Assignment 11

Name: Hitesh Tolani

Roll no: 73

Class: SY-AIDS-A

Title: Write a program to implement a. Network Routing: Shortest path routing, AODV

Theory:

Shortest path routing

Dijkstra's algorithm is a popular method for finding the shortest path between nodes in a graph with non-negative edge weights. It starts from a source node and iteratively explores the neighbouring nodes, updating their tentative distances from the source. At each step, it selects the node with the smallest tentative distance as the next one to explore, ensuring that the path to that node is currently the shortest known. This process continues until all reachable nodes have been visited, guaranteeing the shortest path from the source to every other node in the graph.

AODV

Ad-hoc On-demand Distance Vector (AODV) routing protocol is designed for wireless ad-hoc networks where nodes may dynamically join or leave the network. AODV establishes routes between nodes only when necessary, in response to specific data packets needing delivery. It employs a distributed algorithm where each node maintains routing tables containing information about available routes to other nodes. When a node needs to send a packet to a destination for which it has no route information, it initiates a route discovery process. During route discovery, control packets are broadcasted through the network, and nodes update their routing tables to establish a route. AODV is efficient for dynamic and mobile networks, as it adapts to changes in network topology without requiring periodic updates, conserving bandwidth and resources.

Code

```
class NetworkRouting:
    def __init__(self, graph):
        self.graph = graph

    def shortest_path(self, source, destination):
        visited = set()
        distances = {node: float('inf') for node in self.graph}
        distances[source] = 0
        predecessors = {}
        while len(visited) < len(self.graph):
            current_node = None
            min_distance = float('inf')
```

```
            for node in self.graph:
                if distances[node] < min_distance and node not in visited:
                    min_distance = distances[node]
                    current_node = node
            visited.add(current_node)
            for neighbor, weight in self.graph[current_node].items():
                if distances[current_node] + weight < distances[neighbor]:
                    distances[neighbor] = distances[current_node] + weight
                    predecessors[neighbor] = current_node
        path = []
        current_node = destination
        while current_node != source:
            path.insert(0, current_node)
            current_node = predecessors[current_node]
        path.insert(0, source)
        return path

class AODV(NetworkRouting):
    def __init__(self, graph):
        super().__init__(graph)
        self.route_table = {}  # route table for AODV

    def discover_route(self, source, destination):
        # AODV Route Discovery
        if source not in self.route_table:
            self.route_table[source] = {}
        if destination not in self.route_table[source]:
            shortest_path = self.shortest_path(source, destination)
            if shortest_path:
                self.route_table[source][destination] = shortest_path
            else:
                return None
        return self.route_table[source][destination]

# Example usage
if __name__ == "__main__":
    # Creating a sample network graph
    graph = {
        "A": {"B": 1, "C": 2},
        "B": {"A": 1, "C": 1, "D": 3},
        "C": {"A": 2, "B": 1, "D": 1, "E": 4},
        "D": {"B": 3, "C": 1, "E": 1},
        "E": {"C": 4, "D": 1}
    }

    # Initialize routing protocols
    shortest_path_routing = NetworkRouting(graph)
    aodv_routing = AODV(graph)
```

```
    # Example usage of shortest path routing
    print("Shortest path from A to E:",
shortest_path_routing.shortest_path("A", "E"))

    # Example usage of AODV routing
    print("Route discovered by AODV from A to E:",
aodv_routing.discover_route("A", "E"))
```

Output:

```
Shortest path from A to E: ['A', 'C', 'D', 'E']
Route discovered by AODV from A to E: ['A', 'C', 'D', 'E']
```

Conclusion:

In this assignment, we have learned about network routing and its algorithms such as shortest path and AODV. We have implemented a python program for the same.