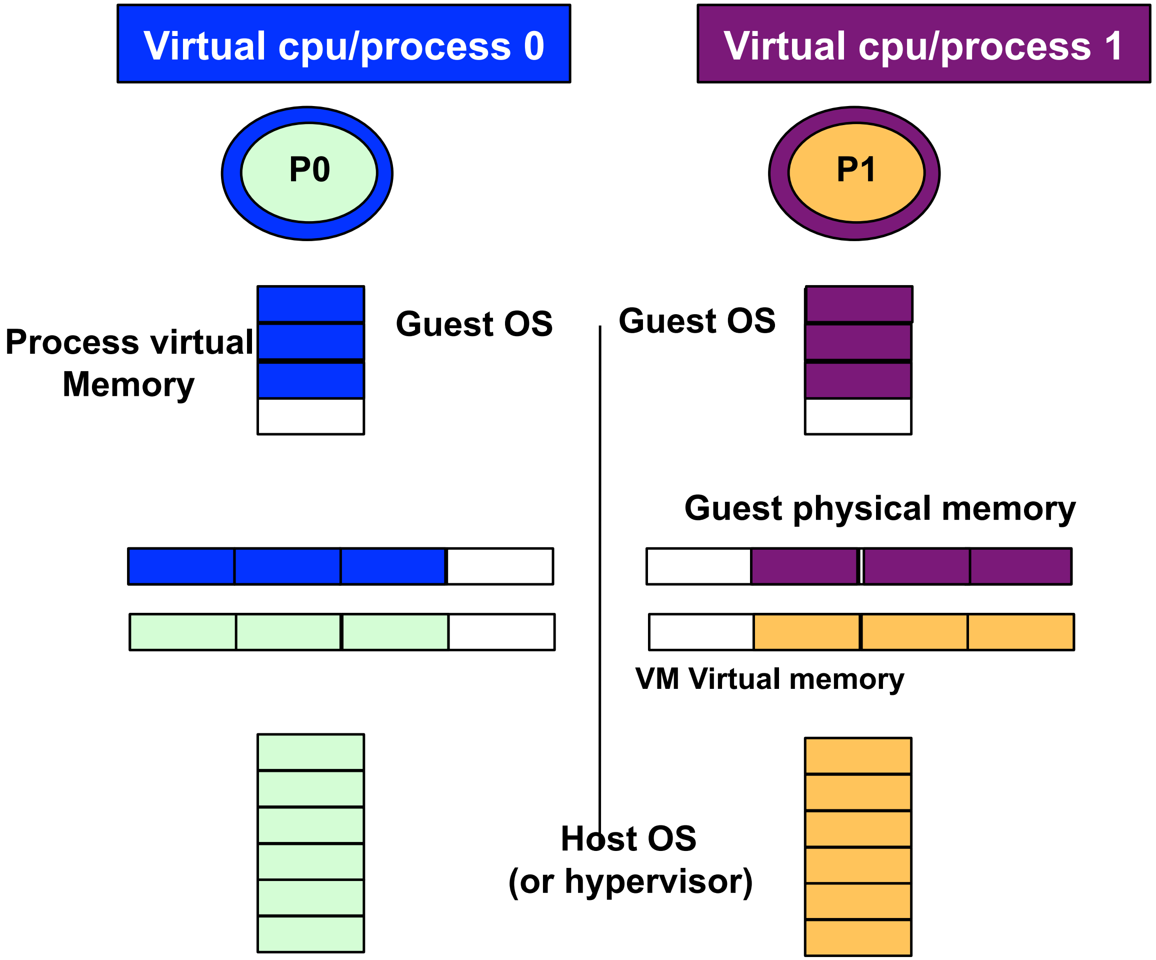
- Is virtualization ready for the primetime?

  Not out of the box

- Performance analysis of virtualized environment.
  - Page Mapping and NUMA Locality
  - IO Performance (full vs. para-virtualization)
- **How to improve the performance of HPC application in virtualized environment!**

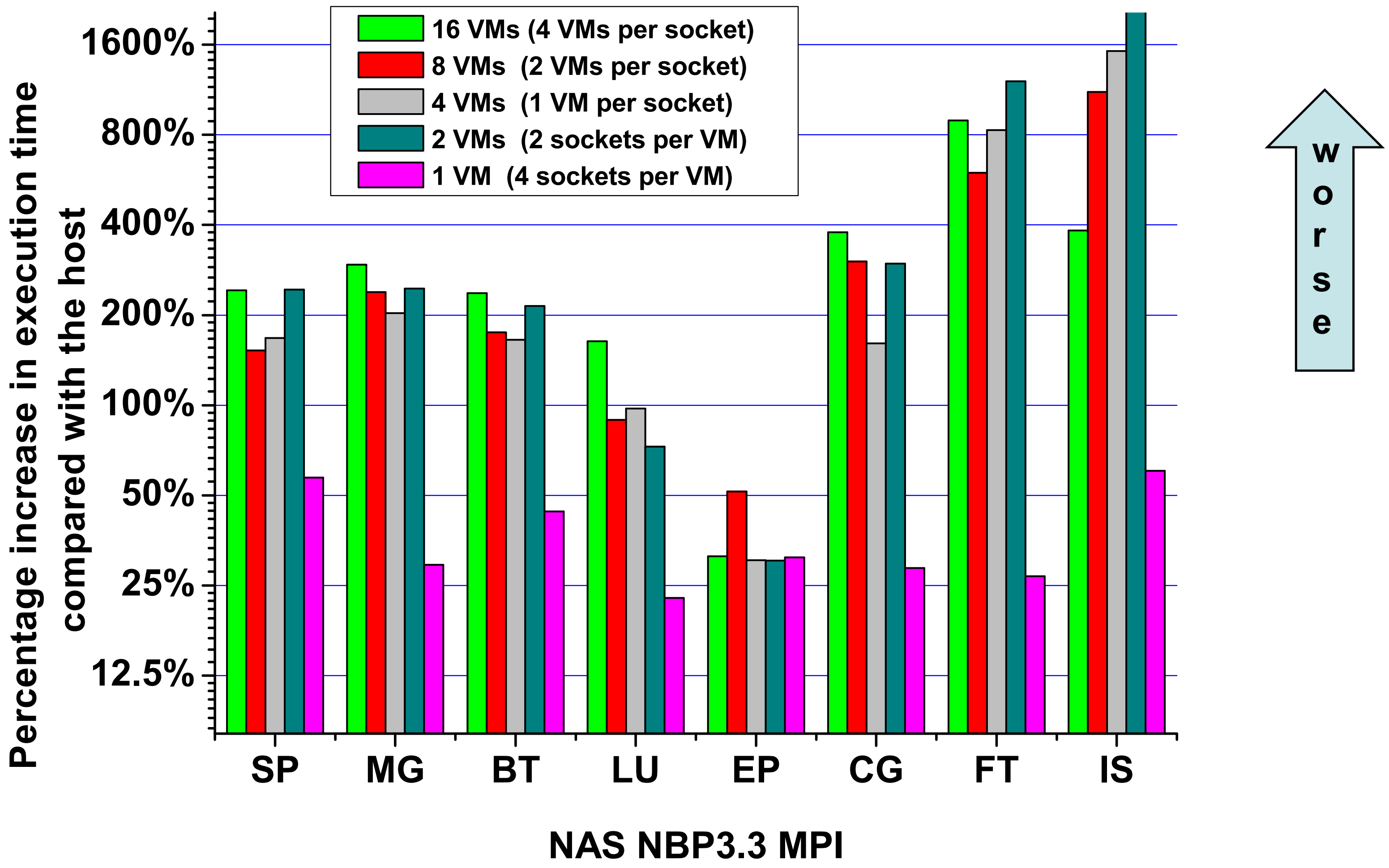
VM Node Confinement (Partitioning)

- Vendors advocate node confinement
  - One VM per NUMA domain
- Performance:
  - Resource Contention
  - Inter-VM communication

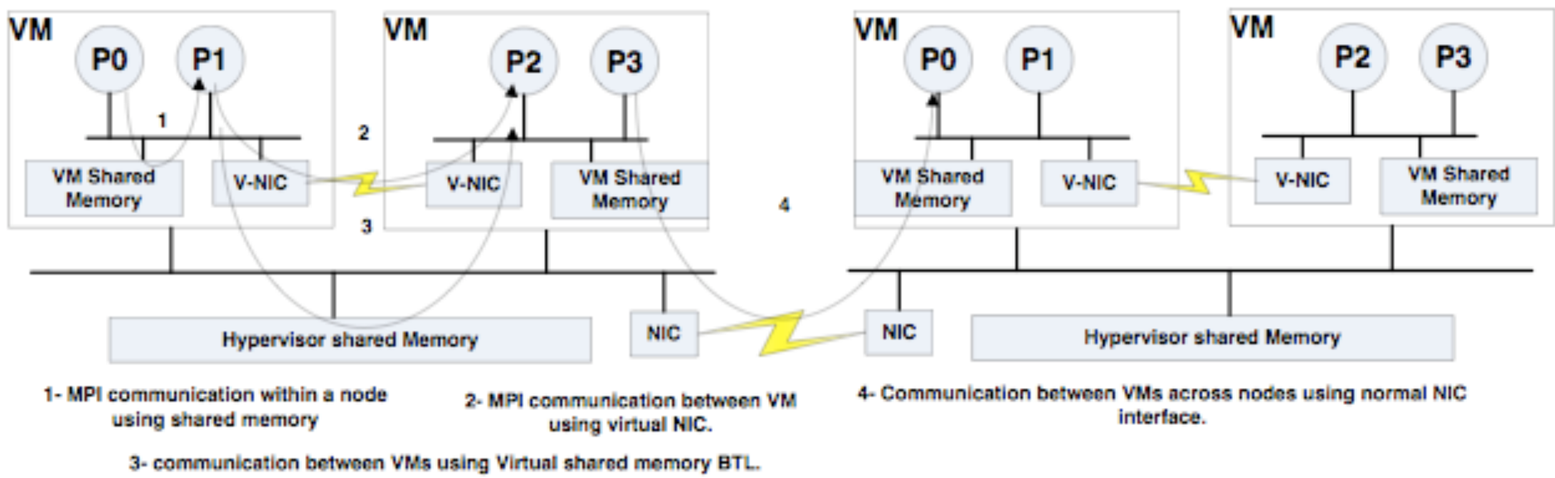


Page reuse results in host only page fault

Out–of-the box Partitioning



Solution: Efficient Inter-VM Communication

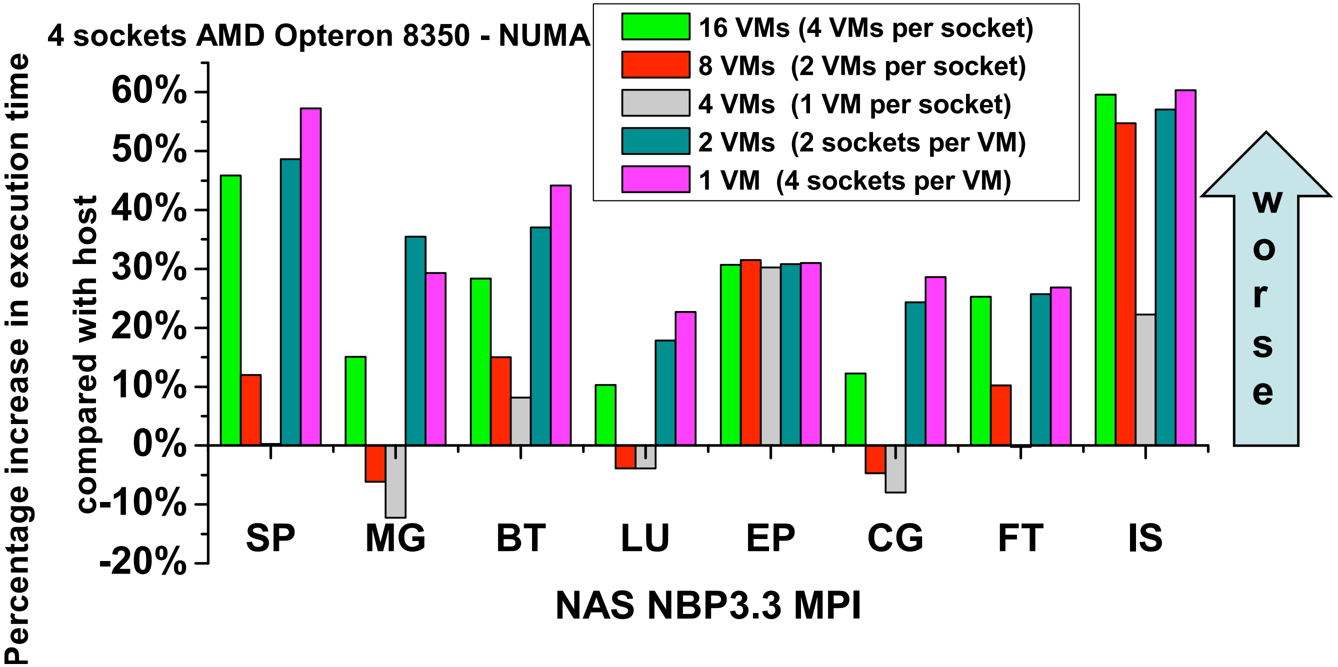


- Shared memory is exposed to guest as a PCI device memory (hypervisor modification).
- Modification to runtime OpenMPI (guest runtime modification)
  - VM Shared memory communication component.
  - VM memory pool communication component.
  - VM collective communication component.
- New selection mechanism for communication component.
- Similar mechanism is implemented for UPC, but has restriction on the dataset sizes.

Performance with Partitioning and Inter-VM Shared Memory

One VM per node (1VM)
Slowdown: 40%

One VM per NUMA domain: (4VM)
Slowdown: 3%

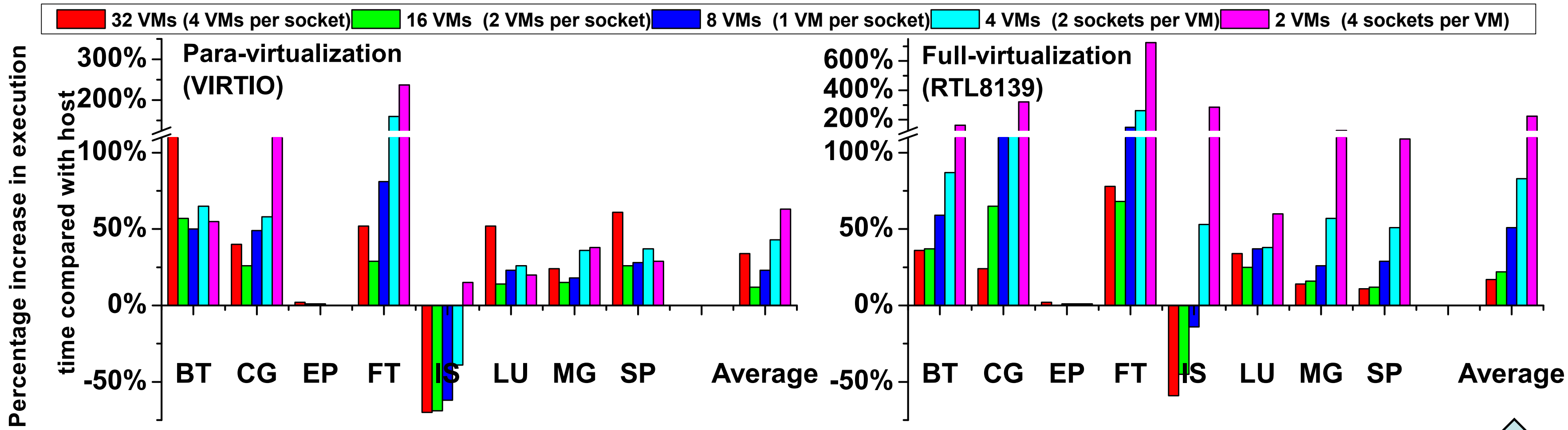


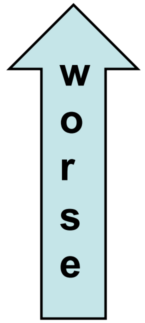
- One VM per socket is usually the best configuration.
- Efficient Inter-VM communication is a key to performance.

Results published in the 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, May 2011.

IO Performance with Partitioning

FUTURE TECHNOLOGIES GROUP





One VM per node (2 VM): Slowdown: 63%

One VM per core: (32 VM): Slowdown: 17%

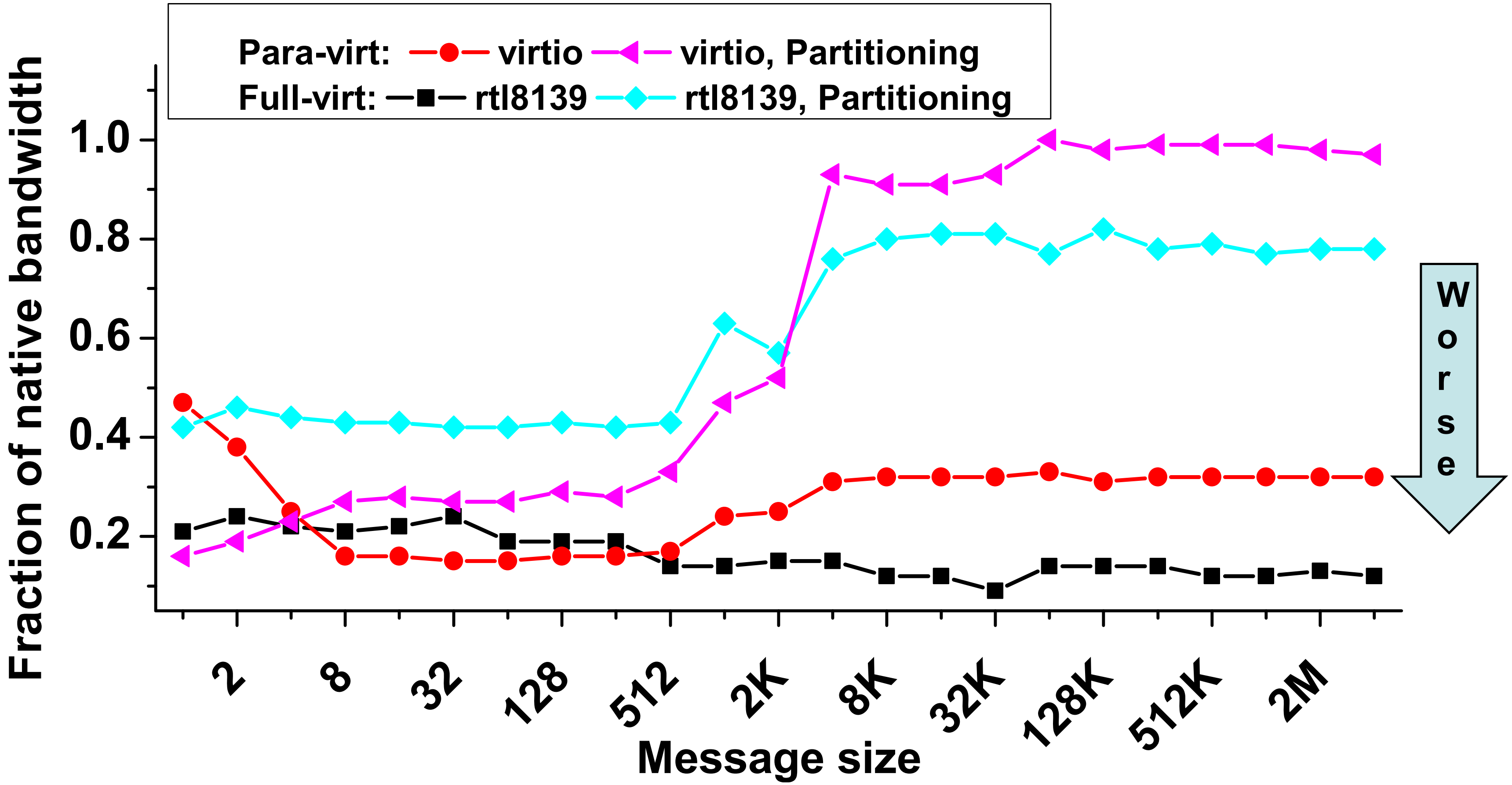
Partitioning improve the IO performance for full and para virtualization

Improvement on full virtualization is higher, even beating para-virtualization

Do we need para-virtualization intervention?!

IO Performance with Efficient Partitioning

FUTURE TECHNOLOGIES GROUP



Conclusion

- Virtualization for HPC Application
  - Out-of-the-box performance disappointing (40% due to NUMA, 63% due to network IO with UMA)
- Efficient partitioning can improve the performance
  - Provide better locality on NUMA
  - Provide IO concurrency
- Requirement for efficient partitioning
  - Modification to the hypervisor to expose shared memory.
  - Modification to the runtime to (MPI, UPC, etc) to exploit them.
- Efficient communication between partitioning reduces the impact of virtualization performance on performance.
  - On Numa nodes 40% -> 3%
  - On Multi nodes 63% -> 17%
  - Efficient Partitioning can render the complex para-virtualization technique unnecessary for Multinodes.