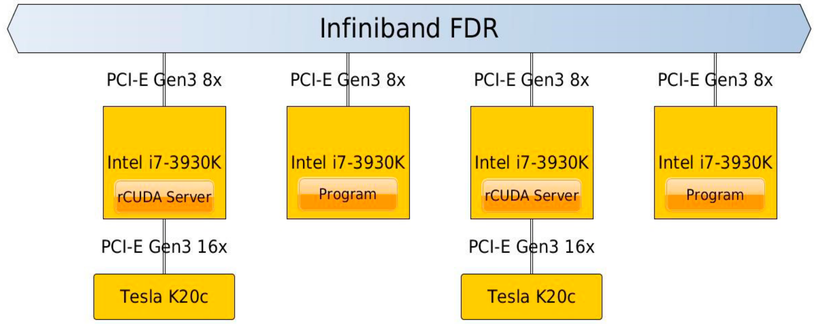
Preliminary Evaluation of GPU Virtualization in Supercomputer for Better Utilization

Pak Markthub[1], Michel Jansen[1,2] and Satoshi Matsuoka[1]

[1] Tokyo Institute of Technology, [2] Delft University of Technology

Background

- Heterogenous supercomputers such as TSUBAME2.5 use GPUs along with CPUs to speed up computation
- GPUs on compute nodes are left idle when the node is assigned to non GPU-enabled jobs
- Total system utilization can be significantly increased if remote nodes can use the idle GPUs
- rCUDA is a virtualization technology that allows idle GPUs to be presented to a remote node as virtual local GPU
- A rCUDA server process runs on the CPU and presents the idle GPU over the network to a remote node



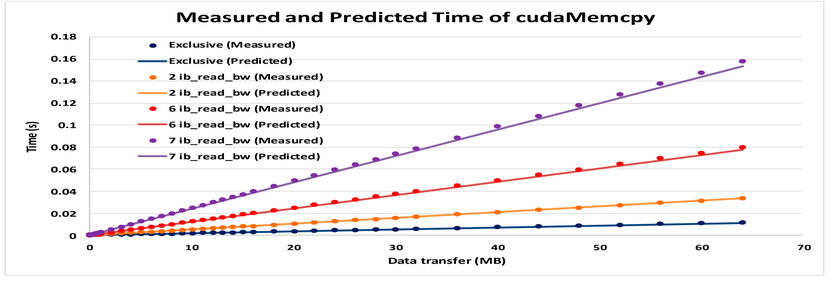
Environment Configuration

Objective

- Evaluate the effect of network contention on jobs that use rCUDA virtualization
- Evaluate the effect of limited CPU resource on nodes presenting the virtual GPUs

Network Contention Effect

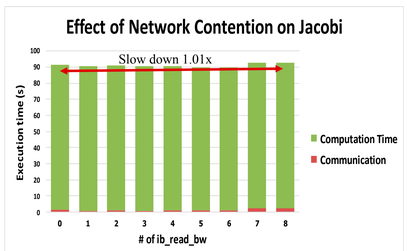
Does network contention affects the performance of applications using rCUDA?

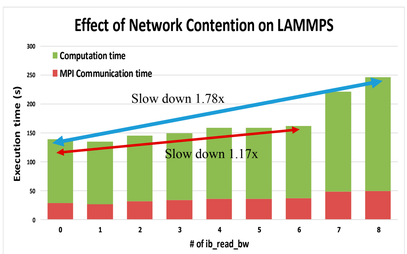


- Generate network contention using *ib_read_bw*
- Measured cudaMemcpy bandwidth using *bandwidthTest* from CUDA SDK

$$time_{rCUDA\ comm} = latency_{network} + latency_{rCUDA} + \frac{data\ size_{application}}{bandwidth_{network}} \cdot overhead_{rCUDA}$$

- Prediction using the equation above
- $overhead_{rCUDA}$ = 1.09 and $latency_{rCUDA}$ = 3.47 ms (from experiments)
- Using IB: $latency_{network}$ << $latency_{rCUDA}$
- Fitness of the model: R^2 = 0.986 (average from all tests)
- Network noise is generated using multiple processes of ib_read_bw



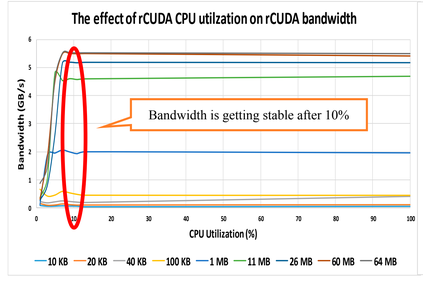


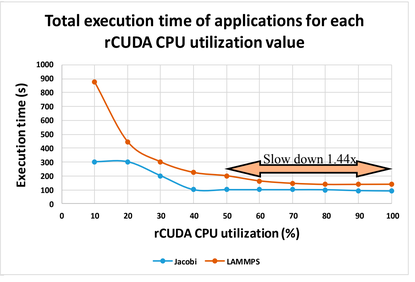
- Network contention doesn't have much impact on the execution time of Jacobi
  - Due to a small portion of time spending on GPU communication
- Use the equation to predict time spending on cudaMemcpy
  - Fitness of the model: R^2 = 0.882
- Network contention also has an impact on LAMMPS
  - More impact than Jacobi due to a lot of CUDA call

# of ib_read_bw	0	1	2	3	4	5	6	7	8
Avg.Bandwidth per process (GB/s)	6.38	3.08	2.04	1.2	1.22	1.02	0.886	0.446	0.386
Network latency (us)	5.97	12.2	12.8	13.6	19.2	18.4	17.0	29.0	108

CPU Contention Effect

Does CPU contention on rCUDA server affect the performance of applications using rCUDA?





- Pinned rCUDA server to 1 core
- Use *cpulimit* to limit rCUDA CPU utilization of that core

- rCUDA consumes about 10% of CPU time to handling cudaMemcpy requests
- Jacobi and LAMMPS suffer more from limiting CPU utilization of rCUDA
  - These programs have a lot of CUDA requests (not only cudaMemcpy)
  - rCUDA needs CPU to process request message header [1]
  - With less than 50% of available CPU time for rCUDA, the total execution time of the program with rCUDA will drastically increase.

Conclusion

- If minimum required bandwidth is available, the effect of network contention is neglegable
- rCUDA server only needs 50% utilization time of one core to service requests

Future Work

- Finding system utilization model that include the impact of rCUDA on other programs and vice versa.
- Finding a way to improve rCUDA by requiring less resources in order to reduce the impact of contention on shared resources.

References

[1] José Duato, Antonio J. Peña, Federico Silla, Rafael Mayo, and Enrique S. Quintana-Ortí. Performance of CUDA virtualized remote GPUs in high performance clusters.In Proceedings of the 2011 International Conference on Parallel Processing (ICPP 2011), Taipei, Taiwan, September 2011.

[2] Carlos Reaño, Rafael Mayo, Enrique S. Quintana-Ortí, Federico Silla, José Duato and Antonio J. Peña. Influence of InfiniBand FDR on the Performance of Remote GPU Virtualization. In Proceedings of the IEEE Cluster 2013 Conference, Indianapolis, IN (USA), September 2013