CSCI-UA.0480-003

Parallel Computing

Homework Assignment 2

1. We discussed briefly how caches are designed. Among cache characteristics are whether a cache is write back (when a cache block is modified, it is written back to the lower level cache only when the block is replaced) or write through (whenever a cache block is updated, it updates also the lower level copy). Discuss the pros and cons of each.

2. Suppose a processor has to wait 10 cycles for its memory system to provide a 64-bit word. What can we do to reduce this delay when we have several loads coming from the processor to the memory?

3. Suppose we have a system with three level of caches: L1 is close to the processor, level 2 is below it, and level 3 is the last level before accessing the main memory. We know that two main characteristics of a cache performance are: cache access latency (How long does the cache take before responding with hit or miss?) and cache hit rate (how many of the cache accesses are hits?). As we go from L1 to L2 to L3, which of the two characteristics become more important? and why?

4. A sequential application with a 20% part that must be executed sequentially, is required to be accelerated five-fold. How many CPUs are required for this task?

5. Coherence protocols are expensive in terms of performance. Why?

6. In slide# 11 of the performance analysis lecture (lecture 6) we see that as the number of processors (or threads or cores) increases, the speedup increases.

a. What the curve of "double size" seems better than the other two?

b. If we keep increasing the number of processes (i.e. the x-axis) what do you think the rest of the curve will look like?

7. In slide# 12 of the performance analysis lecture (lecture 6), efficiency decreases as the number of processes/cores/threads increases, why is that?

8. What is the relationship between synchronization points in a parallel program and load balancing? Explain why do we have such a relationship?

9. In slide 12 of lecture 5, we found that parallelism = 6.25, yet, we can see from the graph that there are 8 paths that can be executed in parallel. How can you explain this discrepancy?

10. Suppose we have two threads doing the same operations but on different data. Also suppose these two threads execute on two different cores. If the two threads start at the same time, can we assume that they will always finish at the same time? Justify.