VMAcS: **V**irtualization and Resource **M**anagement in **Ac**celerator-based Heterogeneous Multi-core **S**ystems

Presented by: Vishakha Gupta

Georgia Tech: Karsten Schwan, Harshvardhan Kharche, Ada Gavrilovska

HP Labs: Niraj Tolia, Vanish Talwar, Partha Ranganathan

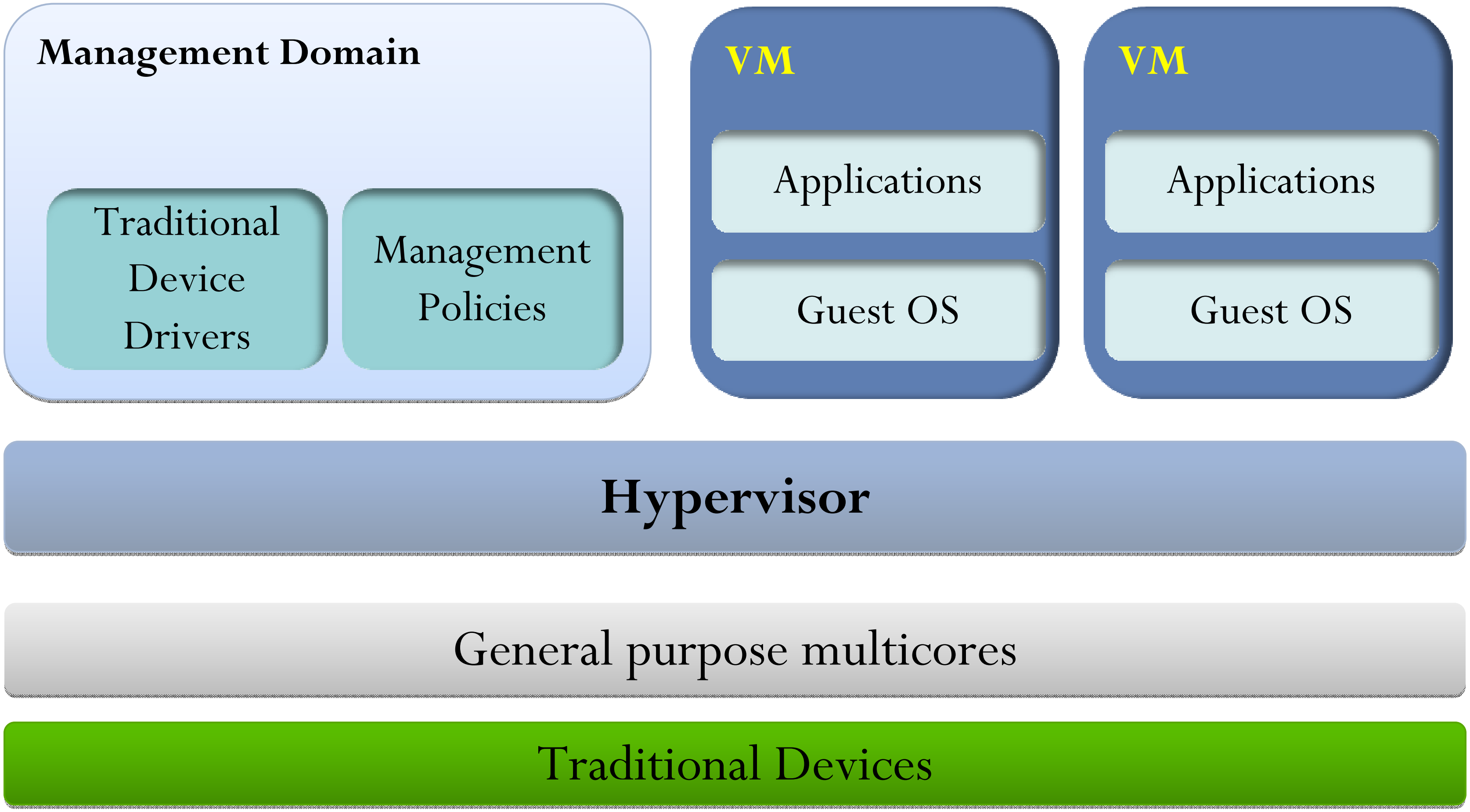
In a Nutshell

- Motivation
  - Industry trend towards heterogeneous multicores
  - Benefits offered by virtualization for broader application domain
- Problem statement
  - Implement and evaluate virtualization of accelerator based systems
  - Provide performance guarantees to guests running on such a system
- Work done
  - Virtualized specific off-chip accelerator – NVIDIA GPU

Outline

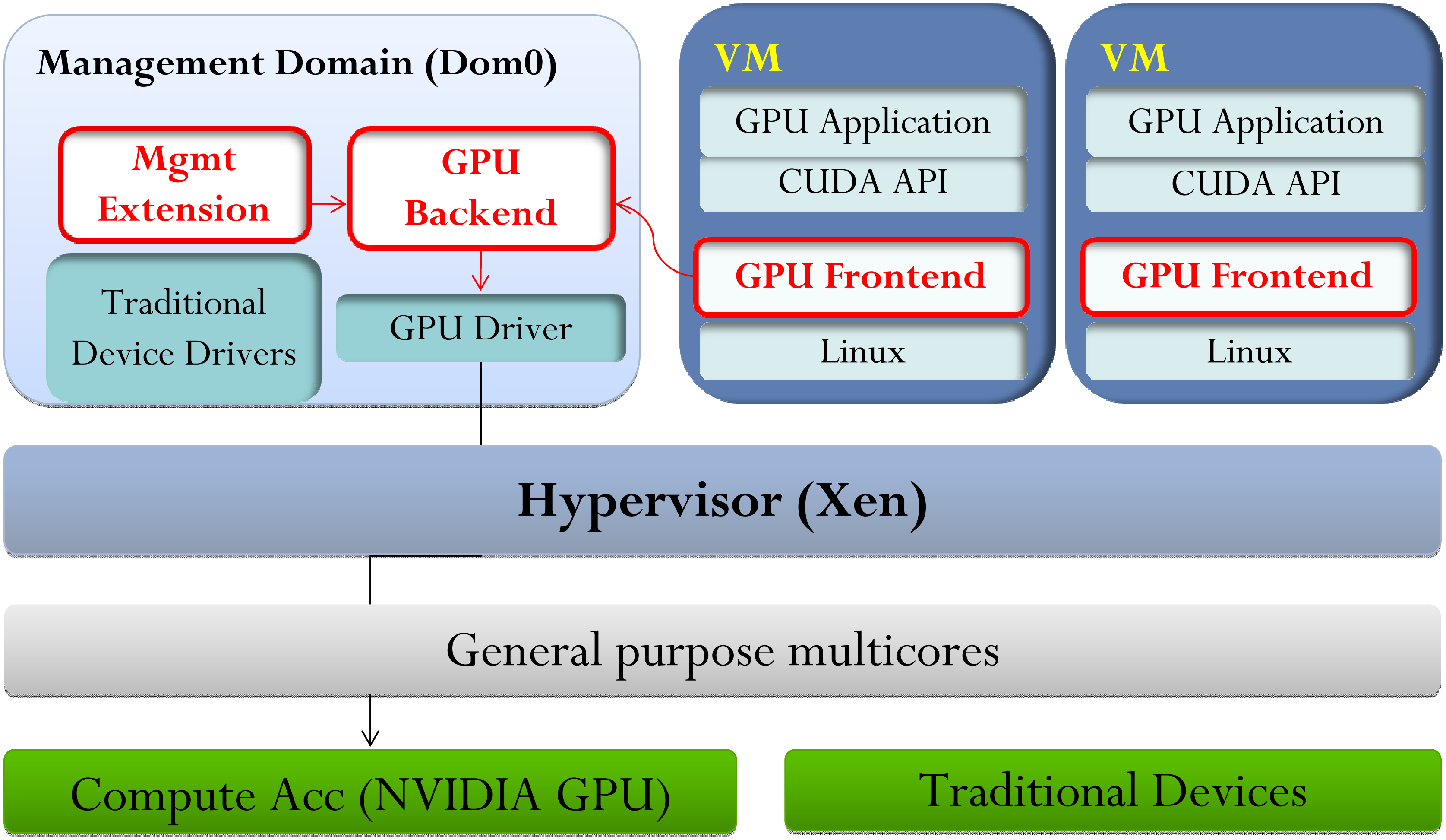
- System architecture
  - Virtualized Homogeneous Multicore Systems
  - Virtualization of Accelerator based Systems
- Management extension
- Evaluation
- Related work
- Future work and conclusion

Virtualized Homogeneous Multicore Systems



Virtualization of Accelerator based Systems

Extending Xen for GPU

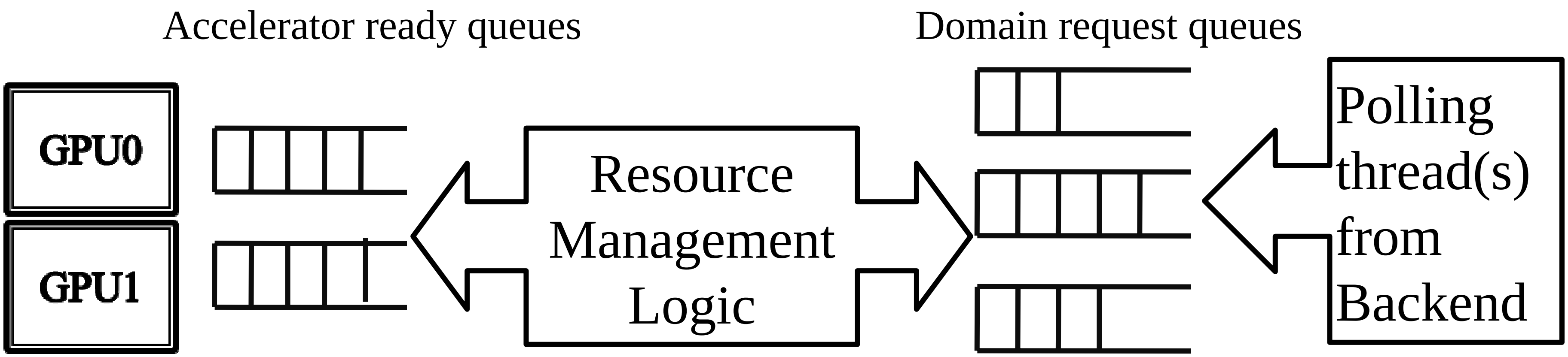


NVIDIA's CUDA – Compute Unified Device Architecture for managing GPUs

Outline

- System architecture
- Management extension
  - Scheduling in Dom0
  - System highlights
- Evaluation
- Related work
- Future work and conclusion

Dom0 – Management extension



Accelerator ready queues

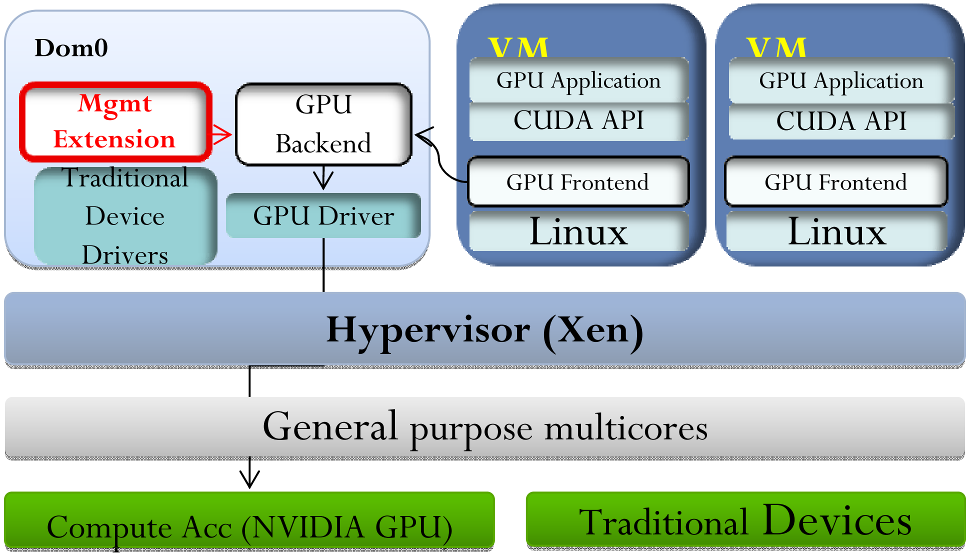
Domain request queues

GPU0

GPU1

Resource Management Logic

Polling thread(s) from Backend



Dom0

Mgmt Extension

GPU Backend

Traditional Device Drivers

GPU Driver

VM

GPU Application

CUDA API

GPU Frontend

Linux

VM

GPU Application

CUDA API

GPU Frontend

Linux

Hypervisor (Xen)

General purpose multicores

Compute Acc (NVIDIA GPU)

Traditional Devices

System Highlights

- Frontend:
  - Applications in guest VM can be multi-threaded
  - Currently one application at a time
  - Call path is synchronous from application's view
- Backend
  - Multiple domains
  - Multiplex resources across multiple GPUs
  - Simple load balancing and admission control
  - A general framework for virtualization and resource management

Outline

- System architecture
- Management extension
- Evaluation
  - Testbed details
  - Benchmarks
  - Preliminary results
  - Discussion
- Related work
- Future work and conclusion

Testbed Details

- Hardware configuration:
  - Xeon quad-core @ 2.5GHz and 2GB memory
  - NVIDIA 8800 GTX PCIe card
- Software configuration
  - Xen 3.2.1 running 2.6.18 Linux kernel
  - CUDA SDK 1.1 with gpu driver 169.09

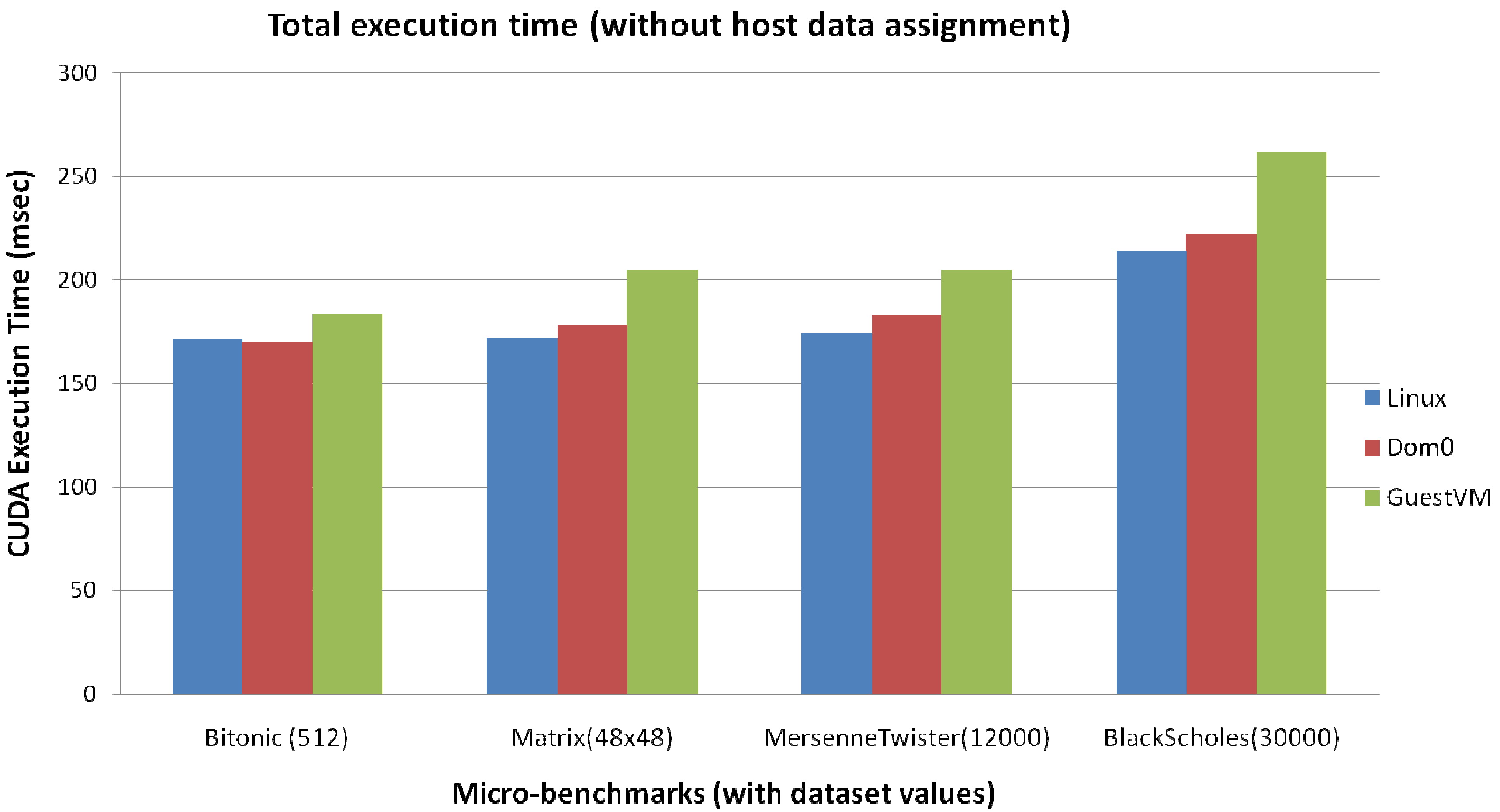
Micro-benchmarks

- Bitonic sorting
  - Smaller data size sort
  - Tested with 512 elements
- Matrix multiplication
  - Used 48x48 floating pt. matrices for the numbers
- MersenneTwister
  - Pseudo-random number generator
  - Tested with generation of 12000 random numbers
- BlackScholes
  - Financial algorithm for call/put option prices
  - 30000 options with 512 iterations

Benchmark - CDO

- Collateralized debt obligation pricing model
- Financial product which takes the risk of a portfolio and segments into several tranches
- Multiple compute kernels involved
  - Generating uniform random numbers
  - Cholesky decomposition
  - Matrix multiplication, Sorting etc.
- Adjustable number of iterations and portfolio size

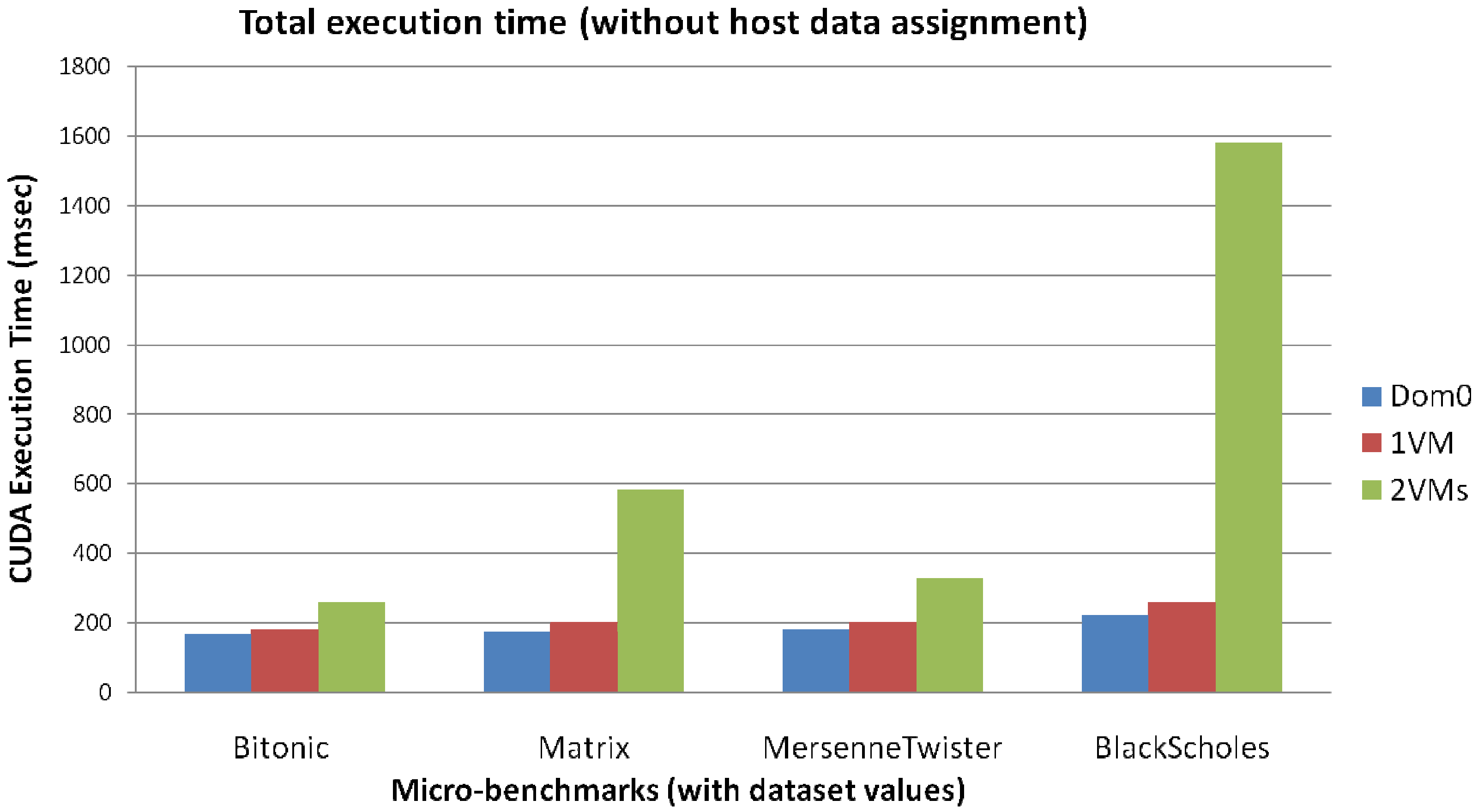
Guest GPU Access Performance



Less than 15% slowdown in the worst case

Multiple Guest Scenario

No Resource Management



Without resource management, calls can get variably delayed due to interference from other domain

Ongoing and Future Work

- Ongoing
  - Optimizations to GPU virtualization code
  - Thorough measurement of benchmarks
  - Scheduling in Dom0 and related measurements

- Future
  - Heterogeneous multicore scheduling
  - SLA management policies
  - Scalability and stability models/analyses
  - Power-awareness in the scheduler

Related Work

- Scheduling extension – [Cypress], [Xen Credit Scheduling], [QoS Adaptive Communication], [Intel Shared ISA Heterogeneity]
  - Differences in (system design – heterogeneity)
- GPU Virtualization: [OpenGL], [VMWare DirectX] [VMGL]
  - Difference in API level virtualization
- Other related work
  - Accelerator Frontend or multi-core programming models: [Georgia Tech Harmony], [Georgia Tech Cellule], [IBM ALF], [Intel QuickAssist], [9p from Plan 9]
  - Some examples: [Intel Tolapai], [Intel Larrabee], [AMD Fusion], [IBM Cell], [LANL Roadrunner]