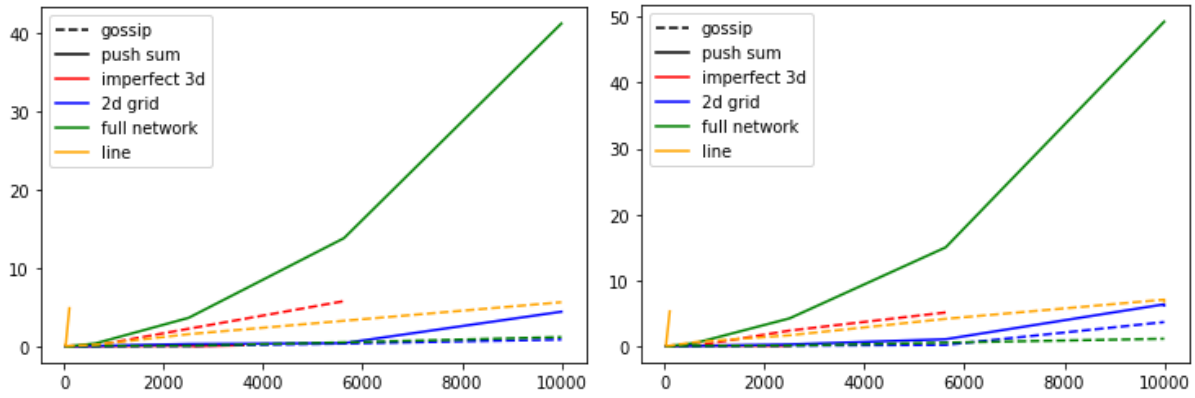
Project 2 Bonus Report

Based upon our gossip and push sum implementations we added failing nodes to the network.

```
Failed = (crypto:rand_uniform(1, 4) rem 4) == 0,
if
    Failed ->
        ok;
    true ->
        lists:nth(getRandomNeighbour(GridType, Index, Nodes, RandAccessDimIndex, RandNodeIndex), Nodes) ! {pushSum, NewSum, NewWeight}
end,
pushSum(GridType, MasterNode, Index, Nodes, NewSum, NewWeight, Iteration + 1, NewRatios, RandAccessDimIndex, RandNodeIndex)
```

Every time a process received a message to propagate or to modify its sum, there is a ¼ chance for the node to fail temporarily and arrest the message propagation.

We ran the application on the same parameters and created the same plots. The left side plot has the failure feature while the right side does not.



Generally, the gossip algorithm is resilient to node failures. The general trend line of speed is retained across different algorithms and topologies. However we did find that interestingly, the best topology overall, 2d grid, is the most affected by the failure nodes. There is in general a 20% increase in run time with the failure nodes for this topology. When we increase the percentage of failure node, this effect become even more apparent.