AMOD 5420H/COIS 4350H HPC

Assignment2

# Theory Questions (1 mark each, 3 total)

Your theory questions should be between 300 and 600 words for each of the 3 main questions and may include diagrams.

1. Textbook 5.5 : (**Sri note: the textbook is SLURM specific, but the same basic problem arises for any scheduler, so don’t bother with SLURM specific details, unless they happen to be useful)** Imagine you are a system administrator for a newly installed computer composed of 260 nodes total. Of those, 64 come equipped with GPUs and 36 have substantially larger memory capacity. Your users execute both regular and (infrequently) high-priority jobs. The latter require exclusive access to non-accelerated hardware resources, but never occupy more than 128 nodes.
   1. Propose a partitioning scheme that provides good utilization of the entire system. So you’ll have different sets of nodes as one logical block. Identify any partition overlaps.(**~~enumerate the types and sizes of Slurm partitions~~** Sri edit: Some partition scheme it doesn’t need to be slurm specific)
   2. What kind of provisions would you implement to facilitate parallel job debugging?
   3. ~~Which Slurm features~~ **Discuss strategies/features of some scheduler, that** would you take advantage of to minimize the impact of conflicts between jobs of different priorities?
2. Textbook 6.4: You are an IT specialist at a small computational research institution. The scientists require a peak 100Tflops machine to conduct their studies. The approved vendor offers two units, rack-mount nodes with two CPU sockets that may accommodate either 12-Core processors clocked at 3.4 GHz or 20 Core processors operating at 2.5 GHz. Each core can perform four floating-point operations per clock cycle. Given up to 32U of floor space for nodes in each rack answer the following:
3. Which type of processor would you recommend to minimise the floor space occupied by the racks?
4. How many racks are needed to reach the required peak throughput?

(**Note you can fit 16 nodes per rack, each rack takes 1U of floor space)**

1. With all racks **(needed in b)** filled what is the final peak computational throughput of the machine?

1. Textbook 6.6: Execution of a 1 million -instruction program takes 2.5 ms on a 2.5GHz core. The Hardware monitor reports a cache miss ratio of 6% for the application. Main Memory access takes on average 80 ns, while cache access has a latency of 800ps. Given that all ALU instructions are executed effectively in a single clock cycle, calculate the following:   
   (**HINT: Equation 6.19 in the text**)



1. The fraction of application instructions that performed ALU operations
2. if the core has 16KB cache and doubling the cache size decreases the miss rate by 1% for that particular application, what would be the required cache size (in powers of 2) to cut the execution time in half?
3. What would the program runtime and resulting speedup be if all accessed data fits in the cache?

# Programming Questions (5 marks for 4, 2 marks for q5)

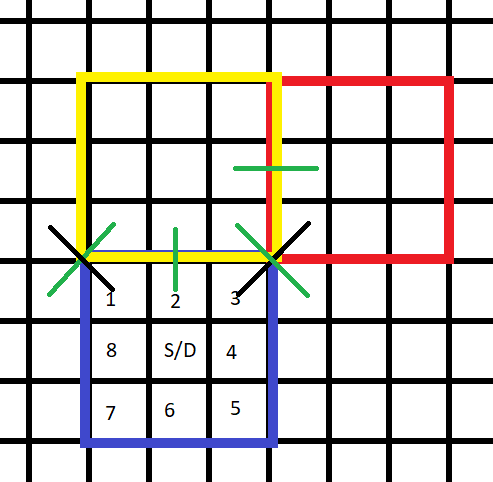
1. Build a large 2D matrix of floats(or doubles), cells are defined in the next paragraph in C++ (large enough that you need to create it on the heap using the keyword “new”, I’d say probably a 100x100 grid is good enough, but for testing start with something like 5x5). Your simulation is just a big loop that changes state each iteration through it.

Your simulation, basic overview: At the start of your simulation every cell starts with a value of zero. At the first frame in the middle of the leftmost edge (position 0, 49 if 0,0 is bottom left) spikes to a value of 100 000.

Each frame, every cell looks at itself and all neighbouring cells, and calculates 5% of the difference between that point and the average of its neighbours (so a cell with a value of 1000, and 8 neighbours each valued 100 does 0.05\*(1000-8\*100/8) =45, and then sends that to each neighbour. After the first iteration the 1000 becomes 1000 – (8\*45) =640, and each neighbour is 145. One thing to consider: each cell should only be processed once in terms of what it passes to neighbours. I will leave it to you to figure out a way to handle if a cell is \*less\* than its neighbours (that is to say it’s a drain) whether it pulls in material from neighbours or not. You should never have a situation where your system has anything other than exactly 100k in it (give or take floating point rounding).   
  
Notice that the exact behaviour this system might depend on the order in which you evaluate cells.

At a wall, essentially nothing happens (ultimately it reflects back, but you don’t really need to do much to make that happen).   
  
Run the simulation until no cell has a value larger than 12. Count the iterations. Optimize using OpenMP

Note that you could write most of the code for this in VS on windows and port it over to Linux if you prefer to work that way, but you should at least know how to compile and debug with gcc, and to use OpenMP and later MPI libraries which would normally run on a Linux Cluster.

  
  
I want you to discuss your strategies for achieving this parallelisation, and how much benefit you get as you write your algorithms. With OpenMP there isn’t that much to talk about.

It should be reasonably obvious this is a basic approximation of the behaviour of a wave created by a fluid pouring into a spot in a container, without the accurate maths. If you want to do a proper Navier stokes model you can, but the point here is really the parallelization, not the physics.

1. Parallelise 4th Order Runge Kutta methods using OpenMP (2 mark)

<https://www.geeksforgeeks.org/runge-kutta-4th-order-method-solve-differential-equation/>

Has a single threaded implementation. There are few implementations for 4th order Runge Kutta’s that you can find using an Internet search. Your goal is to implement your own parallel implementation using OpenMP, but feel free to start from a well understood single threaded starting point.

(Sri Note: I specifically put this question on here because Runge Kutta Methods do not lend themselves to any easy/obvious parallelisation, and in fact your efforts may make things worse.)

Test on functions you can find on the web where you know the right answer. You will want to make sure your step size h is small but not floating-point error sort of small (0.01 is ok 0.000001 is probably not).

You may need to learn what a Runge Kutta is. <https://www.youtube.com/watch?v=hGN54bkE8Ac>, <https://www.youtube.com/watch?v=AT7Olelic8U>