CSE 6010 / CX 4010 Assignment 6
Epidemic Network Model

Ting Liao, GTID: 903278773 | Chin Wang, GTID: 903506612

10/31/2020

In this assignment, we develop create_AdjMatrix_random function to determine the connection between each agent at first. Then we use adjacency matrix in iteration function to update the state of each agent under some specified probabilities in each iteration. In these main functions, probabilities like pInfection, pRecovery, pReplaceRandom and pDisconnect affect the state of each agent. For performing these probabilities, we use if function to check whether the specified probability has effect on agents. Also, srand and rand functions are used to design how the probabilities will happen in each situation.

Sanity check

- To make sure the adjacency matrix is symmetric, we print the adjacency matrix to check whether the Adj[i, j] and Adj[j, i] are 1 if the edge between two agents, i and j, exists and 0 otherwise.
- To implement the experiment, we assume first agent is infected. Then we will check the effects in each iteration. To check the infected agents are infected reasonably through edges, we print adjacency matrix in first several iterations and ensure that the one who just gets infected has entry 1 in matrix with other infected agents.
- To ensure the sum of infected, recovered and susceptible agents are numAgents, we record the state of all agents in each iteration and print. In addition to check the all agents are in experiment, the non-decreasing numbers of recovered agents convince us that the state is reasonably changed from susceptible to infected and to recovered at the end.

Results of experiments

First column: The maximum number of infected individuals in a single iteration
Second column: The iteration number at which the maximum number of infected individuals occurs
Third column: The sum of the number of currently infected and recovered individuals after the last iteration.

	pReplaceRandom = 0		
	pDisconnect = 0		
	27	29	150
	26	57	136
	21	25	64
	20	20	118
	12	16	24
	18	16	53
	14	19	21
	26	37	111
	23	97	127
	27	58	144
MEDIAN	22	27	114.5
MAX	27	97	150
MIN	12	16	21

	pReplaceRandom = 0.25		
	pDisconnect = 0		
	197	31	489
	183	24	488
	177	30	476
	195	32	492
	189	27	488
	185	32	483
	177	39	482
	187	32	487
	202	31	491
	188	32	478
MEDIAN	187.5	31.5	487.5
MAX	202	39	492
MIN	177	24	476

	pReplaceRandom = 0		
	pDisconnect = 0.5		
	9	67	41
	6	5	15
	2	2	6
	7	7	13
	2	2	3
	8	17	42
	8	12	14
	8	24	19
	11	9	35
	8	14	14
MEDIAN	8	10.5	14.5
MAX	11	67	42
MIN	2	2	3

	pReplaceRandom = 0.25		
	pDisconnect = 0.5		
	65	61	293
	59	55	336
	12	27	46
	53	91	225
	57	59	307
	53	57	283
	62	52	305
	39	72	252
	56	53	333
	42	84	232
MEDIAN	54.5	58	288
MAX	65	91	336
MIN	12	27	46

In experiment, we have to control two parameters, pReplaceRandom and pDisconnect. The pReplaceRandom represents the probability that agents can infect any other agent in addition to their closest neighbor. In other words, as pReplaceRandom is greater, the risk of infection is higher. From the results, we can tell that when pReplaceRandom is changed from 0 to 0.25, the data show that the number of infected indiviuals is greater. As for pDisconnect, it represents the probability that connection between each agent might be removed. That is, as pDisconnect is greater, the risk of infection is smaller. From the results, we can tell that when pDisconnect is changed from 0 to 0.5, the data show that the number of infected indiviuals is smaller. Therefore, we can conclude that this experiment works well and satisfies what we expect.

Division of Labor

- At first, Ting developed the network functions (create_array, create_AdjMatrix_random) to build up the network structure. Also, Chin started to build the basic model functions (update_state, calculate and calculate1). Then, we both presented our own ideas to make the iteration function. After that, we found out that Chin's code could perform the result correctly but Ting's not. Therefore, we used the iteration function which was based on the data structures that Chin came up with. After outputting the results, Chin organized the main structure and presented it in the slide. Eventually, Ting divided those functions into two different source file, model.c and network.c. Both of them were connected to the main function via the makefile.
- Ting Liao: network functions, print functions, some suggestions for the iteration function and the makefile steps
- Chin Wang: model functions, most code of the iteration function and the output for the slides.

- All in all, we both contributed great efforts on the Epidemic Network Model program. We appreciate that TA Darby gave us a hand on handling the bug while dividing functions into different source files.