CoNDA:
Efficient Cache Coherence Support for Near-Data Accelerators

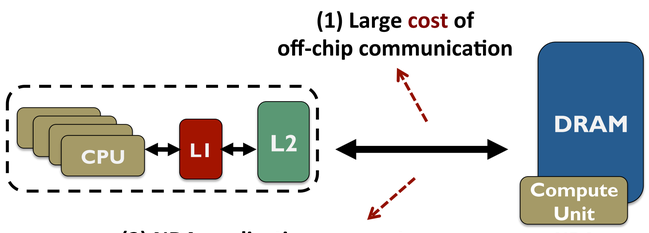
SAFARI

Amirali Boroumand, Saugata Ghose, Minesh Patel, Hasan Hassan, Brandon Lucia, Rachata Ausavarungnirun, Kevin Hsieh, Nastaran Hajinazar, Krishna Malladi, Hongzhong Zheng, Onur Mutlu

Carnegie Mellon · SAMSUNG · SFU · ETH zürich

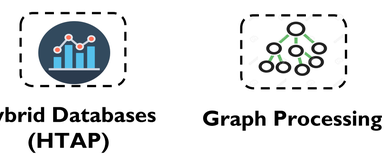
Coherence For NDAs

Challenge: Coherence between NDAs and CPUs



It is impractical to use traditional coherence protocols

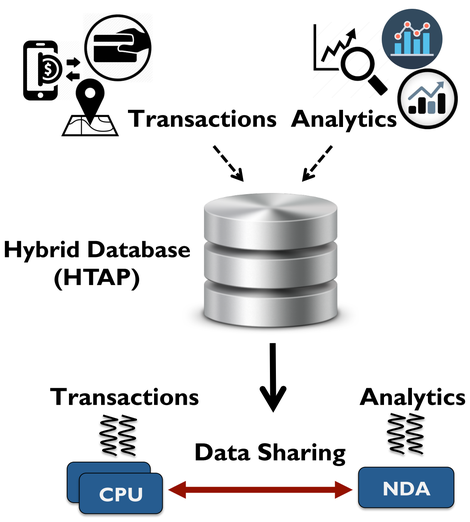
Application Analysis



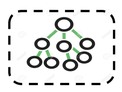
We find not all portions of applications benefit from NDA

1. Memory-intensive portions benefit from NDA
2. Compute-intensive or cache friendly portions should remain on the CPU

1st key observation: CPU threads often concurrently access the same region of data that NDA kernels are accessing which leads to significant data sharing



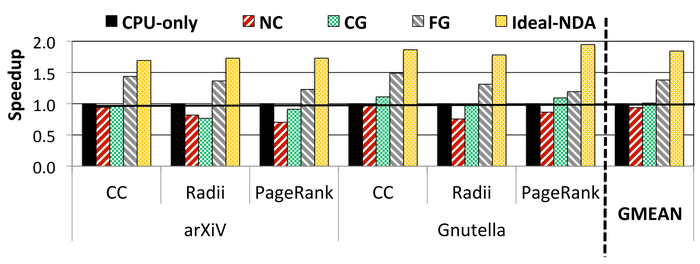
2nd key observation: CPU threads and NDA kernels typically do not concurrently access the same cache lines

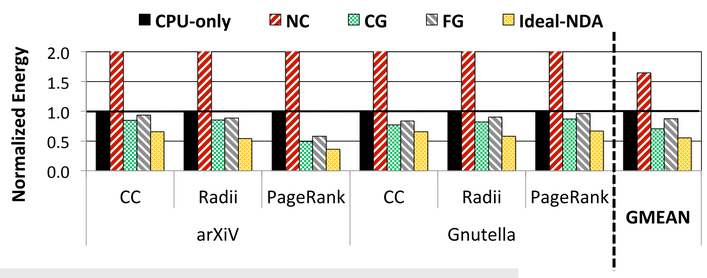


For Connected Components application, only 5.1% of the CPU accesses collide with NDA accesses

CPU threads rarely update the same data that an NDA is actively working on

Analysis of Existing Coherence Mechanisms

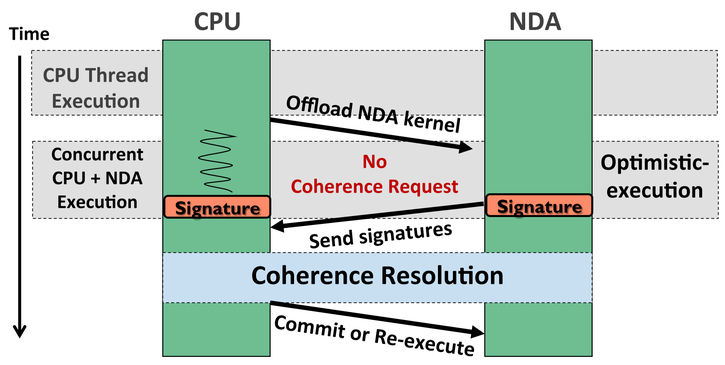




Poor handling of coherence eliminates much of an NDA's performance and energy benefits

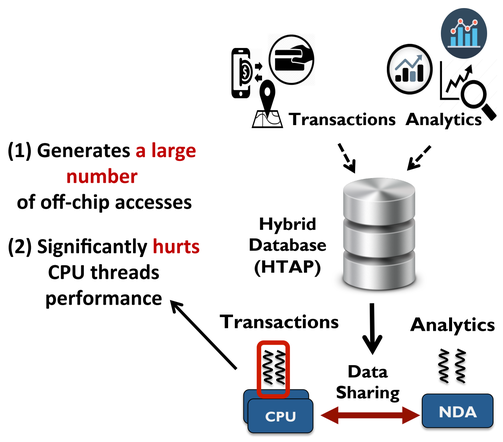
CoNDA

We propose CoNDA, a mechanism that uses optimistic NDA execution to avoid unnecessary coherence traffic



Non-Cacheable Approach

Mark the NDA data as non-cacheable

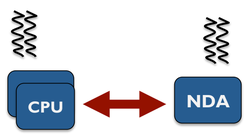


NC fails to provide any energy saving and perform 6.0% worse than CPU-only

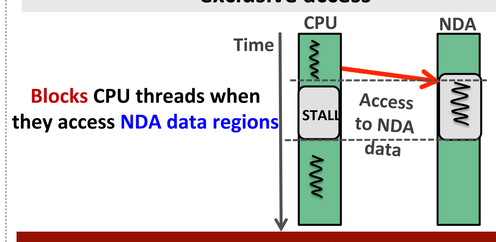
Coarse-Grained Coherence

Get coherence permission for the NDA region

Unnecessarily flushes a large amount of dirty data



Use coarse-grained locks to provide exclusive access

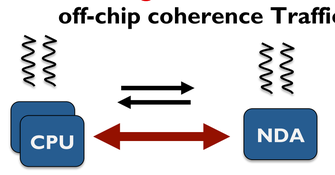


CG fails to provide any performance benefit of NDA and perform 0.4% worse than CPU-only

Fine-Grained Coherence

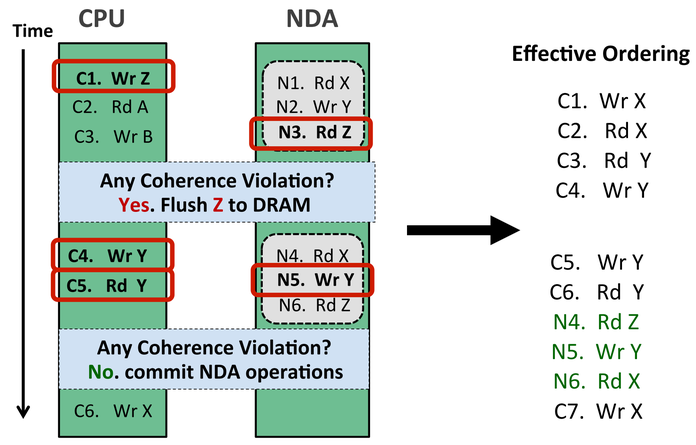
Using fine-grained coherence has two benefits:

1. Simplifies NDA programming model
2. Allows us to get permissions for only the pieces of data that are actually accessed



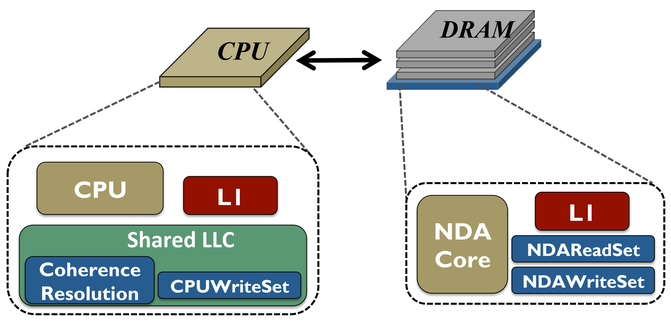
FG eliminates 71.8% of the energy benefits of an ideal NDA mechanism

Identifying Coherence Violations

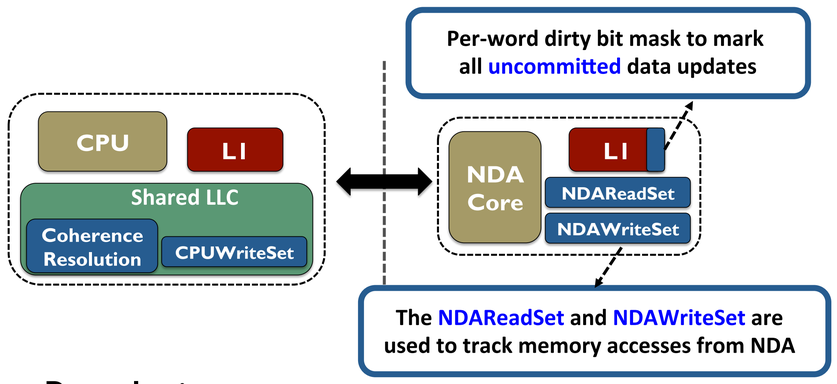


Architecture Support

High Level Architecture of CoNDA



Optimistic Execution



Per-word dirty bit mask to mark all uncommitted data updates

The NDAReadSet and NDAWriteSet are used to track memory accesses from NDA

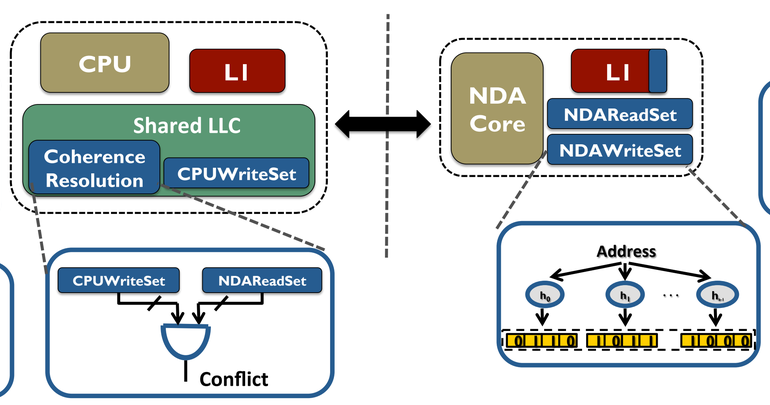
Coherence Resolution

If conflicts happens:

- The CPU flushes the dirty cache lines that match addresses in the NDAReadSet
- NDA invalidates all uncommitted cache lines
- Signatures are erased and NDA restarts execution

If no conflicts:

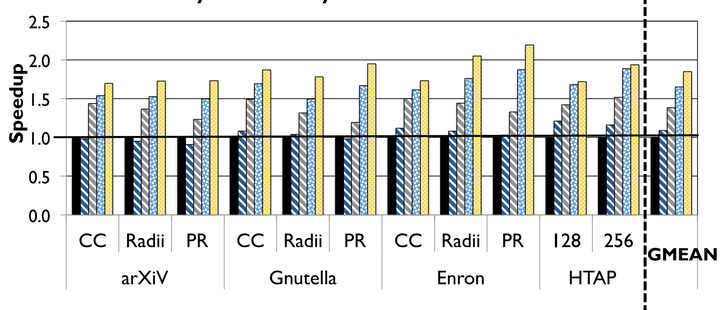
- Any clean cache lines in the CPU that match an address in the NDAWriteSet are invalidated
- NDA commits data updates



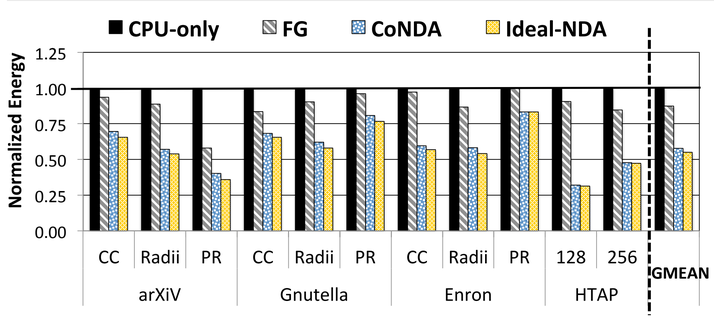
Bloom filter based signature has two benefits:

- Allows us to easily perform coherence resolution
- Allows for a large number of addresses to be stored within a fixed-length register

Evaluation



CoNDA consistently retains most of Ideal-NDA's benefits, coming within 10.4% of the Ideal-NDA performance



CoNDA significantly reduces energy consumption and comes within 4.4% of Ideal-NDA