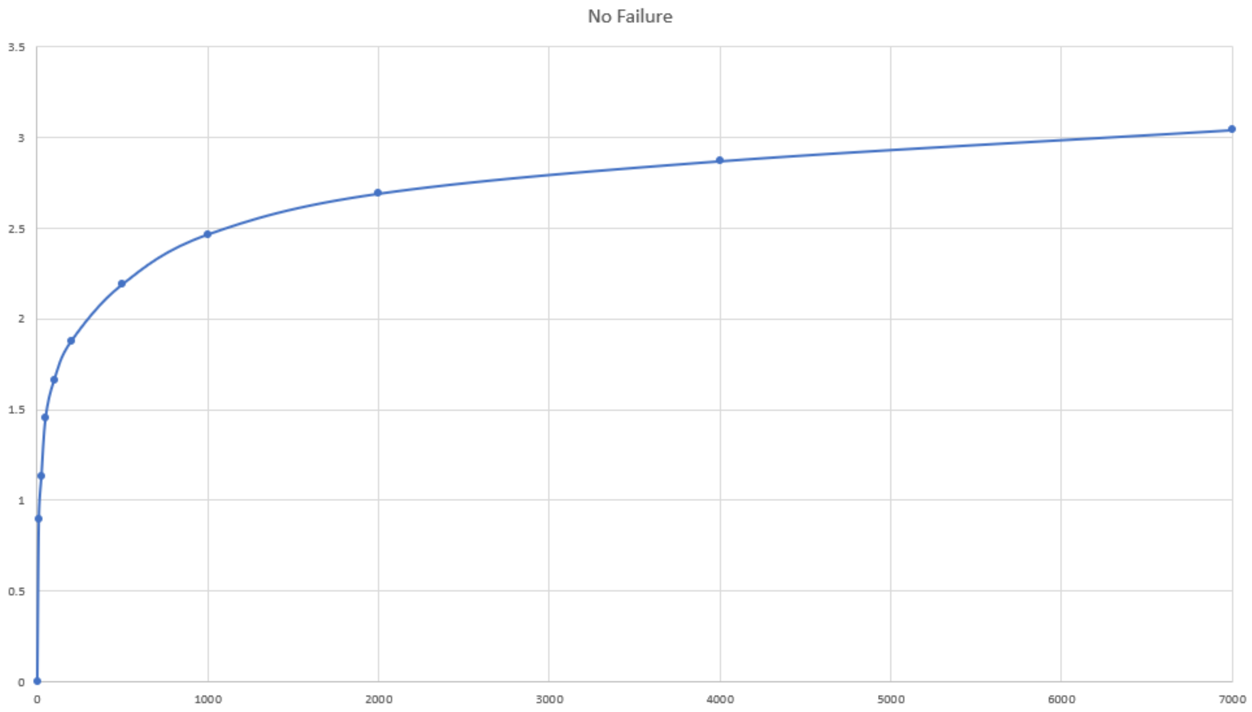
The largest number of node I run is 7000.

Program will initiate node one by one. And node will not start to connect to system until its predecessor finishes. After all nodes finish, server will cast number of requests to each node and they begin to send corresponding number of requests. System finishes when each node completes task and it will wait few seconds for all requests ending. Results include number of each hop and average hops.

1. No Failure:

I tested the program using different number of nodes and got following graph, which shows the average hops to find a key in pastry is O(log(N)). Detailed experiment results are in data.xlsx.

NumNodes	1	10	25	50	100	200	500	1000	2000	4000	7000
Average hops	0	0.896	1.131	1.456	1.663	1.877	2.191	2.466	2.691	2.870	3.041



2. Failure:

In failure mode, program will kill one tenth of nodes after initiating all the nodes. Then each alive node will send 50 messages to make system stable. Then program will measure the result of next several round messages. To show the resilience of pastry, result should reveal all messages are received properly and it should be like the result of nine tenth of nodes that run on no failure mode. Below two tables show experiment results, which proves pastry system is resilient.

numNodes	Average hops			
	no failure	failure	difference	0.9 of numNodes, no failure
100	1.6625	1.601111	0.061389	1.61074
200	1.877	1.835259	0.041741	1.868333
500	2.191	2.152074	0.038926	2.092875
1000	2.4655	2.408629	0.056871	2.416555

Request Send and receive								
NumNodes	Total Request	0(hops)	1(hops)	2(hops)	3(hops)	4(hops)	5(hops)	Received Request
100	1800	19	682	1097	2			1800
200	5400	29	1072	4055	244			5400
500	13500	30	1281	8840	3304	45		13500
1000	27000	34	1589	12866	12346	151	14	27000