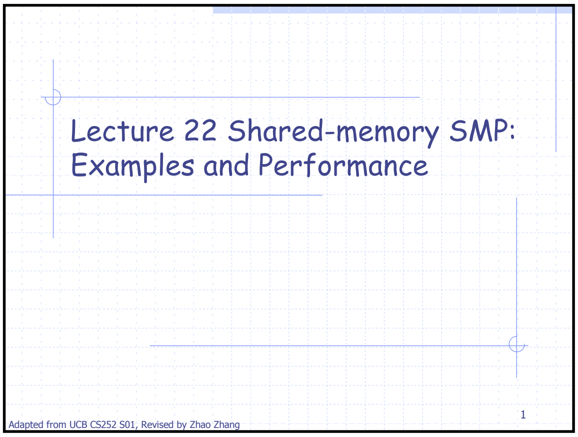
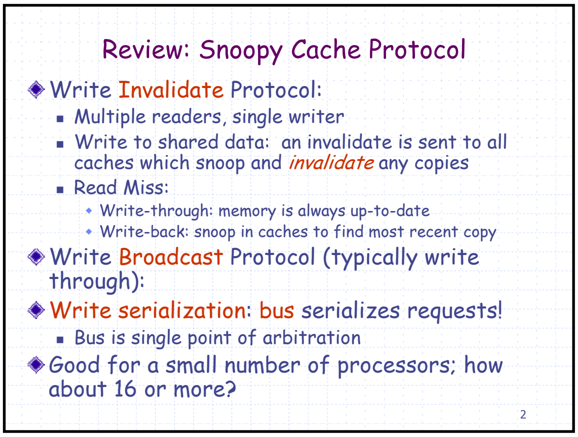


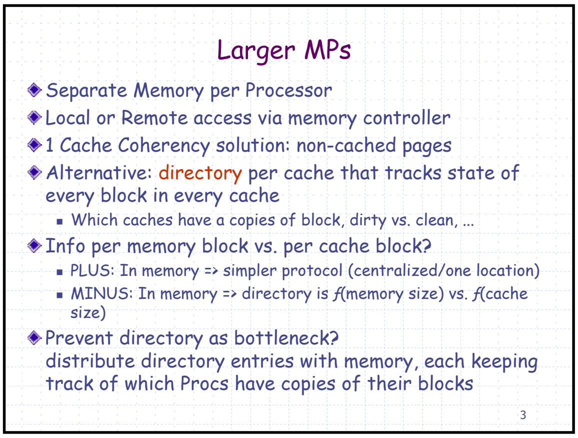
Lecture 22 Shared-memory SMP: Examples and Performance

Adapted from UCB CS252 S01, Revised by Zhao Zhang



Review: Snoopy Cache Protocol

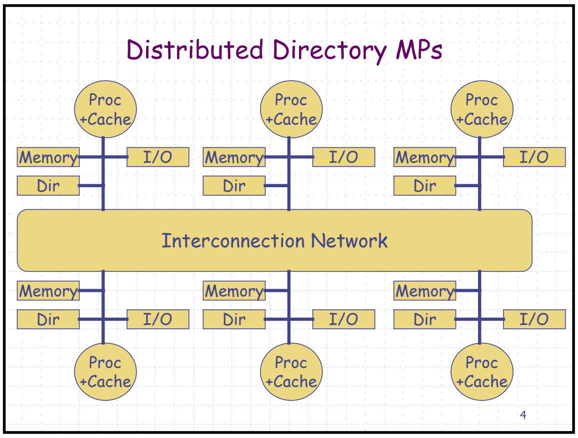
- Write Invalidate Protocol:
  - Multiple readers, single writer
  - Write to shared data: an invalidate is sent to all caches which snoop and *invalidate* any copies
  - Read Miss:
    - Write-through: memory is always up-to-date
    - Write-back: snoop in caches to find most recent copy
- Write Broadcast Protocol (typically write through):
- Write serialization: bus serializes requests!
  - Bus is single point of arbitration
- Good for a small number of processors; how about 16 or more?

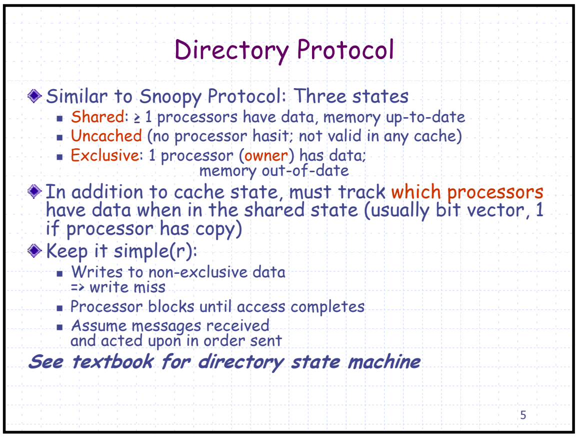


Larger MPs

- Separate Memory per Processor
- Local or Remote access via memory controller
- 1 Cache Coherency solution: non-cached pages
- Alternative: directory per cache that tracks state of every block in every cache
  - Which caches have a copies of block, dirty vs. clean, ...
- Info per memory block vs. per cache block?
  - PLUS: In memory => simpler protocol (centralized/one location)
  - MINUS: In memory => directory is *f*(memory size) vs. *f*(cache size)
- Prevent directory as bottleneck?
  distribute directory entries with memory, each keeping track of which Procs have copies of their blocks

Distributed Directory MPs

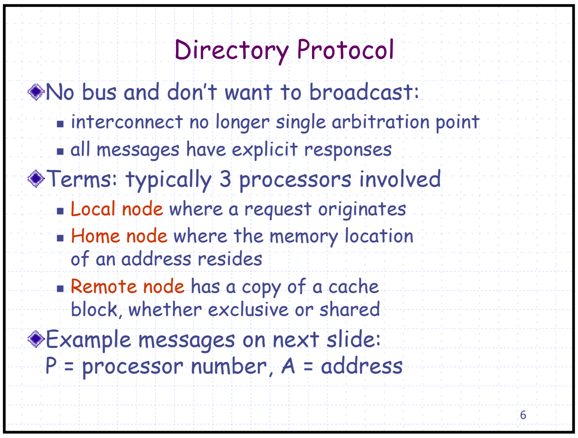




Directory Protocol

- Similar to Snoopy Protocol: Three states
  - Shared: ≥ 1 processors have data, memory up-to-date
  - Uncached (no processor hasit; not valid in any cache)
  - Exclusive: 1 processor (owner) has data;
    memory out-of-date
- In addition to cache state, must track which processors have data when in the shared state (usually bit vector, 1 if processor has copy)
- Keep it simple(r):
  - Writes to non-exclusive data
    => write miss
  - Processor blocks until access completes
  - Assume messages received
    and acted upon in order sent

See textbook for directory state machine



Directory Protocol

- No bus and don't want to broadcast:
  - interconnect no longer single arbitration point
  - all messages have explicit responses
- Terms: typically 3 processors involved
  - Local node where a request originates
  - Home node where the memory location of an address resides
  - Remote node has a copy of a cache block, whether exclusive or shared
- Example messages on next slide:
  P = processor number, A = address

Directory Protocol Messages

Message type	*Source*	*Destination*	*Msg Content*
Read miss	Local cache	Home directory	P, A
▪ *Processor P reads data at address A; make P a read sharer and arrange to send data back*			
Write miss	Local cache	Home directory	P, A
▪ *Processor P writes data at address A; make P the exclusive owner and arrange to send data back*			
Invalidate	Home directory	Remote caches	A
▪ *Invalidate a shared copy at address A.*			
Fetch	Home directory	Remote cache	A
▪ *Fetch the block at address A and send it to its home directory*			
Fetch/Invalidate	Home directory	Remote cache	A
▪ *Fetch the block at address A and send it to its home directory; invalidate the block in the cache*			
Data value reply	Home directory	Local cache	Data
▪ *Return a data value from the home memory (read miss response)*			
Data write-back	Remote cache	Home directory	A, Data
▪ *Write-back a data value for address A (invalidate response)*			

Parallel App: Commercial Workload

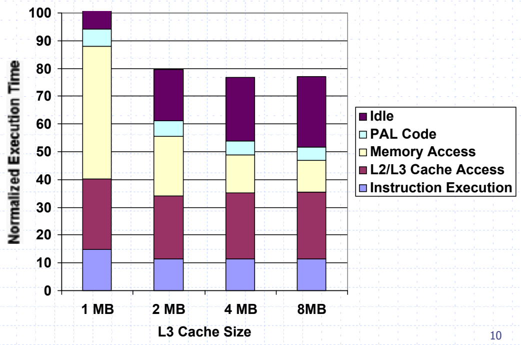
- Online transaction processing workload (OLTP) (like TPC-B or -C)
- Decision support system (DSS) (like TPC-D)
- Web index search (Altavista)

Benchmark	% Time User Mode	% Time Kernel	% Time I/O time (CPU Idle)
OLTP	71%	18%	11%
DSS (range)	82-94%	3-5%	4-13%
DSS (avg)	87%	4%	9%
Altavista	> 98%	< 1%	<1%

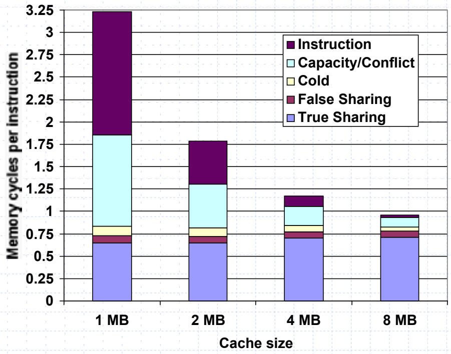
Alpha 4100 SMP

- 4 CPUs
- 300 MHz Apha 211264 @ 300 MHz
- L1$ 8KB direct mapped, write through
- L2$ 96KB, 3-way set associative
- L3$ 2MB (off chip), direct mapped
- Memory latency 80 clock cycles
- Cache to cache 125 clock cycles

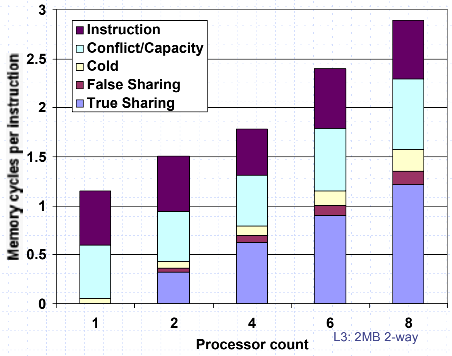
OLTP Performance as vary L3$ size



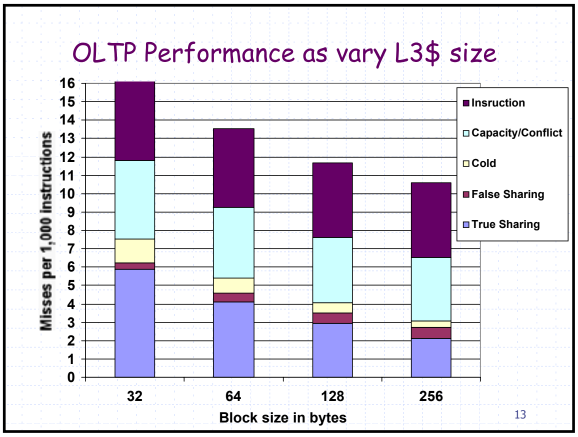
L3 Miss Breakdown



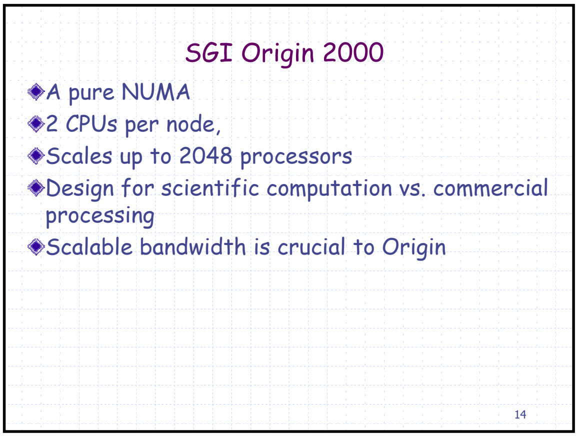
Memory CPI as increase CPUs



L3: 2MB 2-way

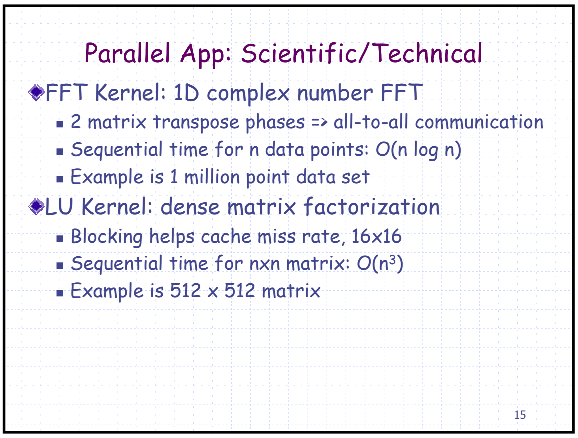


OLTP Performance as vary L3$ size



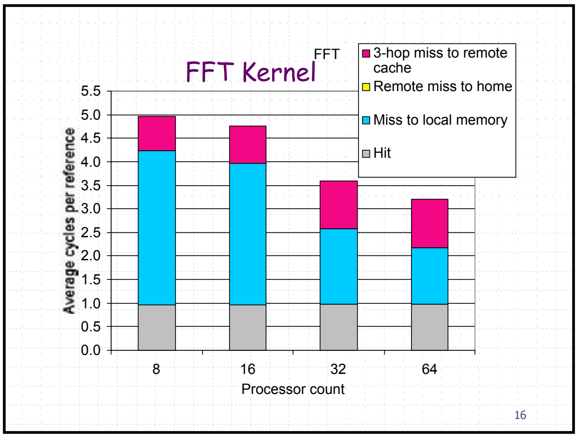
SGI Origin 2000

- A pure NUMA
- 2 CPUs per node,
- Scales up to 2048 processors
- Design for scientific computation vs. commercial processing
- Scalable bandwidth is crucial to Origin

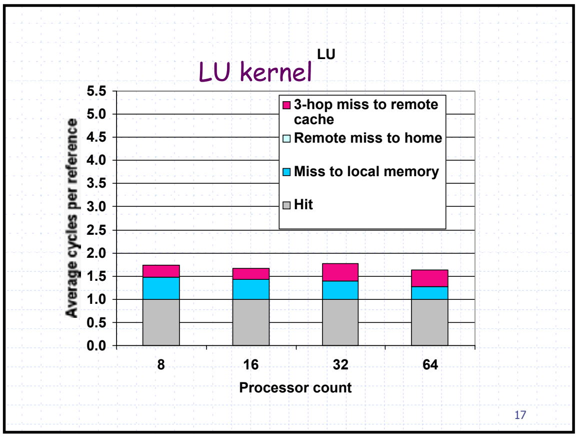


Parallel App: Scientific/Technical

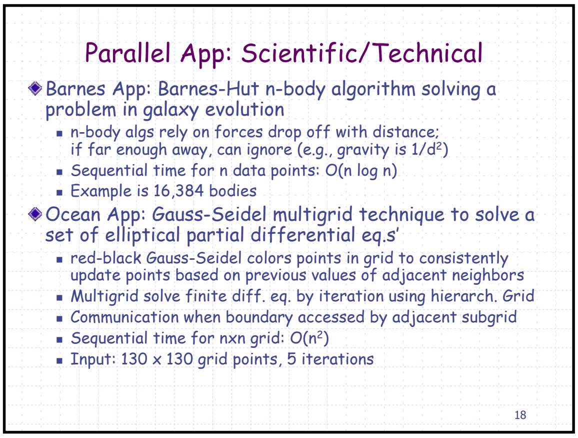
- FFT Kernel: 1D complex number FFT
  - 2 matrix transpose phases => all-to-all communication
  - Sequential time for n data points: O(n log n)
  - Example is 1 million point data set
- LU Kernel: dense matrix factorization
  - Blocking helps cache miss rate, 16x16
  - Sequential time for nxn matrix: O(n^3)
  - Example is 512 x 512 matrix



FFT Kernel



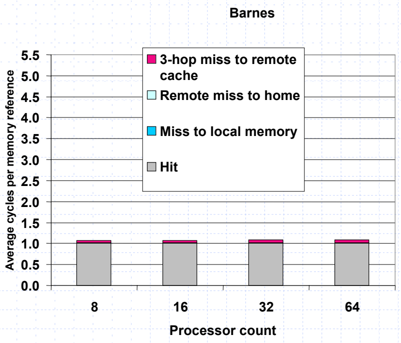
LU kernel



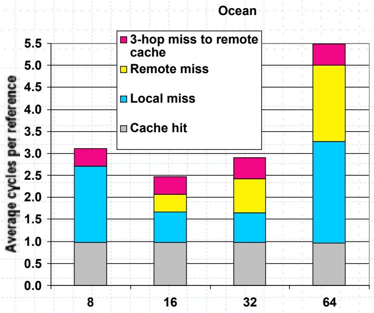
Parallel App: Scientific/Technical

- Barnes App: Barnes-Hut n-body algorithm solving a problem in galaxy evolution
  - n-body algs rely on forces drop off with distance; if far enough away, can ignore (e.g., gravity is 1/d^2)
  - Sequential time for n data points: O(n log n)
  - Example is 16,384 bodies
- Ocean App: Gauss-Seidel multigrid technique to solve a set of elliptical partial differential eq.s'
  - red-black Gauss-Seidel colors points in grid to consistently update points based on previous values of adjacent neighbors
  - Multigrid solve finite diff. eq. by iteration using hierarch. Grid
  - Communication when boundary accessed by adjacent subgrid
  - Sequential time for nxn grid: O(n^2)
  - Input: 130 x 130 grid points, 5 iterations

Barnes App



Ocean App



Example: Sun Wildfire Prototype

1. Connect 2-4 SMPs via optional NUMA technology
   1. Use "off-the-self" SMPs as building block
2. For example, E6000 up to 15 processor or I/O boards (2 CPUs/board)
   1. Gigaplane bus interconnect, 3.2 Gbytes/sec
3. Wildfire Interface board (WFI) replace a CPU board => up to 112 processors (4 x 28),
   1. WFI board supports one coherent address space across 4 SMPs
   2. Each WFI has 3 ports connect to up to 3 additional nodes, each with a dual directional 800 MB/sec connection
   3. Has a directory cache in WFI interface: local or clean OK, otherwise sent to home node
   4. Multiple bus transactions

Example: Sun Wildfire Prototype

1. To reduce contention for page, has Coherent Memory Replication (CMR)
2. Page-level mechanisms for migrating and replicating pages in memory, coherence is still maintained at the cache-block level
3. Page counters record misses to remote pages and to migrate/replicate pages with high count
4. Migrate when a page is primarily used by a node
5. Replicate when multiple nodes share a page

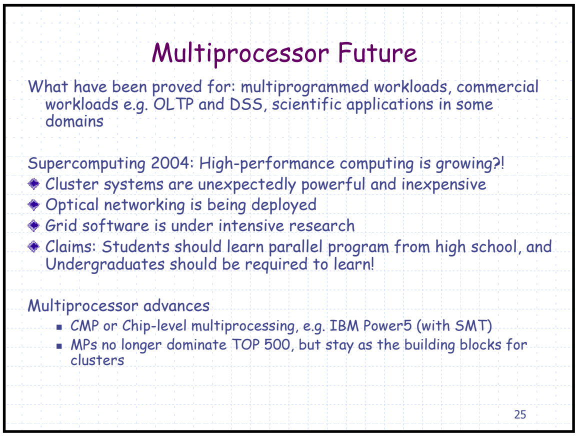
Synchronization

- Why Synchronize? Need to know when it is safe for different processes to use shared data
- For large scale MPs, synchronization can be a bottleneck; techniques to reduce contention and latency of synchronization

Study textbook for details

Fallacy: Amdahl's Law doesn't apply to parallel computers

- Since some part linear, can't go 100X?
- 1987 claim to break it, since 1000X speedup
  - Instead of using fixed data set, scale data set with # of processors!
  - Linear speedup with 1000 processors



Multiprocessor Future

What have been proved for: multiprogrammed workloads, commercial workloads e.g. OLTP and DSS, scientific applications in some domains

Supercomputing 2004: High-performance computing is growing?!

- Cluster systems are unexpectedly powerful and inexpensive
- Optical networking is being deployed
- Grid software is under intensive research
- Claims: Students should learn parallel program from high school, and Undergraduates should be required to learn!

Multiprocessor advances

- CMP or Chip-level multiprocessing, e.g. IBM Power5 (with SMT)
- MPs no longer dominate TOP 500, but stay as the building blocks for clusters