Documentation of Overlay Network in a Simulation

This writing is a documentation of the model of an overlay network in a simulation. The model visually shows the "real" network and the overlay network in operation. This model can accommodate as much node as a user wants, where each node will be connected more than 25% of the other real nodes randomly. The overlay network is structured in a manner that it divides the real network into several regions based on the number of super peers provided, where each super peer connected to a definite group of nodes. A user can provide the required number of super peers as well as a starting and ending node to route in the network during runtime. This model dynamically assigns all the nodes with their assigned super peers. This model provides the route requested by the user from the starting node using Breadth First Search algorithm, which ensures the minimum distance route for the search.

For this simulation, a Graph class having functions add_vertex, add_edge, and breadth_first_Search has been implemented. Edges (network connections between nodes) are added maintaining at least 25% connections with every other node in the real network. To maintain that ratio a for loop is used from the first node to total (N) number of nodes incrementing four each time(see Appendix: pseudocode). To search a route, starting node, ending node and an empty "path" list are fed to the breadth_first_Search function of the Graph class, where "path" stores the whole route until the search finds the ending node. After execution, this model prints out the route from the "path" list.

Same Graph class with all of its functions are also used for the overlay network. However, constructing the overlay network is slightly different than the real network. This model divides the total number of nodes with the provided number of super peers to find the total number of regions including the total number of nodes in a region. After that, the model uses an iteration starting from the first node until it covers the calculated number of nodes in a region and thereby includes all the nodes in a super peer. Similarly, all the super peers are assigned a definite number of nodes chronologically, and at last, all the super peers are connected fully. Searching a route in overlay network is as similar as the searching route in the real network. By using Breadth-first Search algorithm to route a query this model ensured that a search in overlay network starts from the source node to a super peer closed to that source node to a super peer near the destination node to the destination.

Appendix:

Pseudocode:

//Creating peer to peer network

```
Input total number of nodes to N
Iterate i 0 to N-1
    Add node i to graph
Iterate i 0 to N-1
  Iterate j 0 to N-1
     j equal to j+4
     If i equal to j and i not equal to 0 and i not equal to N-1
           Add edge i and j+1 to graph
     Else If i equal to j and i equal to 0
           Add edge i and j+1
     Else If i equal to j and i equal to N-1
          Add edge I and j-1
     Else
          Add edge i and j
```

//Creating peer to peer overlay network with super peer

```
Initializing empty list “normal”
Initializing empty list “superpeer”
Input total number of superpeers to P
Number of total region, r= N/P
Iterate I from r-1 to N
     add node i to overlay network
    i assigned to i+r
    append i to list “superpeer”
Iterate i from 0 to N
```

```
    If i not in “superpeer”
    Add node i to Overlay network
   Append i to list “normal”

Iterate i from 0 to N
    If i not in list “superpeer”
    Add node i to Overlay network
    Append i to list “normal”

Iterate i from 0 to length of list “superpeer”
    If i equal to 0
        Iterate j from 0 to superpeer[i]
                Add edge [superpeer[i], j] to Overlay network
   Else If I equal to “length of list “superpeer” -1”
        Iterate j from superpeer[i-1]+1 to N
                Add edge [superpeer[i-1]+1, j] to Overlay network
   Else
        Iterate j from superpeer[i-1]+1 to superpeer[i]
                Add edge [  superpeer[i], j]

Iterate i from 0 to length of list “superpeer”
        Iterate j from 0 to length of list “superpeer”
                If I not equal to j
                        Add edge [superpeer[i], superpeer[j] ] to Overlay network
```

//Query a route in peer to peer real network and Overlay network

```
Input starting node to A
Input end node to B
```

Initializing an empty list to “path”

Initializing an empty String “output”

Providing argument to function Breadth first search (A, B, path)

While end node B not equal to starting node A

    U assigned to path[B]

    Output assigned to u + v + output

    B assigned to u

Print “Source to destination path “

Flowchart:


Start
Input total number of nodes= N
Input starting node = A
Input end node = B
Input total number of superpeers = P
Initializing an empty list to "path"
Initializing an empty String "output"
Iterate i 0 to N-1
Add node i to graph
Iterate i 0 to N-1
Iterate j 0 to N-1
j equal to j+4
If i equal to j and i not equal to 0 and i not equal to N-1
Add edge i and j+1 to graph
If i equal to j and i equal to 0
Add edge i and j+1
If i equal to j and i equal to N-1
Add edge I and j-1
Add edge i and j
Providing argument to function Breadth first search (A, B,path)
While end node B not equal to starting node A
Print "Source to destination path "
U assigned to path[B]
Output assigned to u + v + output
B assigned to u
End