CS 425: MP 1 - Report

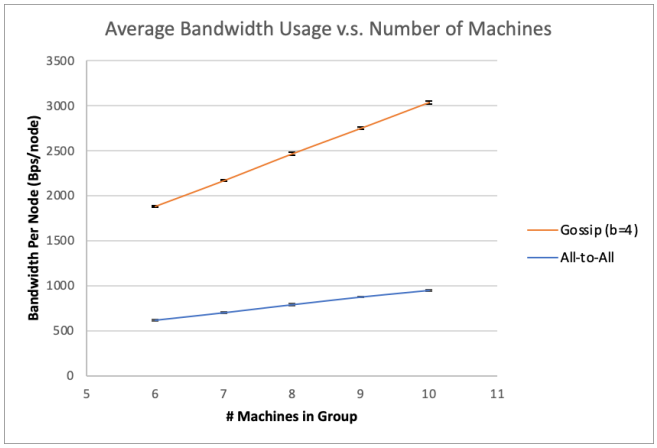
Group 22: Siwei Xian (sxian3), Darci Peoples (darciap2), Ishaan Datta (ishaand2)

Design

In our system, there is one introducer node which can accept join requests from other new nodes. In gossip mode, nodes constantly listen for others' heartbeast/leave group/switch mode messages, periodically gossip their full membership lists (via UDP), and await user commands (i.e. switch, leave, join, list). Nodes gossip via UDP to up to b=4 other nodes, so we can ensure completeness for failure detection for up to 3 simultaneous failures. In all-to-all mode, the difference is that nodes broadcast just their own heartbeat to all other nodes. When sending data between nodes, we marshal using the Python3 json library.

Plots

Bandwidth: Average bandwidth usage (per node) is linearly proportional to the number of nodes. This makes sense as each node gossips it's full membership list. The standard deviation is quite small because of the other parameters during data collection (additional message drop rate = 0%, gossip b = 4, time to heartbeat = 1 s, time to fail = time to cleanup = 4 s). With these parameters, we greatly lower the rate of false positives and message drops, making the membership list size quite stable.



False positive rate: suppose message loss rate for each packet is `p_ml`, waiting time to mark a node as failed is `t_fail`, and the number of gossips we send to each node is `num_gossip`. At time `t0`, the probability of a specific node being marked as failed by the rest of network at `t = t0 + t_fail` is `p = p_ml ^ (t_fail * num_gossip)`. At our default setting `(t_fail = 4, num_gossip = 4, p_ml = 0.2)`, `p = 0.2 ^ 16 = 6.56e-12`, which is extremely small. In the actual test, we didn't find any fail positive situation at these settings for 1 hour.