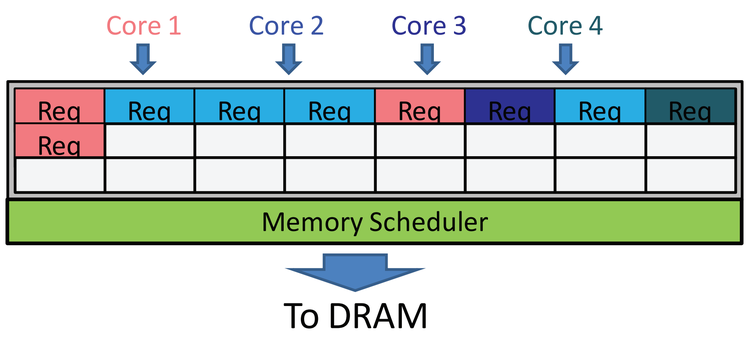
STAGED MEMORY SCHEDULING

Rachata Ausvarungnirun, Kevin Chang, Lavanya Subramanian, Gabriel H. Loh§, Onur Mutlu
(Carnegie Mellon University, §AMD Research)

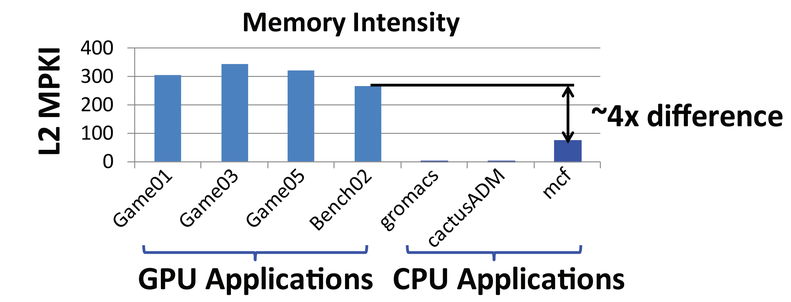
BACKGROUND & PROBLEMS

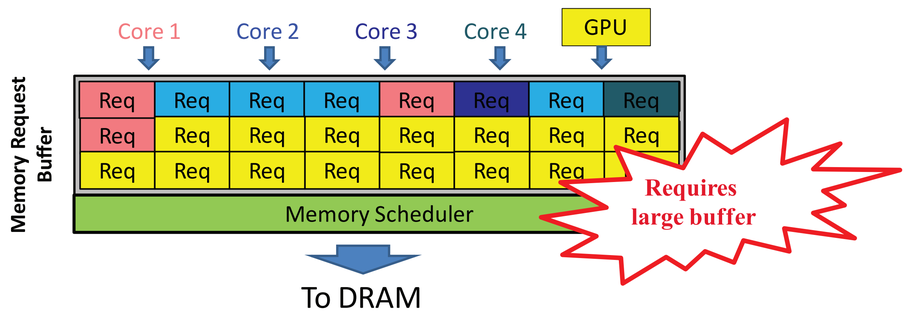
MEMORY SCHEDULING



GOALS: Maximize bandwidth utilization
Provide high throughput and fairness

CPU-GPU HETEROGENEOUS SYSTEM





Problem with large buffer:
- A large buffer requires more complicated logic to:
  - Analyze memory requests
  - Analyze application characteristics
  - Assign and enforce priorities
- This leads to high complexity, high power, large die area

OUR SOLUTION

KEY INSIGHTS

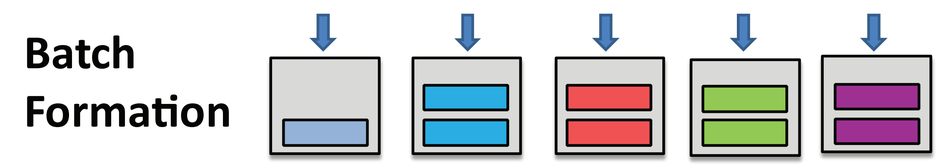
Three key functions of a memory scheduler
- Maximize row buffer hits
- Minimize interference across applications
- Satisfy DRAM timing constraints

Key observation: Memory controller does not have to perform all three at once

Our key idea: Decouple these tasks and distribute them across simpler HW stages

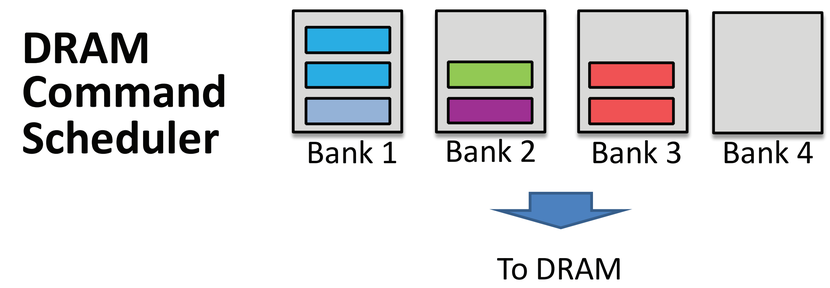
STAGED MEMORY SCHEDULING

1. Batch requests to the same row together



2. Schedule batches probabilistically from two policies:
   1. Shortest Job First: throughput oriented
   2. Round Robin: fairness oriented

3. Schedule requests from an already scheduled order



KEY RESULTS

METHODOLOGY

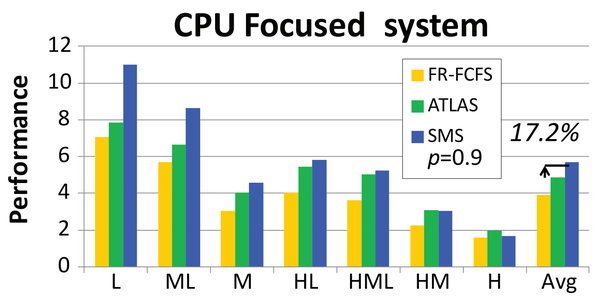
- 16 OoO CPU cores 1 GPU (AMD Radeon 5870)
- DDR3-1600 DRAM

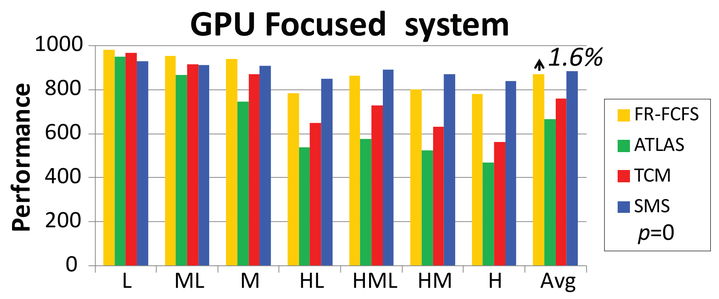
Workloads:
- CPU: SPEC 2006
- GPU: Recent games and GPU Benchmarks

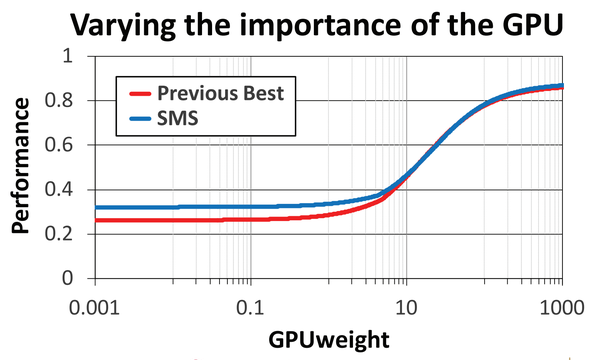
COMPLEXITY

- Compared to a row-hit first scheduler
  - 66% less area
  - 46% less static power

PERFORMANCE & FAIRNESS







Intel Science & Technology Center for Cloud Computing

Carnegie Mellon University · Georgia Tech · intel · Princeton University · UC Berkeley