1. Data structure used to store the graph?

Ans: Adjacency List is being used to store the graph.

Reason : a) Adjacency lists allow quick addition and removal of edges from a graph. b) Adjacency lists facilitate efficient path finding, making algorithms like Dijkstra's effective for traversing graphs. c)Operations on adjacency lists can be parallelized, benefiting multi-threaded applications.

2. What are the additional data structures ? Why did you use them

Ans: The data structures used are:

a) Structure used to store the graph.

```
typedef struct Node
{
    int value; // Node identifier
    int is_landmark; // 1 if node is a landmark, 0 otherwise
    int partition; // Partition identifier
    struct Node *neighbors; // List of neighboring nodes
} Node;

typedef struct Graph
{
    int num_nodes;
    Node *nodes; // Array of nodes
} Graph;
```

b) Structure used to store the partition information of the graph.

```
typedef struct Partition
{
    int num_nodes; // Number of nodes in the partition
    int *nodes; // Array of node identifiers
    int landmark; // Landmark identifier
} Partition;
```

c)Linked list to store the nodes in the graph for the paths

```
typedef struct PathNode
{
    int node;              // Node identifier in the path
    struct PathNode *next; // Pointer to the next node in the path
} PathNode;

typedef struct
```

```
{
    int distance;   // Node identifier
    PathNode *head; // Head of the linked list representing the path
} Path;
```

3. What are the locks and semaphores used(ifany).

Ans: I have used mutex lock to lock the log file while writing to it. This is required because otherwise different threads might try to write simeltaneously to the log file and the output will be garbled.