Cashmere-2L: Software Coherent Shared Memory on a Clustered Remote-Write Network
by Robert Seo, Sathya Devadas, Nicholas Harderline, Galen Hunt, Louridas Kemererlouious, Stevozisam/Parthenay and Michael Scott
Presentation by Daniel Horowitz

Introduction: What is Cashmere-2L?
- CASHMere stands for "Coherence Algorithms for Shared Memory Architectures"
- Provides efficient, scalable, shared memory environment with minimal hardware support.
- 2L – Coherence is provided in two layers, the hardware shared memory within an SMP, and software shared memory within a cluster.

Challenges lie in taking advantage of hardware shared memory for sharing within an SMP while making sure software is only used when activity sharing across SMPs in the cluster.
- Utilizing the low latency remote-write network to communicate only when necessary avoiding messaging overhead.
- Why is this non-trivial?
  - Processors within a node must synchronize very frequently!

Cashmere-2L: What is the motivation?
- Small-scale hardware SMPs are widely available at low cost (4-8cpu) while large-scale systems (64+ cpu) are very expensive.
- Networks of commodity workstations provide similar scale to single systems, but performance is hindered by high latency ethernet.
- Cashmere uses a low-latency remote-write network using DEC’s Memory Channel to overcome latency limitation.

Cashmere-2L: What are the Challenges?
- Challenges lie in taking advantage of hardware shared memory for sharing within an SMP while making sure software is only used when activity sharing across SMPs in the cluster.
- Utilizing the low latency remote-write network to communicate only when necessary avoiding messaging overhead.
- Why is this non-trivial?
  - Processors within a node must synchronize very frequently!

Cashmere-2L: Novel Implementation Techniques
- Cashmere-2L solves the synchronization problem with two-way differencing.
- The technique identifies local and remote changes using:
  - twins
  - diffs
- The coherence protocol itself is very asynchronous
  - no directory (page-table) locks.
  - no intra-node ltb-shutdown.
  - limited remote interrupts.

Cashmere-2L: Presentation Structure
- Memory Channel
- Overview of Protocol
- Principal Consistency Units and Operations
- Alternative Protocols
- Comparative Speedups

Memory Channel
- Provides applications with access to memory on a remote cluster using memory-mapped regions.
- Only writes are possible, reads are employed with write-back requests.
- MC guarantees write ordering and local cache coherence.
Protocol Overview

- "Moderately" Lazy Release Consistency
- Page Consistency
- Synchronization
- Home Node
- Directory Based Coherence
- Exclusive Mode
- Hardware Coherence Exploitation
- Hardware-Software Coherence Interaction

Moderately Lazy Release Consistency

- Multiple writer release consistent protocol
- Shared memory access must be protected by locks and barriers explicitly visible to the runtime system.
- Invalidations take place at the time of the next acquire regardless of whether it is casually related or not (see Treadmarks)

Page Consistency Unit

- Cashmere provides synchronization and fault handling routines at run-time which serve as entry points into the protocol.
- VM faults combine with sync routines to implement consistency throughout the cluster.
  - used to track shared accesses.
  - Coherence granularity is a VM page.

Page Faults

- Upon a page fault, updates are made to the directory
- If no local copy of the page exists, or if the update timestamp precedes the write notice timestamp – the processor fetches a new copy from the home node

Twin Maintenance

- Used instead of TLB Shootdown – Provides mechanism for multiple writer protocol
  - avoids false sharing!
- Twins (clean page copies) are maintained whenever one local processor (not home) has write permissions for the page and the page is not in exclusive mode.
- Used to isolate local and global modifications and avoid interprocess synchronization.

Synchronization Primitives

- Synchronization provided with locks, barriers and flags.
- Implemented two levels with load-linked/store-conditional (ll/sc) acquire with writes to MC space to implement distributed locks.
Syncronization Objects

Memory channel mappings for global directory and synchronization objects

Globally-accessible write notice list has a bin for every node in the system.

Acquires

- The protocol traverses the bins of the nodes global write notice list and distributes the notices to affected local processors.
- eliminates redundant work and book keeping!
- Invalidations occur when the update timestamp proceeds the write notice timestamp.

Releases

- Processor flushes all dirty non-exclusive pages to the home node.
- Then sends write notices to all other nodes (except the home node) with copies of the dirty page.

Home Nodes

- Shared memory pages are mapped to MC space
  - Each page has a single home node containing the master copy of the page.
  - Processors on the home node work directly on its page while remote nodes update the master according to the definition of release consistency and are transparently merged at the home.
  - Each shared page has an entry in the global page directory.
  - Modifications are broadcast using MC’s remote-write

Home Node Page Copies

Memory channel mappings for home node copies of pages. Pages are grouped into supages to reduce total number of mappings.

Directory Based Coherence

- Because MC cannot perform reads, Cashmere uses broadcast to maintain a distributed (replicated) directory of sharing information for each page.
- Page faults are used to trigger requests for an up-to-date copy from the home node.
- Upon a write fault the page is added to a per processor dirty list.
- pages in the dirty list are flushed to the home node at the time of a release (page flush)
- write notices for all pages in the processors dirty list are sent to each node in the sharing set.
Global Directory

- Page table replicated on each node with MC transmit and receive mappings
- Page entries are represented by Nx32Bit words
  - N = number of nodes in the cluster
  - word contains permissions
    - I, RO, RW
  - word contains id of the processor accessing the page in exclusive mode
  - word contains id of the home node

Exclusive Mode

- Optimisation to avoid overhead in the absence of active sharing.
- A node can hold a page in exclusive mode if no other node is accessing the page, effectively acting as if the page were private to eliminate coherence overhead.

Hardware/Software Coherence

- All processors in the same node share the same physical frame for a shared page.
  - Allows protocol transactions from different nodes to be combined
  - reduces communication and overhead
- Protocol tracks temporal ordering of transactions using intra-node timestamps. (logical clock)
- Employs incoming diff page update operation to avoid overhead during a page update operation.
  - keeps the cashmere asynchronous!

Alternative Protocols

- TLB Shootdown – Avoids races between a processor incurring a page fault and concurrent local writers by shooting down (halting) all other write mappings on the node, flushing outstanding changes and creating a new twin on the subsequent page fault.

How does cashmere compare?

Testing Environment

- DEC Alpha 2100 – 4/233 Mhz 256MB – 8 Machine cluster
- 16Kb L1 cache, 1MB L2 cache
- Each node running Digital Unix 4.0D with TruCluster 1.5

Comparative Results

- 1LD v 1L
- 1L v 2L protocols
- 2L v 2LS
- Conclusion: Twins provide most significant speedup
Presentation Conclusion

- We can see that Cashmere-2L exploits hardware cache coherence within SMP nodes and takes advantage of low-latency remote write networks.
- The coherence protocol is based on two way diffing to reduce intra-node synchronization overhead.
- It is able to do this while invoking software when actively sharing across nodes.