Experiment 05

Name :- Kiran K. Patil Sem :- 06
Reg id :- 211070904 Course :- PC. Lab

Aim :- Implement DIJKSTRA's algorithm using OpenMP.

Theory :- Dijkstra's algorithm is a well known algorithm for finding the shortest path between nodes in a graph the algorithm works by maintaing a set of vertices whose shortest distance from the source node is already known.

The algorithm works by maintaining a repeatedly selects the vertex with the smallest distance value from the source node that is not yet in the set and updates the distance values of its neighbors. This process is repeated until all vertices have been added to the set.

OpenMP is a popular API for parallel programming in shared-memory architectures. By using openMP we can parallelize Dijkstra's algorithm and speed up its execution on multicore processors.

To implement Dijkstra's algorithm using OpenMP, we can parallelize the loops that updates the distance values of the neighbors of the selected vertex. Each thread maintains a private copy of the distance array and set of visited vertices.

When a thredd selects a vertex to add to the set, it updates the distance array and the set of visited vertices. When a thread selects a vertex to add to a set it updates the distance array and marks the vertex as visited in its private copy.

After all threddds have finished processing their private copies, the main thread merges all the results by selecting the smallest distance value for each vertex across all private copies.

- To achive correct and efficient parallelization we need to use synchronization primitives such as 'reduction', 'atomic' and 'critical' directives to manage shared and private data and ensure correct results. By parallelizing Dijkstras algorithm using openMP we can - achive significant speedup on multicore multicore processors, especially for large graphs with many vertices and edges.