Lecture 19

Final Exam Review
The Final Exam

• Wednesday Dec 10th
  11.30 AM to 2.30 PM
  Customary classroom (WLH 2205)

• Sign up sheet

• Office hours
  – By appointment
Terms and concepts

• Know the definition and significance of ….
• Parallel speedup and efficiency, super-linear speedup, strong scaling, weak scaling
• Amdahl's law, serial bottlenecks
• Granularity
• Data dependencies, loop carried dependence
Terms and concepts

- PRAMs, CRCW
- SPMD, MIMD, SIMD
- Multiprocessors and multicomputers
- Cache coherence and consistency
- False sharing
- Snooping, directories
- NUMAs and SMPs
- Processor geometry
- Interconnection networks: hypercube, ring, mesh, toroidal mesh, broadcast and reduction algorithms; diameter and bisection bandwidth
- Message startup, half power point $n^{1/2}$, peak bandwidth
Implementation techniques

- Threads
- pthreads
- OpenMP
- Mutexes, semaphores, and barriers
- Critical sections, race conditions
- Red-Black/ Odd-even
- Message passing
Multicore architectures

- **Cell Broadband Engine**
  - 8 fast streaming processors + 1 power PC
  - Each streaming processor
    - Small local store, explicitly managed
    - SIMD instructions
  - DMA between stream processors and external memory

- **Nvidia**
  - Streaming processors, no SIMD instructions
  - Hierarchically organized → hierarchically organized threads
  - Overlap data motion with computation
False sharing in higher dimension arrays

• Compare with distributed memory solution
Practice problems
List all possible outputs that result when the following OpenMP annotated C code is executed:

```c
#pragma omp parallel for shared(j,k)
for ( int i=0, j=0, k=0; i< 2; i++ )
    j = j + 10;
    k = j + 10;
}
cout << “k = “ << k << endl;
```
OpenMP

Now with critical sections

```c
#pragma omp parallel for shared(j,k)
for ( int i=0, j=0, k=0; i< 2; i++ )
    #pragma omp critical
    j = j + 10;
    #pragma omp critical
    k = j + 10;
}
cout << "k = " << k << endl;
```
OpenMP

Consistent with serial program

```c
#pragma omp parallel for shared(j,k)
for ( int i=0, j=0, k=0; i< 2; i++ )

    #pragma omp critical
    {
        j = j + 10;
        k = j + 10;
    }

cout << “k = “ << k << endl;
```
Consistency

What value does thread 1 print, with and without sequential consistency?
What do we need to specify about the memory system to know what value is printed?

Thread 0
A=1;
B=2;
A=0;

Thread 1
B=3;
while (A);
printf(“%d\n”,B);
False Sharing

How many cache misses?

Thread 0
N=1000;
while (N--) a++;

Thread 1
M=1000;
while (M--) b++;
False Sharing

How many cache if a, b, M, and N are on the same cache line?

Thread 0
N=1000;
while (N--) a++;  

Thread 1
M=1000;
while (M--) b++;
Performance

We run a certain application on 16 processors.

The application has two phases.

- The first phase is perfectly parallelizable, and operates at 100% efficiency on the 16 processors.
- The second phase operates at 25% efficiency on 16 processors and consumes a fraction $f$ of the overall running time on one processor (best serial algorithm).
- Let $T_1$ be normalized to 1.

Give an expression for $T_{16}$ in terms of $f$, using an expression of the form $R + Sf$, where $R$ and $S$ are rational numbers, that is, fractions of the form $x/y$. 
Interconnection Networks

Consider a toroidal mesh interconnect where each processor has a direct connection with 4 nearest neighbors, and processors adjacent to an edge have end-around connections.

- What is the bisection bandwidth of this 4x4 toroidal mesh, assuming that the bandwidth of one connection is $B$?
- What is the diameter?