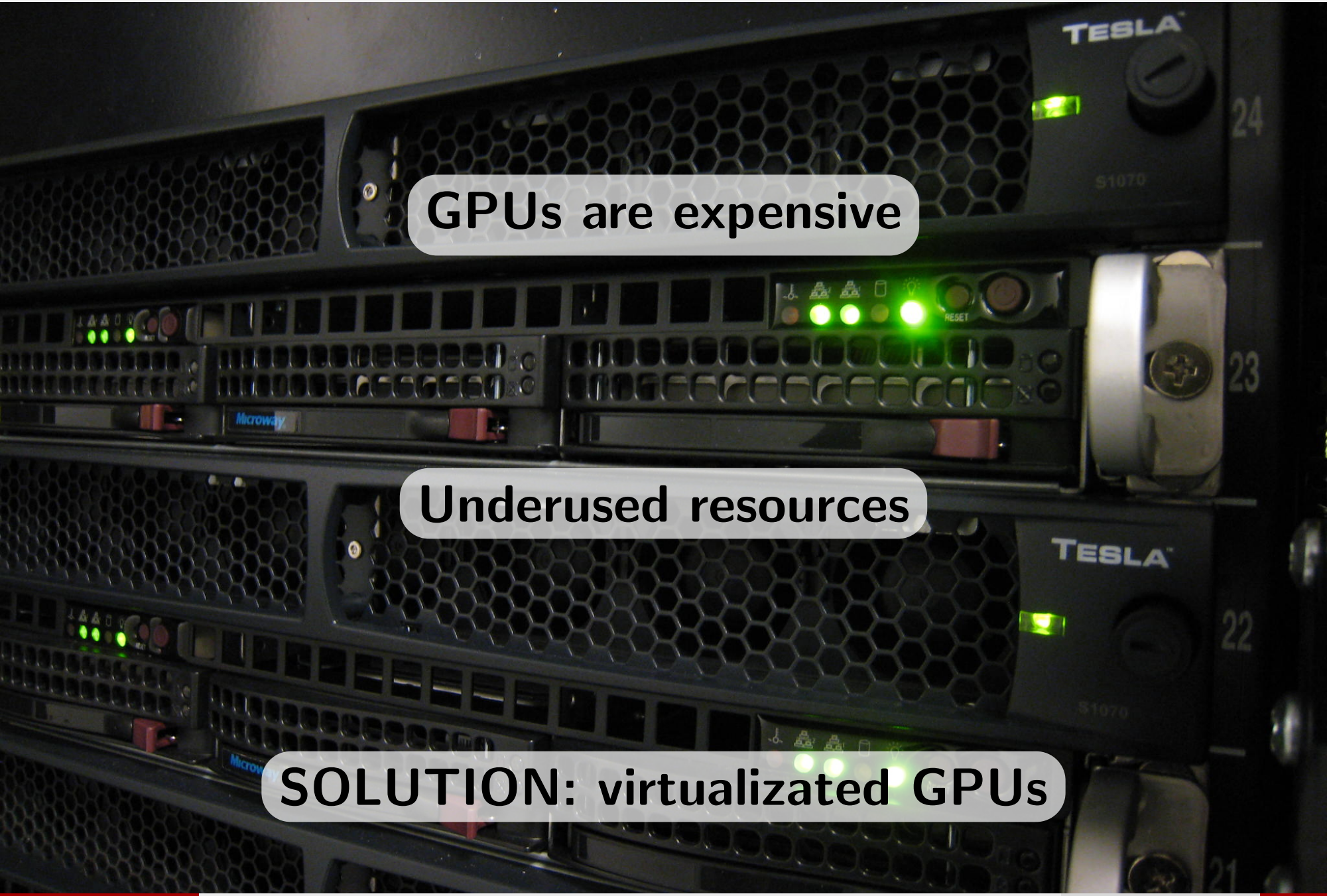
SLURM Support for Remote GPU Virtualization: Implementation and Performance Study

Sergio Iserte, Adrián Castelló, Rafael Mayo,
Enrique S. Quintana-Ortí, Federico Silla,
Jose Duato, Carlos Reaño, Javier Prades

SBAC-PAD 2014

24 October 2014, Paris (France)

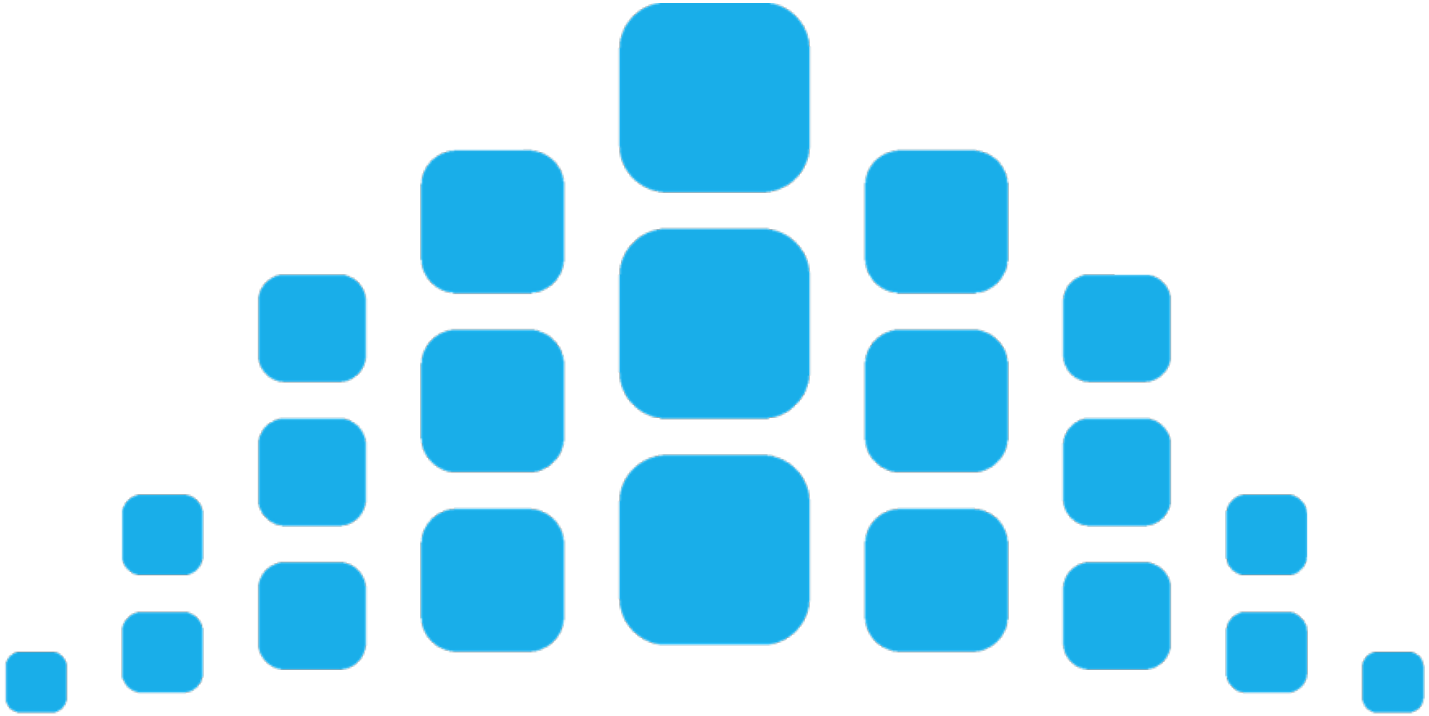
Introduction



GPUs are expensive

Underused resources

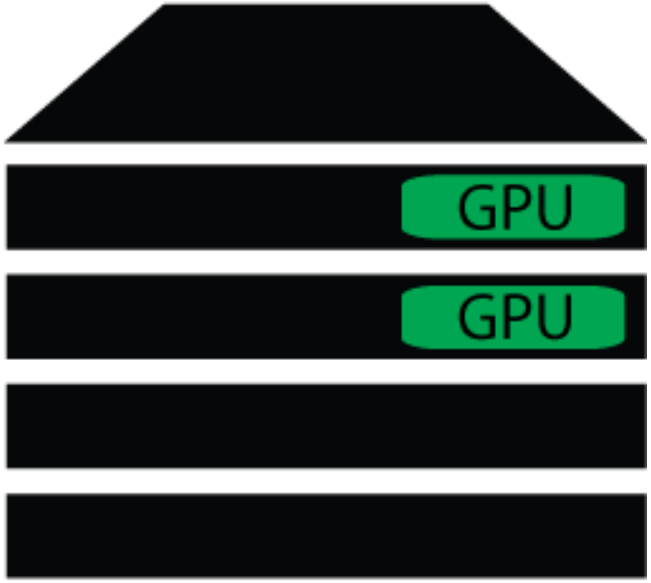
SOLUTION: virtualizated GPUs



slurm

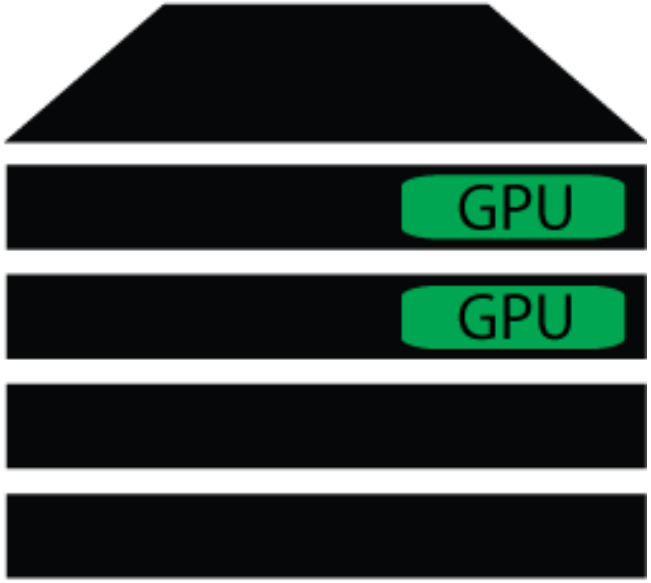
workload manager

Resources Management

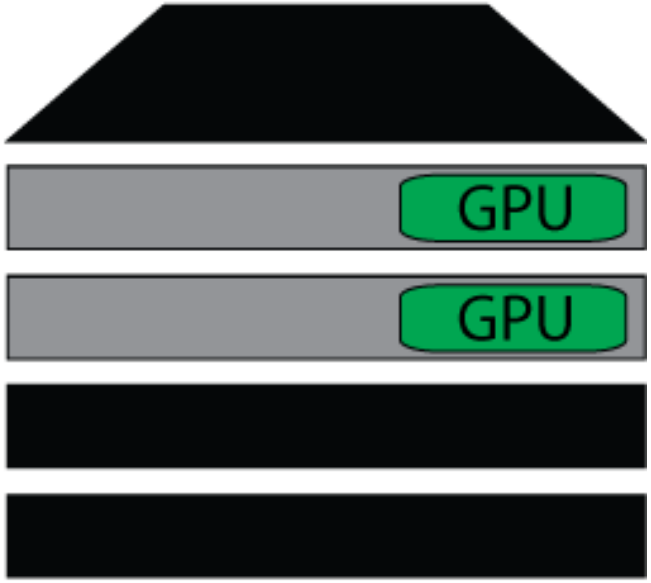


Resources Management

2 nodes

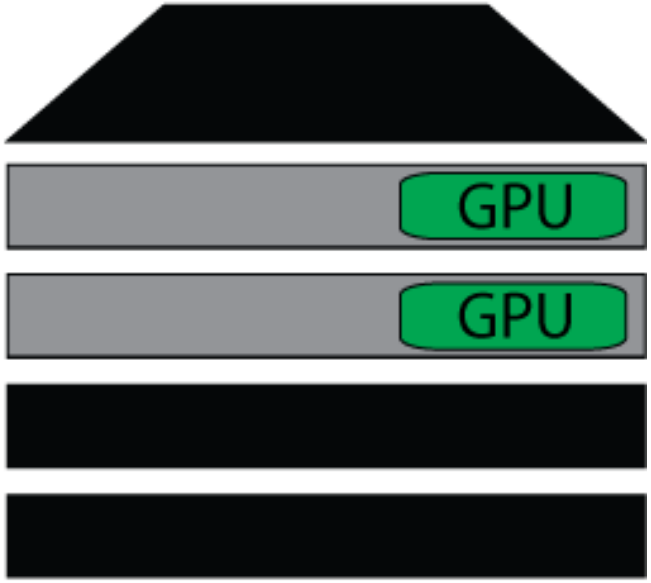


Resources Management



Resources Management

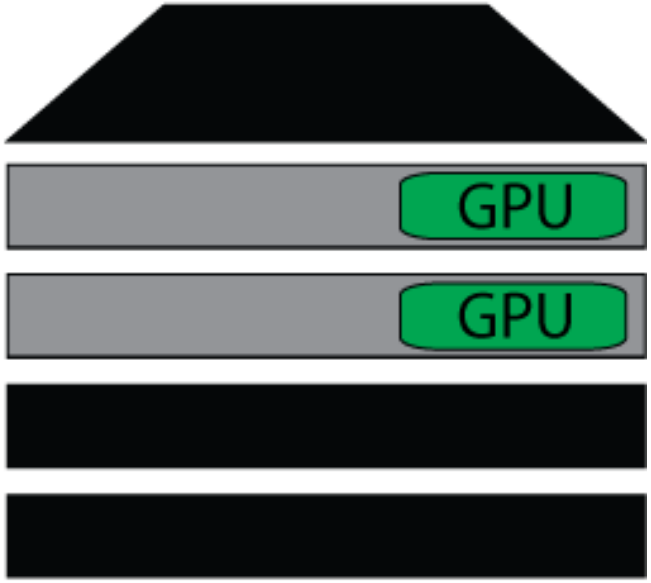
2 nodes
2 GPUs



Resources Management

2 nodes
2 GPUs

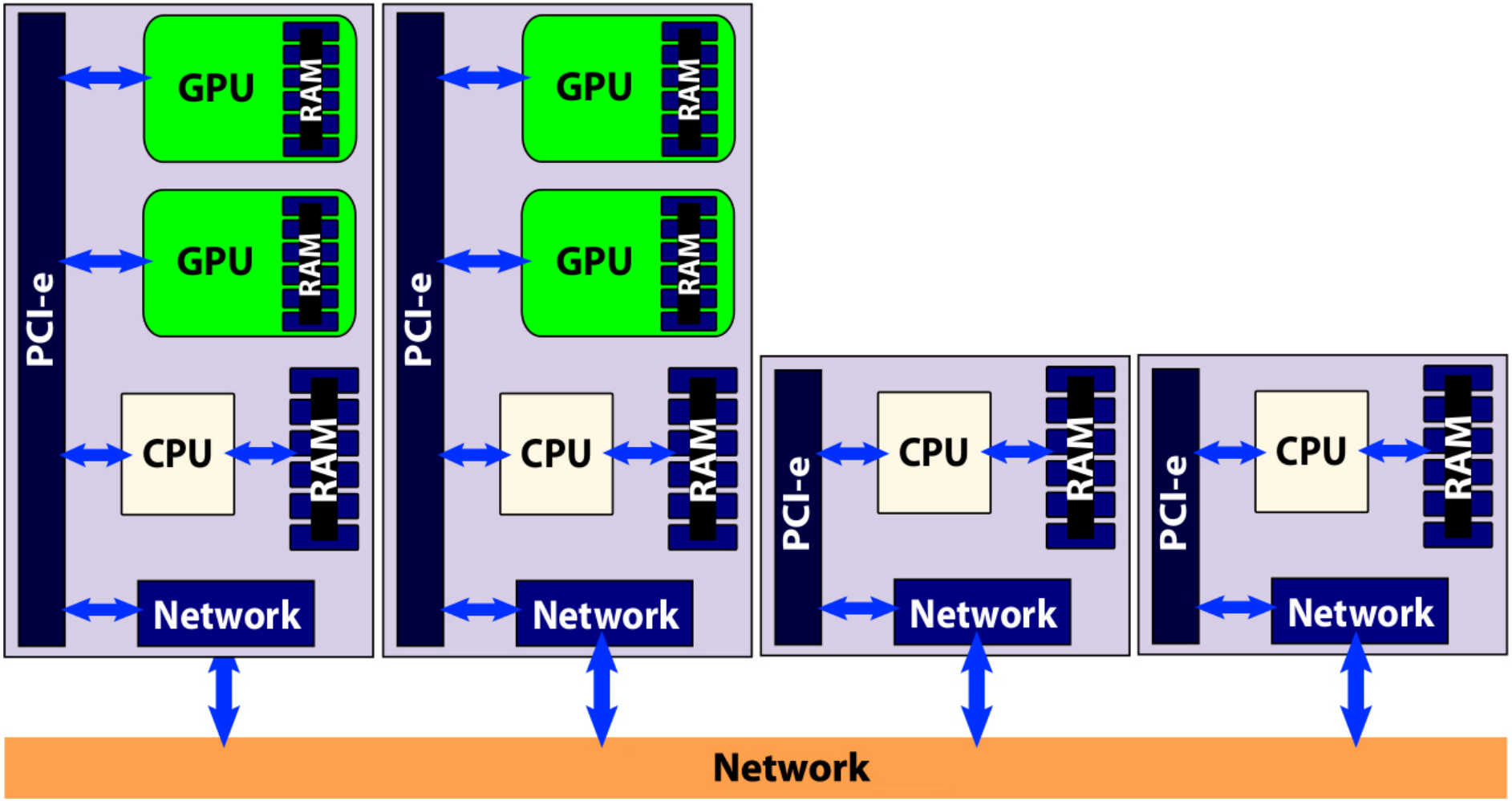
???



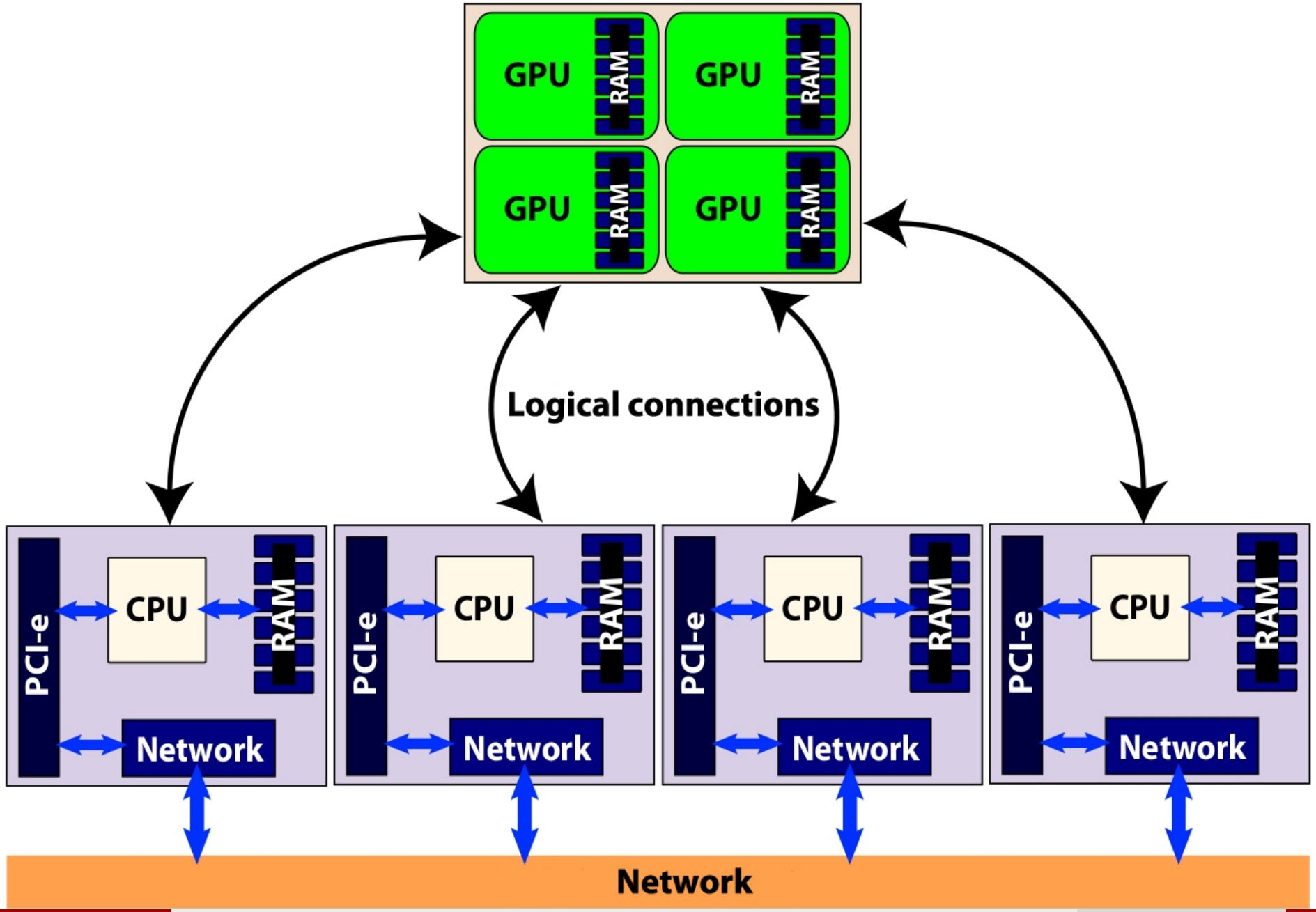
rCUDA

remote CUDA

Cluster without GPU Virtualization

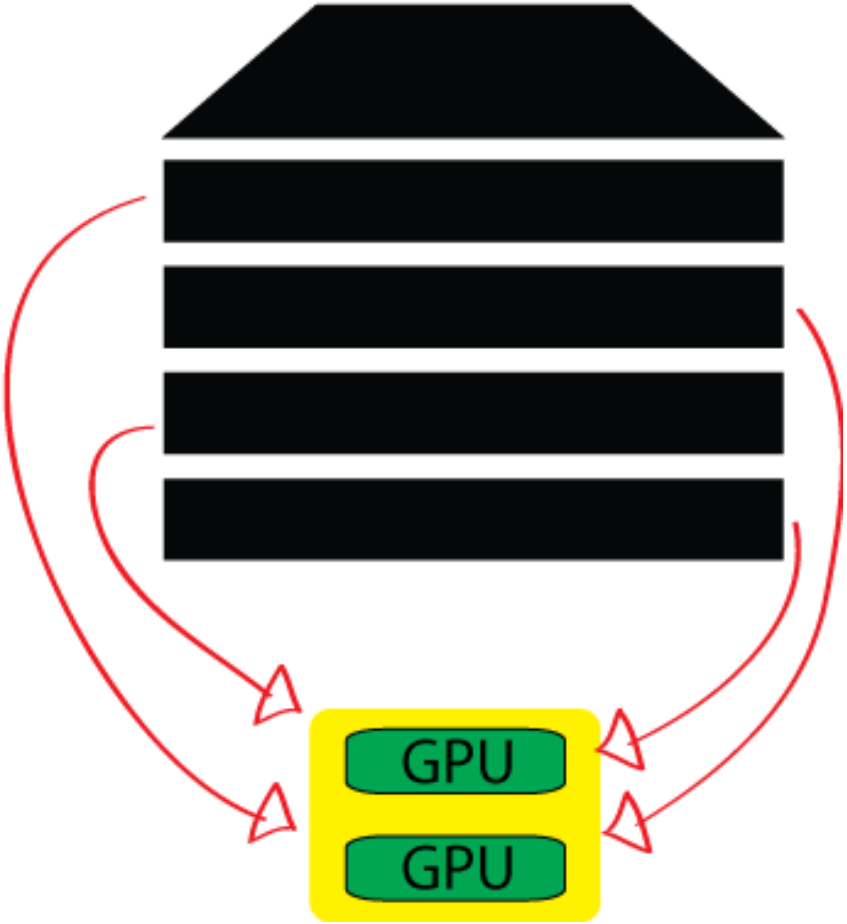


Cluster with rCUDA

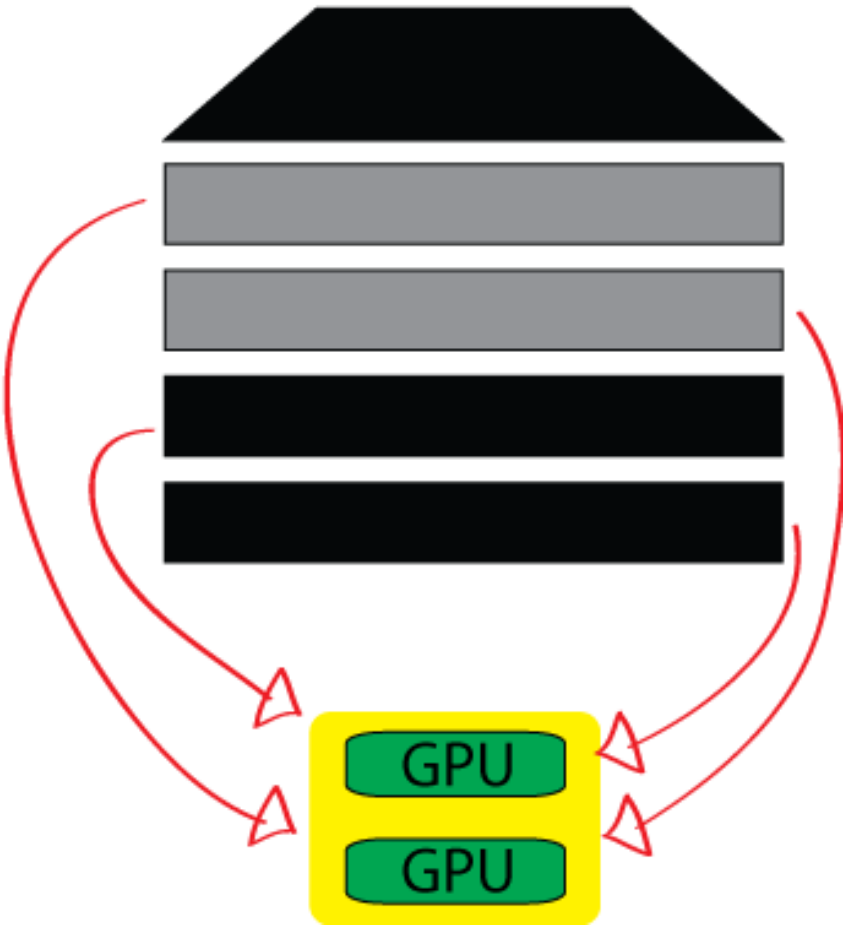


Virtualized Resources Management

2 nodes

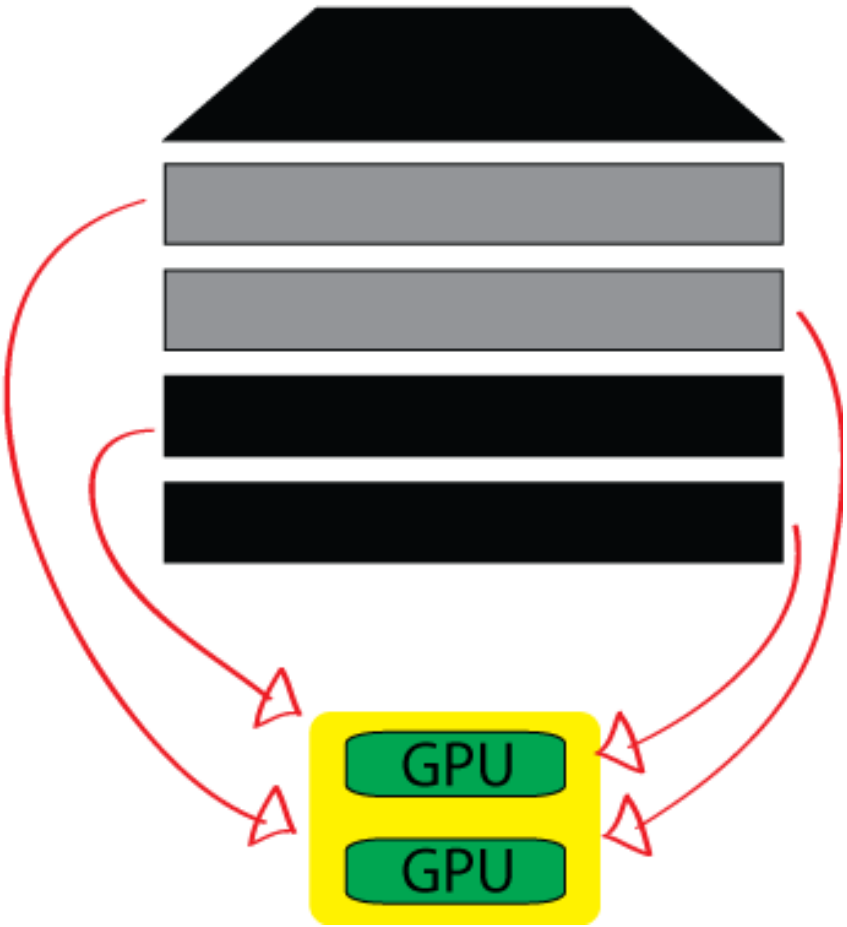


Virtualized Resources Management

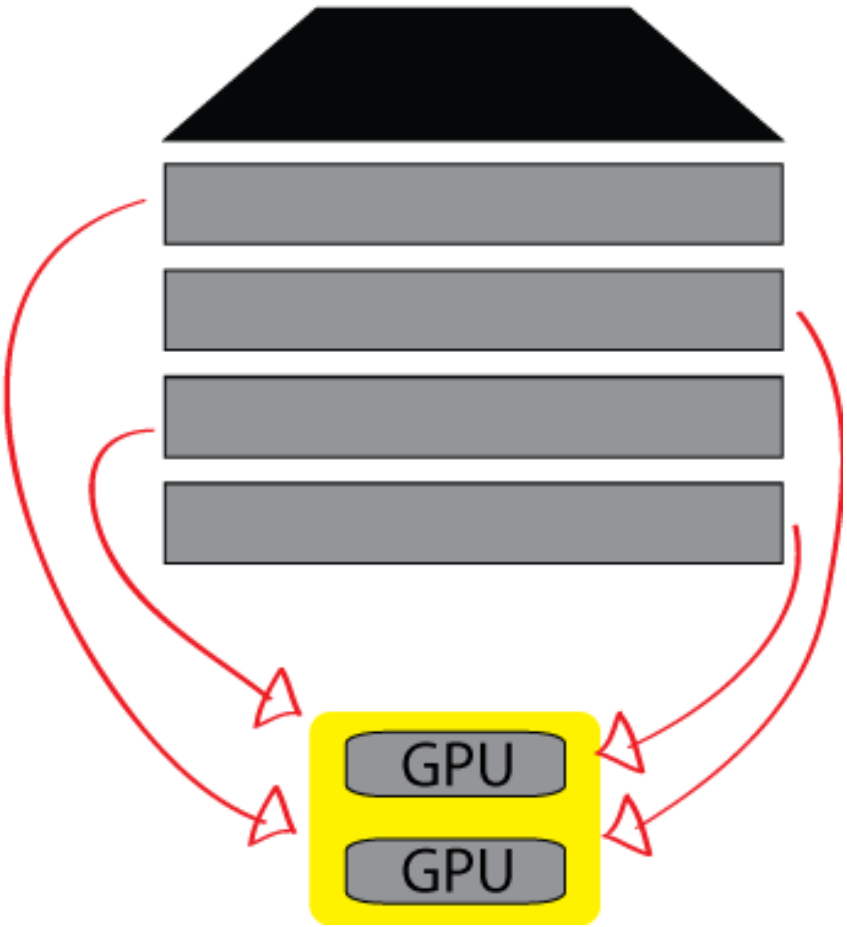


Virtualized Resources Management

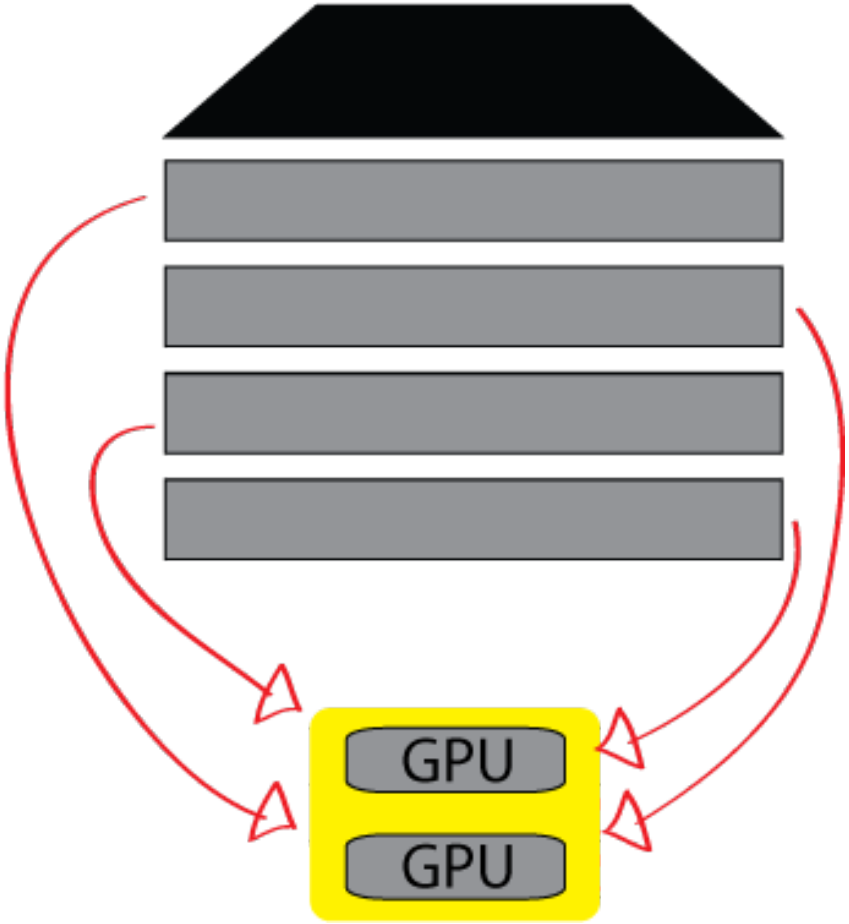
2 nodes
2 GPUs



Virtualized Resources Management



Virtualized Resources Management

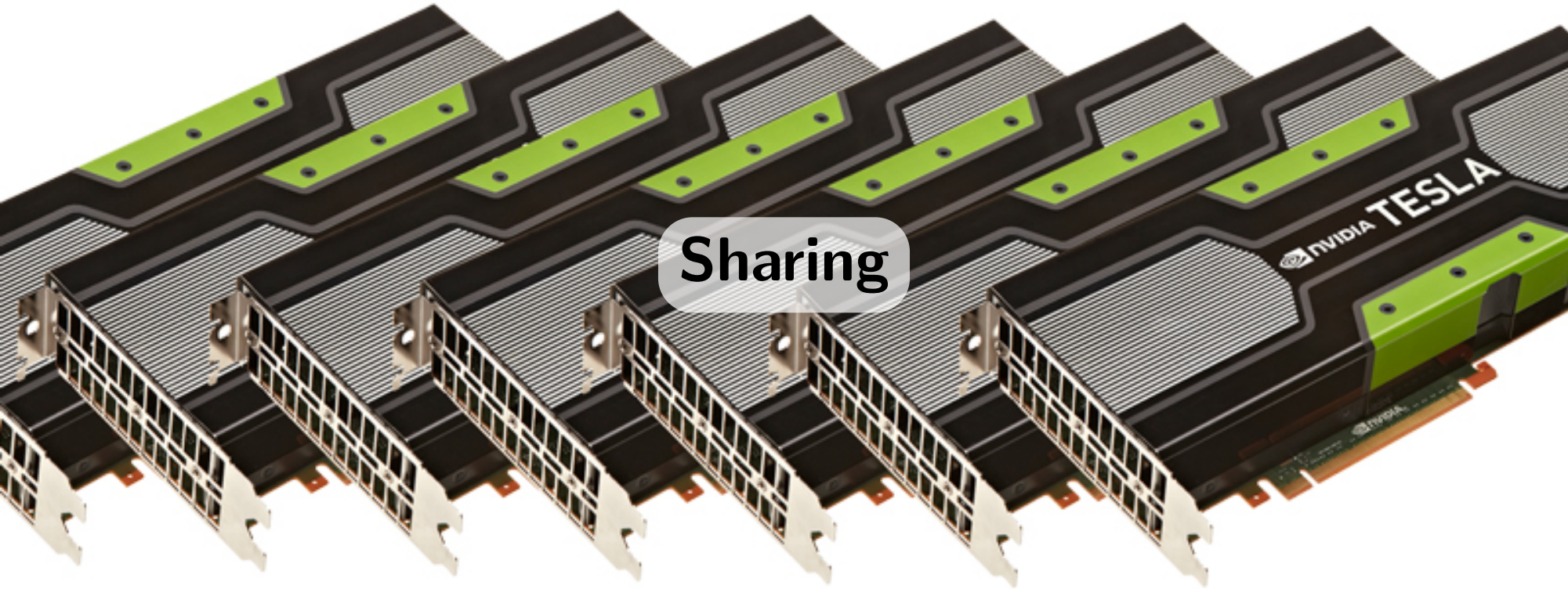


slurm & rCUDA

workload manager

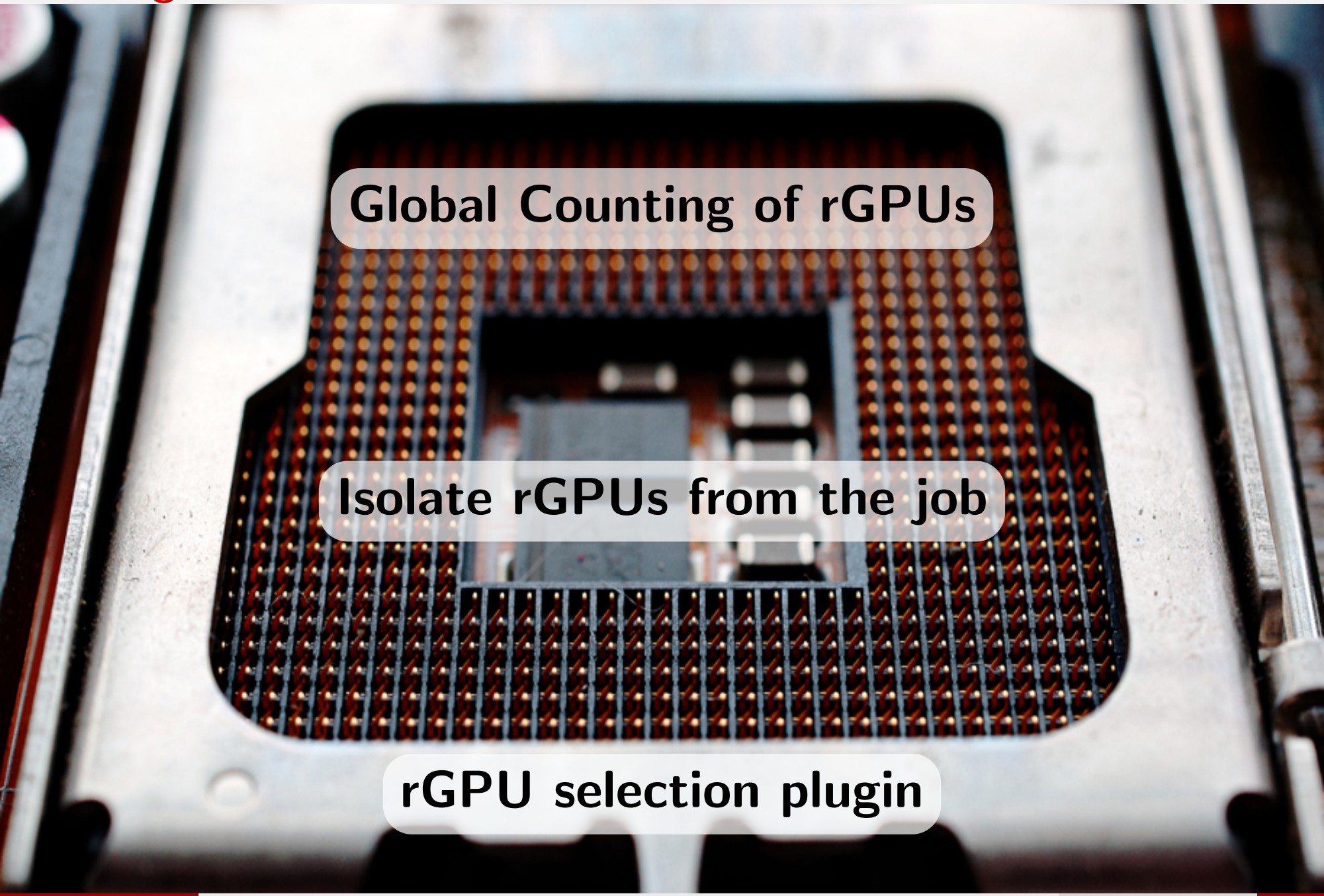
remote CUDA

Special resource: rGPU



Remote Usage

Management daemon *slurmctld*

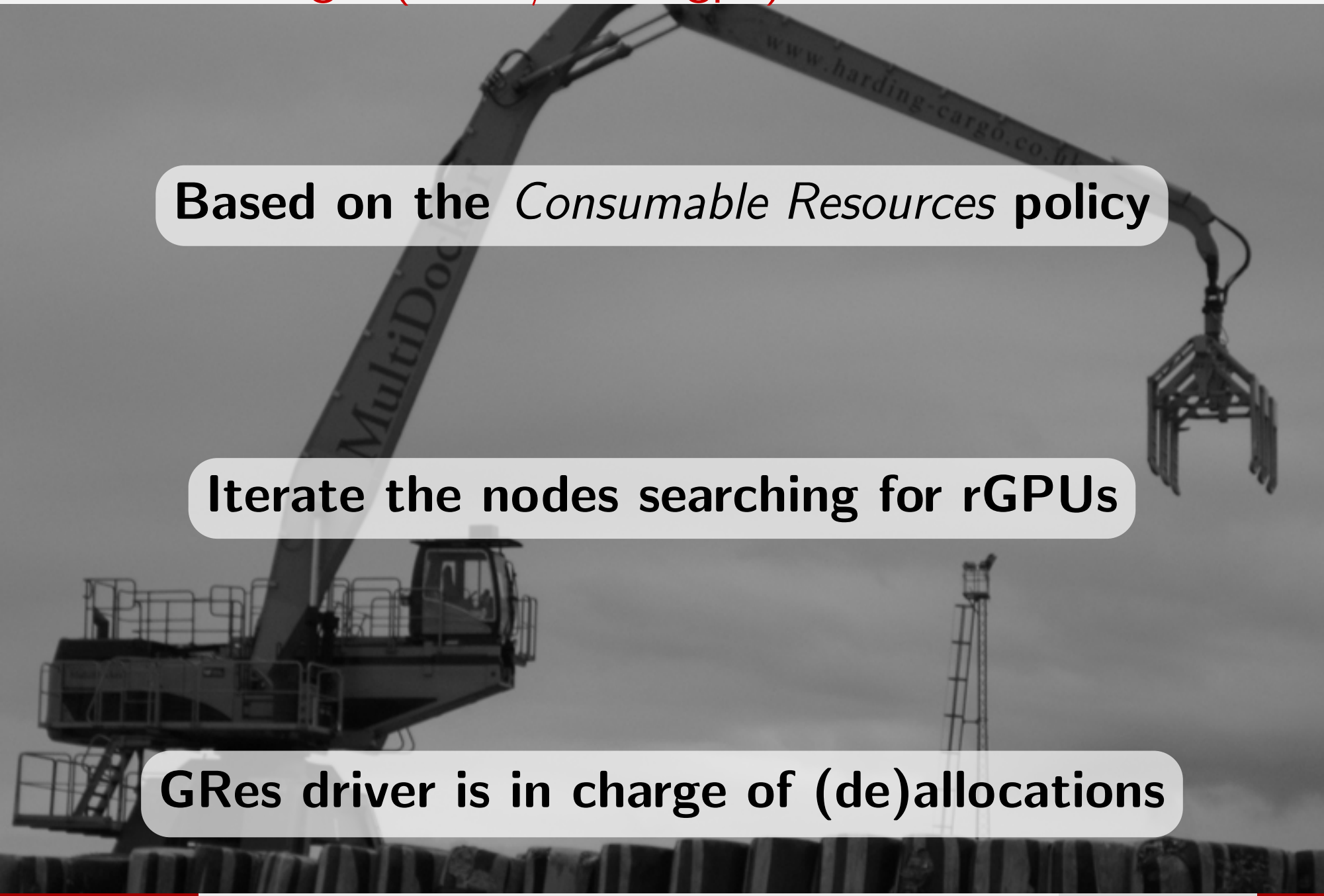


Global Counting of rGPUs

Isolate rGPUs from the job

rGPU selection plugin

Selection Plugin (select/cons_rgpu)



Based on the *Consumable Resources* policy

Iterate the nodes searching for rGPUs

GRes driver is in charge of (de)allocations

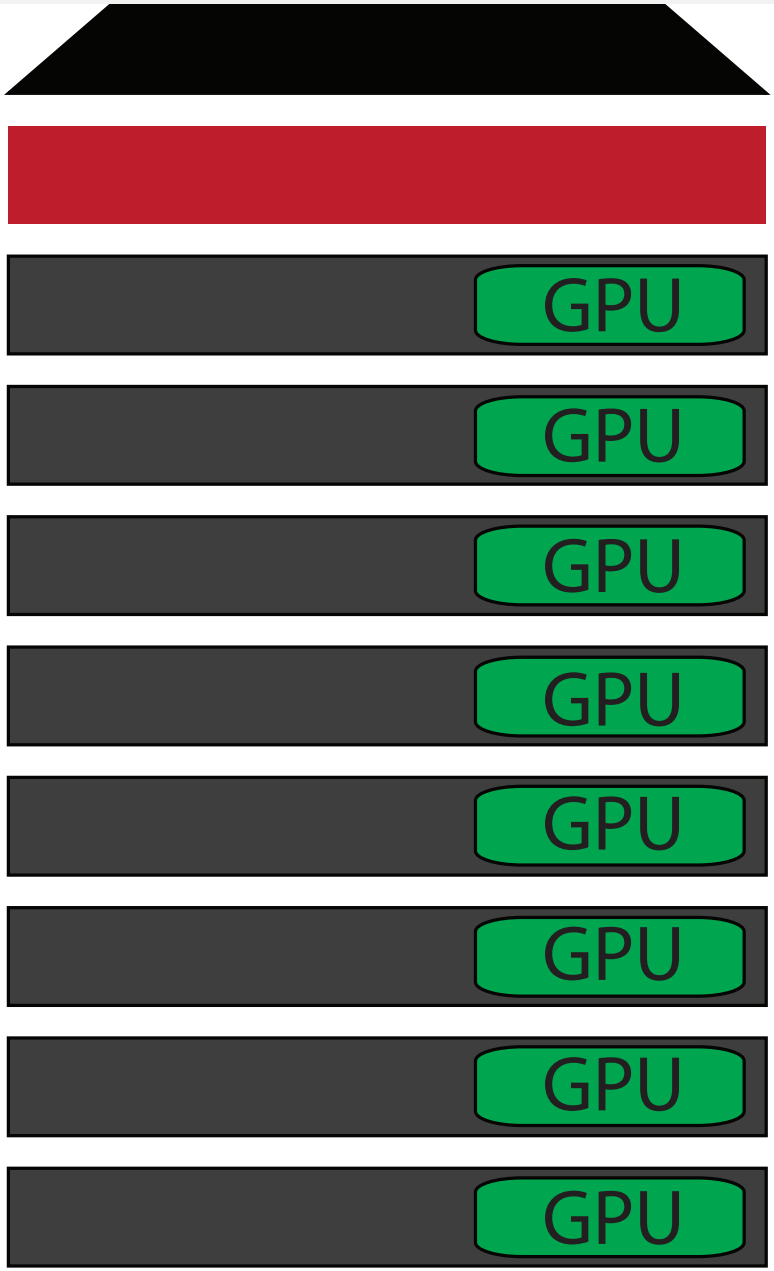
Environment Variables

srun, sbatch, salloc

```
--rcuda-mode=(shared|excl)
--gres=rgpu(:X(:Y)?(:Z)?)?
            X = [1-9]+[0-9]*
            Y = [1-9]+[0-9]*[ kKmMgG]
            Z = [1-9]\.[0-9](cc|CC)
```

experimentation

Platform



CentOS 6.4

2 x Intel Xeon E5-2620 Hexacore

NVIDIA Tesla K20 GPU

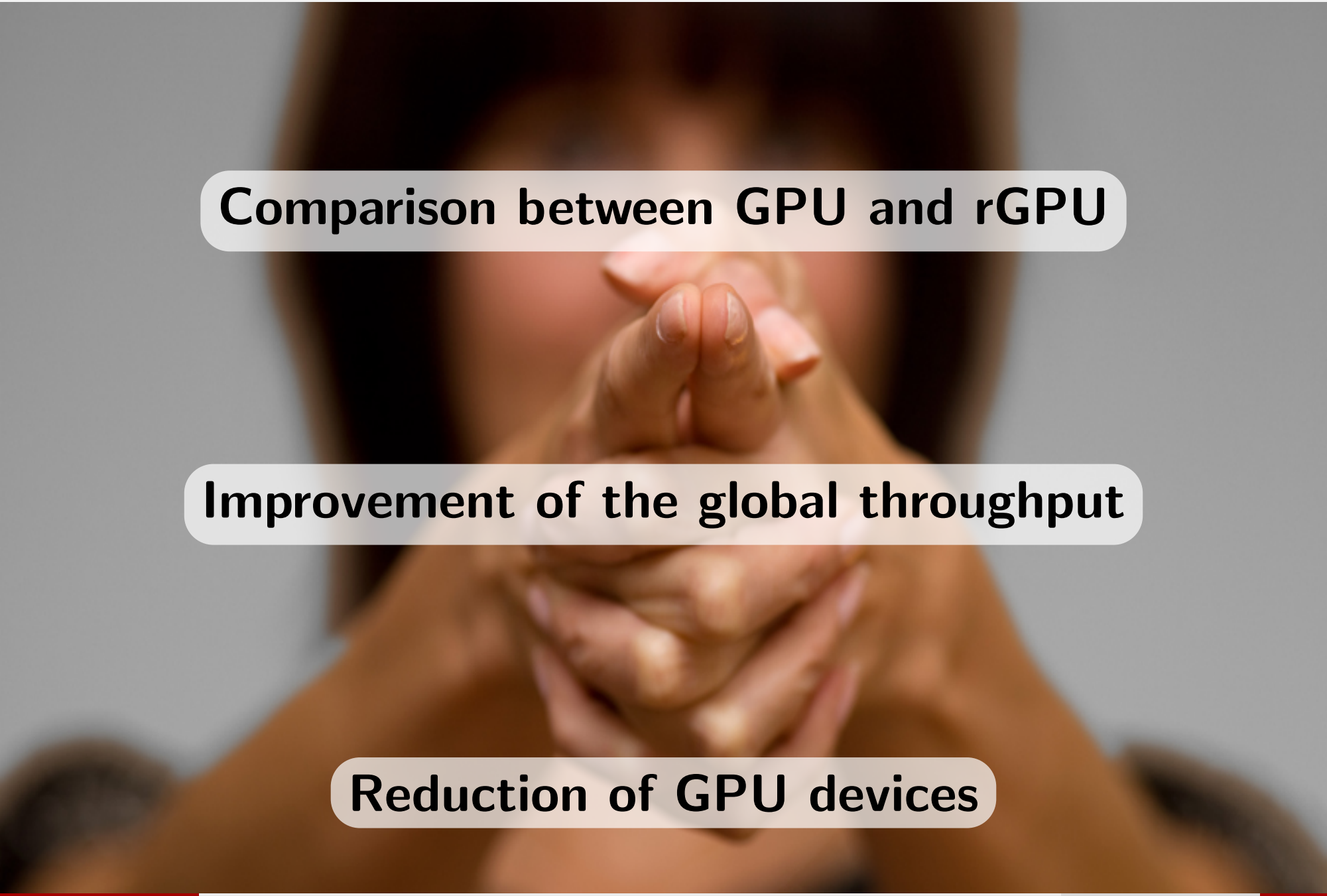
Mellanox SX6025 (FDR)

Applications

Application	Multi-process	Multi-thread	GPU Computational Load
GPU-Blast	-	X	Medium
LAMMPS	X	-	High
MCUDA-MEME	X	X	Medium
GROMACS	X	X	None, only CPU

Workloads

Goals



Comparison between GPU and rGPU

Improvement of the global throughput

Reduction of GPU devices

Maximum Performance

Application	Nodes	Processes	Threads	Generic Resources	
				CUDA	rCUDA
GPU-Blast	1	1	6	gpu:1	rgpu:1:1686M
LAMMPS	5	5	1	gpu:1	rgpu:5:3275M
MCUDA-MEME	4	4	1	gpu:1	rgpu:4:163M
GROMACS	2	2	12	-	-

New approach

High **Performance** *Computing*

HPC $\rightarrow$ **HTC**

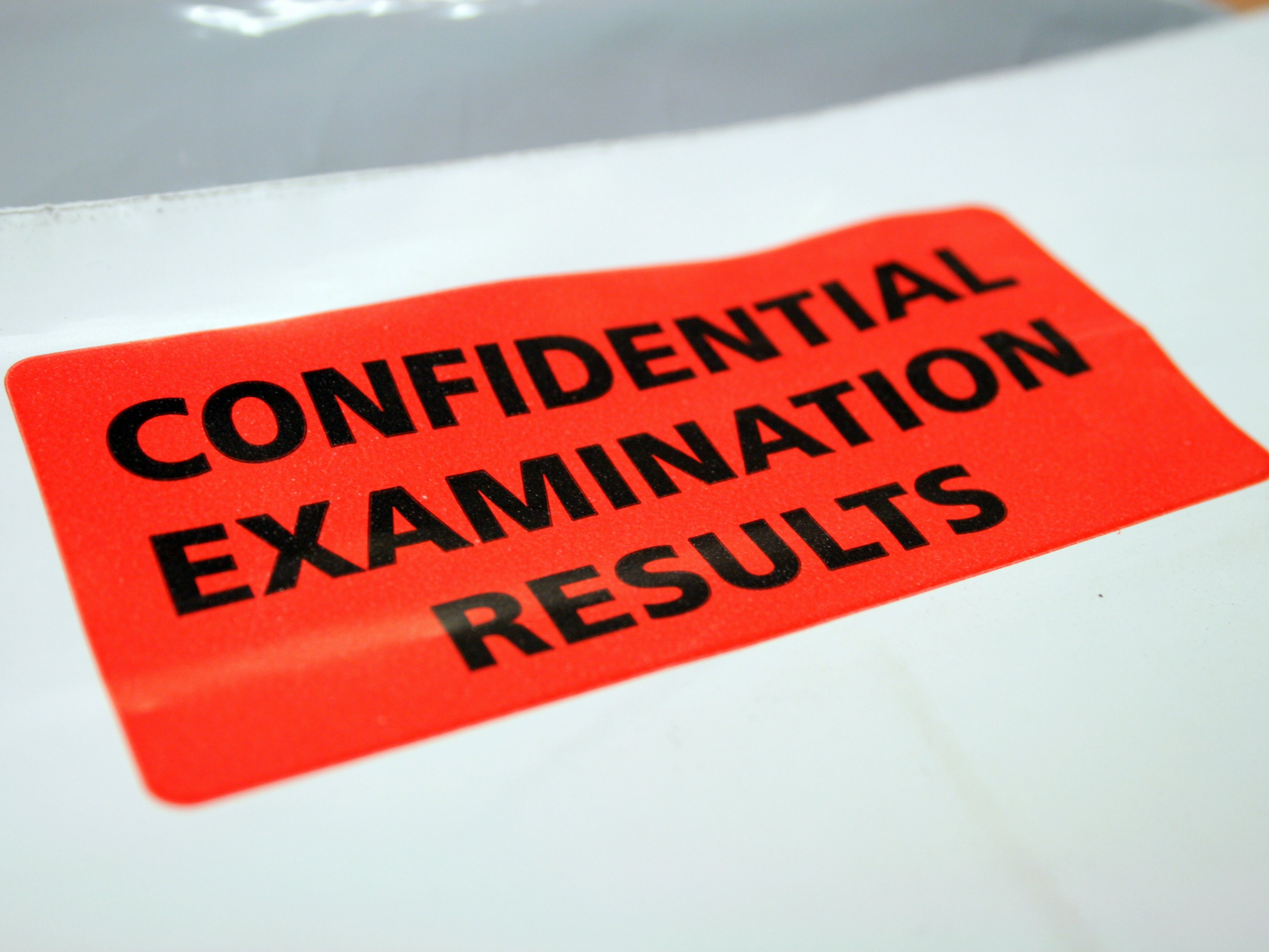
High **Throughput** *Computing*

Maximum Throughput

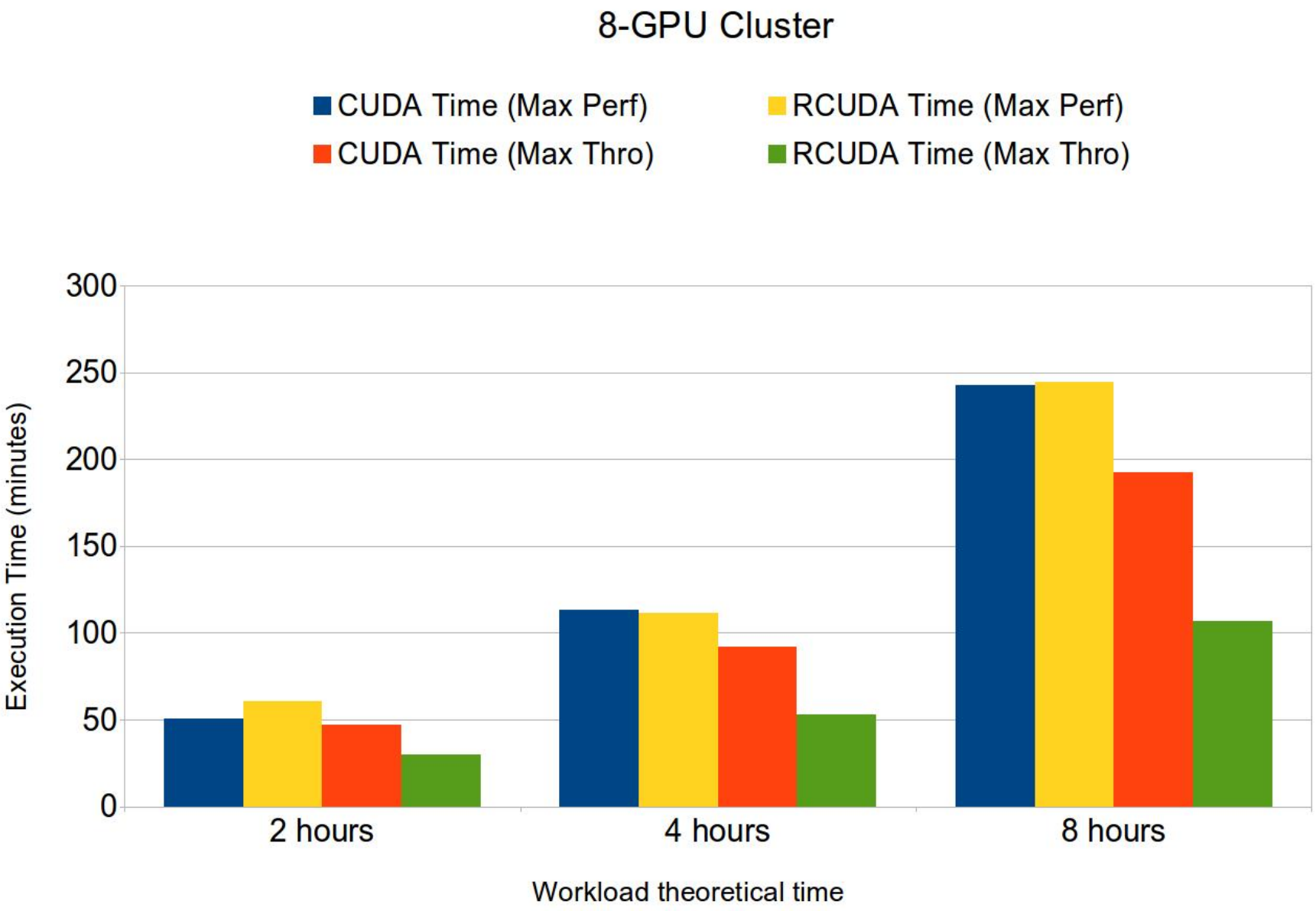
Application	Nodes	Processes	Threads	Generic Resources	
				CUDA	rCUDA
GPU-Blast	1	1	6	gpu:1	rgpu:1:1686M
LAMMPS	~~5~~	5	1	gpu:1	rgpu:5:3275M
MCUDA-MEME	4	4	1	gpu:1	rgpu:4:163M
GROMACS	2	2	12	-	-

Maximum Throughput

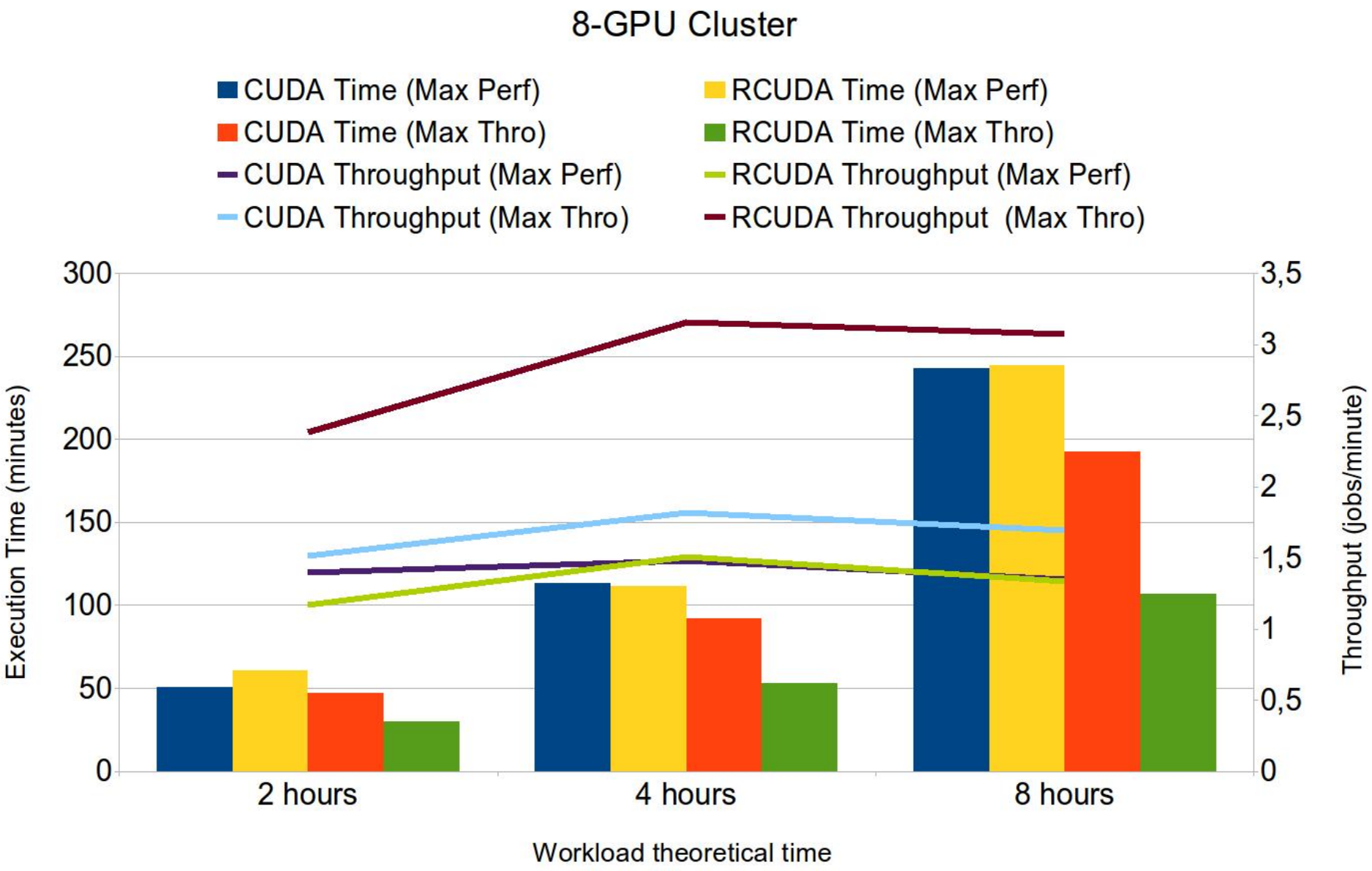
Application	Nodes	Processes	Threads	Generic Resources	
				CUDA	rCUDA
GPU-Blast	1	1	6	gpu:1	rgpu:1:1686M
LAMMPS	~~5~~	5	1	gpu:1	rgpu:5:3275M
MCUDA-MEME	~~4~~	4	1	gpu:1	rgpu:4:163M
GROMACS	2	2	12	-	-


CONFIDENTIAL
EXAMINATION
RESULTS

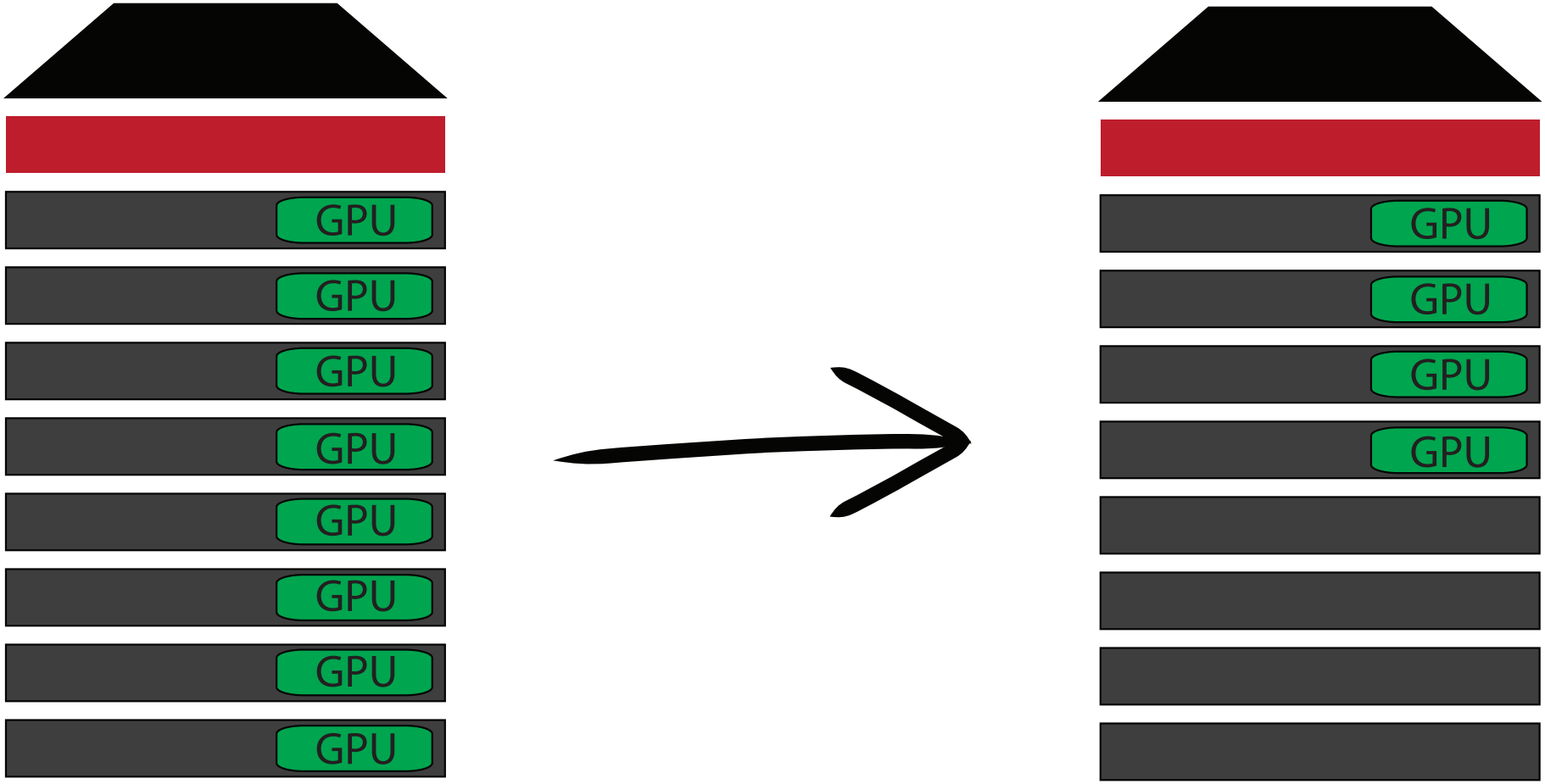
Max. Performance VS Max. Throughput



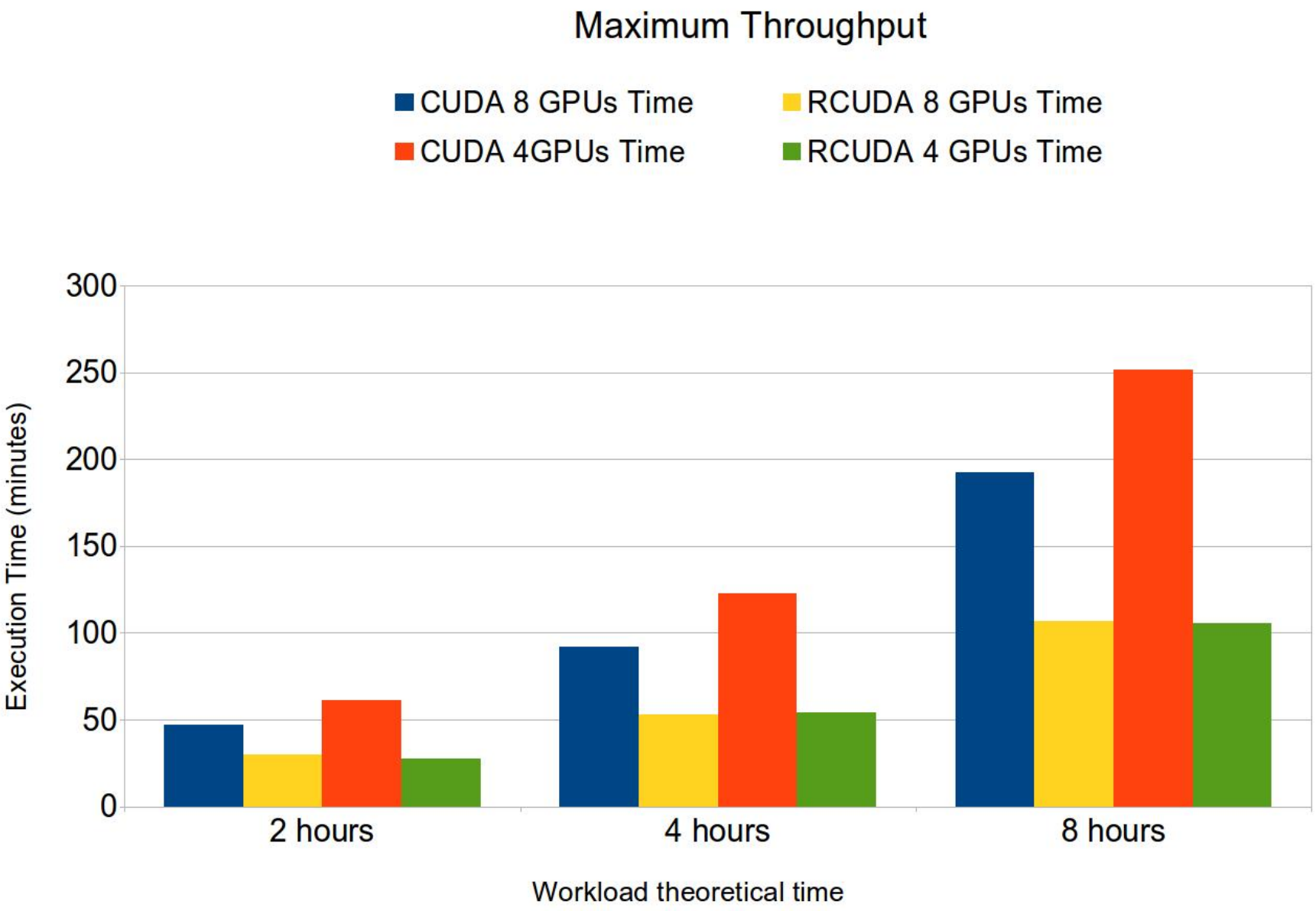
Max. Performance VS Max. Throughput



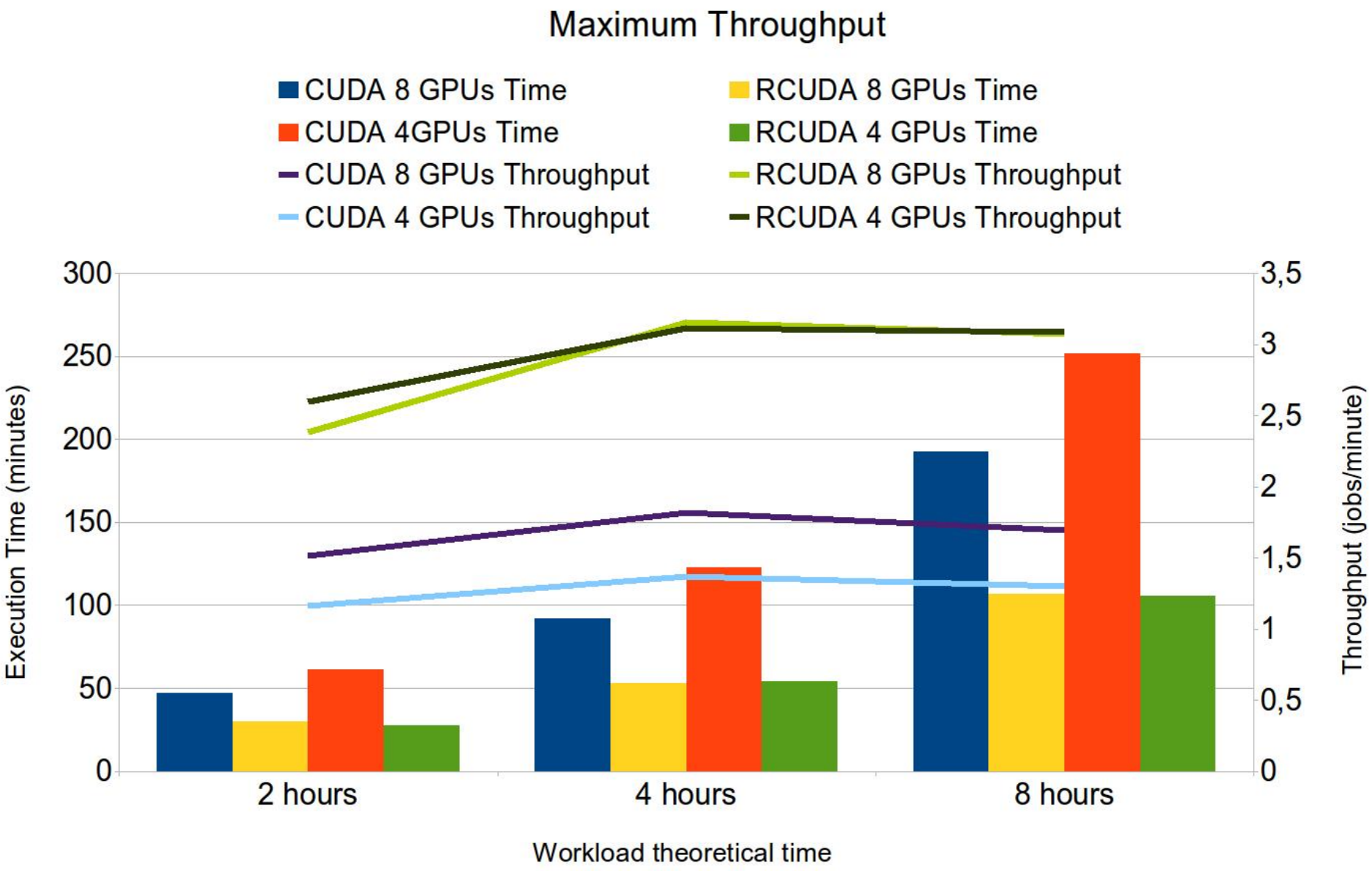
Max. Throughput with 8 and 4 GPUs



Max. Throughput with 8 and 4 GPUs

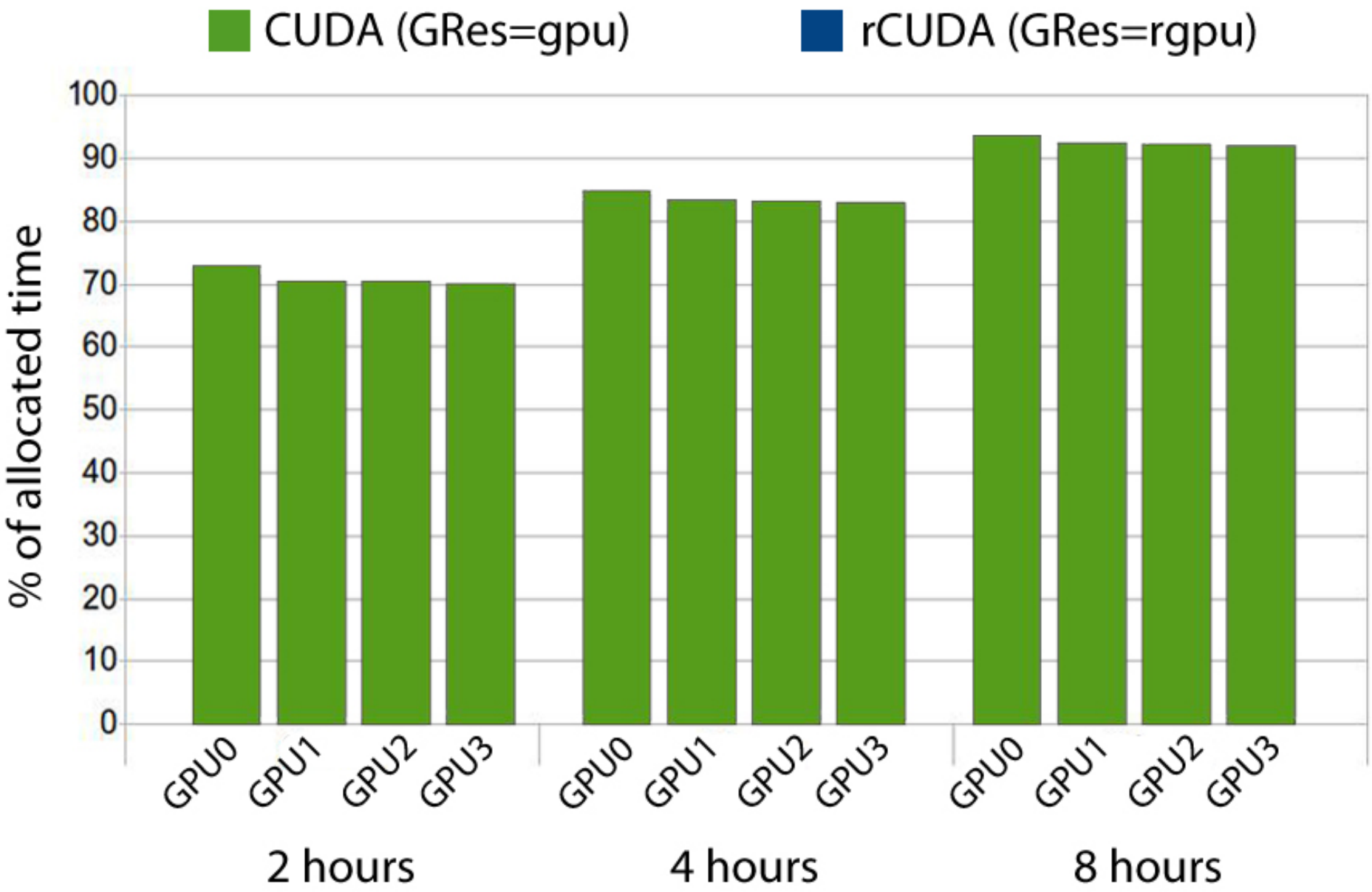


Max. Throughput with 8 and 4 GPUs

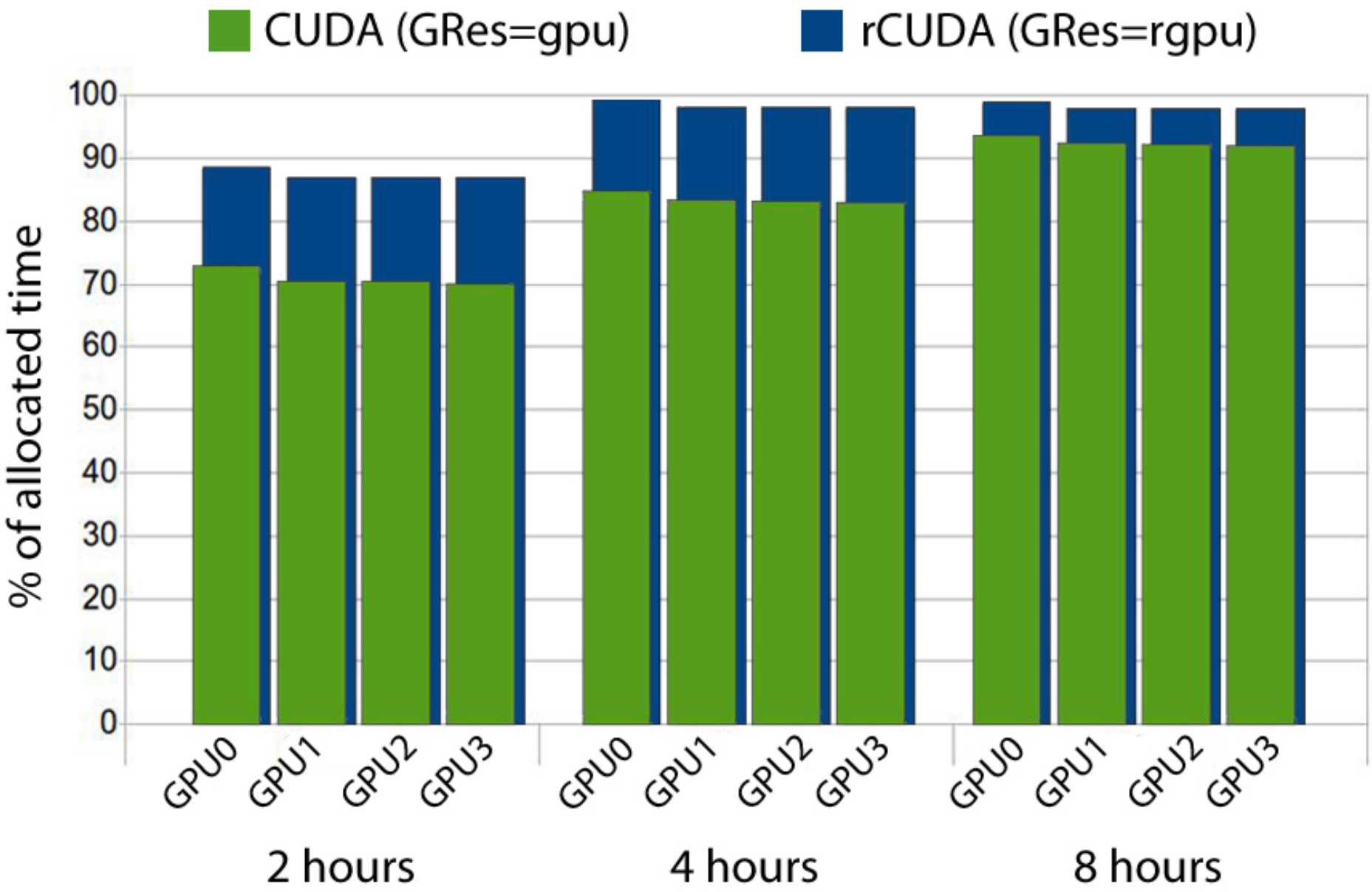


GPU allocation

GPU Allocation

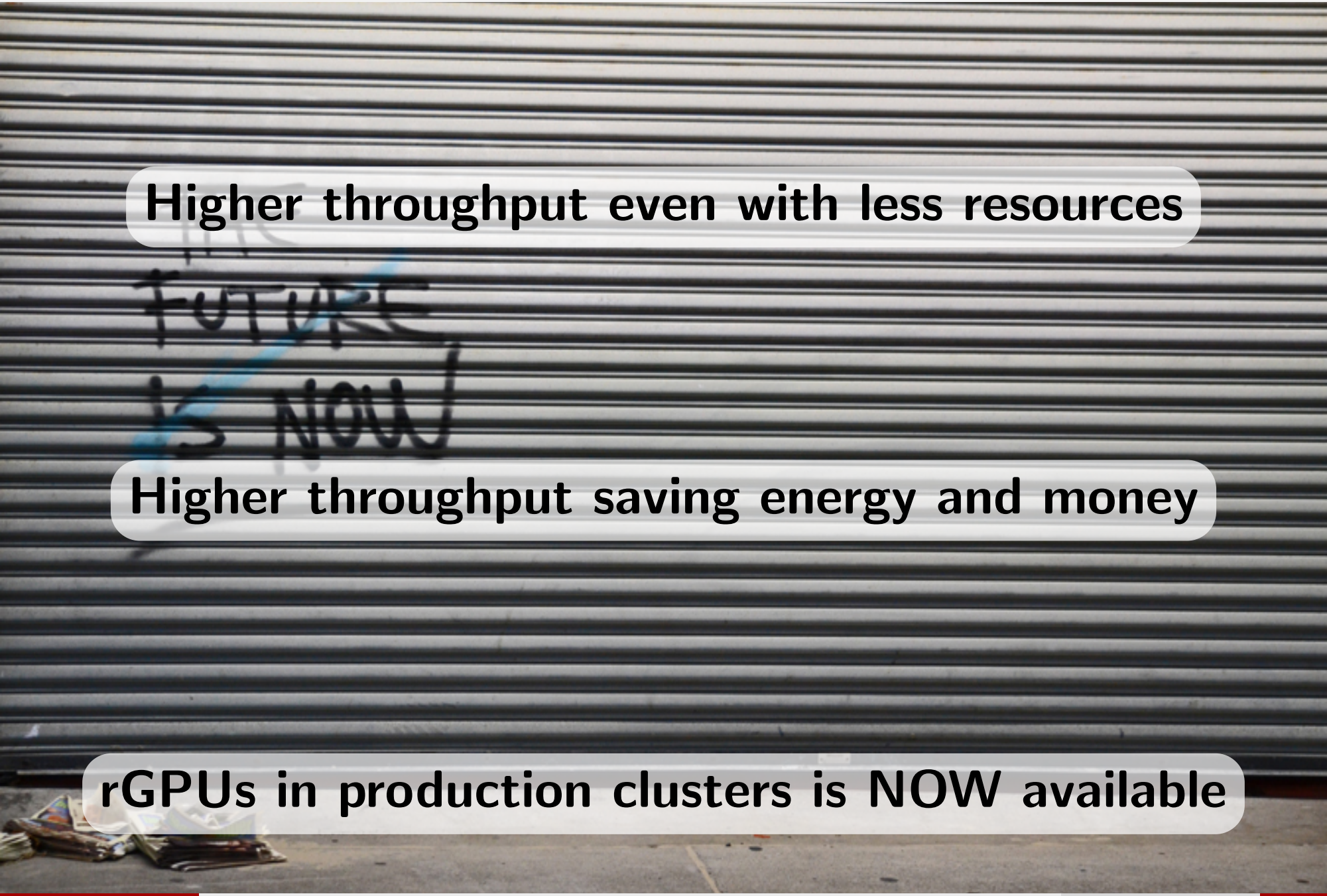


GPU Allocation



Conclusions

Conclusions



Higher throughput even with less resources

Higher throughput saving energy and money

rGPUs in production clusters is NOW available

Make the most of your GPUs with