Design Space Exploration of Memory Model for Heterogeneous Computing

Jieun Lim¹ and Hyesoon Kim²

¹Seoul National University ²Georgia Institute of Technology

Introduction

- Heterogeneous computing has become a major architecture trend
- How to design memory system
  - Strongly coupled with architecture design and programming model
  - Difficult to compare models
- Goal
  - Understand a trade-off in memory system design decisions
  - Evaluate the overhead of design options

Memory Address Space Design Options

(a) Unified space
(b) Disjoint space
(c) Partially-shared space
(d) ADSM

- Unified space: identical address space for CPU and GPU
  - No explicit data transfer, but complicated TDM/MA designs
- Disjoint space
  - Available and easy to implement, but need explicit data transfer
  - Provides a simple space with discrete memories
- Partially-shared: only part of the space is shared
  - Convenience of using shared memory but overhead of managing between spaces
- ADSM: one CPU can access the entire memory, but the other cannot

Code Examples (1)

- Reduction example

Communication Options (1)

- Hardware design options for how data can be transferred between processing units

Communication Options (2)

- Physically shared cache
  - Shared object can be directly updated inside the shared cache
- Cache coherence
- Memory controllers
- Interconnection network + DMA
  - Uses an interconnection network to directly communicate without necessarily going through memory controllers
- I/O
  - System BUS such as PCE, or a processor BUS

Heterogeneous Architecture Summary

- None of the heterogeneous computing systems has employed a unified, fully-coherent, strong-consistent memory system yet
- Most proposed/existing systems have disjoint memory systems

Evaluation

- Benchmarks
- Parameters of modeling communication overhead

Evaluation of Five Heterogeneous Architecture Configuration

- Compare five systems
  - ISD - HETERO, unified and fully coherent
  - CPU+GPGPU: disjoint space + PCE
  - USB: partially-shared space + PCI aperture
  - IBM: ADSM+PCE
  - Fusion: disjoint space in memory controller

Programmability vs. Memory Address Space

- Different programming options affect how easy/difficult it is to write programs
- Use the number of source lines to indicate programmability
  - The number of additional source lines that are required to handle explicit data communication and data handling operations

Programmability vs. Memory Address Space

- Unified < partially-shared < ADSM < disjoint
  - Unified space does not require any special APIs
  - Disjoint memory space requires the most additional source code lines

Conclusion

- We exploited the design space of heterogeneous computing memory systems
- Memory space does not affect performance significantly
- Partially shared memory space is the most promising option
  - Provides many hardware design options and moderately good programmability