CS 5334 Spring 2016 Name ________________________________
Shirley Moore, Instructor
April 19 Class

OpenMP Gotchas

It's easy to insert OpenMP into your programs, and if you make sure to prevent race conditions, the results should be correct. However, it is easy to get poor performance for a number of reasons. One reason we have already looked at with Pthreads is false sharing. Today we'll look at two other reasons for poor performance – unnecessary serialization and poor data allocation.

1. Copy the pi.c file from the instructor's directory on Stampede:

```
cp –r ~tg457571/cs5334/openmp/pi.c .
```

Parallelize the code using OpenMP in two ways. In one version, use a critical section to prevent a race condition on sum. In the second version, make sum a reduction variable. Run the two versions with different numbers of threads and measure the execution time. What did you learn from this exercise?

2. Copy the stream-slow.c file from the instructor's directory on Stampede:

```
cp –r ~tg457571/cs5334/openmp/stream-slow.c .
```

Compile the code and run it using 8 OpenMP threads. You should be able to get about 52 GB/s memory bandwidth but this version is much slower. Find the problem and fix it. What did you learn from this exercise?