CMSC 714
Lecture 10
SGI Origin 2000 and UV

Alan Sussman

Notes

- MPI project grading in progress
  - Grades should be posted within a week
- OpenMP project posted, and timeline for group project also posted along with sample topics from previous years
  - Due Oct. 18, 6PM
- More readings posted soon

Shared Memory Multiprocessors

- Cache coherence
  - to keep different copies of same memory location (data block) the same
  - caching causes the problem, but is needed for performance
- Snooping vs. directory-based coherence
  - shared medium (bus or switched network) vs. distributed directory to keep track of shared data blocks (pages, typically)
  - either way, all memory accesses are to local copies near a processor, and data blocks change state and move around to where they are needed
  - state of each block kept track of with a finite state machine (shared, exclusive, read-only, etc.)
    - A processor (thread) needs exclusive access to write

SGI Origin 2000

- Scalable distributed shared memory (DSM) machine
  - from small building blocks, so scale up and down
  - maximum 512 nodes, mainly limited by interconnection network
- Each node is a dual-processor machine (MIPS processors), with access to local memory (4GB), interconnection network and I/O system
- Nodes connected via "bristled" fat hypercube network (2 nodes per router)
- Cache coherence maintained via directory that keeps track of each data block (page), directory also distributed across all nodes
  - both the state of the cache block and where copies are located
  - protocol appears complicated, but all implemented in hardware, so usually fast – big problem is transitioning to exclusive state for writes, to invalidate copies and TLB entries
  - supports migrating and replicating whole pages across nodes, with OS help
  - Protocol is hardwired to minimize latency and maximize bandwidth

Origin – features for scalability

- SPIDER router chip provides low-latency, high-bandwidth interconnect between nodes, routes memory access requests through the hypercube
  - 2 separate networks for request and response to avoid deadlocks
- HUB chip for on-node routing
  - Between processors, local memory, directory, I/O system, SPIDER router chip
- Coherence protocol optimized to minimize cost of maintaining coherence
  - Directory scheme is optimized to be scalable, but the cost can still be high if a program does not exhibit good spatial and temporal locality in memory accesses
- Memory system includes support for fetch-and-op primitives, to speed up some synchronization operations (locks, barriers, etc.)
  - to avoid cache coherence activity

SGI UV

- UV is a more recent generation SGI, after Origins
  - UV ASIC provides all the functionality beyond what is already available in the Intel Xeon processors
- Scales to 2K hyper-threaded cores (2 sockets/node, up to 8 cores/socket), 64TB memory in 1 global shared memory (GSM), in 1 Linux instance
  - limited by physical (46 bit) and virtual (48 bit) address spaces
  - sockets connected via fat tree
- Scales to much larger configurations, connected via NUMAlink through UV ASIC chips
  - globally addressable memory (GAM) across the Linux instances - think PGAS, or put/get, also good for MPI
  - 53 bit physical memory, 60 bit virtual
- System provides fast MPI implementation (via MPI Offload Engine), fast collective operations, high performance I/O, reliability via error checking and retry, and offloading remote memory accesses to UV ASIC

UV ASIC chip (cont.)

- Provides directory-based cache coherency for GSM and put/get for GAM
- Connects directly to QPI memory interface on Intel Xeon CPUs, not though PCI I/O bus
  - Intel has discontinued support for QPI
  - But connecting memory bus to external devices directly has been done in other processors
  - Global Register Unit (GRU) for global addressing, TLB for address translation across nodes (directories), fast memory initialization w/o CPU aid, fast block copies (good for message passing too), scatter/gather memory ops
- Active Memory Unit (AMU) – cache coherent atomic memory operations, update multicasting for fast collective operations, message queues in cache coherent memory