COP5612 – Fall 2020 / Project 2

TEAM MEMBERS

— Hsiang-Yuan Liao, UFID:4353-5341, hs.liao@ufl.edu
— Tung-Lin Chiang, UFID:9616-8929, t.chiang@ufl.edu

FILES

— project2.fs (main file)
— directory file for sharp project (.fsproj)

GOAL

The goal of this project is to determine the convergence of gossip algorithm and push-sum algorithm in different topologies. Hence, we build an actor-based simulator for the implementation.

ALGORITHM

— ***Gossip algorithm for information propagation*:**

In gossip algorithm, a node would be triggered by the boss actor and starts spreading the rumor to a randomly selected neighbor. Then, each actor tells the rumor to a randomly selected neighbor. For termination, each actor keeps track of the times of receiving the rumors and terminates if the counter reached a threshold (default: 10).

— ***Push-sum algorithm for summation computation*:**

In push-sum algorithm, a ***node i*** maintains two values: ***s = i*** and ***w = 1*** under a specific ***time t***. To start the algorithm, all ***nodes i*** send ***s = i*** and ***w = 1*** to itself. When a node start to send messages at time ***t***, it sums up all the ***s*** and ***w*** value in the messages that it received at time ***t-1***. After that, it sends a message with ***half value of the s*** and ***half value of the w*** to a randomly selected neighbor node and itself. The algorithm moves on until all nodes have a "***converged ratio R"*** at time ***t***, which R = s / w at a specific time ***t***. If the ratio ***R*** of a node has not changed more than 10^-10 in ***"consecutive rounds C"*** (default c = 3). We define that ratio to be 'converged ratio'. Nodes will be terminated once it owns a converged ratio ***R***.

TOPOLOGY

There are four kinds of topologies:

- ***Full****: Every actor is a neighbor of all other actors.*
- ***Line****: All actors are arranged in a line and each one has only 2 neighbors.*
- ***2D Grid****: All actors form a 2D grid and each one has 2~4 neighbors in the 4 directions.*
- ***Imperfect 2D Grid****:*

  All actors form a 2D grid and each one has 3~5 (1 randomly selected) neighbors.

(Note: In 2D based topologies, we round up the **numNodes** to a nearest square number.)

INPUTS

To run the program, we need to provide 3 inputs:

dotnet run numNodes topology algorithm

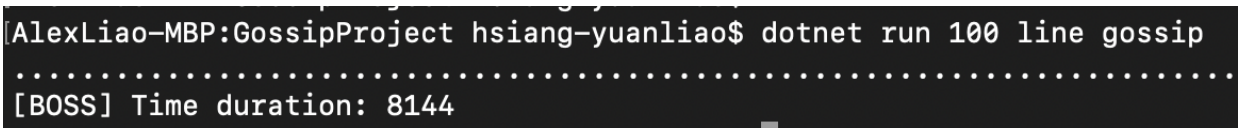
- ***numNodes****:* number of nodes
- ***topology****:* full, 2D, line, imp2D
- ***algorithm****: gossip, push-sum*

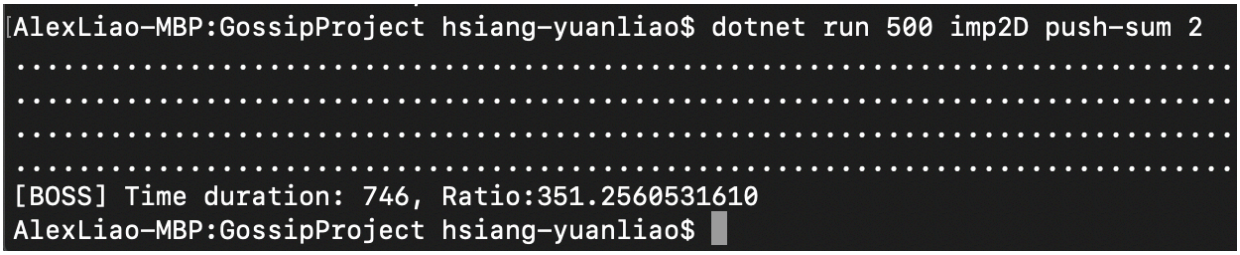
Besides, we find out that the parameter ***"consecutive rounds C"*** for converged ratio determination plays an important part in getting an accurate ratio ***R***, which is the **estimate of the average** for the whole topology. Hence, we add an optional input for push-sum algorithm.

dotnet run numNodes topology push-sum [rounds]

- ***rounds****: number of consecutive rounds (push-sum only)*

OUTPUTS





- ***Time duration:*** shows the millisecond time gap between the start and end of the algorithm
- ***Ratio:*** shows the ratio of the last converged node in push-sum algorithm

IMPLEMENTATION

We use a "BOSS" actor as a supervisor for all node actors. It has two important works to do. First, it triggers both algorithms to start and decides when the algorithms should stop while calculating the time duration. Second, it also works like a trigger of periodic timer. It will receive a message every 50 milliseconds from the main() function. On receiving that periodic message, it triggers all the "active" actor nodes to send rumors in gossip algorithm and all the actor nodes to send (s, w) pair messages in push-sum algorithm.

It makes the both algorithm to have a kind of global timer to send messages almost at a specific time periodically. It seems like to be a little out of the scope of asynchronous Gossip algorithm, but remains some asynchronous factor by using actor model. However, It would much better to implement local periodic timers for each node.

It is important to have this kind of periodic timer to constrain the message sending frequency for each node and message quantities in the whole topology. If we don't implement this feature and let the actor nodes to "free-send" messages, we will easily get unpredictable and incorrect results. More details would be discussed in observation part below.

— *Gossip*

At the beginning, the boss actor receives a message "***Start"***, which starts the algorithm by sending a rumor to the first node. The boss actor monitors the algorithm and node status by keeping a set of nodes and a "***propogationCount***" counter.

The set that maintained by the boss actor is used to keep track of all the "active" nodes. When a node receives its first rumor, it sends a register message to the boss actor, which adds it to the node set. Once a node reaches the threshold of times receiving rumors, it sends a unregister message to boss for removing itself from the node set. With this node set, boss actor is capable of triggering "active" actors to send rumors periodically (default period: 50ms).

The "***propogationCount***" counter adds up every time the boss actor received a register message from a node. When the counter equals the total number of nodes, the algorithm would be terminated immediately and shows the time duration.

Each actor node initials with a neighbor set and a rumor hearing counter. The neighbor set is built according to the topology while the rumor hearing counter adds up every time it receives a rumor. Actor nodes simply receive rumors from neighbors and sending rumors to a randomly selected neighbor periodically.

— *Push-sum*

At the first round (time t: 0), to start the push-sum algorithm the boss actor sends "***Push***" messages which contains (S=i, W=1) to all nodes. As a period of time passed (default period: 50ms), the boss actor triggers all actor nodes to send "***Push***" messages. In addition, boss actor also monitors the converge status of each node by maintaining a counter. The counter adds up when receiving converge messages from nodes. Once the counter reaches the total number of nodes, which means all nodes get a converged ratio ***R***, the boss actor ends the algorithm and shows the time duration.

On the node actor side, it sums up the S and W value in every received "***Push***" message from the last time period (last round). When the next time period comes, it triggers by boss actor. Then, it sends "***Push***" message which contains (1/2*S , 1/2*W) value pair to a random selected neighbor and itself. After sending Push message, it resets the S and W to 0 for the next coming round.

Node actor also checks if the ratio ***R***=S/W changes no more than 10^{-10} in ***C*** consecutive rounds (default:C=3) before sending any Push message. If it satisfied the converge condition, then it sends converge message with that ratio ***R*** to boss actor.

RESULT (Gossip) :

- **T1~ T5: Time Duration (ms)**
- **Std: Standard Deviation**

Gossip & FULL

numNodes	T1	T2	T3	T4	T5	Average	Std
50	498	498	449	497	551	**498.6**	**36.09**
100	699	599	548	598	648	**618.4**	**57.27**
300	752	752	701	754	753	**742.4**	**23.16**
500	810	811	809	808	911	**829.8**	**45.41**

Gossip & LINE

numNodes	T1	T2	T3	T4	T5	Average	Std
50	3997	4813	5248	5602	4698	**4871.6**	**606.97**
100	9388	10943	10469	10057	10751	**10321.6**	**619.25**
300	30276	29054	30807	28658	29922	**29743.4**	**880.37**
500	47266	50084	51903	53723	48428	**50280.8**	**2600.04**

Gossip & 2D GRID

numNodes	T1	T2	T3	T4	T5	Average	Std
50	1244	1151	1297	1449	1168	**1261.8**	**120.08**
100	1599	1733	1748	1748	1954	**1756.4**	**126.99**
300	2936	2747	2748	3024	2939	**2878.8**	**124.96**
500	3507	3821	3600	3650	3719	**3659.4**	**118.86**

Gossip & IMPERFECT 2D

numNodes	T1	T2	T3	T4	T5	Average	Std
50	698	748	748	662	798	**730.8**	**52.24**
100	1087	749	751	847	997	**886.2**	**151.03**
300	1196	1114	1153	952	1003	**1083.6**	**102.70**
500	1107	1168	1102	1307	1005	**1137.8**	**111.14**

RESULT (Push-sum, consecutiveRounds = 3) :

- R1 ~ R5: Ratio (s/w)
- MSE: Mean Squared Error

Push-sum & FULL

#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	1097	25.53	1148	25.67	1098	25.47	1148	25.65	1297	25.63	1157.6	25.59	25.50	0.09	0.01
100	1250	50.7	1200	50.22	1250	50.63	1250	50.60	1098	50.52	1209.6	50.53	50.50	0.19	0.03
300	1255	150.16	1192	150.69	1303	150.93	1304	150.84	1204	151.06	1251.6	150.74	150.50	0.35	0.15
500	1258	251.15	1255	251.11	1159	251.20	1325	250.09	1262	250.27	1251.8	250.76	250.50	0.54	0.30

Push-sum & LINE (consecutiveRounds = 3)

#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	2048	23.71	2570	12.64	2447	12.06	2299	18.32	2047	31.95	2282.2	19.74	25.50	8.31	88
100	2598	28.96	1650	38.95	1997	39.25	1950	32.07	1500	72.04	1939	42.25	50.50	17.23	306
300	2458	95.17	1999	20.78	2809	14.29	2005	117.32	1948	194.90	2243.8	88.49	150.50	74.65	8303
500	3463	361.58	2765	472.94	2361	17.71	2354	225.79	2609	418.54	2710.4	299.31	250.50	182.31	28972

Push-sum & 2D GRID (consecutiveRounds = 3)

#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	1298	39.11	1966	14.49	1548	28.78	1748	19.19	1599	45.55	1631.8	29.42	25.50	13.06	152
100	1848	15.16	1832	34.25	1749	32.81	1800	85.02	1649	21.30	1775.6	37.71	50.50	27.62	774
300	1707	119.08	1808	154.65	1755	83.91	1705	53.66	2003	130.91	1795.6	108.44	150.50	39.86	3040
500	2020	63.71	1864	348.58	3218	498.85	1759	358.09	2315	322.26	2235.2	318.30	250.50	158.06	24583

Push-sum & IMPERFECT 2D (consecutiveRounds = 3)

#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	1348	34.74	1247	35.36	1447	36.35	1448	32.33	1448	33.6	1387.6	34.48	25.50	1.56	83
100	1550	56.8	1548	48.1	1648	50.71	1731	50.0	1395	53.54	1574.4	51.83	50.50	3.40	11
300	1652	163.61	1645	163.65	1753	149.18	1854	173.55	1600	167.97	1700.8	163.59	150.50	9.03	237
500	1612	201.56	2014	241.22	1560	252.83	1863	274.87	1704	301.4	1750.6	254.38	250.50	37.40	1134

RESULT (Push-sum, consecutiveRounds = 5) :

- R1 ~ R5: Ratio (s/w)
- MSE: Mean Squared Error

Push-sum & FULL

#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	3897	25.49	3877	25.49	3797	25.51	4066	25.49	3857	25.50	3898.8	25.50	25.50	0.01	0.00
100	4302	50.49	4101	50.48	4250	50.53	4103	50.49	4352	50.48	4221.6	50.49	50.50	0.02	0.00
300	4459	150.50	4198	150.48	4545	150.50	4511	150.48	4417	150.52	4426	150.50	150.50	0.02	0.00
500	4648	250.44	4911	250.54	5147	250.50	4527	250.35	4657	250.49	4778	250.46	250.50	0.07	0.01

Push-sum & LINE

#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	8451	28.46	13714	14.57	10107	9.05	11253	14.88	24152	35.78	13535.4	20.55	25.50	11.12	123
100	11871	15.59	18489	52.87	11149	68.99	13146	53.24	15379	25.64	14006.8	43.27	50.50	21.97	438
300	13628	115.40	13645	168.12	15223	125.82	11712	285.38	13993	99.95	13640.2	158.93	150.50	75.07	4580
500	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	250.50	NA	NA

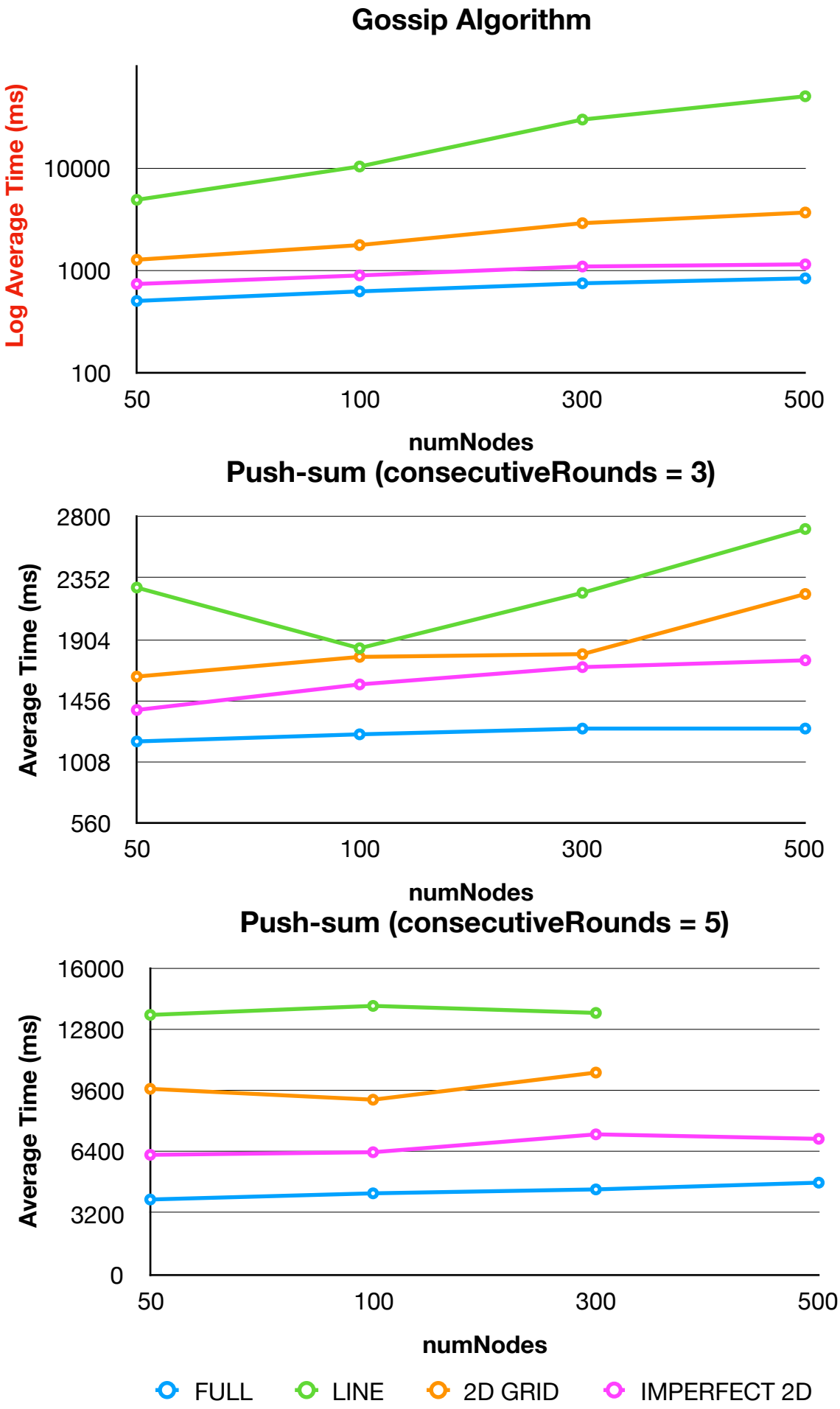
Push-sum & 2D GRID (consecutiveRounds = 5)

#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	9903	30.03	9298	39.67	9777	27.58	9201	26.29	10205	32.43	9676.8	31.20	25.50	5.29	55
100	8828	71.66	9242	73.02	8105	40.87	9614	27.35	9758	55.74	9109.4	53.73	50.50	19.74	322
300	10531	214.91	8895	98.52	9810	58.40	12633	101.52	10760	220.84	10525.8	138.84	150.50	74.16	4536
500	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	250.50	NA	NA

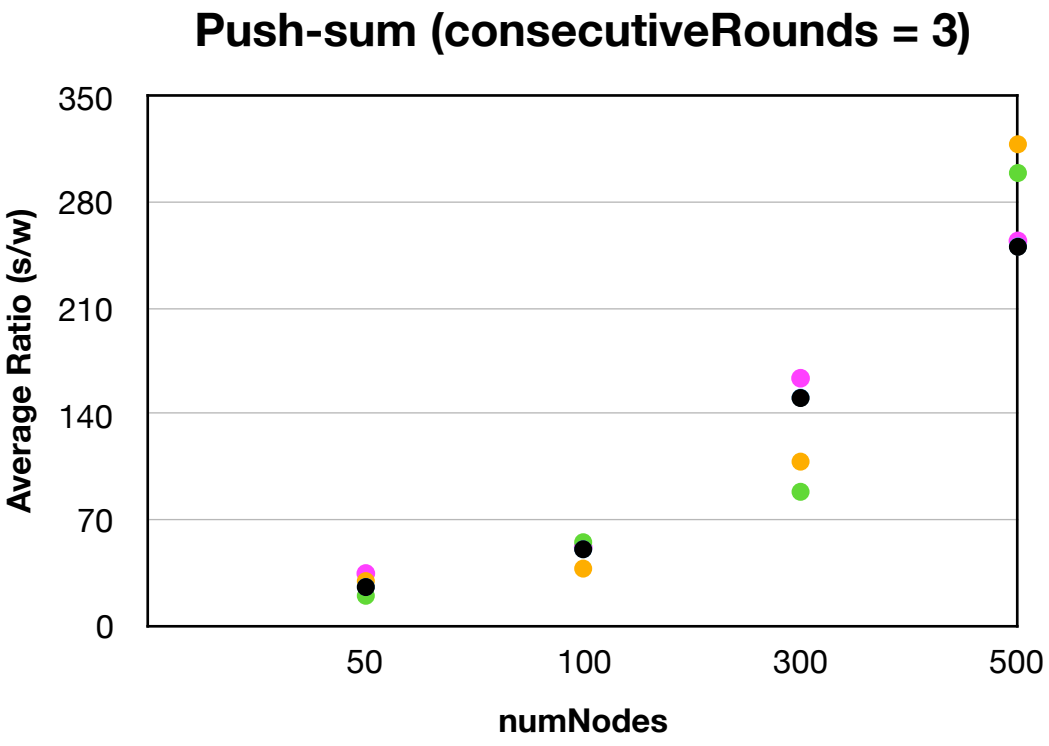
Push-sum & IMPERFECT 2D (consecutiveRounds = 5)

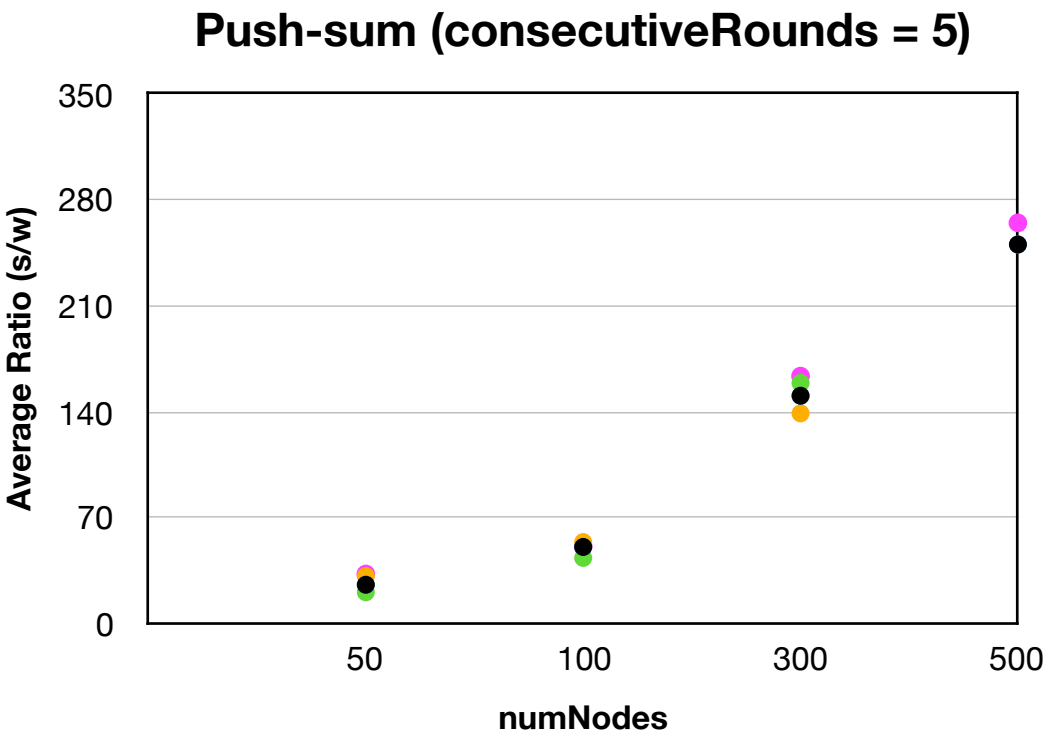
#	T1	R1	T2	R2	T3	R3	T4	R4	T5	R5	Avg. T	Avg. R	Ideal	Std. R	MSE
50	6797	33.40	5151	32.43	7254	32.58	6303	32.91	5652	31.85	6231.4	32.63	25.50	0.57	51
100	6652	51.14	7453	50.78	6310	51.16	5810	51.15	5612	49.78	6367.4	50.80	50.50	0.59	0
300	6159	163.34	9268	166.35	7180	162.41	6285	163.79	7610	161.49	7300.4	163.48	150.50	1.83	171
500	7725	263.04	6799	262.79	6866	262.81	7472	267.00	6468	268.37	7066	264.80	250.50	2.68	210

RESULT (Average time) :



RESULT (Average Ratio) :





IDEAL FULL LINE 2D GRID IMPERFECT 2D

CONCLUSION

From the results above, we have some interesting observations:

Gossip Average time (ms): FULL < IMPERFECT 2D < 2D < LINE

Push-sum Average time (ms): FULL < IMPERFECT 2D < 2D < LINE

Different topologies have different spreading speeds. (see the page 7)

First, considering the different topologies, both algorithms have a much higher speed spreading the information among the nodes in a full topology. With an extra random neighbor, imperfect 2D topology also gets a better spreading speed than 2D topology. This is quite intuitive that the more neighbors a node has, the faster information is spread. That is, the randomness from just one extra neighbor in imperfect 2D makes a significant impact on the spreading speed of information.

Second, there is a possibility that the gossip algorithm cannot be ended, due to some nodes are able to receive any rumor at all. In this situation, those “never activated” nodes are isolated from their “already terminated” neighbors, and network topology may be a crucial factor for this “node isolation” phenomenon. In other words, when a node has a small number of neighbors and its neighbors are localized, it gets a higher chance to be isolated than the node having a great number of neighbors distributed more widely. Note that a localized neighbor means that it is a nearby neighbor of a node. Take 2D and Line topology for example, they are more likely to have an isolated node than imperfect 2D and full topology. Furthermore, we believe that even though a node is in a full topology, there is still a very small chance to be isolated.

Std in Average Ratio: FULL < IMPERFECT 2D < 2D < LINE

MSE in Average Ratio: FULL < IMPERFECT 2D < 2D < LINE

Estimate average bias is smaller with stricter converge condition. (see the page 8)

Third, we find out that the accuracy of ratio estimation within push-sum algorithm is affected a lot by two factors. One is the type of topology and the other one is the consecutive rounds of the condition for convergence (termination). A full topology gets the best estimation of the ratio, which is really close to the ideal value. We think it is the topology that not only helps in spreading out the information but also has the advantage of receiving more pushes in a single round. Although a node in such a topology is hard to converge quickly since the node receives more messages to deal with, it also means the node has more information to make estimations as the algorithm goes on, which ends up with an accurate estimation. In contrast, if the nodes converge too early, it normally gets a bad estimation. Hence, the nodes having localized neighbors in 2D and Line topology give an inaccurate estimation.

On the other hand, we think setting a more strict converge condition can help the topologies which have a great bias. To be specific, a more strict converge condition means a higher number of **consecutiveRounds** or a smaller ratio difference (10^{-10}). For example, 2D and Line would get a more accurate estimate average simply by given a stricter converge condition. It is just like our life. If we would like to get better performance, we have to pay a higher price. No pain, no gain.

INTERESTING OBSERVATIONS

1. While spreading the rumor, the speed is slow in the beginning. Then, it starts accelerating to a high speed level when most of the actors are still active, and finally, it slows down as the number of active actors decreases.

2. If the frequency of spreading rumor is too high, we may encounter some unpredictable problems of convergence. It is crucial to design a rumor-sending mechanism used by active node. If there is only a small delay between two sent rumors. That active node might have sent too many rumors to its neighbors in a small time period. Because the actor model is message driven. It won't able to deal with messages synchronously. Those rumors might piled up pretty fast in one's mailbox. This makes a specific node to deactivate shortly with just sending out 10 rumors (default: 10 times). It really decreases the rumor spreading chances, and also increases the probability of node isolation.