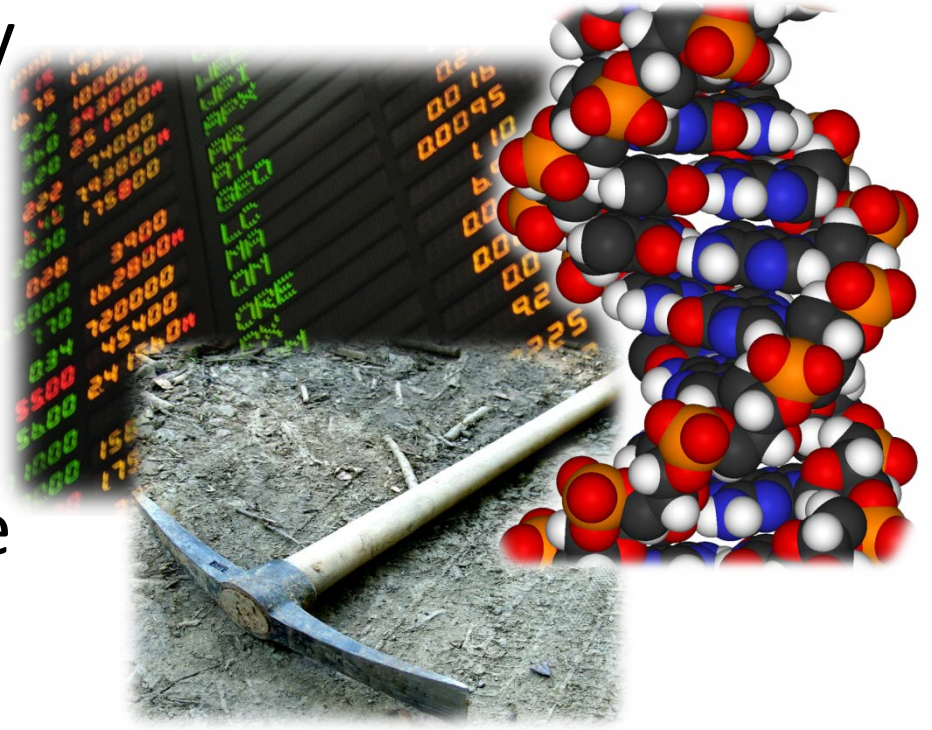
Accelerating The Cloud
with Heterogeneous Computing

Sahil Suneja, Elliott Baron, Eyal de Lara, Ryan Johnson

GPGPU Computing

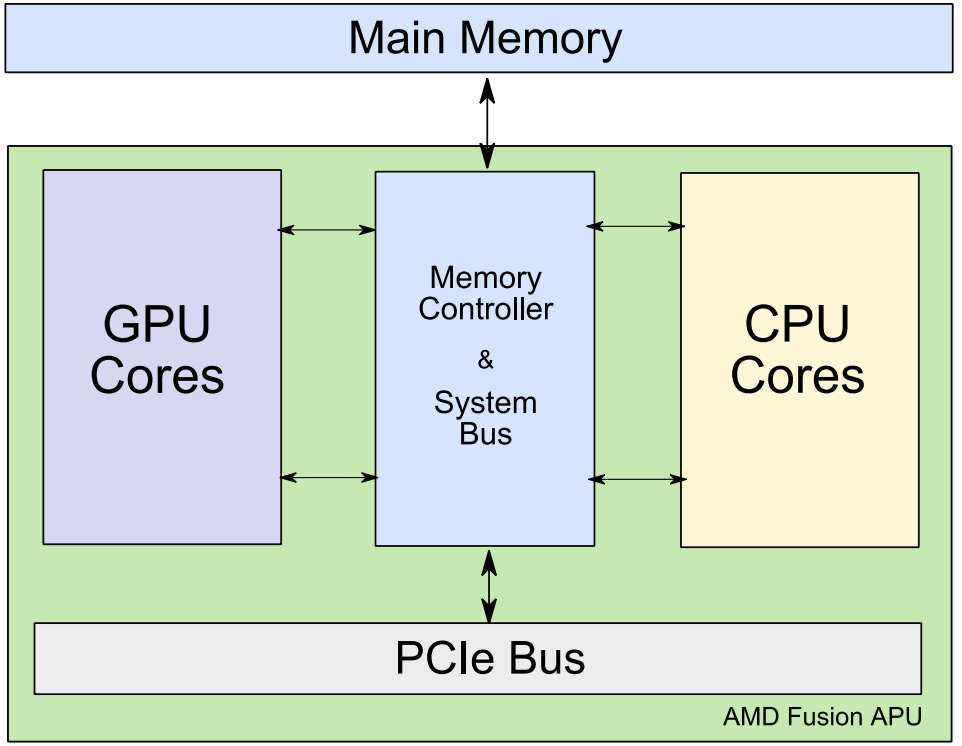
- Data Parallel Tasks
  - Apply a fixed operation in parallel to each element of a data array
- Examples
  - Bioinformatics
  - Data Mining
  - Computational Finance
  - NOT Systems Tasks
    - High-latency memory copying



UNIVERSITY OF TORONTO

Game Changer – On-Chip GPUs

- Processors combining CPU/GPU on one die
- AMD Fusion APU, Intel Sandy/Ivy Bridge
- Share Main Memory
- Very Low Latency
- Energy Efficient

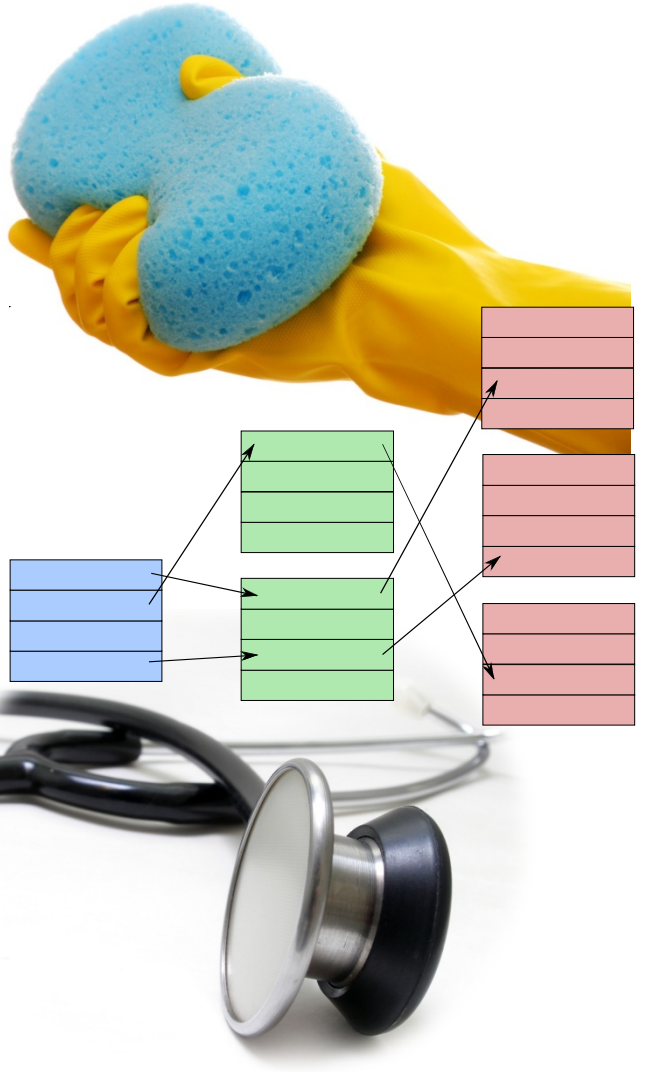


Accelerating The Cloud

- Use GPUs to accelerate Data Parallel Systems Tasks
  - Better Performance
  - Offload CPU for other tasks
  - No Cache Pollution
  - Better Energy Efficiency (Silberstein et al, SYSTOR 2011)
- Cloud Environment particularly attractive
  - Hybrid CPU/GPU will make it to the data center
  - GPU cores likely underutilized
  - Useful for Common *Hypervisor Tasks*

Data Parallel Cloud Operations

- Memory Scrubbing
- Batch Page Table Updates
- Memory Compression
- Virus Scanning
- Memory Hashing



UNIVERSITY OF TORONTO

Hardware Management

- Complications
  - Different Privilege Levels
  - Multiple Users
- Requirements
  - Performance Isolation
  - Memory Protection

Hardware Management

- Management Policies
  - VMM Only
  - Time Multiplexing
  - Space Multiplexing

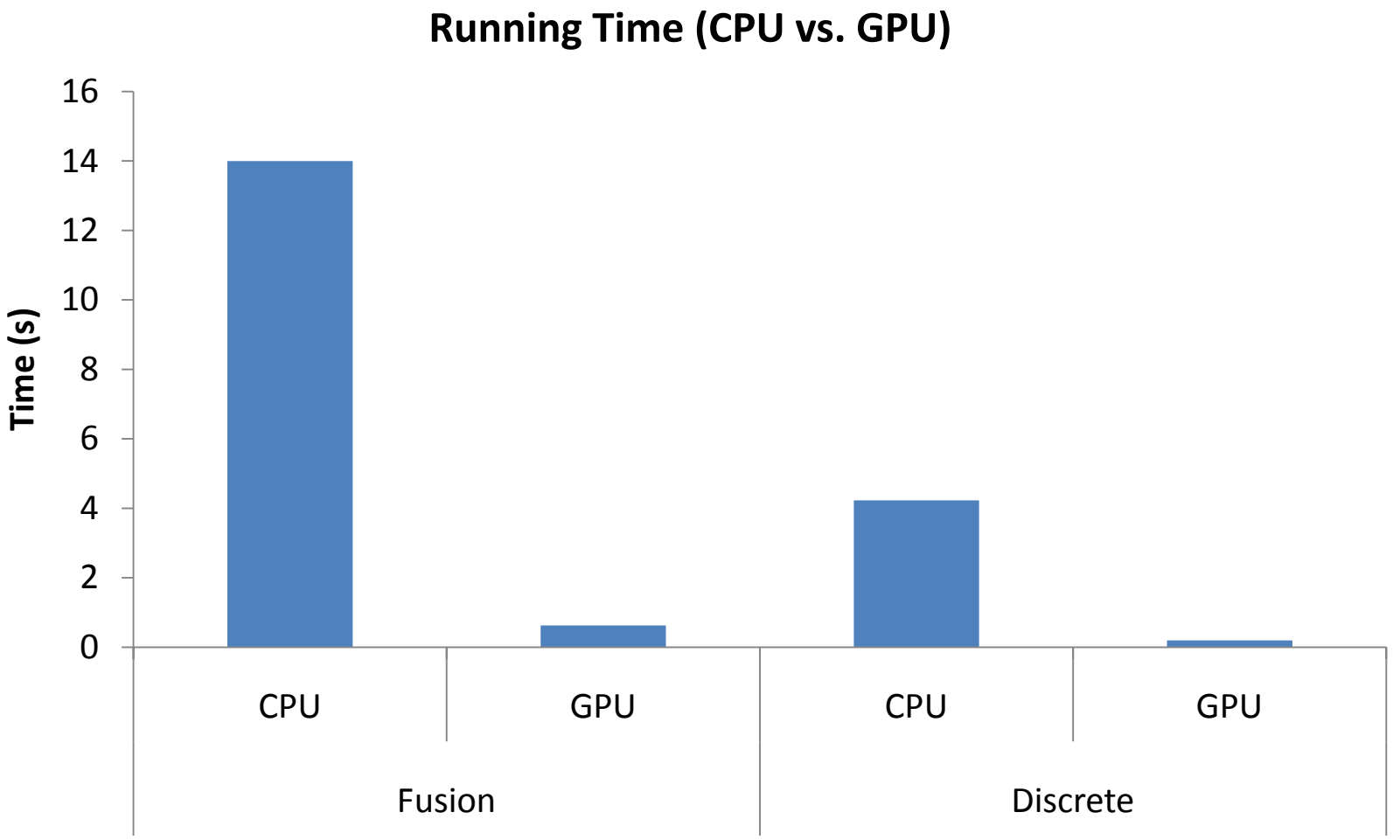
Memory Access

- All Tasks mentioned assume GPU can Directly Access Main (CPU) Memory
  - Many require Write Access
- Currently, CPU <-> GPU copying required!
  - Even though both share Main Memory
- Makes some tasks infeasible on GPU, others less efficient

Case Study – Page Sharing

- “De-duplicate” Memory
- Hashing identifies sharing candidates
- Remove all, but one physical copy
- Heavy on CPU
- Scanning Frequency ∝ Sharing Opportunities

Memory Hashing Evaluation



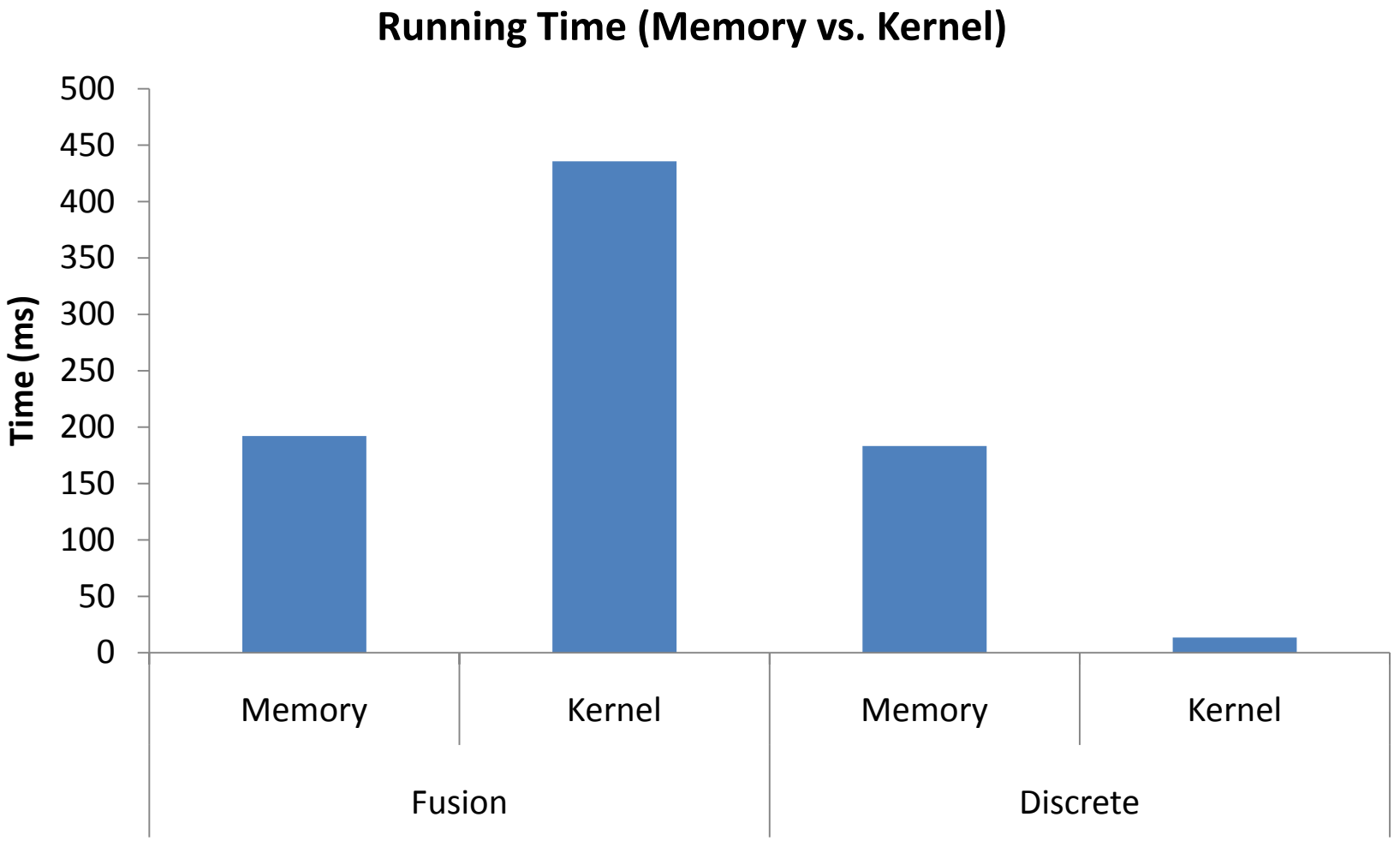
UNIVERSITY OF TORONTO

Conclusion/Summary

- Hybrid CPU/GPU Processors Are Here
- Get Full Benefit in Data Centres
  - Accelerate and Offload Administrative Tasks
- Need to Consider Effective Management and Remedy Memory Access Issues
- Memory Hashing Example Shows Promise
  - Over Order of Magnitude Faster

Extra Slides

Memory Hashing Evaluation



UNIVERSITY OF TORONTO

CPU Overhead

- Measure performance degradation of CPU-Heavy program
- Hashing via CPU = 50% Overhead
- Hashing via GPU = 25% Overhead
  - Without Memory Transfers = 11% Overhead