Improving the Efficiency of your GPU-accelerated Cluster with rCUDA

F. Silla
Technical University of Valencia
Spain
Improving application performance

• The complexity of current applications may cause their execution time to be extremely high

• There is the trend to accelerate parts of their code by using GPUs
GPU computing has noticeably grown

- Initially for video games
- Now pervasive

<table>
<thead>
<tr>
<th>Rank</th>
<th>Site</th>
<th>Computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RIKEN Advanced Institute for Computational Science (AICS) Japan</td>
<td>K computer, SPARC64 VIIfx 2.0GHz, Tofu interconnect / 2011 Fujitsu</td>
</tr>
<tr>
<td>2</td>
<td>National Supercomputing Center in Tianjin China</td>
<td>Tianhe-1A - NUDT TH MPP, X5670 2.93GHz 6C, nVidia GPU FT-1000 SC / 2010 NUDT</td>
</tr>
<tr>
<td>3</td>
<td>DOE/SC/Oak Ridge National Laboratory United States</td>
<td>Jaguar - Cray XT5-HE Opteron 6-core 2.6 GHz / 2009 Cray Inc.</td>
</tr>
<tr>
<td>4</td>
<td>National Supercomputing Centre in Shenzhen (NSCS) China</td>
<td>Nebulae - Dawning TC3600 Blade, Intel X5650, nVidia Tesla C2050 GPU / 2010 Dawning</td>
</tr>
<tr>
<td>5</td>
<td>GSIC Center, Tokyo Institute of Technology Japan</td>
<td>TSUBAME 2.0 - HP ProLiant SL390s G7 Xeon 6C X5670, nVidia GPU, Linux/Windows / 2010 NEC/HP</td>
</tr>
</tbody>
</table>
Building a node with GPUs

GPUs inside the CPU box

GPUs outside the CPU box

PCI-e
Building a cluster with GPUs

From the programming point of view:

- A set of nodes, each one with:
  - One or more CPU sockets (multiple cores per socket)
  - One or more GPUs
Software support for GPU computing

Two main approaches in GPU computing development environments:

• CUDA → nVidia proprietary
• OpenCL → open standard
Getting benefits from GPUs is not magic

GPUs only bring benefits for the right kind of code:

• There must be data parallelism in the code: this is the only way of taking benefit from the hundreds of processors in a GPU
  • Code with low amount of data parallelism
  • Code with high amount of data parallelism
  • Code with moderate amount of data parallelism
Getting benefits from GPUs is not magic

GPUs only bring benefits for the right kind of code:

- There must be data parallelism in the code: this is the only way of taking benefit from the hundreds of processors in a GPU
  - Code with low amount of data parallelism
  - Code with high amount of data parallelism
  - Code with moderate amount of data parallelism

BAD for GPU computing

No GPU is needed, just proceed with the tradiçional HPC strategies
Getting benefits from GPUs is not magic

GPUs only bring benefits for the right kind of code:

- There must be **data parallelism in the code**: this is the only way of taking benefit from the hundreds of processors in a GPU
  - Code with low amount of data parallelism
  - Code with high amount of data parallelism
  - Code with moderate amount of data parallelism

**GOOD for GPU computing**

Add several GPUs to each node in the system and rewrite the applications to use them
Getting benefits from GPUs is not magic

GPUs only bring benefits for the right kind of code:

- There must be **data parallelism in the code**: this is the only way of taking benefit from the hundreds of processors in a GPU
  - Code with low amount of data parallelism
  - Code with high amount of data parallelism
  - Code with moderate amount of data parallelism
    
    - Application has a moderate level of data parallelism, typically between 40% and 80%
For applications with moderate levels of data parallelism, the GPUs of a CUDA-enabled cluster may be idle for long periods of time.

- **Waste** of resources and energy
  - Electricity: GPUs keep consuming power
  - Space: GPUs reduce CPU density

Leak of money in current clusters?
Looking for an efficient solution

- A way of avoiding this inefficiency is by reducing the number of GPUs and sharing the remaining ones among the CPU nodes in the cluster.
- This would increase GPU utilization also reducing power consumption.
- Saving costs by doing better.
Looking for an efficient solution

- A way of avoiding this inefficiency is by reducing the number of GPUs and sharing the remaining ones among the CPU nodes in the cluster.
- This would increase GPU utilization also reducing power consumption.
- Saving costs by doing better:
  - Doing better by spending less money in GPUs at the initial investment and therefore reducing TCO.
Looking for an efficient solution

- A way of avoiding this inefficiency is by reducing the number of GPUs and sharing the remaining ones among the CPU nodes in the cluster.

- This would increase GPU utilization also reducing power consumption.

- Saving costs by doing better
  - Doing better by **spending less money** in GPUs at the initial investment and therefore reducing TCO.
  - Doing better by **deploying rCUDA** into your new cost-effective cluster.
rCUDA makes more flexible your GPU cluster

- rCUDA allows **having less GPUs** than nodes in the cluster
  - **Add only the required GPUs** giving you the necessary computational power!
rCUDA makes more flexible your GPU cluster

• rCUDA allows **having less GPUs** than nodes in the cluster
  - **Add only the required GPUs** giving you the necessary computational power!

• rCUDA **makes them accessible** from every node
rCUDA further improves application performance

• rCUDA also helps you **providing more GPUs** to a given application

• **Currently**, an application can make use of **6 GPUs**
rCUDA further improves application performance

- rCUDA also helps you **providing more GPUs** to a given application
- **Currently**, an application can make use of **6 GPUs**
- rCUDA makes **all GPUs visible** to a given application
rCUDA is a middleware that enables **seamlessly remote CUDA usage**

- The rCUDA **client** at every node
- The rCUDA **server** only on those nodes having a GPU

**Client side**
- Application
- rCUDA library
- Network interface

**Server side**
- rCUDA daemon
- Network interface
- CUDA library
How rCUDA works

• rCUDA is a middleware that enables **seamlessly** remote CUDA usage
  
  • The rCUDA **client** at every node
  
  • The rCUDA **server** only on those nodes having a GPU
Infiniband is the perfect partner for rCUDA

Why InfiniBand?

- InfiniBand is the most used HPC network
- Low latency and high bandwidth
- High performance can be expected
Execution time for a matrix-matrix product

(matrix dimension = 4096)

- GeForce 9800 GTX
- Intel Xeon E5645
Remember

• rCUDA adds value to your cluster:
  • makes it more flexible
  • use only the GPUs you really need
    • less … or more
  • reduces power consumption
  • has almost no impact on performance
• rCUDA makes you save money