Improving the Efficiency of your GPU-accelerated Cluster with rCUDA

F. Silla
Technical University of Valencia
Spain
Make your GPUs flexible!
Detach, share, aggregate them!
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.
Building a node with GPUs

GPUs inside the CPU box

GPUs outside the CPU box

GPUs

Main Memory

CPU

Network

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
  • Applications can only use the GPUs in their node
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** amounts of data parallelism
  • Code with **high** amounts of data parallelism
  • Code with **moderate** amounts 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 amounts of data parallelism
  • Code with high amounts of data parallelism
  • Code with moderate amounts of data parallelism

BAD for GPU computing

No GPU is needed, just proceed with the traditional 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 amounts of data parallelism
  - Code with moderate amounts 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 amounts of data parallelism
  - Code with high amounts of data parallelism
  - Code with moderate amounts 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
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

**Diagram:**

Detach and share GPUs!
rCUDA further improves application performance

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

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

- rCUDA also helps you providing more GPUs to a given application
  - Currently, an application can make use of a few GPUs
  - rCUDA makes all GPUs visible to a given application
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
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

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

**Server side**
- rCUDA daemon
- Network interface
- CUDA library
- **device**
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 = 13824)
Execution time for the LAMMPS application

(scaled by a factor of 5)

LAMMPS: Molecular dynamics simulator
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
• rCUDA is available for free:

www.rcuda.net

• Binary compatible with your current code
• Supports CUDA 5
• Linux and Windows platforms