Improving the Efficiency of Future Exascale Systems with rCUDA

Carlos Reaño, Javier Prades and Federico Silla
Technical University of Valencia (Spain)

4th IEEE International Workshop of High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB 2018)
Vienna, Austria, 24-Feb, 2018

Outline

- Introduction
- GPU Virtualization
- Approach proposed
- The NVIDIA Jetson TX1 system
- Performance evaluation
- Conclusions and future work

Outline

- Introduction
- GPU Virtualization
- Approach proposed
- The NVIDIA Jetson TX1 system
- Performance evaluation
- Conclusions and future work

Introduction

- Current supercomputers:
  - √ Performance
  - ✘ Energy consumption

Introduction

- Current supercomputers:
  - √ Performance
  - ✘ Energy consumption
- **Desired** future supercomputers:
  - √ Performance: exaFLOPS
  - √ Energy consumption: 20 MW

Introduction

- Current high-performance processors (Xeon):
  - √ Performance
  - ✘ Energy (~165W)

Introduction

- Current high-performance processors (Xeon):
  - √ Performance
  - ✘ Energy (~165W)
- ...plus high-performance accelerators (GPU):
  - √ Performance
  - ✘ Energy (~100W)

Introduction

- Current high-performance processors (Xeon):
  - ✓ Performance
  - ✘ Energy (~165W)
- ...plus high-performance accelerators (GPU):
  - ✓ Performance
  - ✘ Energy (~100W)
- Current low-power processors (ARM):
  - ✓ Energy consumption (~15W)
  - ✘ Performance

Introduction

- How improving efficiency of future exascale systems?

Introduction

- How improving efficiency of future exascale systems?
- Our proposal: heterogeneous platforms
  - Low-power processors (ARM)
  - Accelerated with remote HPC servers (Xeon + GPUs)
  - Remote HPC servers shared among all low-power processors

Introduction

- How improving efficiency of future exascale systems?
- Our proposal: heterogeneous platforms
  - Low-power processors (ARM)
  - Accelerated with remote HPC servers (Xeon + GPUs)
  - Remote HPC servers shared among all low-power processors
- How sharing? GPU virtualization
  - ✓ GPUs transparently shared among all cluster nodes
  - ✘ Virtualization framework overhead
  - ✘ Network overhead (high-performance interconnection!!)

Outline

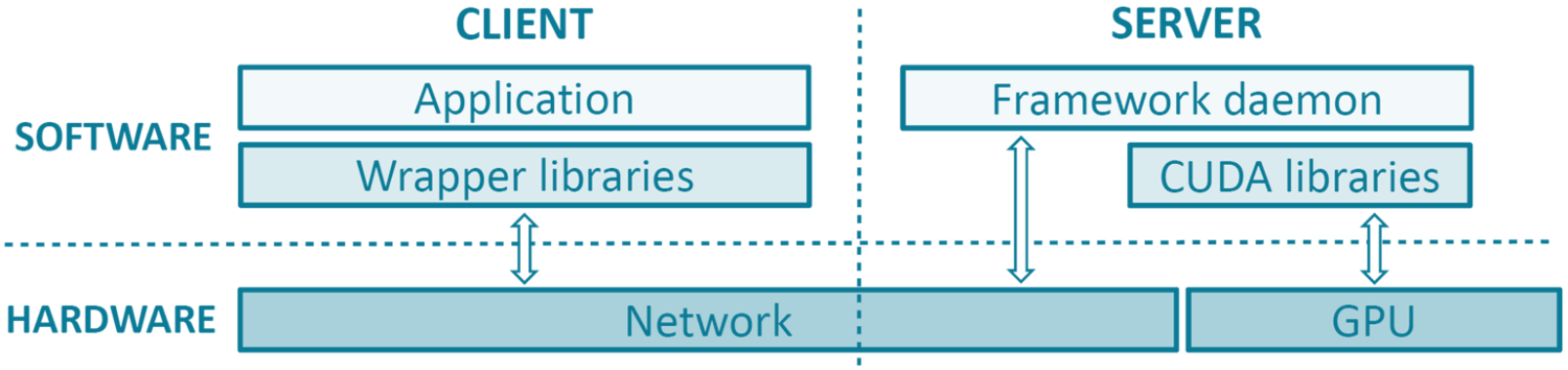
- Introduction
- GPU Virtualization
- Approach proposed
- The NVIDIA Jetson TX1 system
- Performance evaluation
- Conclusions and future work

GPU Virtualization

- GPGPU: OpenCL or CUDA

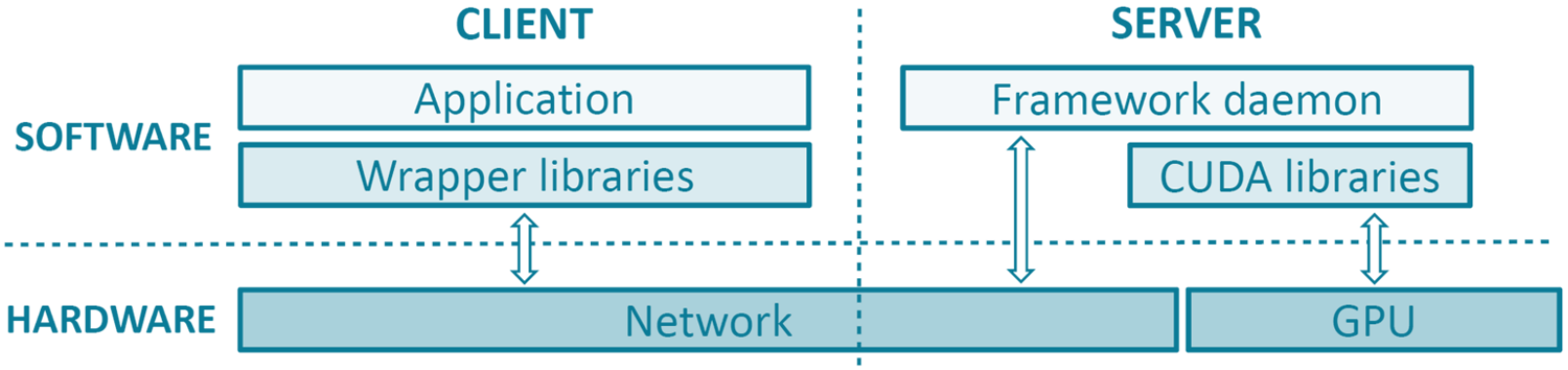
GPU Virtualization

- GPGPU: OpenCL or **CUDA**
- General architecture remote GPU solutions:



GPU Virtualization

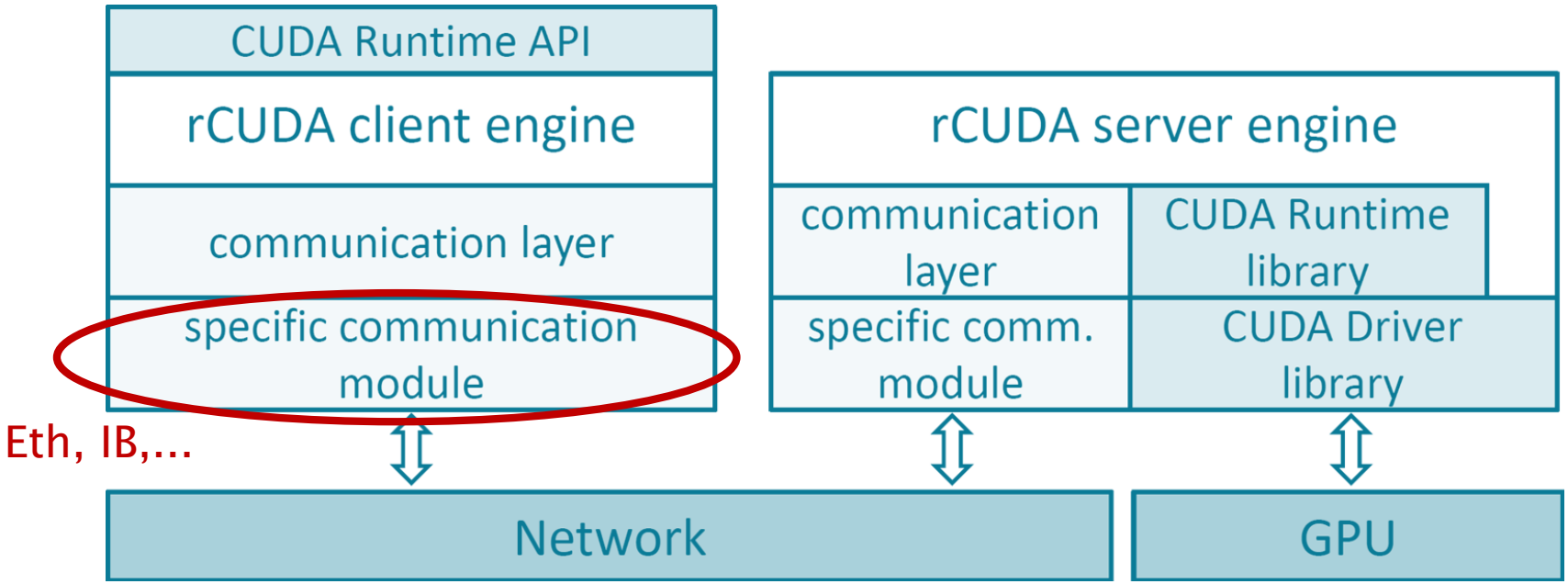
- GPGPU: OpenCL or **CUDA**
- General architecture remote GPU solutions:



- Existing solutions: **rCUDA (www.rcuda.net)**
  - rCUDA: freely available, supports CUDA 8

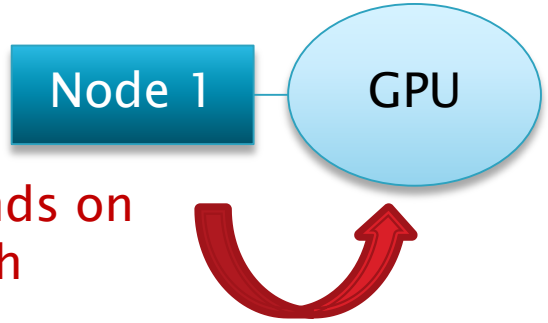
GPU Virtualization

- rCUDA communication architecture:

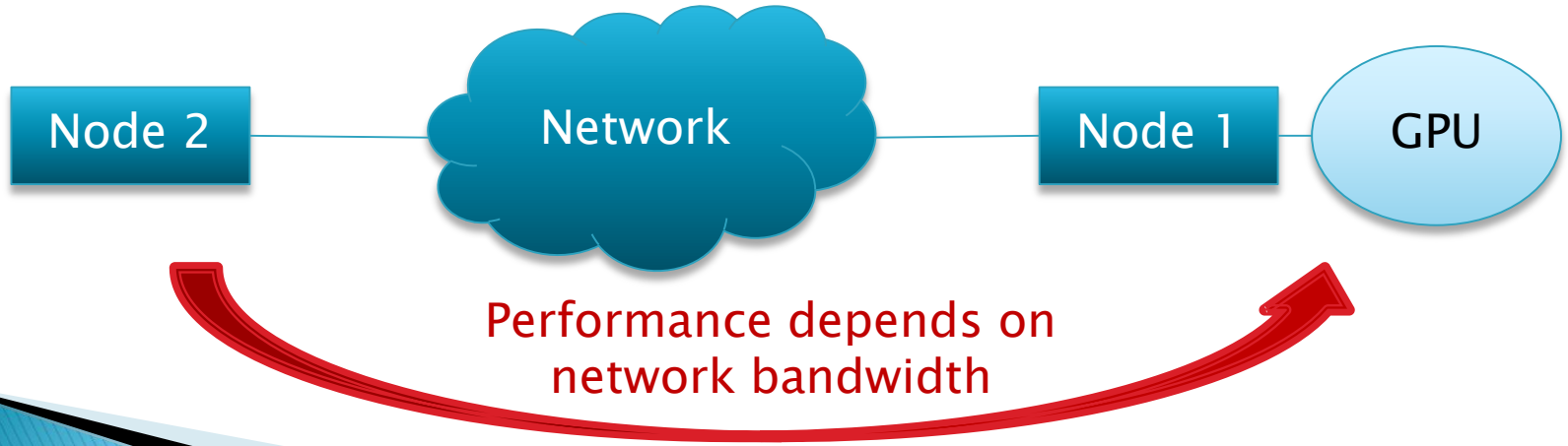


GPU Virtualization

- Local GPU:



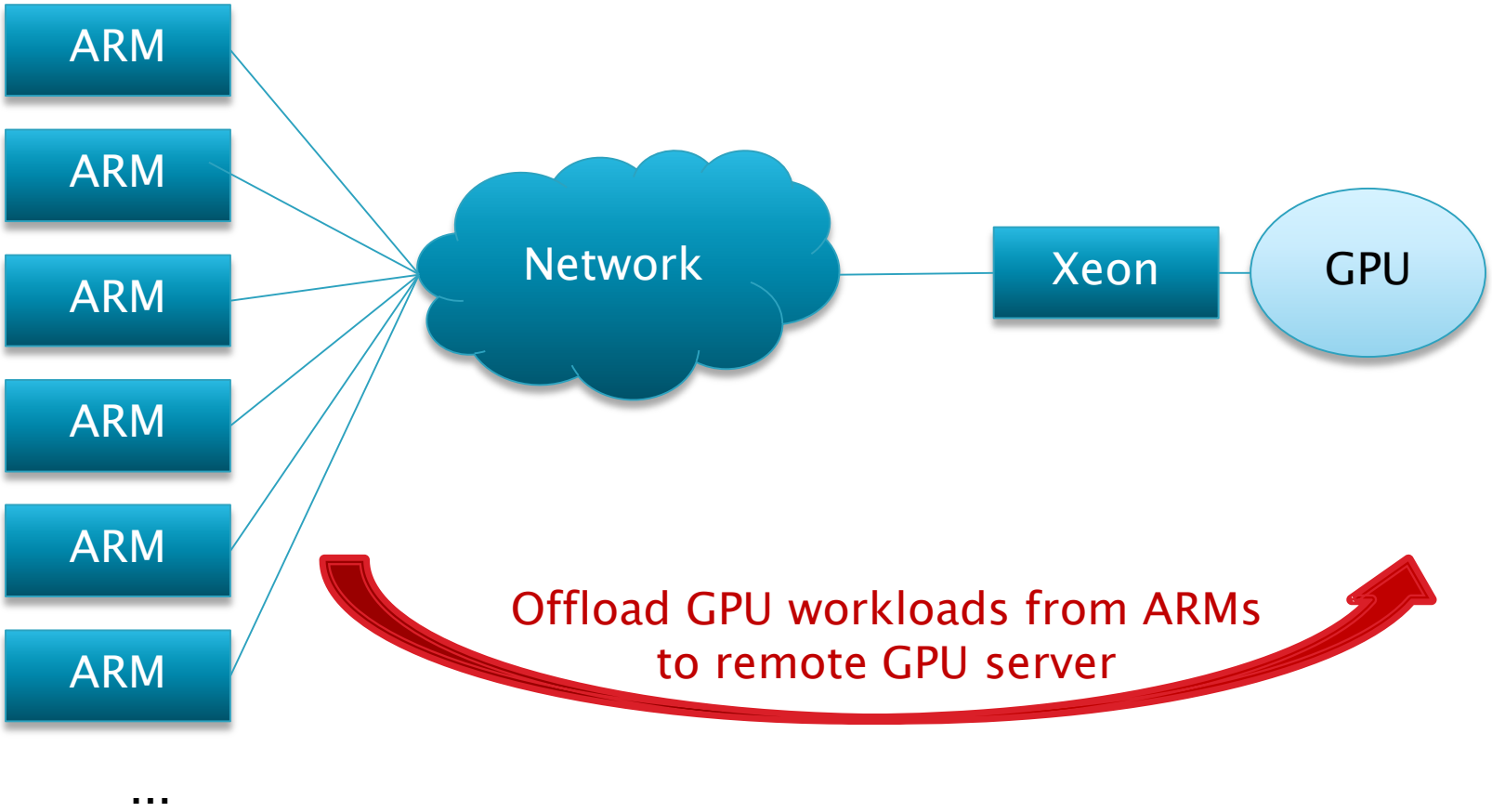
- Remote GPU:



Outline

- Introduction
- GPU Virtualization
- **Approach proposed**
- The NVIDIA Jetson TX1 system
- Performance evaluation
- Conclusions and future work

Approach proposed

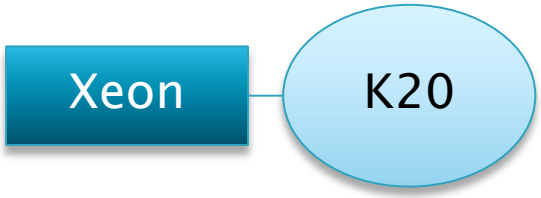


Approach proposed

ARM

- NVIDIA Jetson TX1:
  - 4-core ARM processor
  - 256-core CUDA compatible GPU

- Supermicro 1027GR-TRF:
  - 2 hexa-core Xeon processors
  - NVIDIA Tesla K20 GPU with 2496 CUDA cores



Outline

- Introduction
- GPU Virtualization
- Approach proposed
- The NVIDIA Jetson TX1 system
- Performance evaluation
- Conclusions and future work

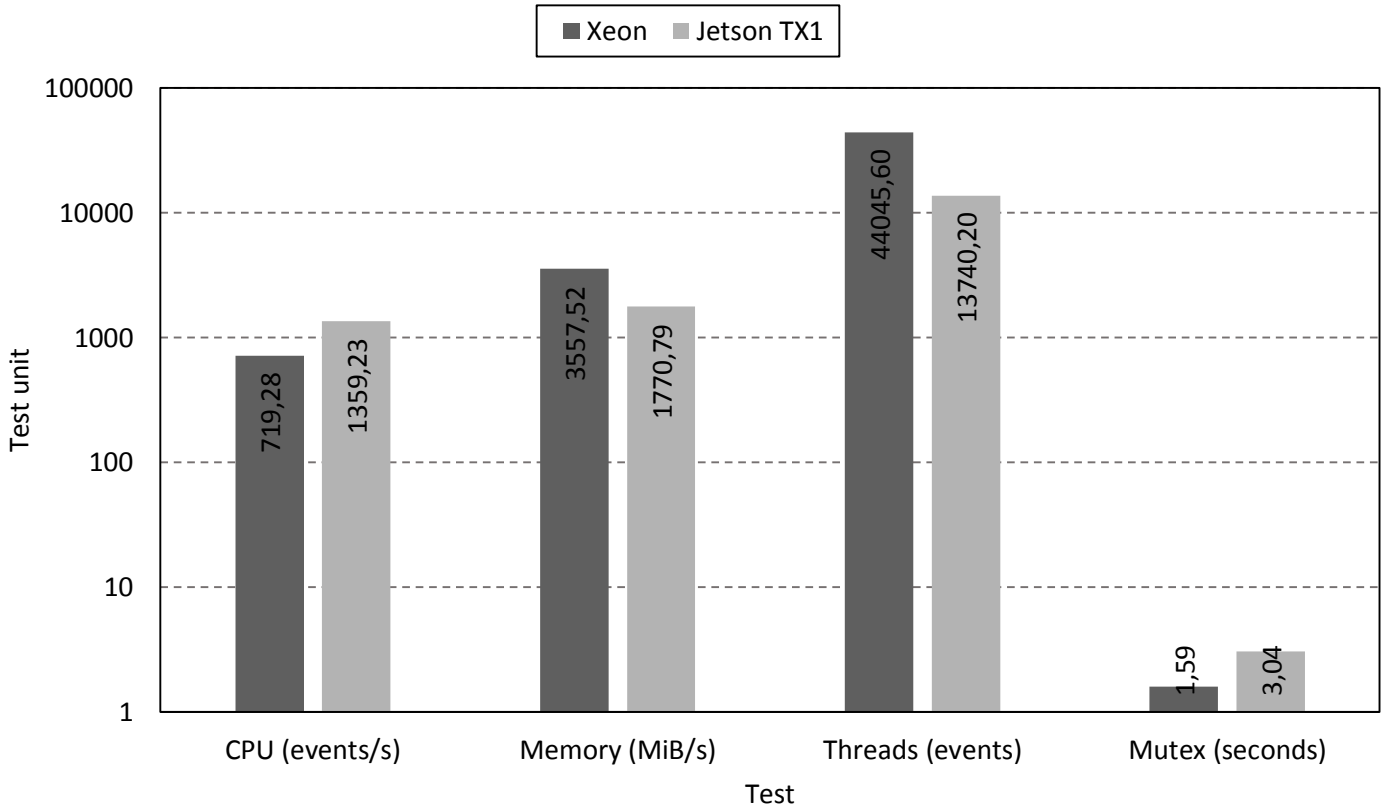
The NVIDIA Jetson TX1 system

- Performance comparison:
  - ARM CPU vs. Xeon
  - ARM GPU vs. K20

The NVIDIA Jetson TX1 system

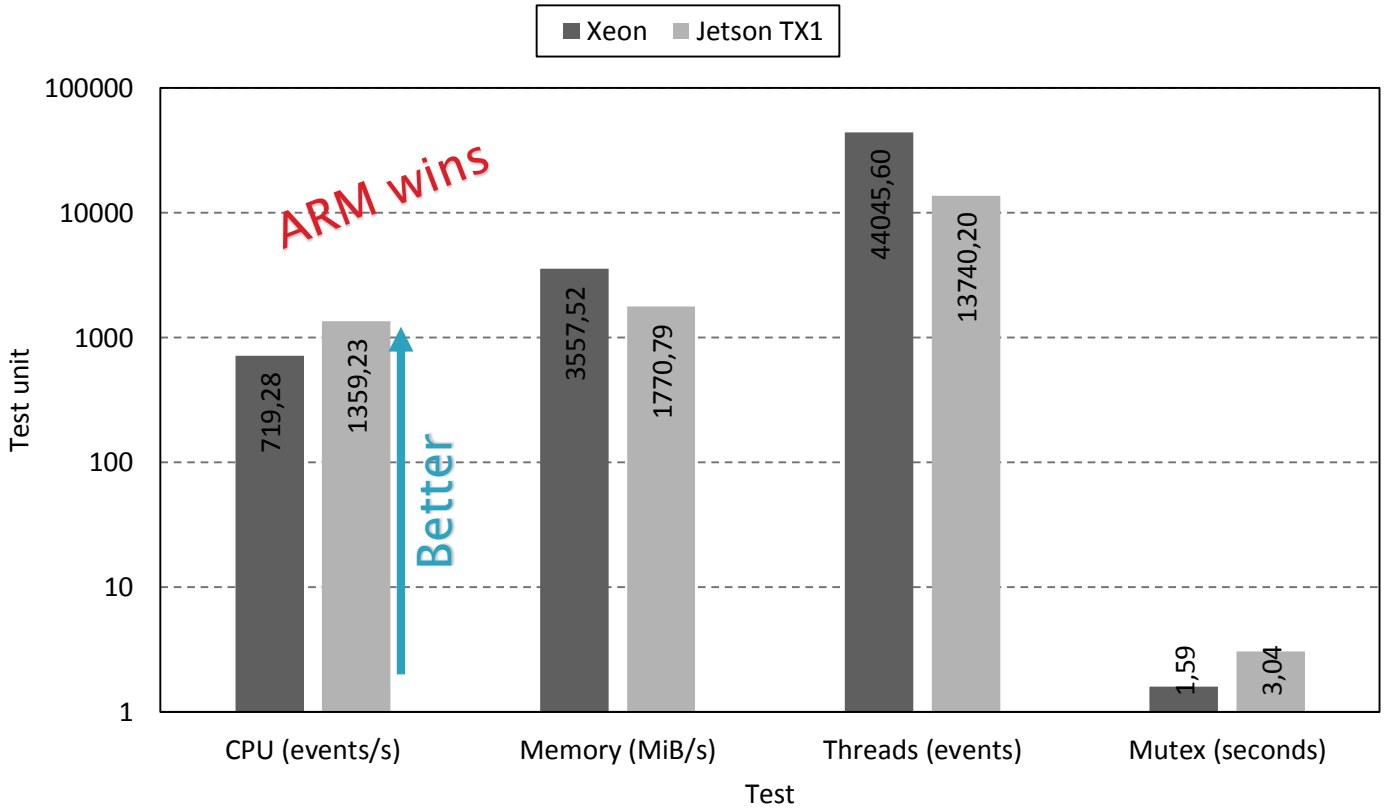
- Performance comparison:
  - ARM CPU vs. Xeon
  - ARM GPU vs. K20

The NVIDIA Jetson TX1 system



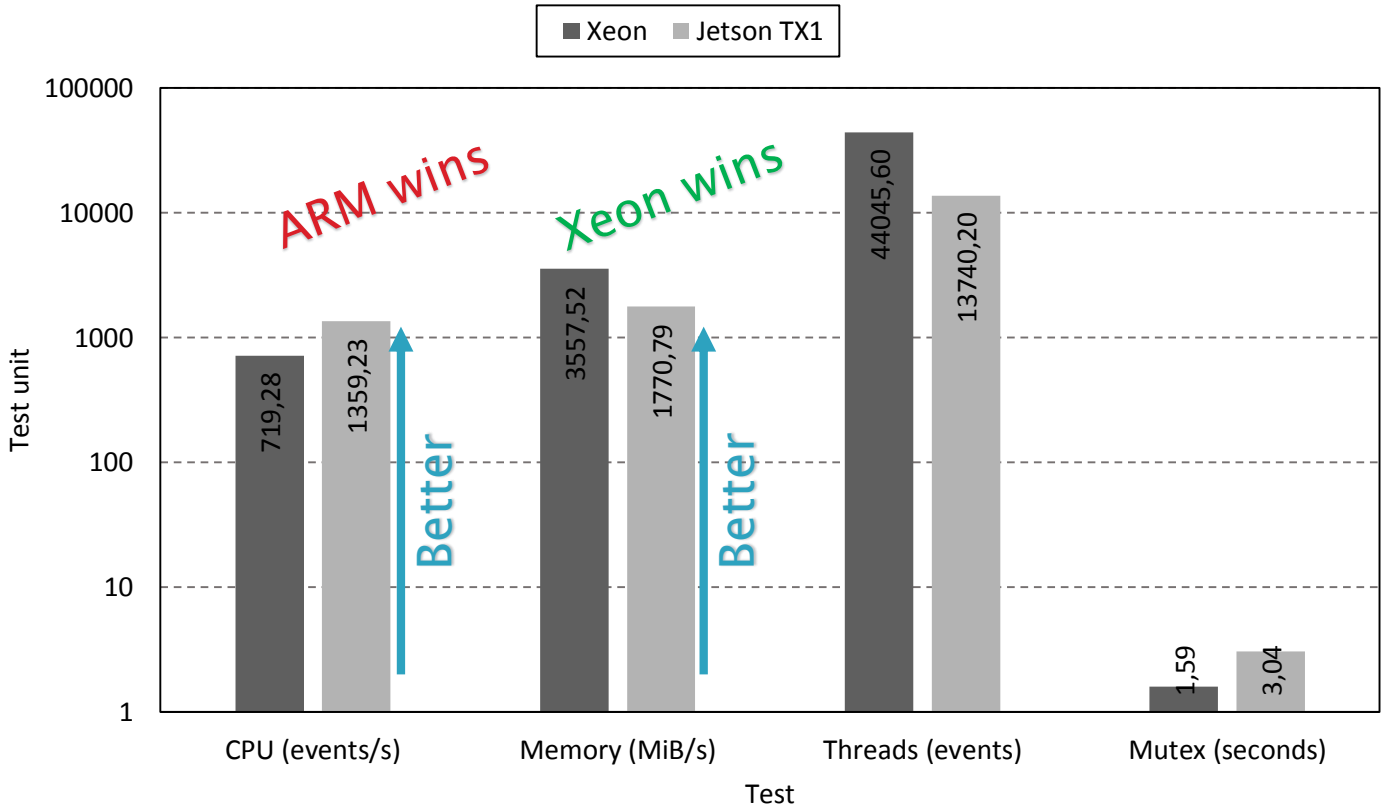
Performance comparison of NVIDIA Jetson TX1 and Supermicro 1027GR-TRF systems using the sysbench benchmark. Notice that the Y-axis is in logarithmic scale.

The NVIDIA Jetson TX1 system



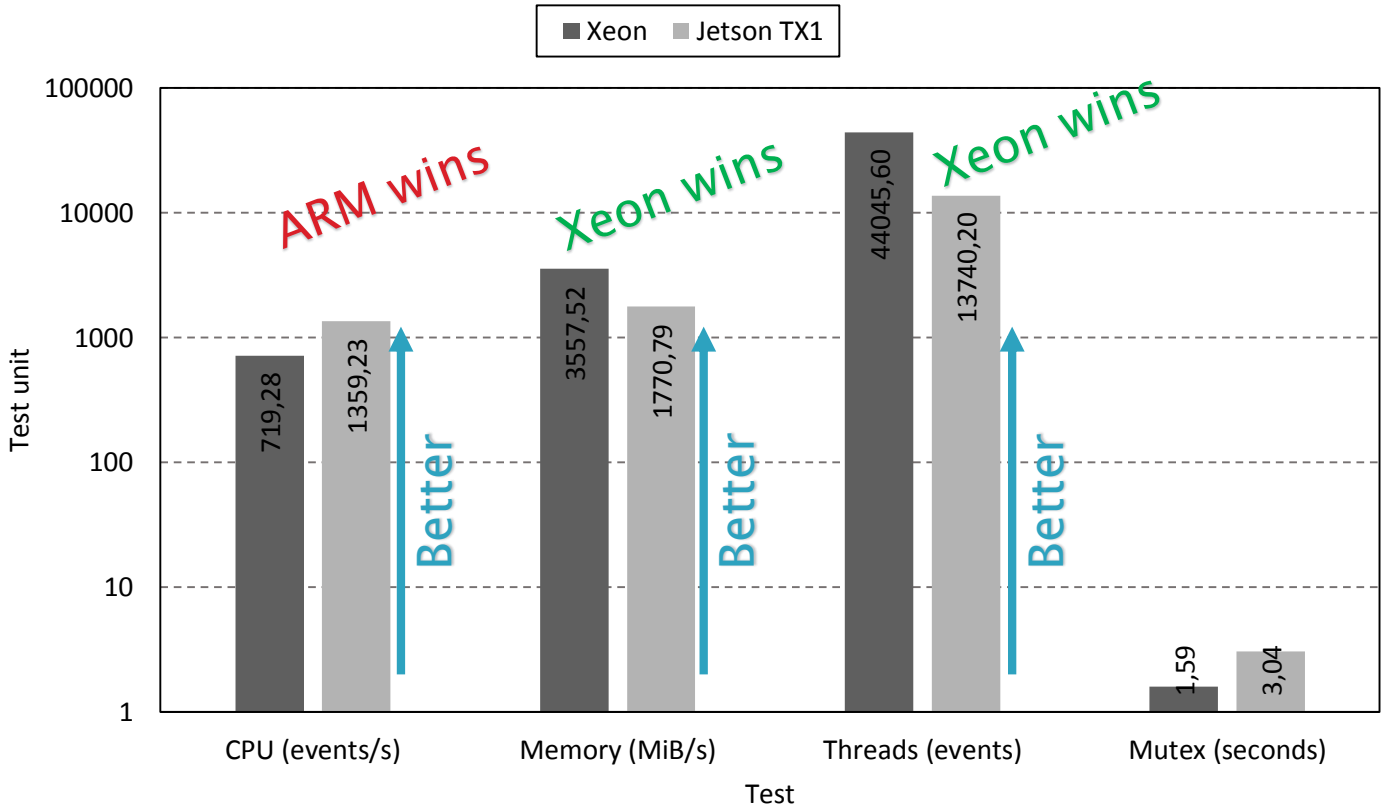
Performance comparison of NVIDIA Jetson TX1 and Supermicro 1027GR-TRF systems using the sysbench benchmark. Notice that the Y-axis is in logarithmic scale.

The NVIDIA Jetson TX1 system



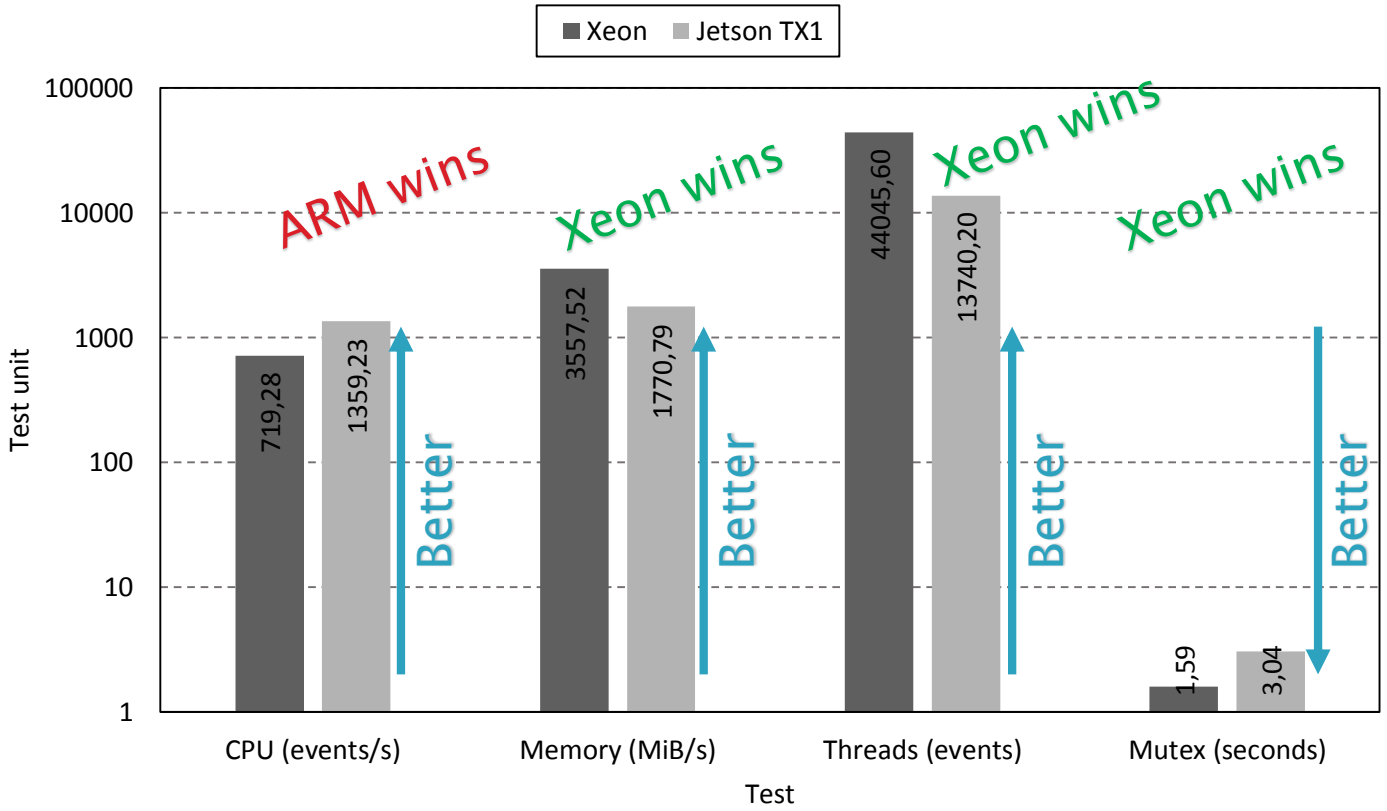
Performance comparison of NVIDIA Jetson TX1 and Supermicro 1027GR-TRF systems using the sysbench benchmark. Notice that the Y-axis is in logarithmic scale.

The NVIDIA Jetson TX1 system



Performance comparison of NVIDIA Jetson TX1 and Supermicro 1027GR-TRF systems using the sysbench benchmark. Notice that the Y-axis is in logarithmic scale.

The NVIDIA Jetson TX1 system



Performance comparison of NVIDIA Jetson TX1 and Supermicro 1027GR-TRF systems using the sysbench benchmark. Notice that the Y-axis is in logarithmic scale.

The NVIDIA Jetson TX1 system

- Performance comparison:
  - ARM CPU vs. Xeon
  - **ARM GPU vs. K20**

The NVIDIA Jetson TX1 system

Attribute	GPU included in Tegra TX1 chip	Tesla K20m
CUDA Capability Major/Minor version number	**5.3**	3.5
Total amount of global memory	3,983 MBytes (4,176,248,832 bytes)	**4,742 MBytes** (4,972,412,928 bytes)
Multiprocessors (MP) - CUDA Cores	2MP - 256 CUDA Cores	**13MP - 2,496 CUDA Cores**
GPU Max Clock rate	**998 MHz (1.00 GHz)**	706 MHz (0.71 GHz)
Memory Clock rate	1600 Mhz	**2,600 Mhz**
Memory Bus Width	64-bit	**320-bit**
L2 Cache Size	262,144 bytes	**1,310,720** bytes
Maximum Layered 1D Texture Size, (num) layers	1D=(16,384), 2,048 layers	1D=(16,384), 2,048 layers
Maximum Layered 2D Texture Size, (num) layers	2D=(16,384, 16,384), 2,048 layers	2D=(16,384, 16384), 2,048 layers
Total amount of constant memory	65,536 bytes	65,536 bytes
Total amount of shared memory per block	49,152 bytes	49,152 bytes
Total number of registers available per block	32,768	**65,536**
Warp size	32	32
Maximum number of threads per multiprocessor	2,048	2,048
Maximum number of threads per block	1,024	1,024
Max dimension size of a thread block (x,y,z)	(1,024, 1,024, 64)	(1,024, 1,024, 64)
Max dimension size of a grid size (x,y,z)	(2,147,483,647, 65,535, 65,535)	(2,147,483,647, 65,535, 65,535)
Maximum memory pitch	2,147,483,647 bytes	2,147,483,647 bytes
Texture alignment	512 bytes	512 bytes
Concurrent copy and kernel execution	Yes with 1 copy engine(s)	**Yes with 2 copy engine(s)**
Run time limit on kernels	**Yes**	**No**
Integrated GPU sharing Host Memory	**Yes**	**No**
Support host page-locked memory mapping	Yes	Yes
Alignment requirement for Surfaces	Yes	Yes
Device has ECC support	Disabled	**Enabled**
Device supports Unified Addressing (UVA)	Yes	Yes

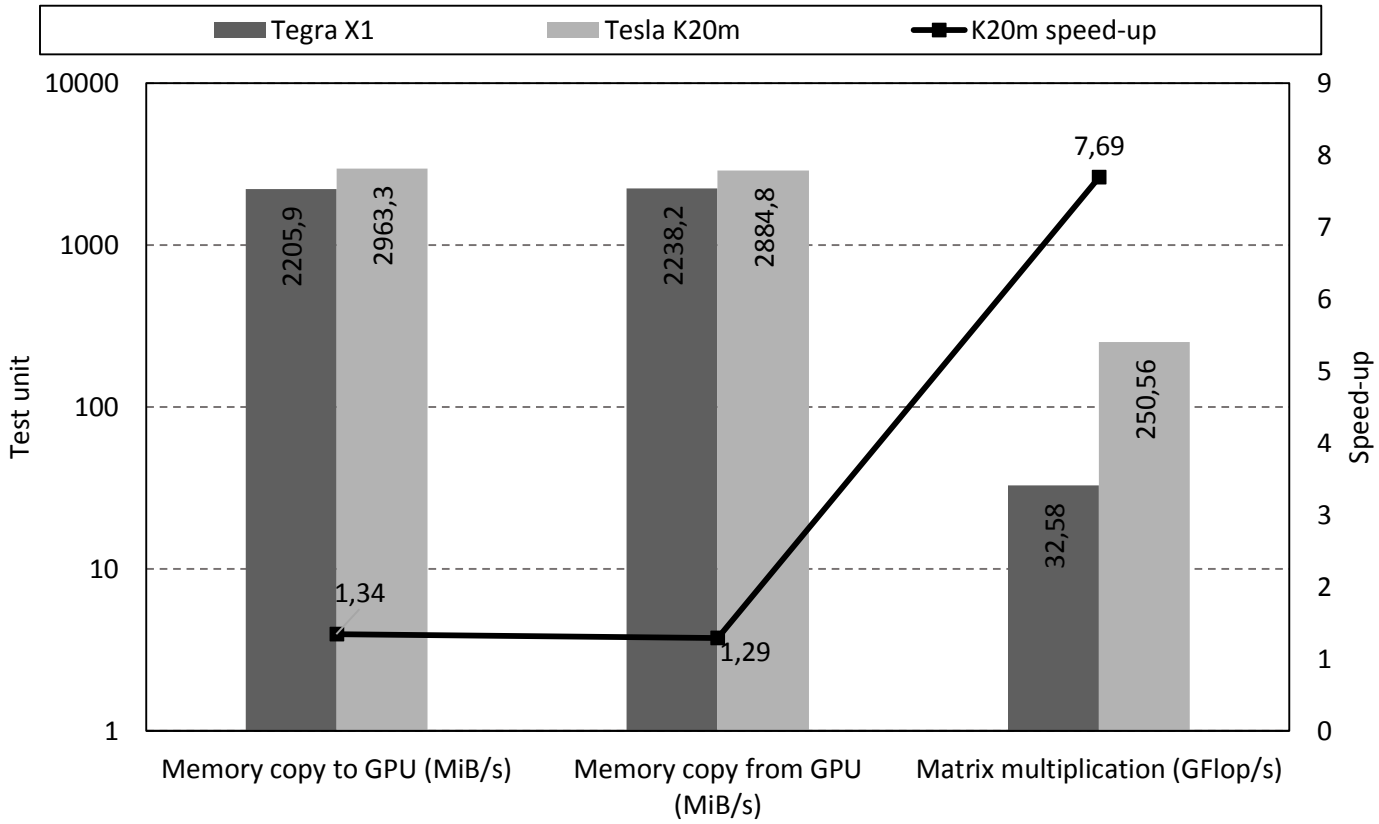
The NVIDIA Jetson TX1 system

Attribute	GPU included in Tegra TX1 chip	Tesla K20m
CUDA Capability Major/Minor version number	**5.3**	3.5
Total amount of global memory	3,983 MBytes (4,176,248,832 bytes)	**4,742 MBytes** (4,972,412,928 bytes)
Multiprocessors (MP) - CUDA Cores	2MP - 256 CUDA Cores	**13MP - 2,496 CUDA Cores**
GPU Max Clock rate	**998 MHz (1.00 GHz)**	706 MHz (0.71 GHz)
Memory Clock rate	1600 Mhz	**2,600 Mhz**
Memory Bus Width	64-bit	**320-bit**
L2 Cache Size	262,144 bytes	**1,310,720** bytes
Maximum Layered 1D Texture Size, (num) layers	1D=(16,384), 2,048 layers	1D=(16,384), 2,048 layers
Maximum Layered 2D Texture Size, (num) layers	2D=(16,384, 16,384), 2,048 layers	2D=(16,384, 16384), 2,048 layers
Total amount of constant memory	65,536 bytes	65,536 bytes
Total amount of shared memory per block	49,152 bytes	49,152 bytes
Total number of registers available per block	32,768	**65,536**
Warp size	32	32
Maximum number of threads per multiprocessor	2,048	2,048
Maximum number of threads per block	1,024	1,024
Max dimension size of a thread block (x,y,z)	(1,024, 1,024, 64)	(1,024, 1,024, 64)
Max dimension size of a grid size (x,y,z)	(2,147,483,647, 65,535, 65,535)	(2,147,483,647, 65,535, 65,535)
Maximum memory pitch	2,147,483,647 bytes	2,147,483,647 bytes
Texture alignment	512 bytes	512 bytes
Concurrent copy and kernel execution	Yes with 1 copy engine(s)	**Yes with 2 copy engine(s)**
Run time limit on kernels	**Yes**	**No**
Integrated GPU sharing Host Memory	**Yes**	**No**
Support host page-locked memory mapping	Yes	Yes
Alignment requirement for Surfaces	Yes	Yes
Device has ECC support	Disabled	**Enabled**
Device supports Unified Addressing (UVA)	Yes	Yes

The NVIDIA Jetson TX1 system

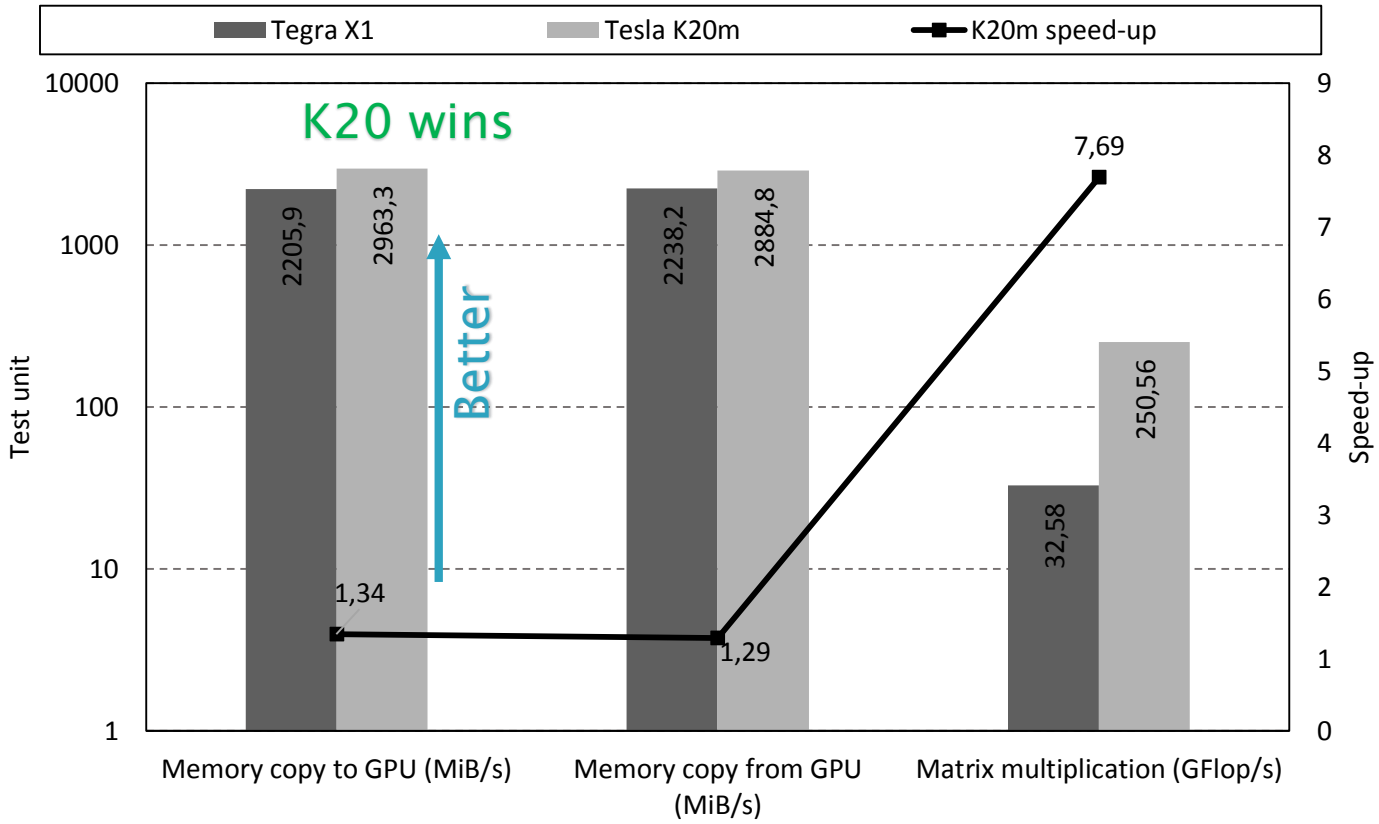
Attribute	GPU included in Tegra TX1 chip	Tesla K20m
CUDA Capability Major/Minor version number	**5.3**	3.5
Total amount of global memory	3,983 MBytes (4,176,248,832 bytes)	**4,742 MBytes (4,972,412,928 bytes)**
Multiprocessors (MP) - CUDA Cores	2MP - 256 CUDA Cores	**13MP - 2,496 CUDA Cores**
GPU Max Clock rate	**998 MHz (1.00 GHz)**	706 MHz (0.71 GHz)
Memory Clock rate	1600 Mhz	**2,600 Mhz**
Memory Bus Width	64-bit	**320-bit**
L2 Cache Size	262,144 bytes	**1,310,720** bytes
Maximum Layered 1D Texture Size, (num) layers	1D=(16,384), 2,048 layers	1D=(16,384), 2,048 layers
Maximum Layered 2D Texture Size, (num) layers	2D=(16,384, 16,384), 2,048 layers	2D=(16,384, 16384), 2,048 layers
Total amount of constant memory	65,536 bytes	65,536 bytes
Total amount of shared memory per block	49,152 bytes	49,152 bytes
Total number of registers available per block	32,768	**65,536**
Warp size	32	32
Maximum number of threads per multiprocessor	2,048	2,048
Maximum number of threads per block	1,024	1,024
Max dimension size of a thread block (x,y,z)	(1,024, 1,024, 64)	(1,024, 1,024, 64)
Max dimension size of a grid size (x,y,z)	(2,147,483,647, 65,535, 65,535)	(2,147,483,647, 65,535, 65,535)
Maximum memory pitch	2,147,483,647 bytes	2,147,483,647 bytes
Texture alignment	512 bytes	512 bytes
Concurrent copy and kernel execution	Yes with 1 copy engine(s)	**Yes with 2 copy engine(s)**
Run time limit on kernels	**Yes**	**No**
Integrated GPU sharing Host Memory	**Yes**	**No**
Support host page-locked memory mapping	Yes	Yes
Alignment requirement for Surfaces	Yes	Yes
Device has ECC support	Disabled	**Enabled**
Device supports Unified Addressing (UVA)	Yes	Yes

The NVIDIA Jetson TX1 system



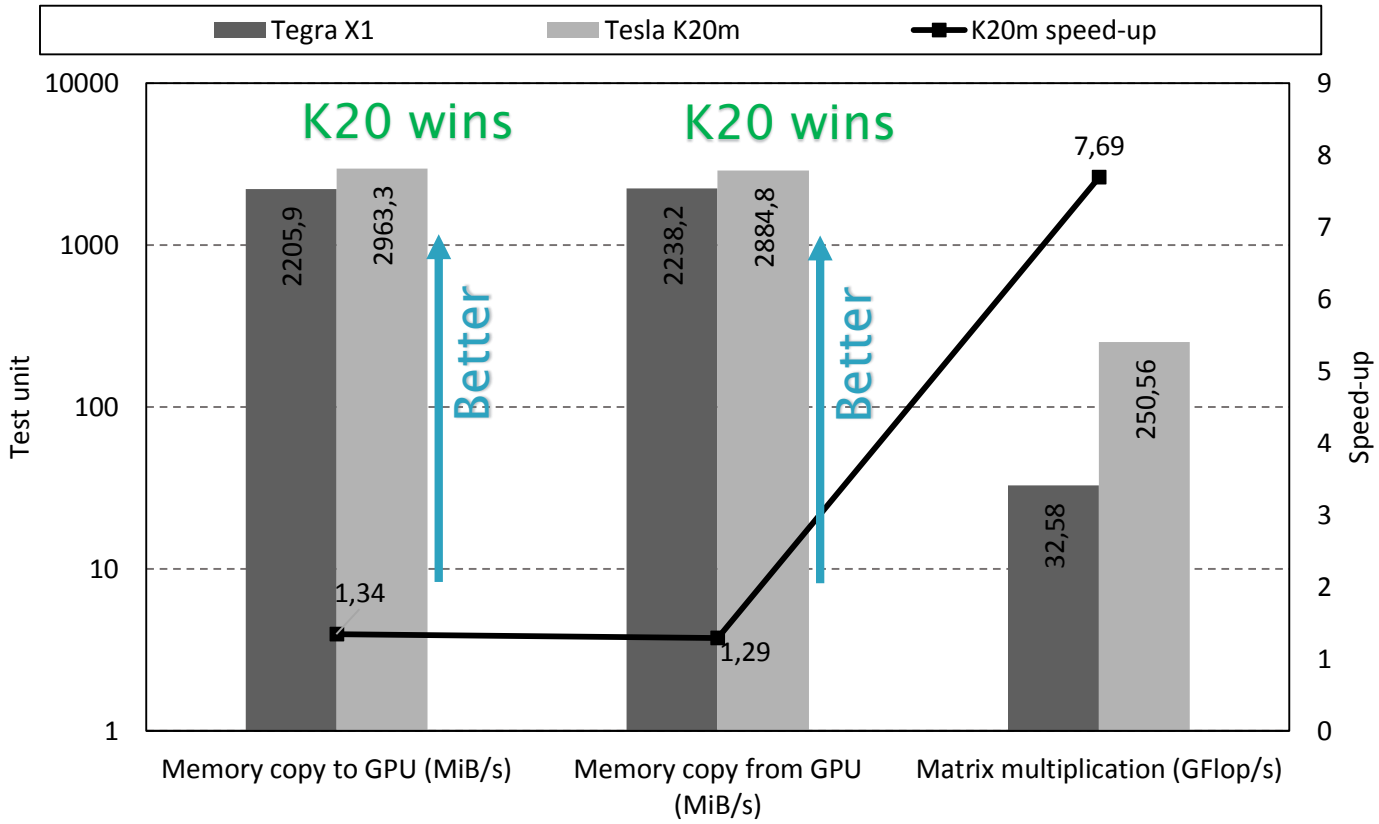
Performance comparison of the GPU included in the Tegra X1 chip and the Tesla K20m GPU in terms of data transfer bandwidth and computational power. Notice that primary Y-axis is in scale.

The NVIDIA Jetson TX1 system



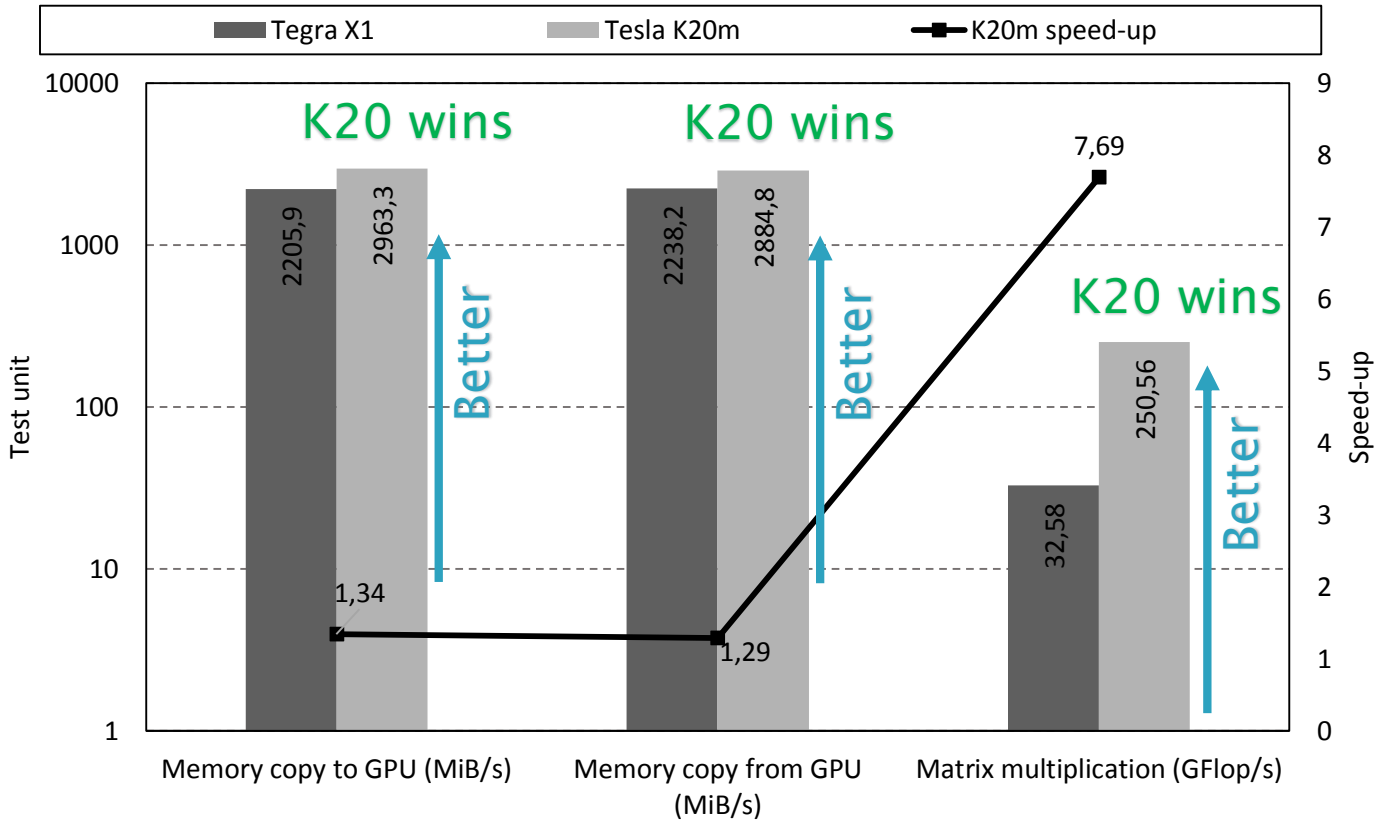
Performance comparison of the GPU included in the Tegra X1 chip and the Tesla K20m GPU in terms of data transfer bandwidth and computational power. Notice that primary Y-axis is in scale.

The NVIDIA Jetson TX1 system



Performance comparison of the GPU included in the Tegra X1 chip and the Tesla K20m GPU in terms of data transfer bandwidth and computational power. Notice that primary Y-axis is in scale.

The NVIDIA Jetson TX1 system



Performance comparison of the GPU included in the Tegra X1 chip and the Tesla K20m GPU in terms of data transfer bandwidth and computational power. Notice that primary Y-axis is in scale.

Outline

- Introduction
- GPU Virtualization
- Approach proposed
- The NVIDIA Jetson TX1 system
- **Performance evaluation**
- Conclusions and future work

Offloading GPU workload from ARMs to remote Xeon GPU servers

- Three scenarios:
  1. ARM
  2. Xeon+GPU
  3. ARM+remote GPU

- Performance comparison:
  - Rodinia benchmarks
  - Gaussian benchmark varying problem size
  - Gaussian benchmark, fixed problem size, sharing GPU among multiple ARM clients

Offloading GPU workload from ARMs to remote Xeon GPU servers

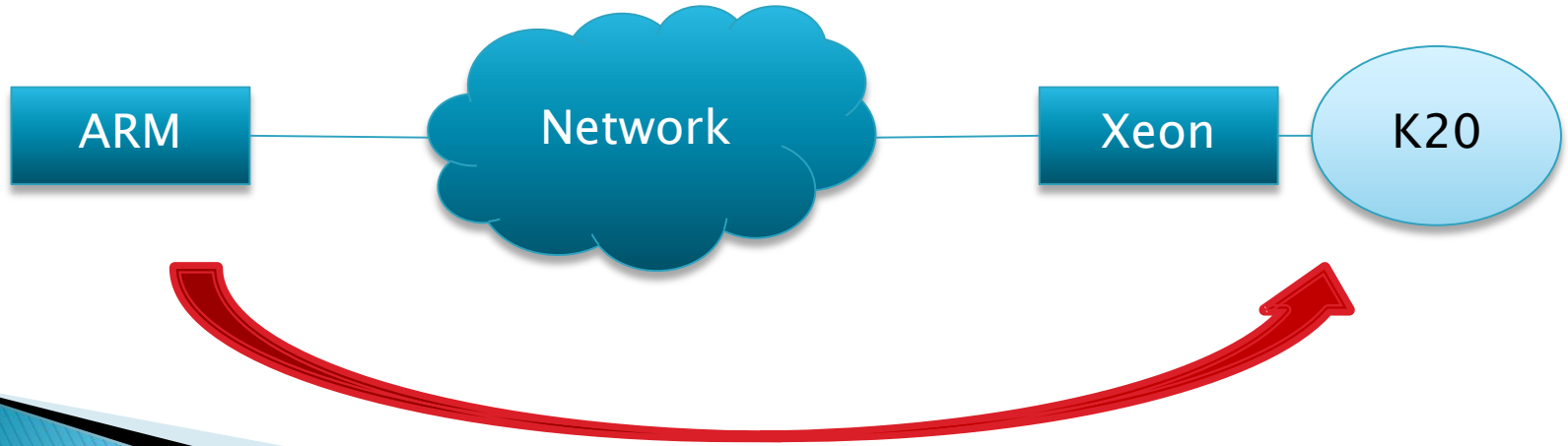
- Three scenarios:
  1. ARM
  2. Xeon+GPU
  3. ARM+remote GPU

- Performance comparison:
  - **Rodinia benchmarks**
  - Gaussian benchmark varying problem size
  - Gaussian benchmark, fixed problem size, sharing GPU among multiple ARM clients

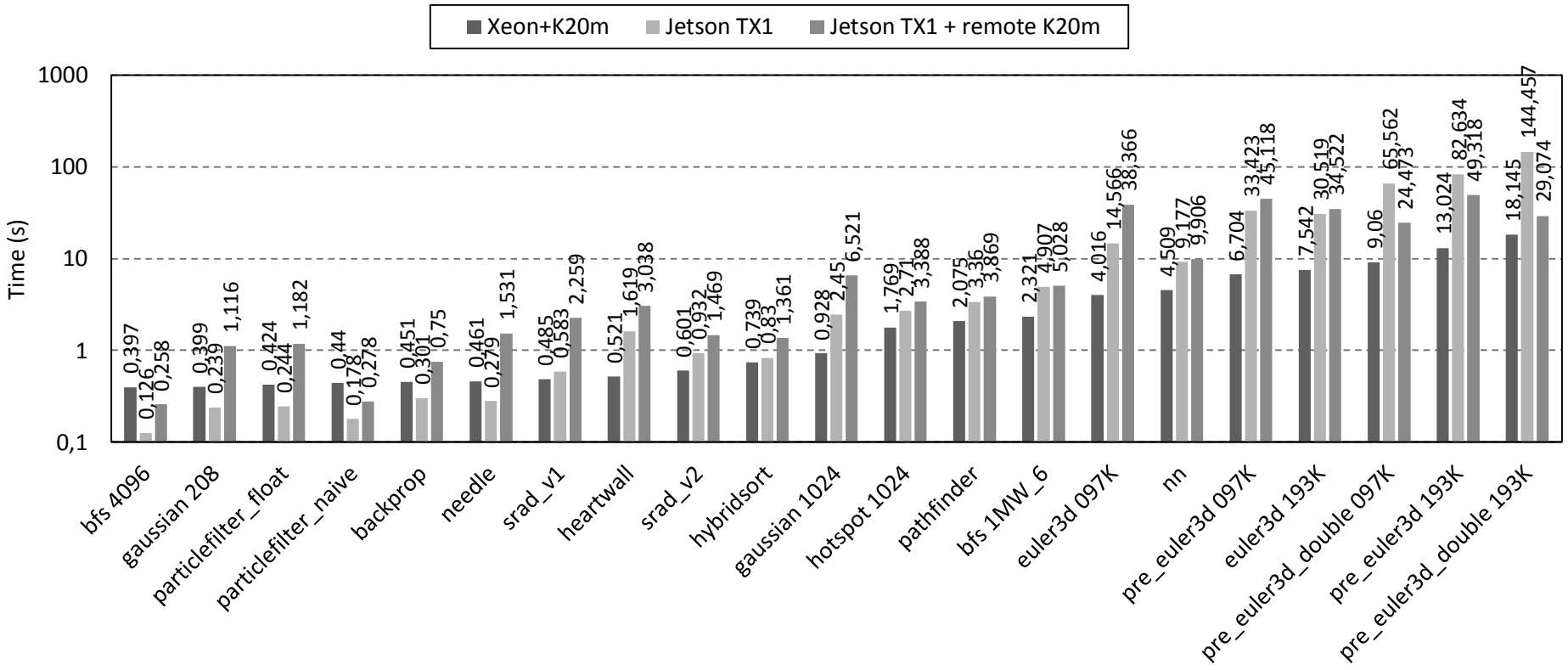
Offloading GPU workload from ARMs to remote Xeon GPU servers

- Local GPU:

- Remote GPU:

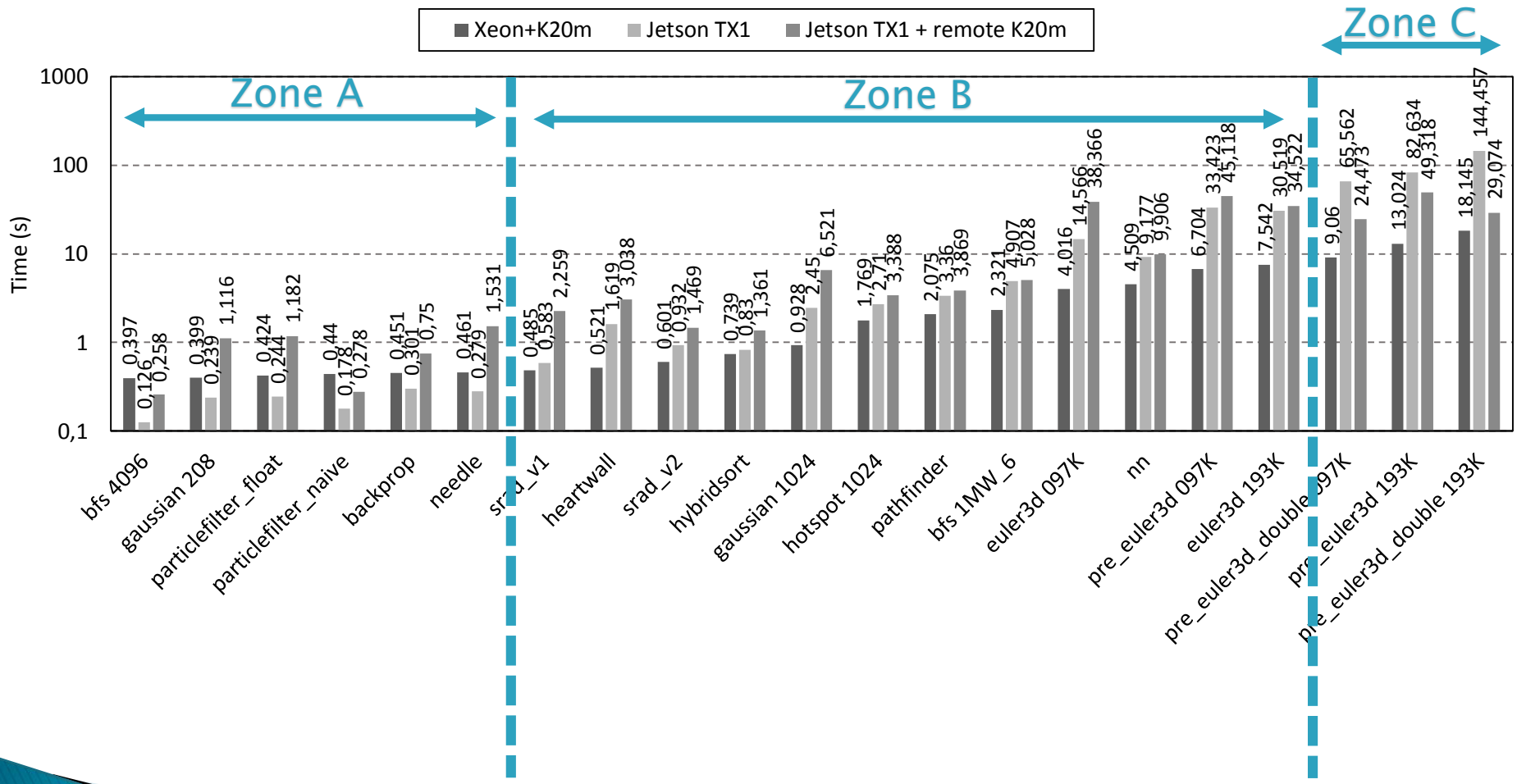


Offloading GPU workload from ARMs to remote Xeon GPU servers

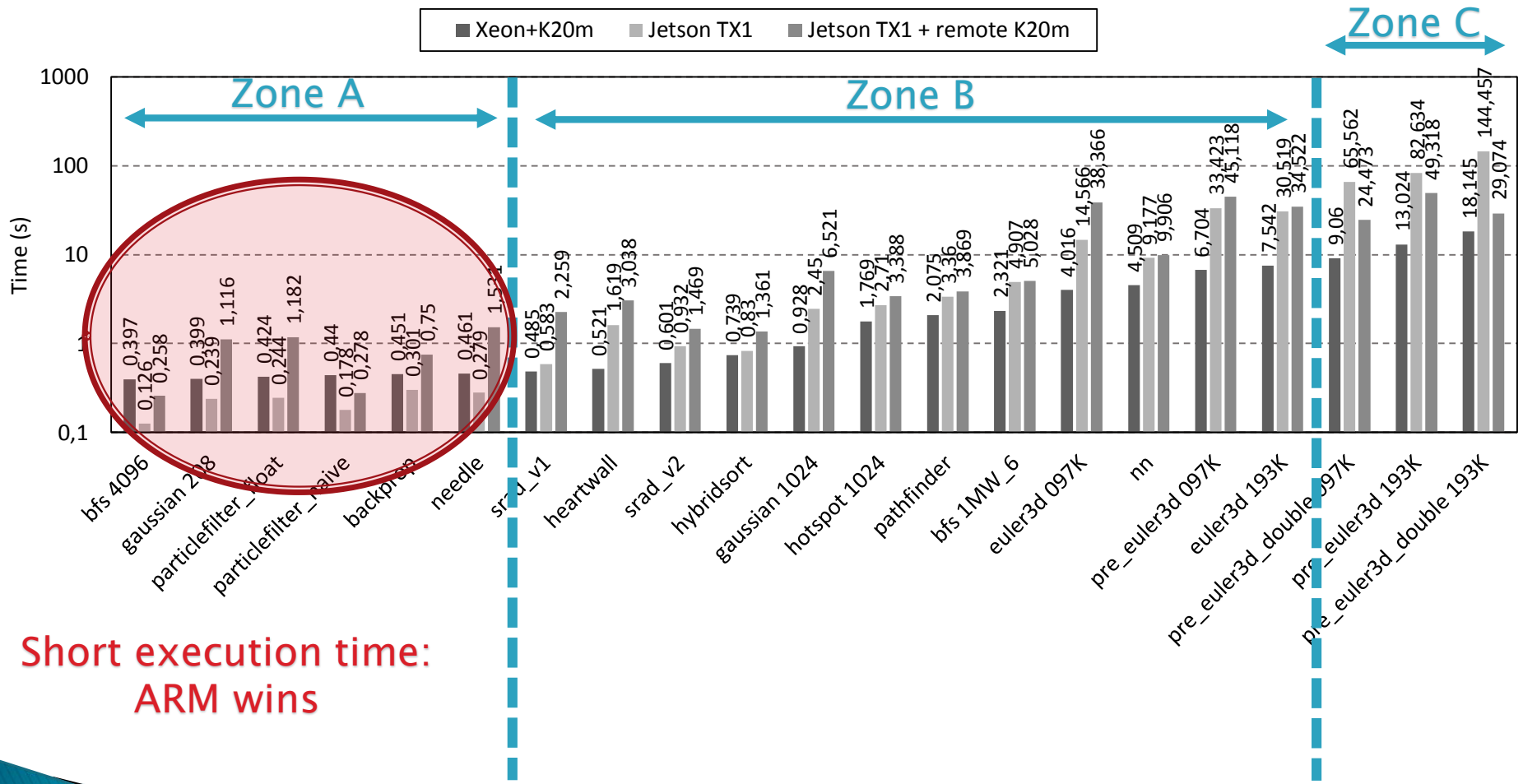


Comparison of Rodinia benchmarks executed in three different scenarios: (i) a regular server with one NVIDIA K20 GPU, (ii) an NVIDIA Jetson TX1, and (iii) an NVIDIA Jetson TX1 offloading GPU work to a remote regular server with one NVIDIA K20 GPU.

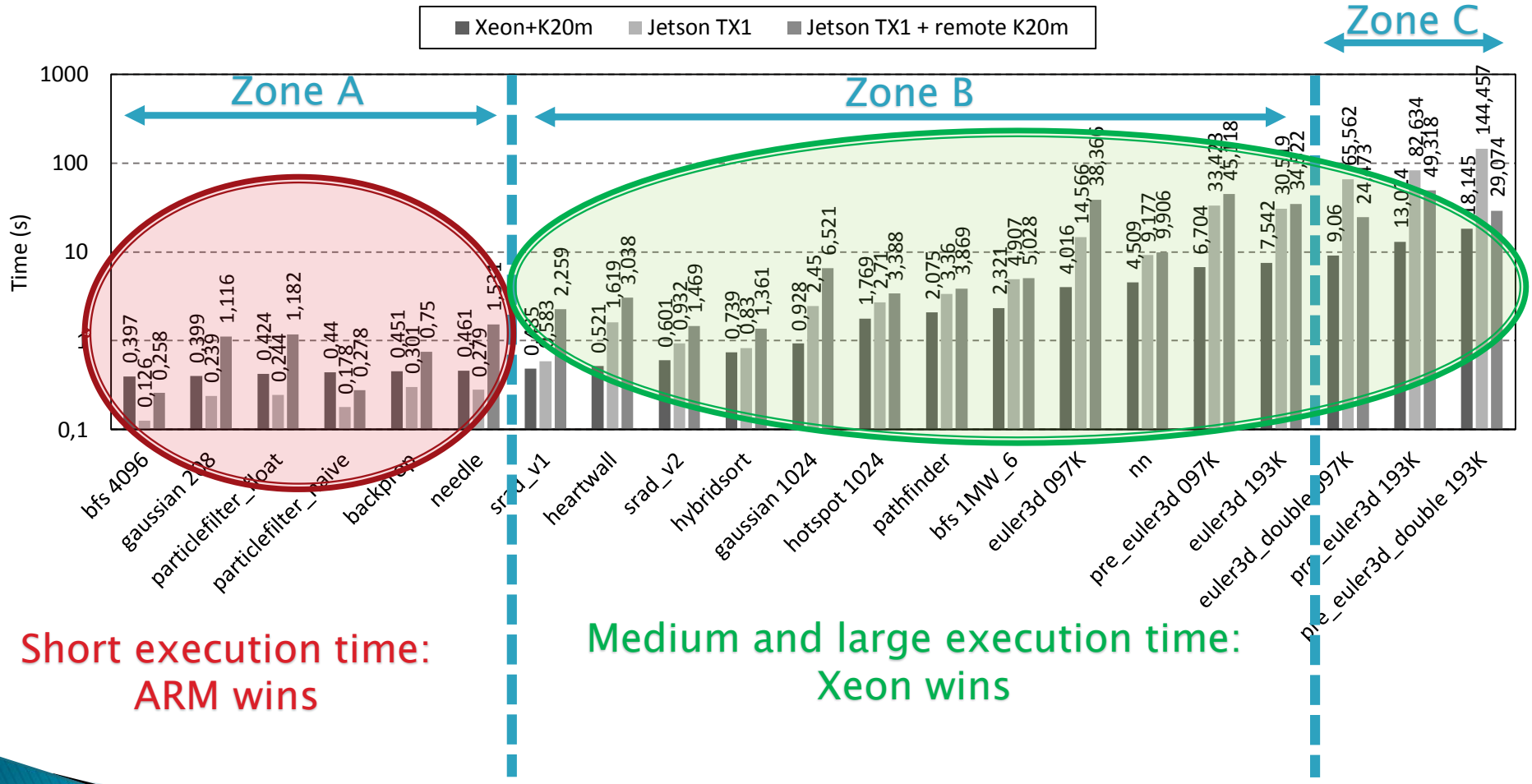
Offloading GPU workload from ARMs to remote Xeon GPU servers



Offloading GPU workload from ARMs to remote Xeon GPU servers



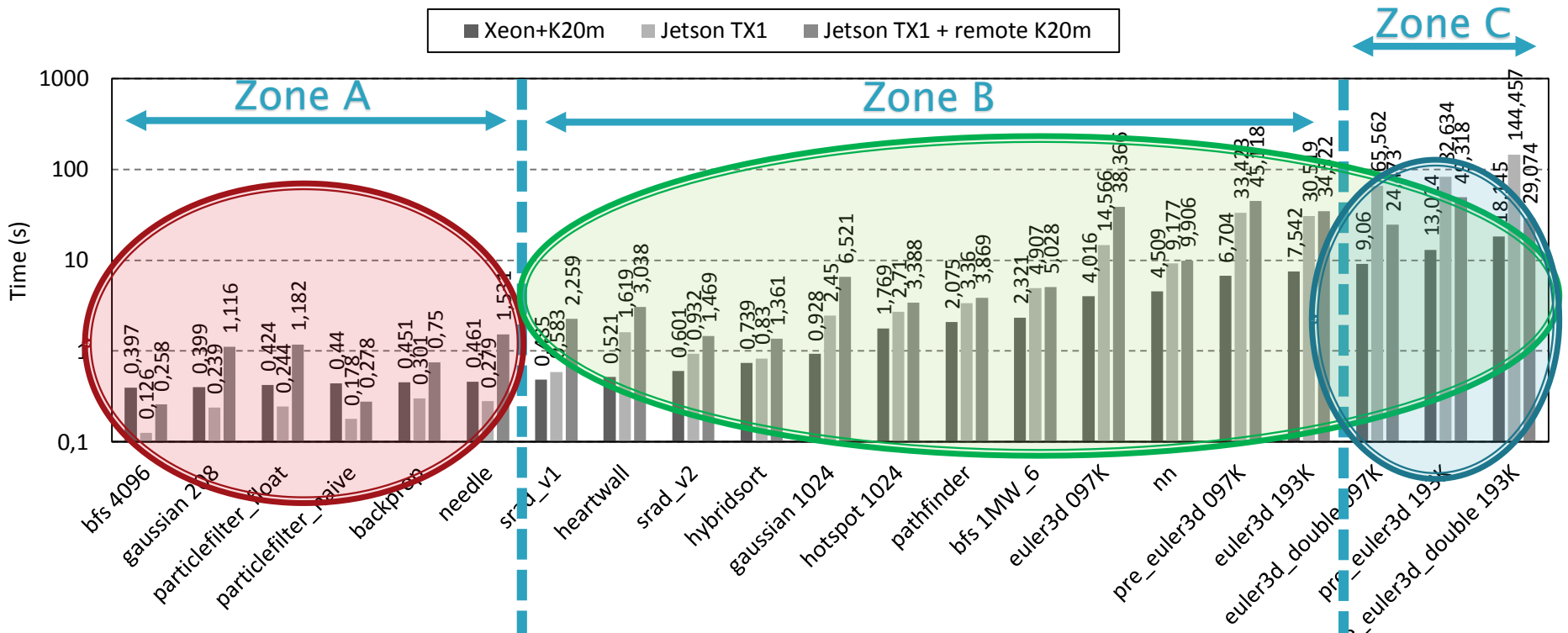
Offloading GPU workload from ARMs to remote Xeon GPU servers



Short execution time:
ARM wins

Medium and large execution time:
Xeon wins

Offloading GPU workload from ARMs to remote Xeon GPU servers



Short execution time:
ARM wins

Medium and large execution time:
Xeon wins

Large execution time:
remote GPU better than ARM

Offloading GPU workload from ARMs to remote Xeon GPU servers

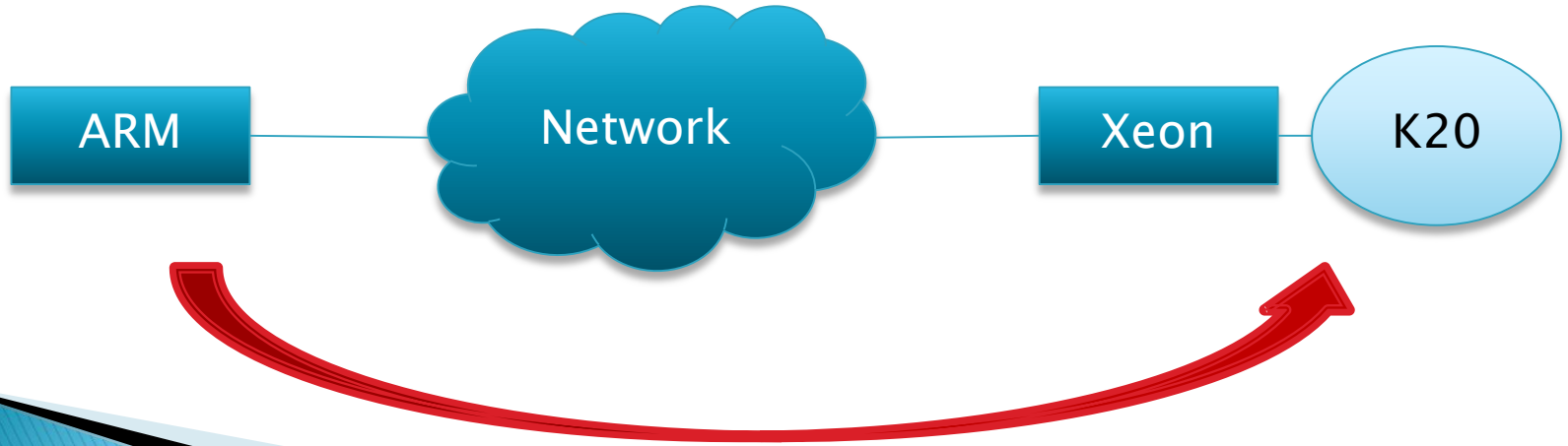
- Three scenarios:
  1. ARM
  2. Xeon+GPU
  3. ARM+remote GPU

- Performance comparison:
  - Rodinia benchmarks
  - **Gaussian benchmark varying problem size**
  - Gaussian benchmark, fixed problem size, sharing GPU among multiple ARM clients

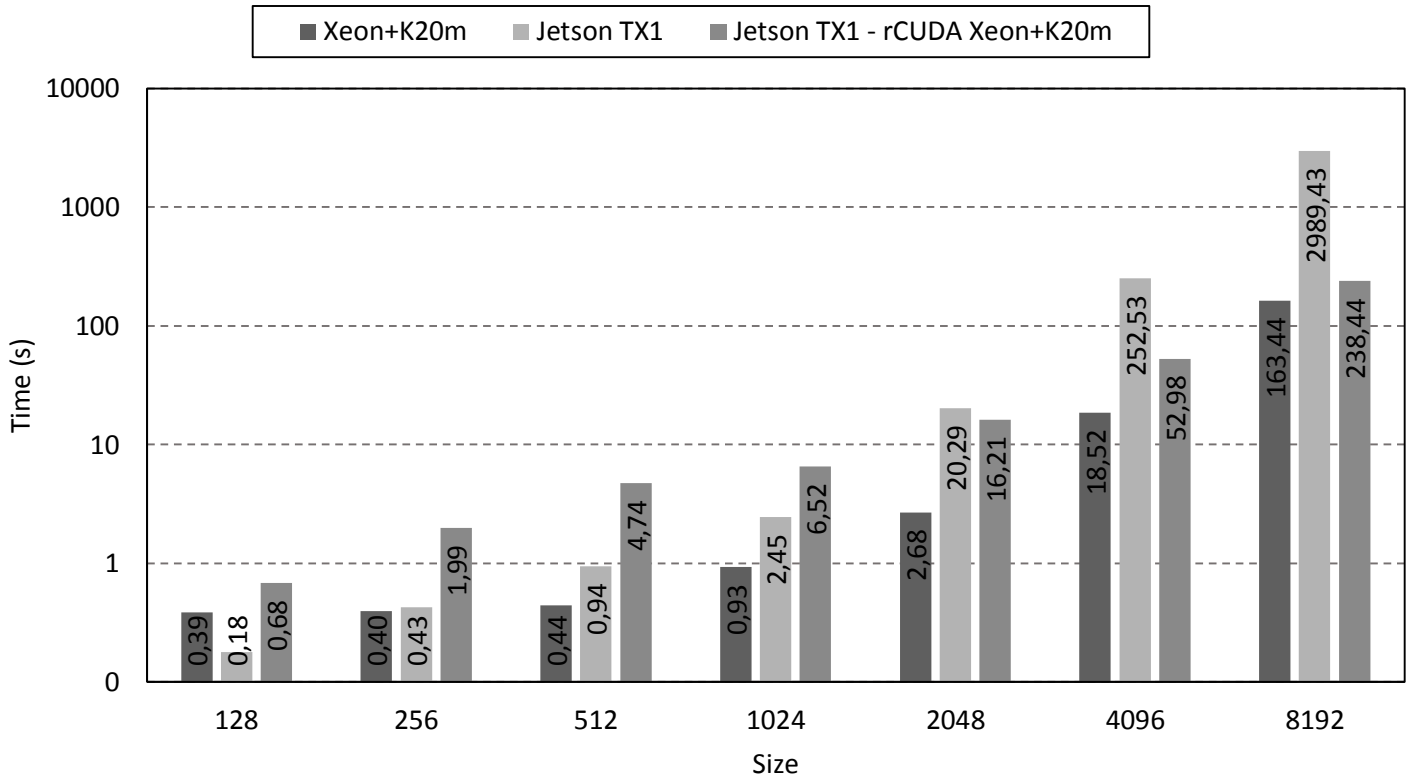
Offloading GPU workload from ARMs to remote Xeon GPU servers

- Local GPU:

- Remote GPU:

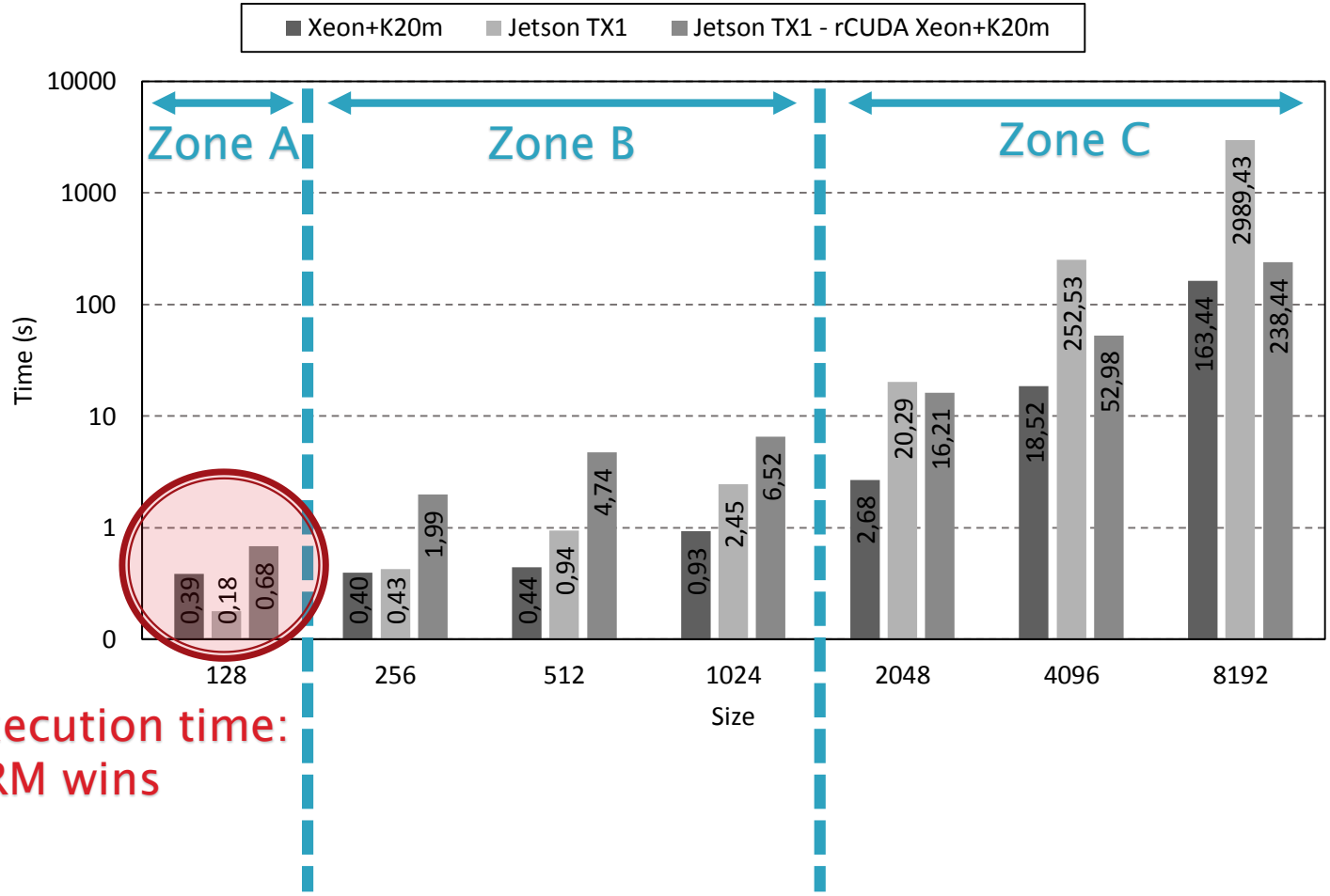


Offloading GPU workload from ARMs to remote Xeon GPU servers



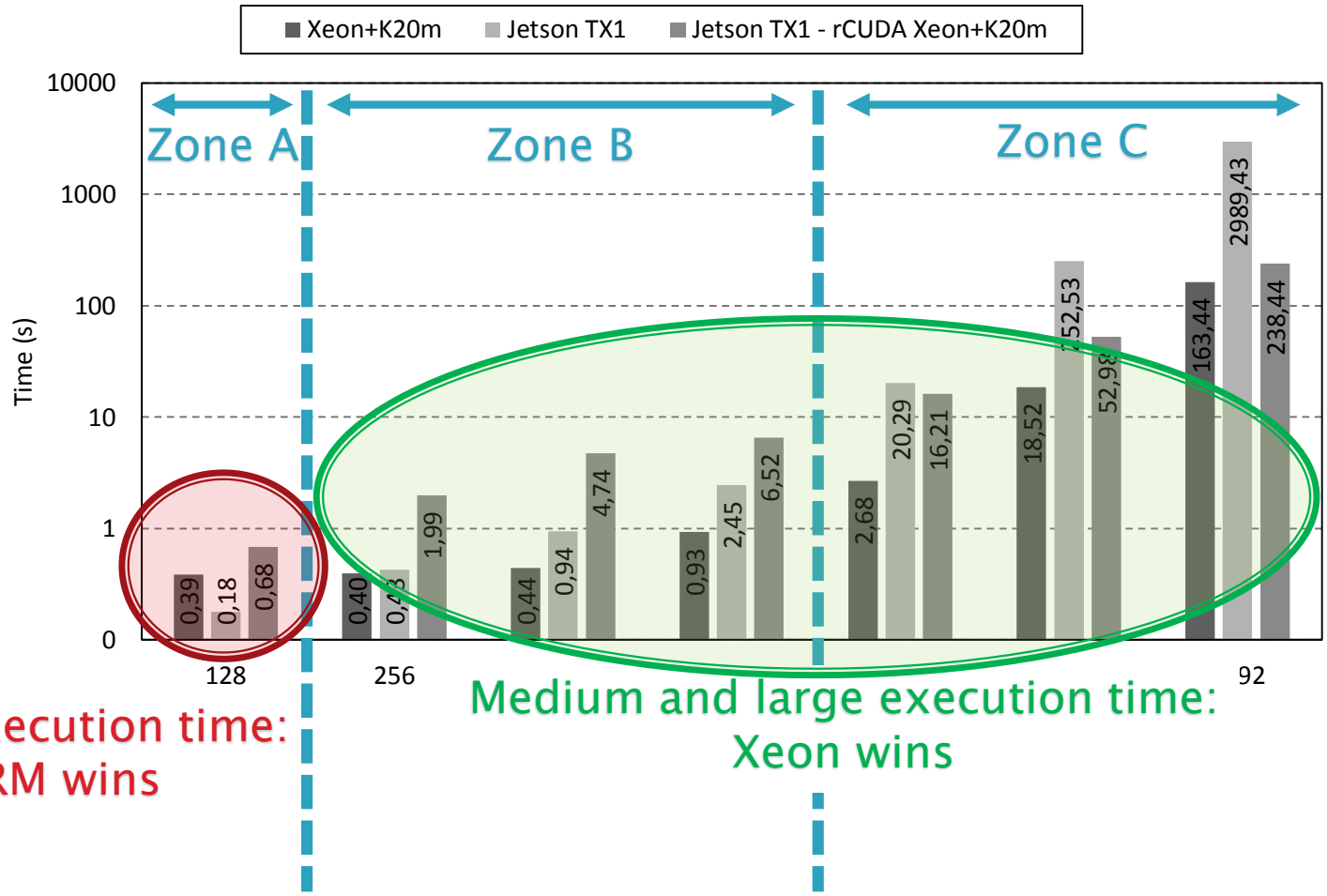
Comparison of Rodinia benchmark "gaussian" varying the problem size, and executed in three different scenarios: (i) a regular server with one NVIDIA K20 GPU, (ii) a NVIDIA Jetson TX1, and (iii) a NVIDIA Jetson TX1 offloading GPU work to a remote regular server with one NVIDIA K20 GPU.

Offloading GPU workload from ARMs to remote Xeon GPU servers

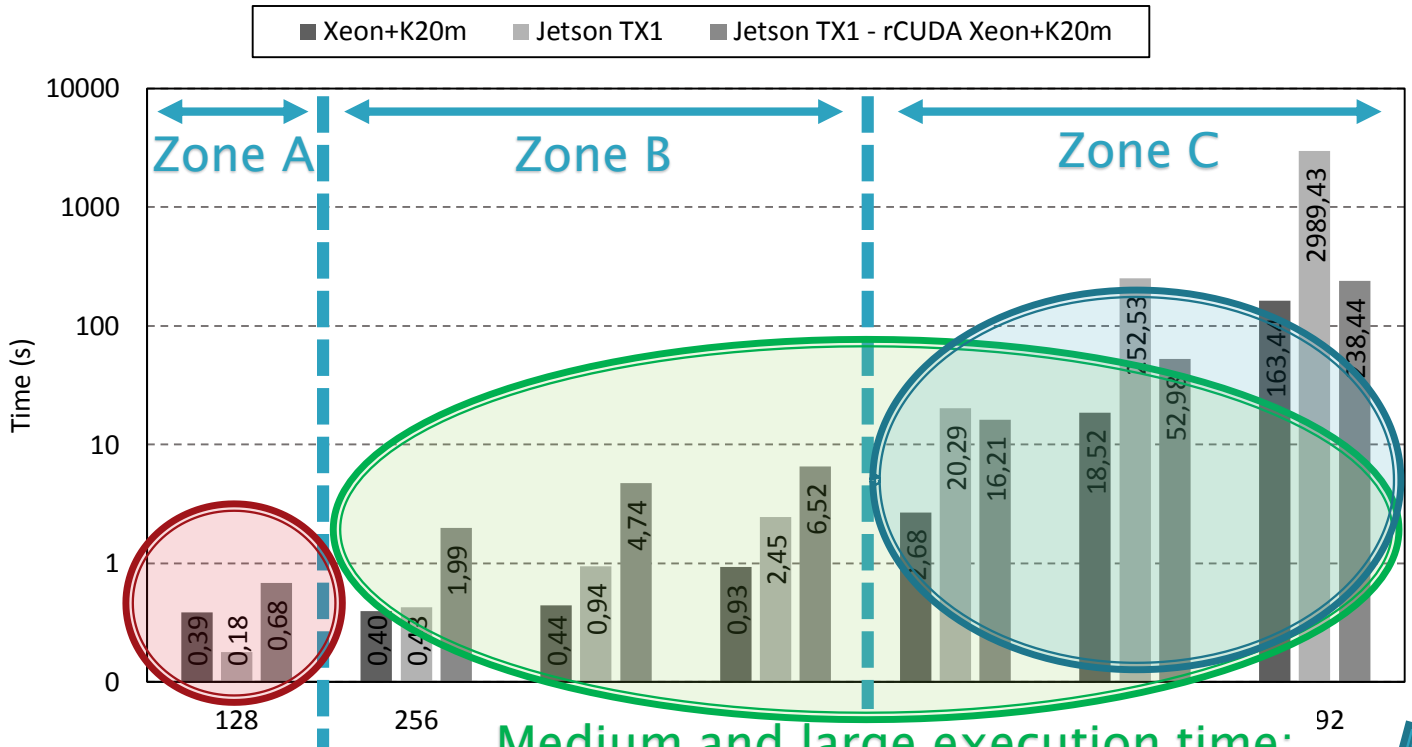


Short execution time: ARM wins

Offloading GPU workload from ARMs to remote Xeon GPU servers



Offloading GPU workload from ARMs to remote Xeon GPU servers



Short execution time: ARM wins

Medium and large execution time: Xeon wins

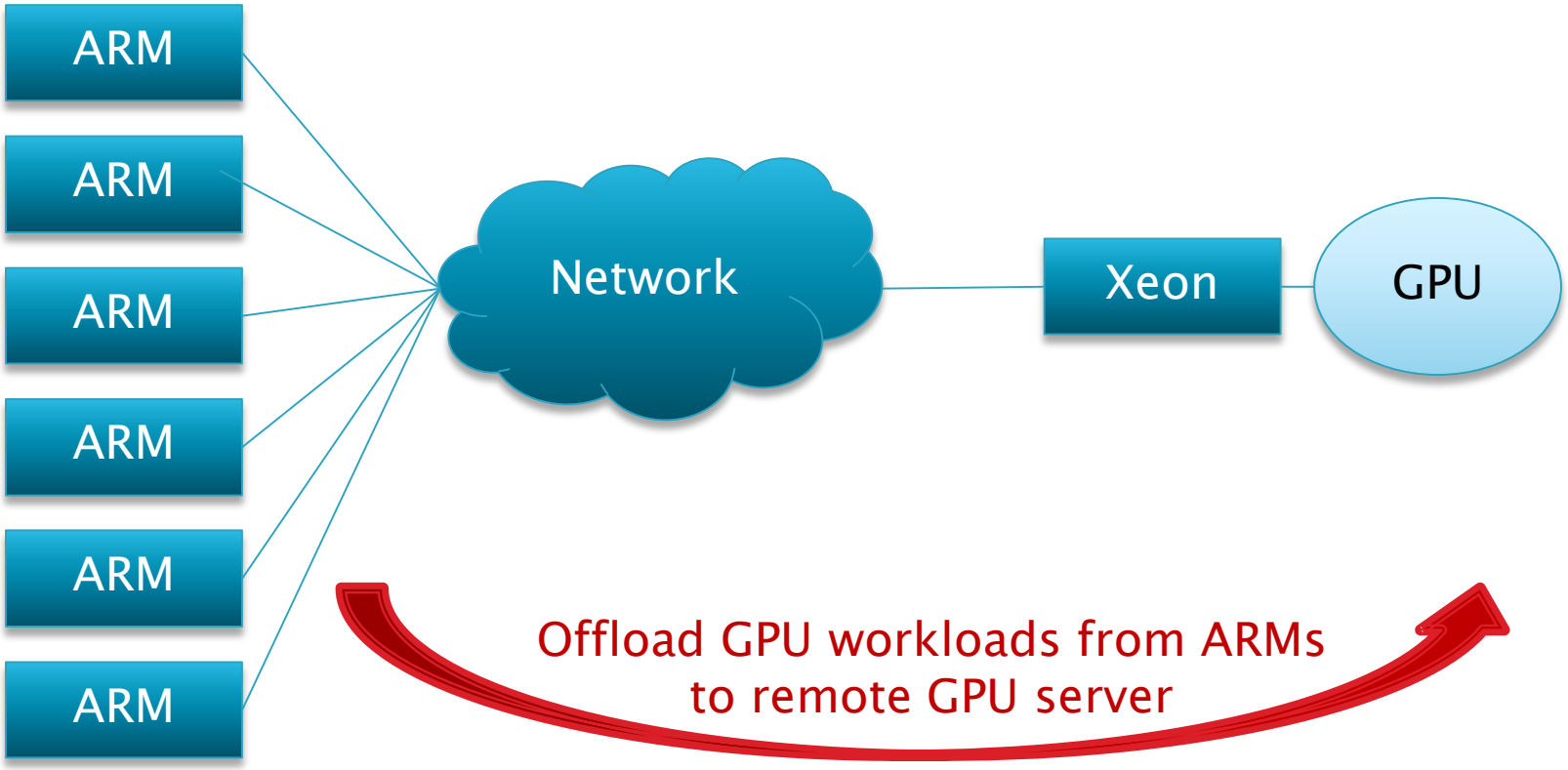
Large execution time: remote GPU better than ARM

Offloading GPU workload from ARMs to remote Xeon GPU servers

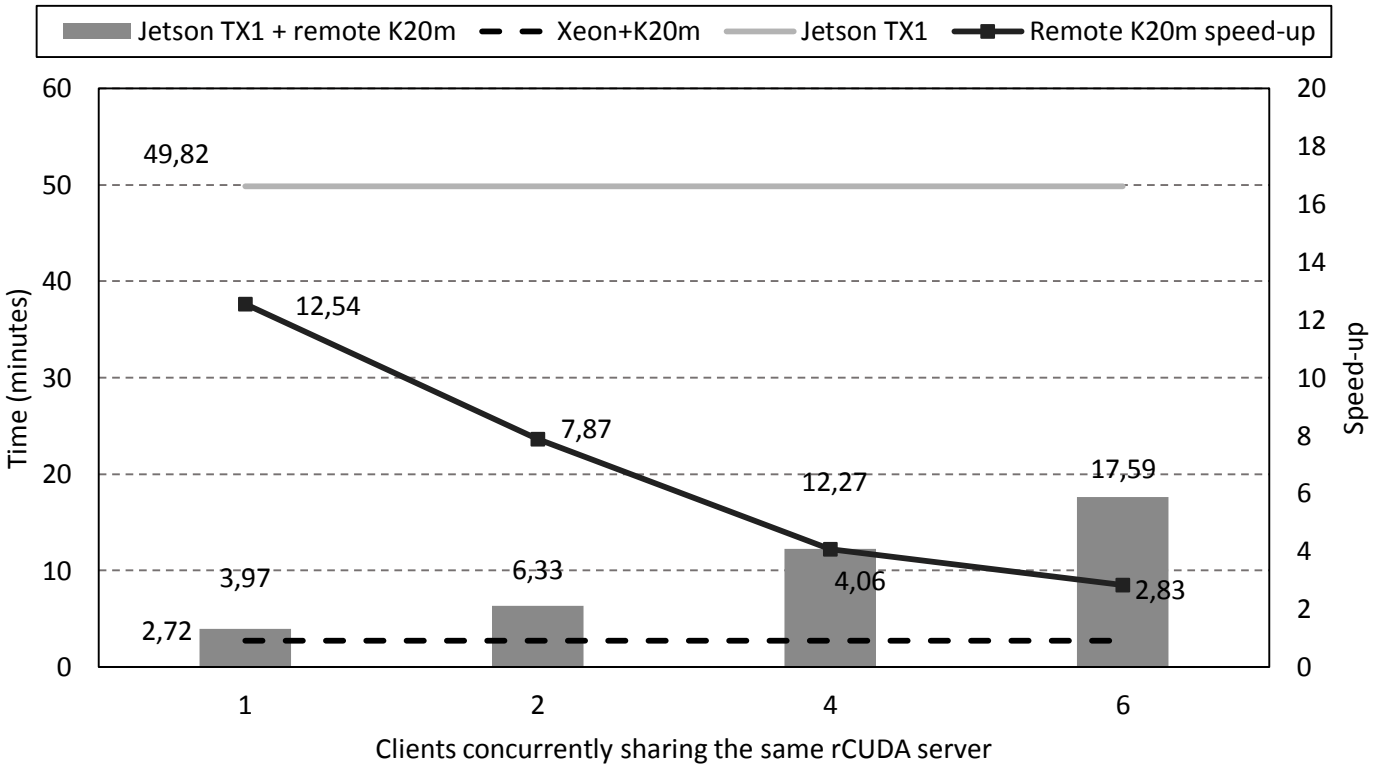
- Three scenarios:
  1. ARM
  2. Xeon+GPU
  3. ARM+remote GPU

- Performance comparison:
  - Rodinia benchmarks
  - Gaussian benchmark varying problem size
  - **Gaussian benchmark, fixed problem size, sharing GPU among multiple ARM clients**

Offloading GPU workload from ARMs to remote Xeon GPU servers

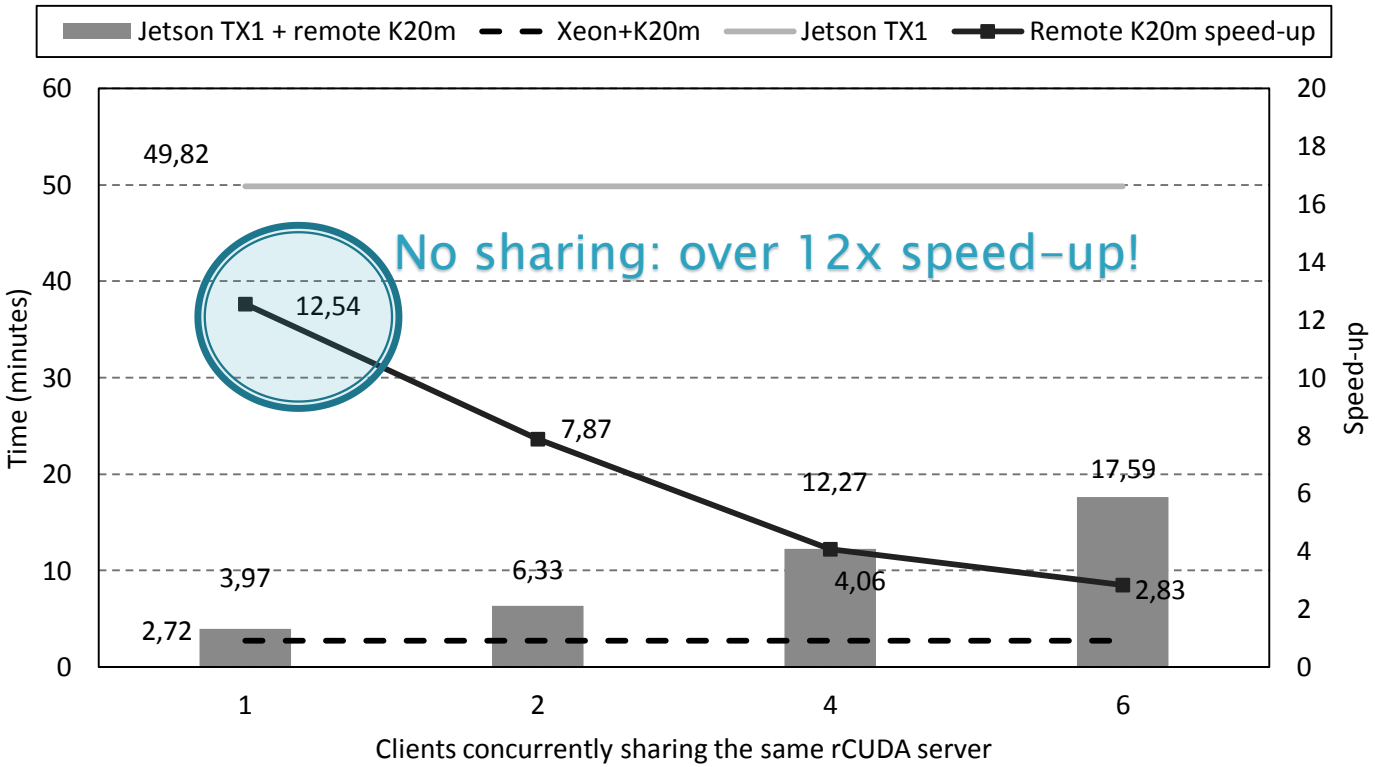


Offloading GPU workload from ARMs to remote Xeon GPU servers



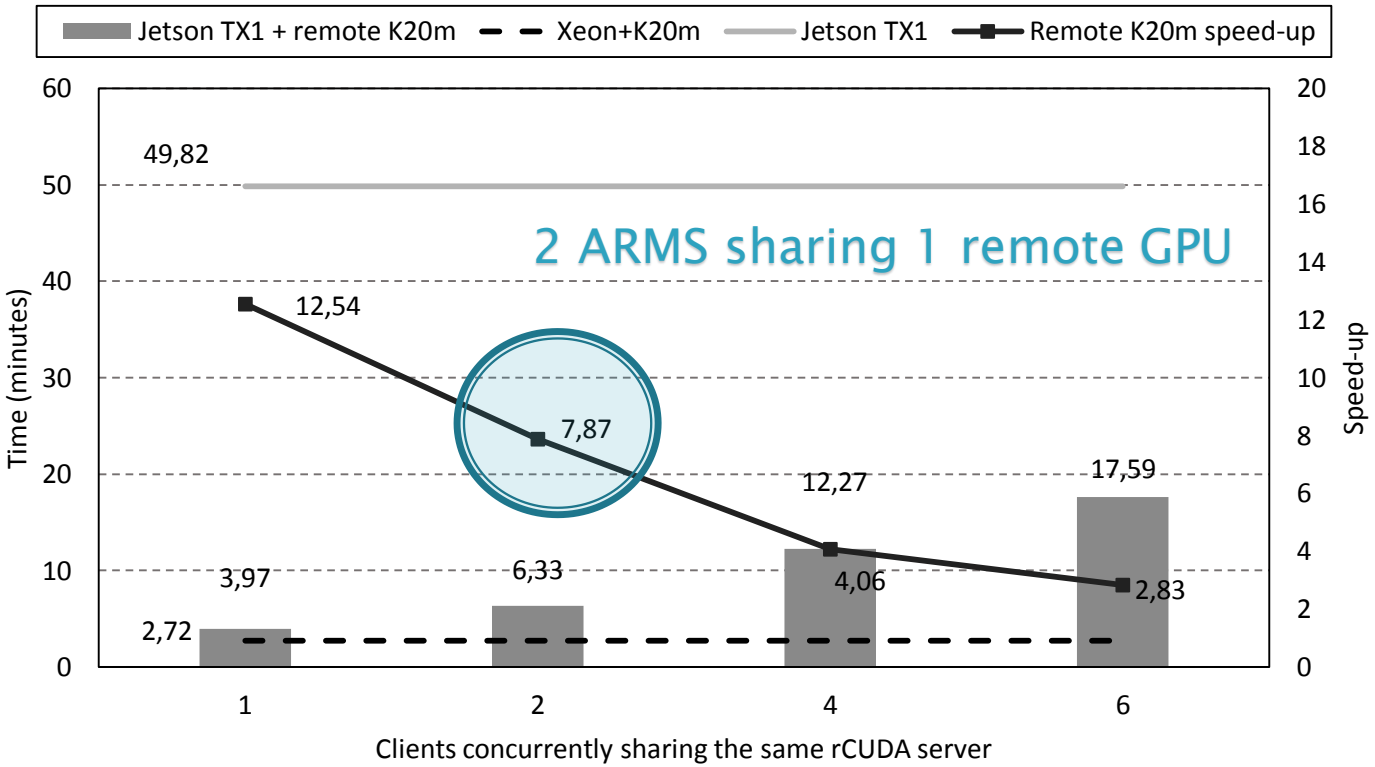
Comparison of Rodinia benchmark "gaussian" with problem size 8,192, executed in three different scenarios: (i) server with one K20, (ii) Jetson TX1, and (iii) Jetson TX1 offloading GPU work to a remote server with one K20 (in this scenario, the number of concurrent clients sharing the same remote server varies from 1 up to 6).

Offloading GPU workload from ARMs to remote Xeon GPU servers



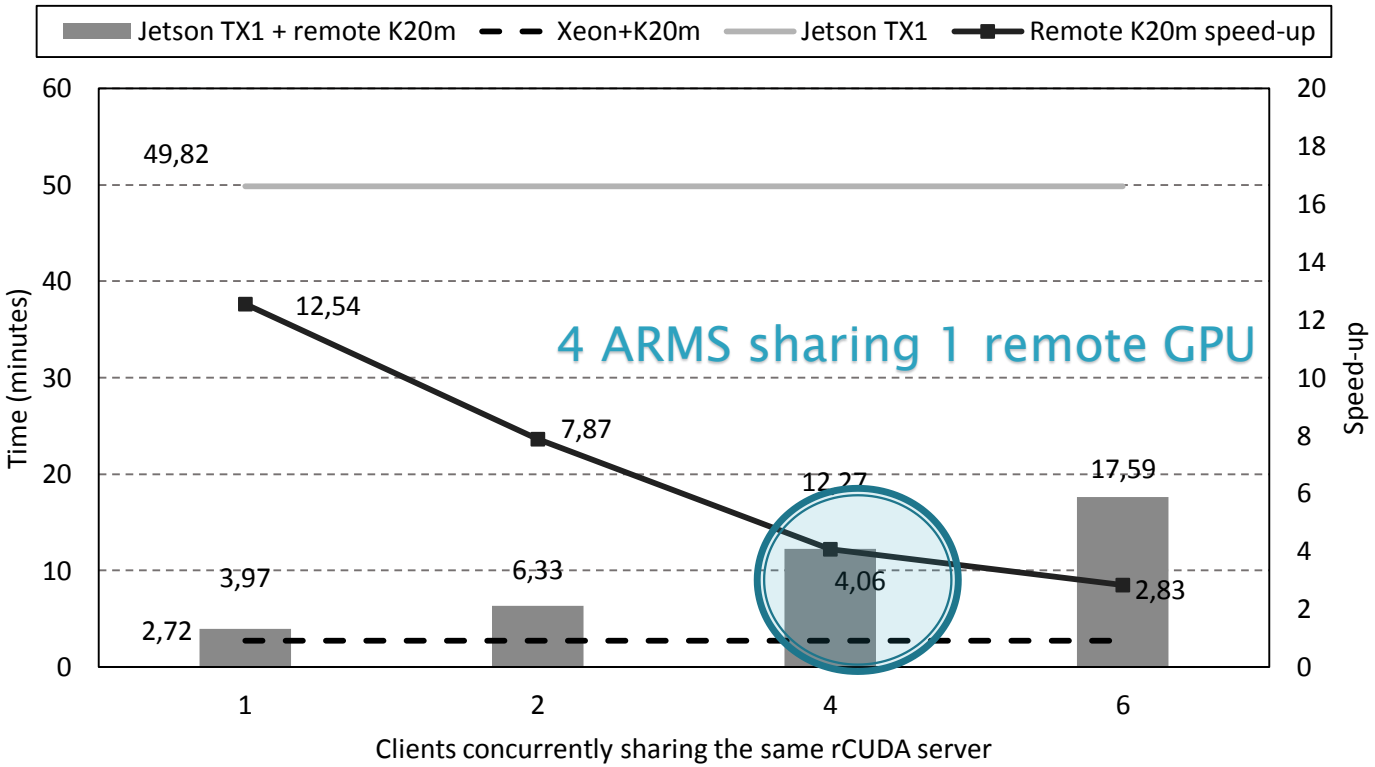
Comparison of Rodinia benchmark "gaussian" with problem size 8,192, executed in three different scenarios: (i) server with one K20, (ii) Jetson TX1, and (iii) Jetson TX1 offloading GPU work to a remote server with one K20 (in this scenario, the number of concurrent clients sharing the same remote server varies from 1 up to 6).

Offloading GPU workload from ARMs to remote Xeon GPU servers



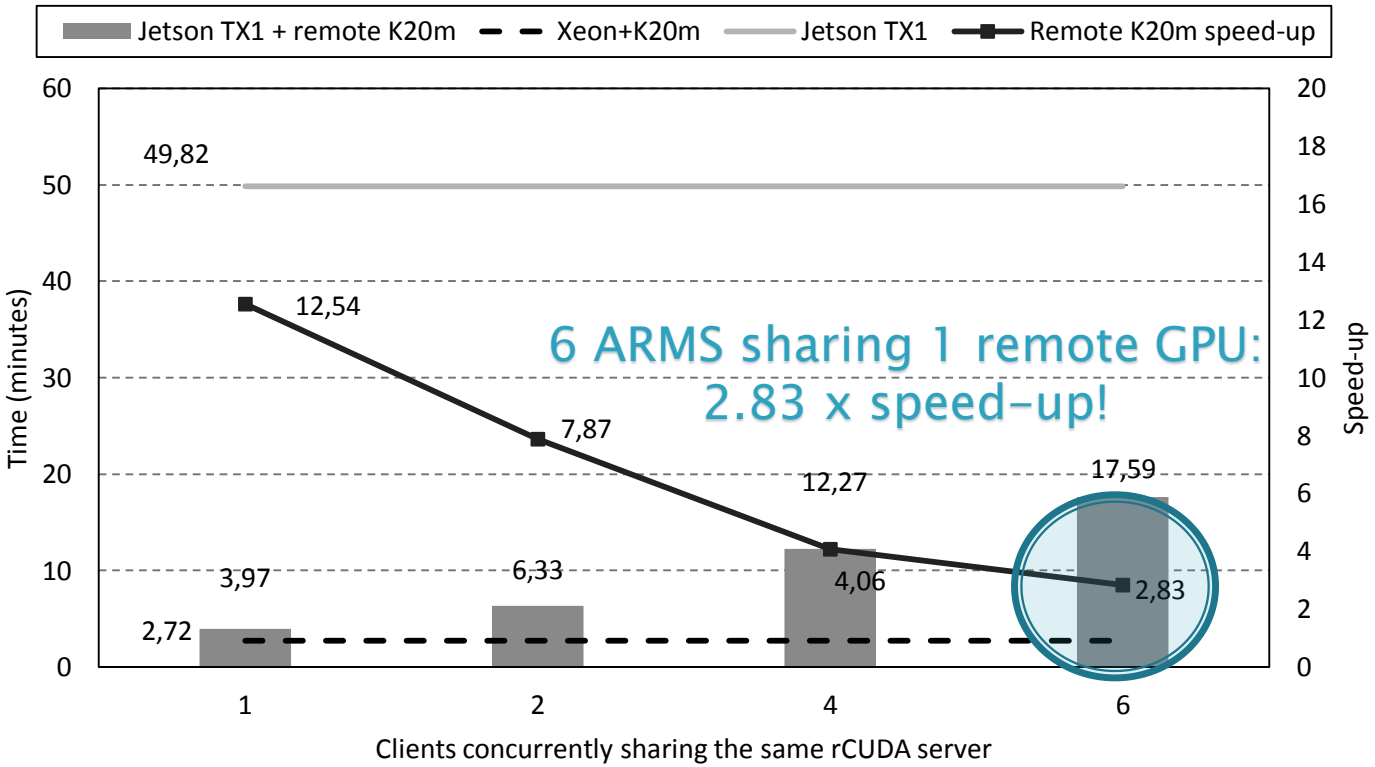
Comparison of Rodinia benchmark "gaussian" with problem size 8,192, executed in three different scenarios: (i) server with one K20, (ii) Jetson TX1, and (iii) Jetson TX1 offloading GPU work to a remote server with one K20 (in this scenario, the number of concurrent clients sharing the same remote server varies from 1 up to 6).

Offloading GPU workload from ARMs to remote Xeon GPU servers



Comparison of Rodinia benchmark "gaussian" with problem size 8,192, executed in three different scenarios: (i) server with one K20, (ii) Jetson TX1, and (iii) Jetson TX1 offloading GPU work to a remote server with one K20 (in this scenario, the number of concurrent clients sharing the same remote server varies from 1 up to 6).

Offloading GPU workload from ARMs to remote Xeon GPU servers



Comparison of Rodinia benchmark "gaussian" with problem size 8,192, executed in three different scenarios: (i) server with one K20, (ii) Jetson TX1, and (iii) Jetson TX1 offloading GPU work to a remote server with one K20 (in this scenario, the number of concurrent clients sharing the same remote server varies from 1 up to 6).

Outline

- Introduction
- GPU Virtualization
- Approach proposed
- The NVIDIA Jetson TX1 system
- Performance evaluation
- Conclusions and future work

Conclusions

- Proposal for improving efficiency of future exascale systems
- Using rCUDA to provide ARMs access to high-end remote GPUs
- Execution time speed-up by up to 12x
- ARM power efficiency is leveraged along with the high-end accelerators
- Accelerators dynamically shared among multiple ARMs

Future work

- Analysis of this proposal with HPC apps
- Include energy and power measurements
- Validate if our proposal is able to contribute to energy efficiency in future exascale systems

rCUDA
remote CUDA

Get a free copy of rCUDA at

http://www.rcuda.net

@rcuda_