Parallel to the max

I'd like you to use the LINDA model we worked on in class as an inspiration for wringing as much parallelism as you can from your cluster. Here is what I would like you to do using the mqtt framework:

1. Choose one node to be supervisor. This node is keeper of the BAG.

2. The supervisor places a set of contiguous ranges in the BAG, i.e., it outs the ranges. The goal is to count the primes in each range and come up with total.

3. Workers contact the supervisor when they are free and ask for a new range to work on, i.e., they do ins. The supervisor supplies them with a range waiting in the BAG.

4. Each worker will get a range from the BAG. That worker should then further divide that range into 4ths and use all 4 of its cores to compute. It will sum up and send result back to supervisor.

5. When a worker has a result, it gives it back to the supervisor. The supervisor keeps a running sum. I considered placing the worker's result back in the BAG as a separate summation task, but decided to keep it simple.

6. When the BAG is empty, the supervisor publishes a FINISHED message. Workers should subscribe and pay heed to it.

In the end, this should give us roughly3x4 cores dedicated to the problem!

Note 1: I am not following the LINDA spec exactly. Likewise, you do not have to follow my spec exactly. If your group can come up with better way of handling the 12 cores at your disposal, go for it. But also see the extra credit below: I want to keep the ability to add or remove workers easily and that is what LINDA gives you.

Note 2: I am keeping the problem simple by having homogenous tasks in the BAG. But I could foresee putting different size ranges in the BAG and marking them with a second parameter EASY/MEDIUM/HARD. Workers could decide if they have the processing power to take on HARD problems and do the appropriate in.

Warning: when we tried this out we noticed that once you put all 4 cores in action, you lose mqtt publishes, i.e. callbacks are not exectured. We tried using only 3 cores hoping the 4^{th} core would still be able to do asynch communication. Still did not work: mqtt callbacks not active. I don't see an issue with your system: a worker will only get a published message from supervisor when the worker is idle. But keep it in

mind. Could be a real flaw in the multiprocessing library if it locks out asynch activity until its processing completes.

Extra credit: handle a worker going down while working on a task. An idle worker going down should be no issue. Ditto adding new workers to the mix. That's what makes the LINDA model elegant. But if a worker does an in on a task and crashes before outing a result, then you have problems. Come up with a means of not losing results.