Parallel Computing
Homework Assignment 1
[Total: 40 points]

1. [6 points, 3 for each point]
   - Tree-like summation gets better performance due to exploiting some parallelism. At each level of the tree (i.e. each step in the summation calculation) more than one processor is doing work. Even if not all of them are working, it is still better than one processor doing all the work.
   - Communication is also better. If all processors (or cores) are sending to the same processor there may be contention on the interconnection network, negatively affecting the performance. But when processors are sending to different processors (e.g. processor 0 sends to 1; processors 3 sends to 2; …) the communication is much faster as it can be done in parallel.

2. [6 points, 2 for each point.]
   - Load balancing ensures that we have the most parallelism because all processors will be working. But with load-unbalancing, some processors may finish before others, reducing the amount of parallel execution and not making the best use of these processors.
   - If the load is not balanced, the program becomes very sensitive (i.e. easily affected) by synchronizing points (e.g. barriers). This is because each time a synchronization point is encountered, the faster processors are slowed down.
   - Processors which are doing more work can become very warm and hence the hardware will slow down the execution a bit to cool it down.

[Note: For this question, we use the words processors to mean either processors or cores.]

3. [9 points, 3 for each item: 1 point for #threads and 2 points for justification]

a) Each core is superscalar but not hyperthreading	Eight threads maximum	Without hyperthreading, each core can execute one thread maximum at a time.
b) Each core is superscalar and two-way hyperthreading	16 threads maximum	Each core can execute up to two threads simultaneously.
c) Each core is neither superscalar nor hyperthreading	Eight threads maximum	Like the first case, each core can execute one thread at a time. But without superscalar, it will be slower than the first case.

4. [6 points total: 2 for needed/not needed and 4 for justification]

   Speculative execution is not needed for correct working for superscalar. Superscalar means executing several instructions at the same time, which the superscalar can do with speculative execution. However, speculative execution (i.e. branch prediction) makes superscalar more effective if there are many conditional branches.

5. [4 points]
MIMD can implement SIMD. However, with dedicated SIMD, we can have much more execution units than with MIMD. This is because for MIMD, the execution units used will be full-fledged cores while in SIMD, like GPUs, they are just execution units and hence we can pack many of them in the chip getting better performance for the applications that are SIMD-friendly.

6. [4 points, 2 for each point]

- The increase in processor frequency
- The ability of a single core processor to exploit instruction-level parallelism using techniques such as: pipelining, superscalar capability, speculative execution, etc.

7.
Coherence protocols affect performance negatively, [2 points]
for several reasons: [3 points, 1 for each point.]

- Any write to the memory is delayed till the protocol ensures exclusive ownership of the block.
- Coherence protocol causes extra cache misses due to the invalidation.
- Coherence protocol results in more traffic due to the messages sent to invalidate/update/acknowledge, etc.