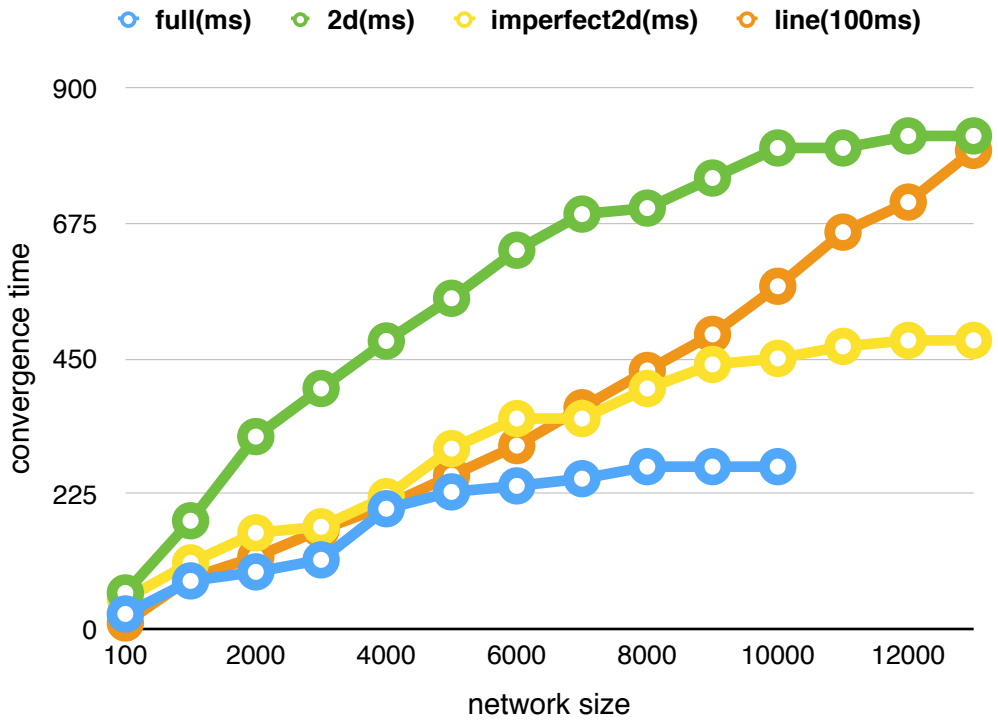
Report of Homework2

1. Gossip Algorithm

1.1 Result

	full(ms)	2d(ms)	imperfect2d(ms)	line(100ms)
100	25	60	50	10
1000	80	180	110	85
2000	95	320	160	120
3000	115	400	170	165
4000	200	479	220	205
5000	228	550	300	255
6000	238	630	350	305
7000	250	690	350	368
8000	270	700	400	430
9000	270	750	440	490
10000	270	800	450	570
11000		800	470	660
12000		820	480	710
13000		820	480	796



1.2 Observation

Some definitions:
n: size of the network
t: convergence time(ms)
maxT: the maximum gossip each actor hears before it shutdown itself

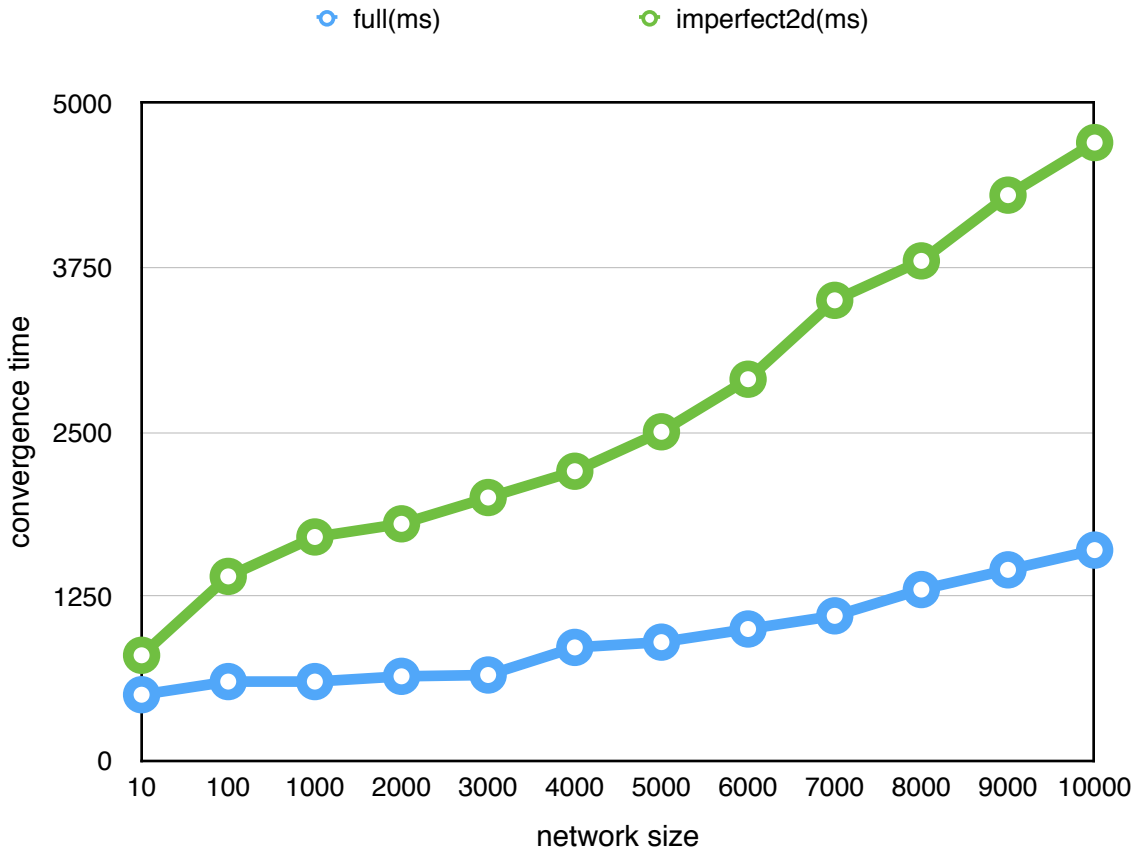
We found that:

- In general, We can notice that in full network, 2d network, imperfect2d network: t = O(log(n)). In line network:t = O(n). And full network is fastest while line network is slowest.
- imperfect2d is always faster than 2d network.
- full network is fast but more difficult to build. And imperfect2d network is also fast while easier to build. So, imperfect2d network may be a great choice if we want to balance the efficiency and management cost.
- During the experiments, we found that t isn't stable, especially when n is small. For example, when n = 5000 and maxT = 10, t can vary from 160 to 320. So, We did a lot of experiments and recorded the average result. This is reasonable since actor always sends the message to an arbitrary neighbor.
- if maxT is too small like maxT = 2, the convergence time can be greatly influenced by maxT, ie, the increase of maxT can significantly reduce t. But when maxT is large enough, like maxT = 10, the increase of maxT has little to do with t.

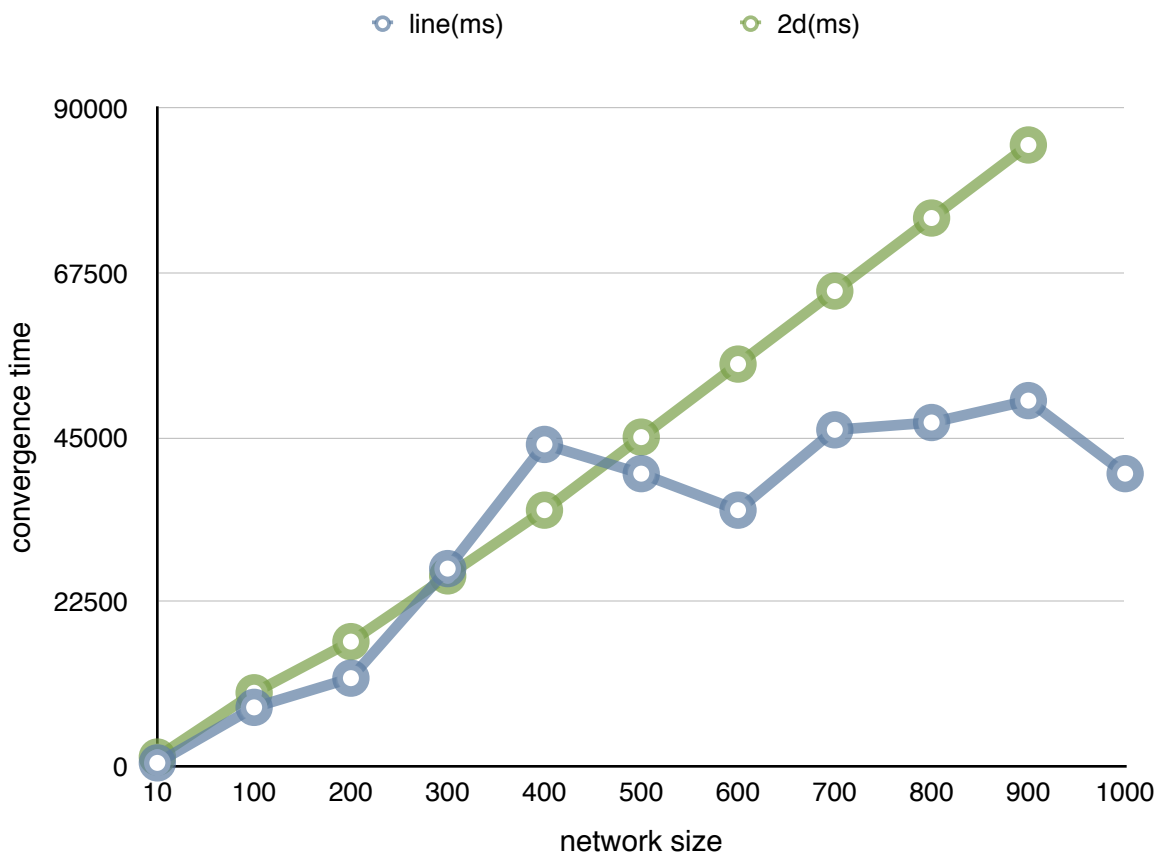
2. PushSum Algorithm

2.1 Result

	full(ms)	imperfect2d(ms)
10	500	800
100	600	1400
1000	600	1700
2000	640	1800
3000	650	2000
4000	860	2200
5000	900	2500
6000	1000	2900
7000	1100	3500
8000	1300	3800
9000	1450	4300
10000	1600	4700
16000		7000



	2d(ms)	line(ms)
10	1200	400
100	10000	8000
200	17000	12000
300	26000	27000
400	35000	44000
500	45000	40000
600	55000	35000
700	65000	46000
800	75000	47000
900	85000	50000
1000		40000



2.2 observation

Some definitions:

n: size of the network
t: convergence time(ms)
maxT: When an actor calculate v/w and gets the same value for continuously maxT times, it shutdown itself.

2.2.1 We found that:

- When n is small($n < 3000$), in full network, $t = O(\log(n))$
- When n is large, in full network, $t = O(n)$
- imperfect2d network shares the similar property as full network
- full network and imperfect2d network is much faster than 2d and line network.
- 2d network is the slowest, even slower than line.

2.2.2 Besides, we found that pushsum algorithm is greatly influenced by maxT.
If maxT is too small, there are potential problems. Firstly, some actors may shutdown itself quickly, as a result, other actors may keep waiting for the incoming message forever. For example, when n = 1000 and maxT = 3, I find that when more than 980 actors have finished their work, the remaining 20 actors just wait for the message and are unable to shutdown. Secondly, since some actors may shutdown very fast, some actors may get the wrong result because they cannot collect enough information.

There are three solutions to solve this problem. Firstly, increase maxT. According to my observation, if maxT is large enough(for instance, n = 2000 and maxT = 20), every actor can work and shutdown properly. But if the maxT is too large, convergence time t will increase greatly since it becomes harder for actors to shutdown themselves. So finding a proper maxT is a big problem. Secondly, when an actor gets the same answer for continuously maxT times, it doesn't shutdown itself, instead, it keeps sending messages randomly to neighbors. Thirdly, although it's possible that some actors can't finish their task, they are only the minority of all actors. So, we don't have to wait for all actors to finish their task to shutdown the system. Instead, we can shutdown the system that when 95 percent of actors finished their tasks.

When running my program, the 7th argument can be set as **nonstop**(strategy 2) or **mostfinish**(strategy 3).

The result I got was based on the strategy 2.

2.2.3
Another interesting finding is that, when n is large, like n = 1000, we can see from the table that in line network, the convergence time is 40000 ms. In fact, when system reported that 100 actors got their results, more than 30000ms time has been spent, and those left 900 actors will got their results in only 10000 ms. This is to say, if some actors get their results, other actors will benefit a lot from them. **This phenomenon demonstrates how powerful flooding is!**