GPU VIRTUALIZATION
WITH POWER PROCESSOR

Federico Silla
Universitat Politècnica de València
REMOTE GPU VIRTUALIZATION WITH POWER PROCESSOR

Federico Silla
Universitat Politècnica de València
Heterogeneous Clusters
Heterogeneous$^2$ Clusters
What is remote GPU virtualization?
Basics of GPU computing

Remark:
GPUs can only be used within the node they are attached to

Basic behavior of CUDA

Application

CUDA libraries

GPU
Remark:
GPUs can only be used within the node they are attached to.
A different approach: remote GPU virtualization

A software technology that enables a more flexible use of GPUs in computing facilities

No GPU

Network

rCUDA … remote CUDA

rCUDA is a development by Universitat Politecnica de Valencia
Basics of rCUDA

Access to remote GPU is transparent to applications: no source code modification is needed.

rCUDA is a development by Universitat Politecnica de Valencia.
Basics of rCUDA

Access to remote GPU is transparent to applications: no source code modification is needed.

rCUDA is a development by Universitat Politecnica de Valencia.
rCUDA envision

- **rCUDA allows a new vision** of a GPU deployment, moving from the usual cluster configuration …

… to the following one:
Heterogeneous$^2$ environments
rCUDA availability

rCUDA is available for the x86, POWER and ARM processors.
Performance of rCUDA on POWER systems
Several testbeds used

CUDA

rCUDA client

#1

rCUDA server

#2

network fabric is EDR InfiniBand

#3

x86 @ 2.1 GHz
DDR3 1600 MHz

x86 @ 3.5 GHz
DDR4 2400 MHz
Performance of data movements to/from GPU

CUDA

rCUDA
Performance of data movements to/from GPU

- **H2D**
  - CUDA Minsky
  - rCUDA Minsky-EDR-Minsky
  - rCUDA x86 @ 2.1GHz -EDR-Minsky
  - rCUDA x86 @ 3.5GHz -EDR-Minsky

- **D2H**
  - CUDA Minsky
  - rCUDA Minsky-EDR-Minsky
  - rCUDA x86 @ 2.1GHz -EDR-Minsky
  - rCUDA x86 @ 3.5GHz -EDR-Minsky

Higher is better
Performance of data movements among GPUs

Higher is better
Application performance

Several applications have been analyzed in this study:

1. BarraCUDA
2. CUDA-MEME
3. CUDASW++
4. GPU-Blast
5. Gromacs
6. GPU-LIBVSM
7. Magma
8. NAMD
Unfortunately, we could not run all the applications in the Minsky system:

1. BarraCUDA: this application includes intrinsics headers
2. CUDA-MEME: successfully compiled and executed
3. CUDASW++: this application includes intrinsics headers
4. GPU-Blast: we were not able to compile it
5. Gromacs: successfully compiled and executed
6. GPU-LIBVSM: successfully compiled and executed
7. Magma: successfully compiled and executed
8. NAMD: we were not able to compile it
Application performance

Graph showing execution times for various applications and configurations:
- **MAGMA**
- **GPU-LIBSVM (1)**
- **GPU-LIBSVM (2)**
- **CUDA-MEME**
- **GROMACS**

**CUDA Minsky**
**Minsky-EDR-Minsky**
**rCUDA x86 @ 2.1GHz -EDR-Minsky**
**rCUDA x86 @ 3.5GHz -EDR-Minsky**
Throughput instead of performance
One rCUDA box serves multiple clients
One rCUDA box serves multiple clients

1. BarraCUDA
2. CUDA-MEME
3. CUDASW++
4. GPU-Blast
5. Gromacs
6. GPU-LIBVSM
7. Magma
Get a free copy of rCUDA at http://www.rcuda.net
More than 900 requests world wide

@rcuda_

rCUDA is a development by Universitat Politècnica de València, Spain
Funded by Agencia Valenciana de la Innovación, Generalitat Valenciana