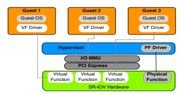
Designing and Building Efficient HPC Cloud with Modern Networking Technologies on Heterogeneous HPC Clusters

Jie Zhang zhanjie@cse.ohio-state.edu
Advisor: Dhabaleswar K. (DK) Panda panda@cse.ohio-state.edu

Introduction

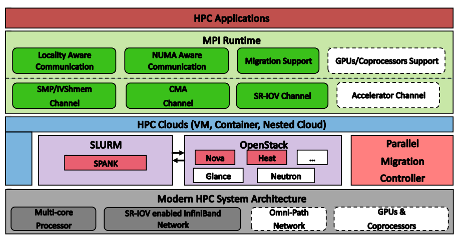
- Cloud Computing has been widely adopted in industry computing environment due to its attractive features, like resource sharing, live migration, easy management
- Virtualization is the key technology in Cloud
- More and more people are trying to build HPC clouds to gain both high flexibility and powerful computing capability
- Single Root I/O Virtualization (SR-IOV) delivers near-native I/O performance in virtualized environment
- SR-IOV does not have high performance virtualization support, such as locality aware and NUMA aware communication support
- SR-IOV based virtual networks prevent the VM migration with current generation of hypervisors and SR-IOV drivers
- HPC clouds cannot support running multiple concurrent jobs without properly isolate and manage critical virtualized resource through resource management and job scheduling system

(Cloud Computing)

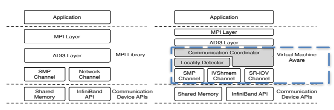


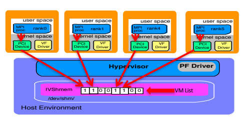
(SR-IOV Architecture)

Research Framework

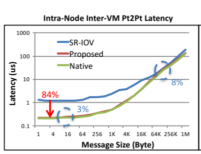


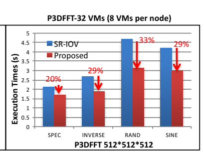
Locality-aware MPI Communication

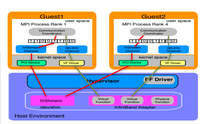




- Redesign MPI library in virtualized environment [4,6,7]
  - Support shared-memory channels (SMP, IVShmem) and SR-IOV channel
  - Add Locality Detector
  - Add Communication Coordinator
  - Optimize different communication channels (SMP, IVShmem, SR-IOV; RC, UD)
- Create a VM List structure on IVShmem region of each host
- Each MPI process writes its own membership information into shared VM List structure according to its global rank
- One byte each, lock-free, O(N)

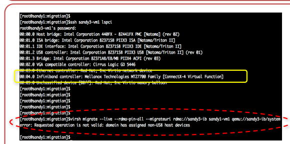


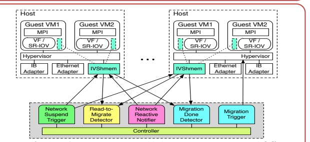




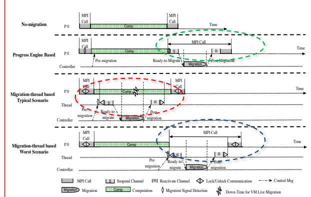
- Retrieve VM locality detection information
- Reschedule communication channels based on VM locality information

SR-IOV enabled VM Migration Support

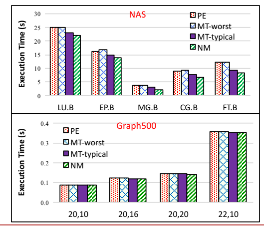




(High Performance SR-IOV enabled VM Migration Framework [1])

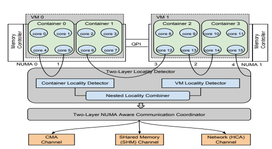


- Propose high performance SR-IOV enabled VM migration framework
  - External Controller coordinates VM during migration with multiple parallel libraries (detach/reattach SR-IOV/IVShmem, migrate VMs, track migration status)
  - MPI runtime handles IB connection suspending and reactivating
- Propose Progress Engine (PE) and Migration Thread based (MT) in MPI runtime to handle IB connection
  - PE: Leverage MPI progress engine thread to detect migration signal, handle IB connection suspension/reactivation
  - MT: Create a dedicated thread, to detect migration signal, handle IB connection suspension/reactivation, which allows overlapping migration with computation (MT-typical)

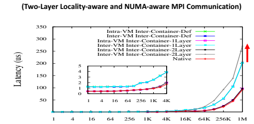


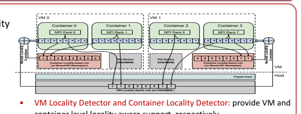
Two-Layer Locality-aware and NUMA-aware MPI Communication

- Nested Virtualization is gaining momentum with both portability and isolation
- There are multiple data paths existing on multi-core processor
- Propose Two-Layer Locality-aware and NUMA-aware MPI Communication Support [2]
  - Add Two-layer Locality Detector
  - Add Two-Layer NUMA Aware Communication Coordinator

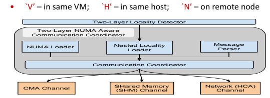


(Two-Layer Locality-aware and NUMA-aware MPI Communication)

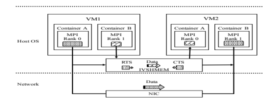




- VM Locality Detector and Container Locality Detector: provide VM and container level locality aware support, respectively
- Nested Locality Combiner combines locality info from both VM and container locality aware lists
- `V' – in same VM; `H' – in same host; `N' – on remote node

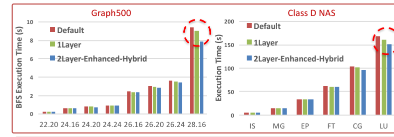


- Reschedule communication channels based on nested locality info (Nested Locality Loader), NUMA info (NUMA Loader), and message attributes (Message Parser)



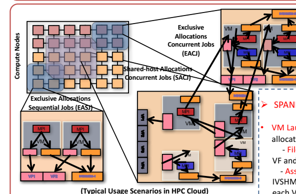
(Enhanced Hybrid)

- ISSUE Inter-Socket – Inter-VM Inter-Container-2Layer (datapath 3) delivers near-native performance for small message, but clear overhead (67%) for large message, compared with *-1Layer
- Enhanced Hybrid – small and control (RTS, CTS) messages go with shared memory channel, ONLY data payload of large message goes through network channel



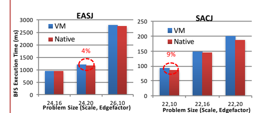
- 256 processes across 64 containers on 16 nodes
- Compared with Default, 2Layer-Enhanced-Hybrid reduces up to 16% (28,16) and 10% (LU) of execution time for Graph 500 and NAS, respectively
- Compared with the 1Layer, 2Layer-Enhanced-Hybrid brings up to 12% (28,16) and 6% (LU) performance benefit

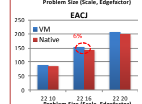
Slurm-V: Co-design with resource management and job scheduling system



(Typical Usage Scenarios in HPC Cloud)

- HPC cloud can have multiple typical usage scenarios
- Critical virtualized resources need to be properly managed and isolated
- Propose Slurm-V framework, which extends Slurm to provide virtualization-oriented capabilities [3,5]

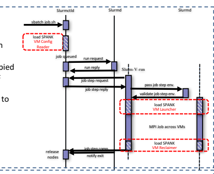




- Graph500 with 64 Procs on 32VMs across 8 Nodes on Chameleon
- EASJ – Only 4% overhead
- SACJ, EACJ – less than 9% overhead, when running NAS as concurrent job with 64 Procs

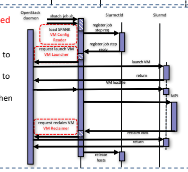
➢ SPANK Plugin based Approach

- VM Launcher – Setup VMs on each allocated nodes
  - File based lock to detect occupied VF and exclusively allocate free VF
  - Assign a unique ID to each IVSHMEM and dynamically attach to each VM
- Fast coordination
- High Scalability
- High Security



➢ SPANK Plugin over OpenStack based Approach

- VM Launcher, VM Reclaimer – Offload to underlying OpenStack infrastructure
  - PCI Whitelist to passthrough free VF to VM
  - Extend Nova to enable IVSHMEM when launching VM
- Easy Management
- Component Optimization
- High Scalability
- High Performance



Conclusion and Future Work

- The proposed framework enables users to build efficient HPC clouds on HPC clusters
- Provides efficient virtualization support for various cloud environments (VM, Docker Container, Singularity, Nested virtualization), high performance VM migration mechanism, and proper isolation and management for critical virtualized resource in resource management and job scheduling system
- In the future, we will propose efficient communication support for accelerator like GPUs in virtualization environment

References

[1]. **Jie Zhang**, Xiaoyi Lu and Dhabaleswar K. Panda,*High-PerformanceVirtualMachineMigrationFrameworkfor MPI Applications on SR-IOV enabled InfiniBand Clusters*, The 31st IEEE International Parallel and Distributed Processing Symposium (**IPDPS '17**), May 2017
[2]. **Jie Zhang**, Xiaoyi Lu and Dhabaleswar K. Panda, *Designing Locality and NUMA Aware MPI Runtime for Nested Virtualization based HPC Cloud with SR-IOV Enabled InfiniBand*, The 13th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (**VEE '17**), April 2017
[3]. **Jie Zhang**, Xiaoyi Lu, Sourav Chakraborty and Dhabaleswar K. Panda, *SLURM-V: Extending SLURM for Building Efficient HPC Cloud with SR-IOV and IVShmem*, The 22nd International European Conference on Parallel and Distributed Computing (**Euro-Par '16**), Aug 2016
[4]. **Jie Zhang**, Xiaoyi Lu and Dhabaleswar K. Panda, *High Performance MPI Library for Container-based HPC Cloud on InfiniBand Clusters*, The 45th International Conference on Parallel Processing (**ICPP '16**), Aug 2016
[5]. **Jie Zhang**, Xiaoyi Lu, Mark Arnold and Dhabaleswar K. Panda, *MVAPICH2 over OpenStack with SR-IOV: An Efficient Approach to Build HPC Clouds*, The 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, (**CCGrid '15**), May 2015
[6]. **Jie Zhang**, Xiaoyi Lu, Jithin Jose and Dhabaleswar K. Panda, *High Performance MPI Library over SR-IOV Enabled InfiniBand Clusters*, The International Conference on High Performance Computing (**HiPC'14**), Dec 2014
[7]. **Jie Zhang**, Xiaoyi Lu, Jithin Jose, Rong Shi and Dhabaleswar K. Panda, *Can Inter-VM Shmem Benefit MPI Applications on SR-IOV based Virtualized InfiniBand Clusters?*, The 20th International European Conference on Parallel and Distributed Computing (**Euro-Par '14**), Aug 2014

Software Distribution

- The designs proposed in this research are available for the community in MVAPICH2-Virt release http://mvapich.cse.ohio-state.edu/userguid/virt/
- Complex Appliances available on Chameleon Cloud https://www.chameleoncloud.org/appliances/29 (MPI bare-metal cluster) and https://www.chameleoncloud.org/appliances/28 (MPI+SRIOV KVM cluster)

Acknowledgements

Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu

Department of Computer Science and Engineering

The Ohio State University
College of Engineering

This research is supported in part by National Science Foundation grant #CCF-1519123, #ACI-1450440, #CNS-1513120, and #CCF-1565414.