Distributed Computing [EEL 6935]

Shreyas Gaadikere Sreedhara

‘b’ topology

The nodes are distributed on the x-y axis in the form of b, with the last node at the top. The last node has infinite distance to all the nodes except the first and last but one node. In the program I have first checked the topology and decided which part of the code is to be executed. Then I will execute the ‘b’part of code if the input is ‘b’. First, in initialization we pick random ‘k’ elements ranging from 1 to N and fill the neighbor list of each node. In my implementation, each node is a structure. Each node will thus have its own variables. Then after initialization I have calculated the angle, x and y for all the nodes and saved it in their structure. Then the evolution phase starts. In evolution phase each node sends and receives the neighbor list from a randomly selected node from its neighbor list. Then it will merge both its neighbor list and the received list. Then it will calculate the distance for each node in the combined list and arrange it in the ascending order of their distance from the node which is under consideration. Then we pick the first ‘k’ elements from the combined list and update the neighbor list of that node. We also pick the corresponding ‘k’ distances and update the structure of the node. This happens for all the N nodes and then we add all the distances got from all N nodes and then print the sum_of_distances for that cycle. This happens for 40 cycles and for each cycle we get sum_of_distances.

In my code the main function calls and passes the parameters to initialization function and that function in turn calls the random_numbers. The random number returns a vector to the initialization function and the initialization function returns that vector to main function. The evolution part of the code is inside the main function.

For node 1, if it is any other node other than 1 or N, I have used the distance formula given. If it is the same node then the distance for same node is 5000 (infinity). If it is Nth node then the distance is 1. Same rule is followed for N-1 node also. For all other node the distance to N node is set as 5000 and to itself it is set as 4000. This ensures that those neighbors are not picked when picking k smallest elements.

Spectacle topology.

The same rule is followed for spectacle also. But here there is no Nth node. Hence no need to take care of that. The distance formula has been calculated according to the topology. All other things remain the same for the spectacle topology also. There are 5 semi circles in the spectacle. The nodes are distributed evenly on these nodes. Hence each semicircle will have N/5 nodes. We shift the center each time by 2 so that we get the circles such that they meet each other.

The first and second circles will have 2N/5 nodes and middle semi-circle will have N/5 nodes. There will be a total of (5*pi)/N angles in all the circles.

For the middle semi-circle, we have calculated the y by taking negative of sin because we want the semi-circle on the top and not on the bottom. The x-axis keeps shifting by 2 values for each circle. The nodes on the first circle ranges from 1 to 2N/5 and the node ranges from (2N/5) +1 to 3N/5 on the semicircle and on the last circle it ranges from (3N/5)+1 to N.

For 1st circle,

Angle = (node_id - 1) *((5*pi)/N)

X = 1 + cos(angle)

Y = sin(angle)

For semicircle,

Angle = pi + ((node_id-(2*N/5)) * ((5*pi)/(N+5))

X = 3 + cos(angle)

Y = -sin(angle)

For 2nd circle,

Angle = pi + ((node_id-(3*N)/5)) * ((5*pi)/N)

X = 5 + cos(angle)

Y = -sin(angle)

Question and answers.

1) Since there is only one semicircle in the ‘b’ topology, the nodes won’t be separated anyways. The angle starts from (pi/2) and goes on decreasing. In the spectacle, there are 3 structures and we need to make sure that the structures meet each other. Hence, we are assuming that the radius of the circle is 1 and keeping the centers of the circles 2 units apart so that they meet each other.
2) Yes, the nodes can be repeated in the merged list of the node when some node it had in its neighbor list is in the neighbor list of the node which it is contacting. We must eliminate such repeated nodes from the merged array because it doesn’t make sense to have the same node twice in the same spot in the network. So, in conclusion it can’t have repeated elements in its neighbor list.