**Name : Amir Patel Email :** [**amir.patel76@gmail.com**](mailto:amir.patel76@gmail.com)

**Ventursity user code : 59643**

**Design document for GRAPH algorithms**

**Input Specifications :**

1> we will provide following input to through command line

a> No. of threds needs to be created.

b> Each thread’s vertex no. , no. Of edges it want to have

c> For each vertex forming an edge with thread’s vertex, we will provide weights.

2> This data will be written to a file in the form of (u, v) = w .

where,

u = thread vetex no. which is running

v = vetex no. to which it wants to form an edge

w = weight

In this way whatever thread is run, initially it should update it’s neighbouring data into file

**Output Specification :**

**Algorithm :**

1> Read How many threads we need to create , suppose n

2> Read input for i = 0 to n

i> Thred vertex no.

ii> No. Of edges it need to form, and their corresponding weights

3> Write Data data to gnode structure, which is each thread specific and this will be going to back passed as thread Arg while thread creation.

3> A global file which is shared among all process will contain all data related to graph

in following form :

4 6

0 1 2

0 2 3  
 1 2 4   
 1 3 5  
 2 3 7

Where, In the first line, 4 is no of vertices, 6 is edges

2nd line onwards we have data in following form,

vertex-X vertex-Y edge

NOTE: This file represents whole network graph.

4> When each thread gets created, It performs following things.

a> Read data from arg and update it in Global file. Duplication handling is also done over here.

b> register interrrupt handler for each thread, so that it can handle signal from pthred\_kill().

Upon re­ception of signal, thread will remove it entries from Global file.

c> periodically read file and update GRAPH data with neighbours.

d> File handling is doen with preacuation with avoiding race conditions.

5> Reading the file to form GRAPH

i> This will be a periodic activity in all thread or in while(1) loop.

ii> We will basically create an array of linked list to form an adjecency list for all nodes.

iii> Let the array be arr[], so each entry in array arr[i] will represents list of vertices which are adjecent to vertex ‘i’.

iv> Each node represents a vertex no with weight and link to next node.

v> By reading each line from we file, we will form a graph.

6> Finding a SHORTEST PATH

i> read to vertices to find shortest path between them.

i> For this, we will be using Dijkstra’s algorithm.

ii> We will create a data structure Minheap which is of size n. i.e no. of vertices.

iii> Each node in Minheap will contain vertex number and minimum distance value of the index.

iv> Initailze Minheap with starting vertex with 0 and value for all other vertices is INF.

v> Iterate loop till Min heap is not empty

a> Extract the vertex with minimum distance value node from Minheap. Let the extracted vertex be x.

b> For every adjacent vertex v of u, check if v is in Minheap. If v is in Min Heap and distance value is more than weight of u-v plus distance value of u, then update the distance value of v.

7> Find the minimum spanning tree

i> Prim’s algorithm will be used in for finding minimum spanning tree.

ii> Create a Minheap of size V where V is the number of vertices. Every node of Minheap contains vertex number and key value of the vertex.

iii> Initialize Minheap with first vertex as root , key value assigned to it is 0. The key value assigned to all other vertices is INF .

iv> While Minheap is not empty, do following

a> Extract the min value node from Minheap. Let the extracted vertex be u.

b> For every adjacent vertex v of u, check if v is in Min Heap (not yet included in MST). If v is in Min Heap and its key value is more than weight of u-v, then update the key value of v as weight of u-v.

8> Fault confinement in threads

If any threads dies, with any signal say, SIGKILL or SIGTERM we will handle those signals in handler and remove the information of current node from file before exiting. This way, when next time a process creates GRAPH by reading values form file, it will update appropriately.