The more processes you create, the less likely it is to get a new coordinator with each new process using the bully algorithm. Since the coordinator always has the highest id, eventually it becomes one with a value far above the average.

Didn’t bother to try removing downed processes from the process list on S3. This ended up causing problems when a new process would start up and take the IP/Port of the downed process then try sending election messages to itself, getting stuck in an infinite loop. Would also cause weird behavior when a process starts up taking the IP and port of a dead process that someone had sent a request to. The new process accepts the request once its socket is bound, even if it doesn’t make sense.

Had issues with process state when getting out of sync with a lot of processes coming up. Process got stuck trying to request work from a process that was no longer the coordinator, or would fail to submit the work results and wouldn’t request new work.

Some issues with the fact that Nodejs is single threaded. When running on a single machine, a lot of messages would time out because a process is busy computing edit distances and can’t respond to election requests.

Some errors on windows with zmq when running a lot of nodes. Apparently there are some built in limits that need to be manually changed in zmq and rebuilt. On a positive note, the random failures were great for verifying the correctness of the election algorithm. Also had a few cases where multiple processes somehow bound to the same port on my local machine, or at least appeared to.

Issue with all nodes thinking they are the coordinator.