SIMULATION OF DSDV ROUTING PROTOCOL: REACTION TO LINK FAILURES AND ROUTING TABLE CONVERGENCE

A Micro Project Report

Submitted to the APJ Abdul Kalam Technological University

in partial fulfillment of requirements for the award of degree

Master of Technology

in

COMMUNICATION SYSTEMS

by

ARDRA V GANESH

under the Guidance of

Prof.Sanoj Viswasom

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

COLLEGE OF ENGINEERING TRIVANDRUM

THIRUVANANTHAPURAM

October 2025

ABSTRACT

The Destination Sequenced Distance Vector (DSDV) is a proactive routing protocol for Mobile Ad-Hoc Networks (MANETs) that maintains loop-free routes to all known destinations by using sequence numbers. When a link failure occurs, the protocol marks the affected destinations as unreachable by assigning an infinite metric and increments their sequence numbers to broadcast updated route information. These updates propagate through the network, prompting all nodes to review their routing tables to reflect the new, consistent topology. The process, known as routing table convergence, is driven by both periodic and triggered updates that carry the latest sequence numbers. By simulating this behavior, it becomes possible to quantify key performance metrics, including convergence time and routing overhead, which reveal how effectively DSDV adapts to dynamic network conditions and maintains stable communication in mobile environments. Such simulations are crucial for understanding protocol performance in scenarios where node mobility and frequent topology changes can severely disrupt communication. The study also emphasizes the trade-off between timely convergence and control overhead, highlighting the challenges of ensuring reliable communication in highly dynamic MANETs. The insights gained from this evaluation contribute to a better understanding of proactive routing protocols and provide a foundation for exploring potential improvements, such as enhanced versions of DSDV designed to reduce packet loss and improve efficiency under frequent topology changes.

Contents

List of Figures

Chapter 1

Introduction

Mobile Ad-Hoc Networks (MANETs) have emerged as an essential communication paradigm in scenarios where fixed infrastructure is either unavailable, impractical, or destroyed. Unlike traditional wired or wireless networks that rely on centralized infrastructure such as routers or access points, MANETs are composed of self configuring mobile nodes that communicate with each other over wireless links. Each node functions not only as a host but also as a router, forwarding packets for other nodes. This decentralized and infrastructure-less nature makes MANETs extremely useful in applications such as disaster recovery, emergency operations, battlefield communications, vehicular networks, and remote area connectivity.

However, the highly dynamic and unpredictable topology of MANETs introduces significant challenges in designing efficient and reliable routing protocols. As nodes frequently join, leave, or move within the network, the topology changes rapidly, resulting in frequent route breakages. This demands routing protocols that can quickly adapt to topology changes, maintain loop-free routes, and ensure timely delivery of packets with minimal overhead. Consequently, routing in MANETs has become one of the most critical research areas in wireless networking. Routing protocols in MANETs are generally classified into three categories: proactive (table-driven), reactive (on-demand), and hybrid protocols. Proactive protocols, such as Destination Sequenced Distance Vector (DSDV), maintain fresh and consistent routing tables by periodically exchanging control messages, ensuring that routes to all destinations are always available. Reactive protocols, such as AODV (Ad hoc On-Demand Distance Vector) and DSR (Dynamic Source Routing), create routes only when required by a source node, thereby reducing control overhead but introducing route discovery delays. Hybrid protocols combine the advantages of both proactive and reactive approaches. Among these, proactive protocols like DSDV are especially important in scenarios where continuous connectivity and low latency are required, as they eliminate the delay associated with route discovery. The Destination Sequenced Distance Vector (DSDV) protocol, proposed as an enhancement of the classical Bellman–Ford algorithm, was one of the earliest and most influential routing protocols designed for MANETs. It introduces the concept of sequence numbers to guarantee loop free routes and avoid the well known count-to-infinity problem of distance vector routing. Each node maintains a routing table containing information about all reachable destinations, their next hops, hop counts, and sequence numbers. Updates are exchanged periodically in the form of either full dumps or incremental updates, ensuring that all nodes share a consistent view of the network. When a link failure occurs, DSDV reacts by assigning an infinite metric to the affected route and incrementing its sequence number, broadcasting this information to notify all nodes. The updated information propagates throughout the network until all nodes adjust their routing tables, a process known as routing table convergence. Despite its ability to provide loop free routes and relatively fast convergence compared to traditional distance vector algorithms, DSDV also introduces challenges. Frequent topology changes in MANETs may cause the protocol to generate high routing overhead due to periodic and triggered updates, leading to increased bandwidth consumption. Moreover, while convergence ensures consistency, the transient phase before stabilization can result in packet loss, increased delay, and reduced delivery ratio. These limitations underline the importance of analyzing and simulating the behavior of DSDV under dynamic conditions, especially in response to link failures, which are the most common cause of route disruption in MANETs.

The motivation for this work stems from the need to evaluate how effectively DSDV adapts to frequent topology changes and how quickly it can reconverge to a stable state after a failure. Through controlled simulations, critical performance metrics such as convergence time, routing overhead, and packet delivery ratio are measured to evaluate the efficiency and reliability of the protocol. This analysis also helps in understanding the trade-offs between proactive route maintenance and control overhead, thereby offering valuable insights into the suitability of DSDV for different MANET scenarios. Accordingly, this report presents a detailed simulation study of DSDV under link failure conditions, with particular emphasis on the process of routing table convergence. The findings are expected to highlight the protocol's strengths in maintaining stable communication as well as its limitations in highly dynamic environments.

Chapter 2

Literature Review

Mobile ad-hoc networks (MANETs) have been widely studied due to their ability to provide communication without fixed infrastructure. One of the major research areas in MANETs is the design and performance evaluation of efficient routing protocols. Several studies have been carried out to analyze and compare protocols such as DSDV, AODV, and DSR under different network conditions.

In [1], the authors present a comparative evaluation of three widely used MANET routing protocols, namely AODV, DSR, and DSDV. Their work highlights the fundamental differences between proactive and reactive routing, showing that while proactive protocols like DSDV maintain up-to-date routes with low delay, they incur higher control overhead compared to on-demand protocols. This trade-off remains central to evaluating routing performance in dynamic topologies. A comparative study on routing protocols for MANETs and wireless sensor networks was reported in [2], where the authors emphasized the importance of efficient protocol selection depending on mobility and density of nodes. They concluded that DSDV is suitable for smaller or moderately dynamic networks due to its faster convergence, but may not scale as efficiently in highly mobile scenarios. The convergence behavior of routing protocols has also been studied in detail. In [3], convergence times of various MANET algorithms were evaluated, and DSDV was observed to provide relatively stable convergence due to its sequence number mechanism. However, the study also indicated that frequent topology changes increase overhead, which in turn impacts packet delivery efficiency. Research in [4] proposed improvements to the standard DSDV protocol by optimizing its update mechanisms. Their results demonstrated that enhanced versions of DSDV can achieve lower routing overhead and faster adaptation to topology changes, thereby improving packet delivery ratio in dynamic environments. Similarly, [5] introduced a new variant of DSDV with modified update strategies to further reduce packet loss after link failures, showing promising performance gains compared to the original protocol. To address the efficiency problem, [6] presented an Efficient DSDV (E-DSDV) variant that improves the handling of triggered updates, thereby reducing unnecessary control traffic. The results showed better throughput and packet delivery, highlighting the importance of refining update mechanisms. Another performance comparison between AODV and DSDV under varying node densities was carried out in [7], where it was found that while AODV adapts better to highly mobile scenarios, DSDV provides lower latency in relatively stable topologies. More recent work in [8] analyzed the performance of topology based protocols under different node speeds. The study demonstrated that DSDV suffers performance degradation with increasing mobility due to the cost of frequent updates, yet it still offers predictable convergence and loop-free routes, which remain advantageous in applications requiring consistency. Finally, [9] and [10] provide broader comparisons of routing protocols in MANETs, reinforcing the notion that no single protocol is universally optimal. These studies highlight that proactive protocols like DSDV are better suited for networks requiring low-latency route availability, whereas reactive protocols perform better in highly dynamic networks where minimizing overhead is critical. In summary, the literature establishes that DSDV remains a fundamental benchmark in MANET routing research due to its proactive nature and loop free routing guarantees. However, frequent link failures and high mobility challenge its efficiency by increasing control overhead. This motivates further investigation into its performance under link failure scenarios, particularly focusing on routing table convergence, convergence time, and routing overhead, which form the core objectives of the present work.

Chapter 3

Methodology

The objective of this study is to evaluate the performance of the Destination-Sequenced Distance Vector (DSDV) routing protocol in mobile ad-hoc networks (MANETs) under varying network conditions. The methodology followed can be divided into several steps:

3.1 Network Setup

The network simulations were performed using a standard network simulator. The simulation environment consists of mobile nodes randomly deployed in a specified area. Each node acts as both a host and a router, forwarding packets according to the DSDV protocol.

- The number of nodes was varied to observe the impact on network performance.
- Nodes move according to the random waypoint mobility model.
- Transmission range, speed, and pause time of the nodes were kept constant to isolate the effect of network size.

3.2 Traffic Generation

- Constant Bit Rate (CBR) traffic sources were used for generating packets between randomly chosen source-destination pairs.
- Packet sizes and sending rates were fixed to ensure uniform traffic load.
- Simulation time was sufficiently long to capture steady-state behavior.

3.3 Performance Metrics

The following performance metrics were evaluated to assess the DSDV protocol:

1. **Packet Delivery Ratio (PDR):** The ratio of successfully delivered packets to the total packets sent.
2. **Average Delay:** The average end-to-end delay experienced by packets.
3. **Routing Overhead:** The total number of routing control packets transmitted during the simulation.
4. **Convergence Time:** The time taken for routing tables to stabilize after topology changes.

3.4 Simulation Procedure

1. The network topology and traffic parameters were defined.
2. DSDV routing was enabled on all nodes.
3. Simulations were run for varying number of nodes (e.g., 10, 20, 30, 40).
4. For each scenario, the performance metrics were recorded.
5. Multiple runs were performed to ensure statistical reliability, and average values were calculated.

3.5 Data Analysis

The collected data was analyzed to observe the trends of each performance metric with increasing network size. Both tabular summaries and graphical plots were generated to visualize the behavior of DSDV under different conditions.

Chapter 4

System Overview

Block Diagram

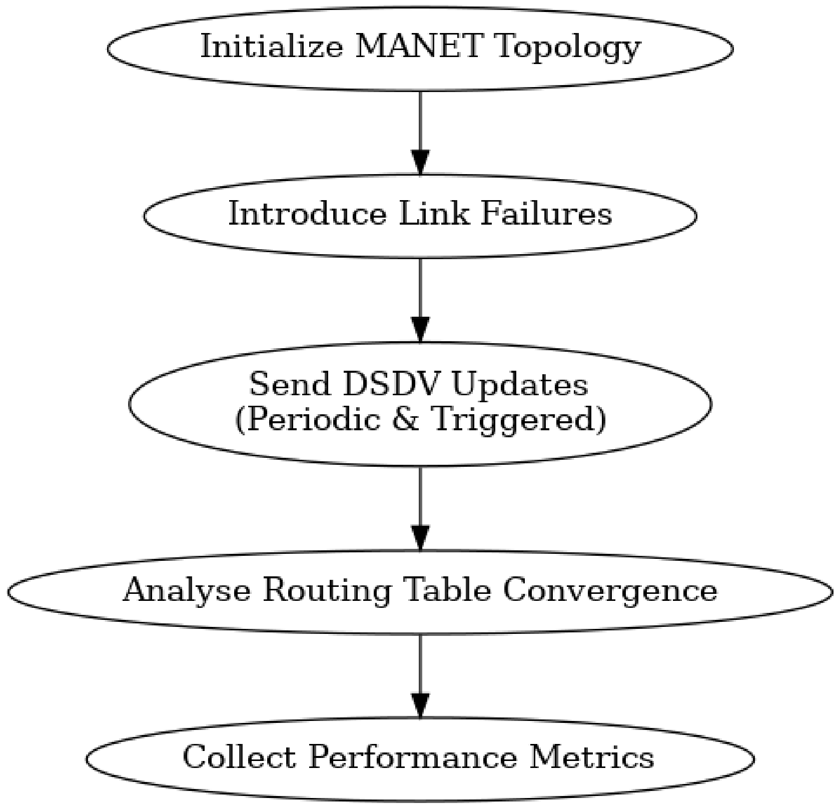
Initialize MANET Topology
Introduce Link Failures
Send DSDV Updates (Periodic & Triggered)
Analyse Routing Table Convergence
Collect Performance Metrics

Figure 4.1: Block diagram of the DSDV routing protocol simulation process.

The block diagram illustrates the step-by-step process of simulating the DSDV routing protocol to study its reaction to link failures and routing table convergence. The simulation begins with the initialization of a Mobile Ad Hoc Network (MANET) topology, where multiple mobile nodes are configured, and each node maintains a DSDV routing table with information about destinations, next hops, hop counts, and sequence numbers. Once the network stabilises under normal conditions, controlled link failures are introduced to emulate the effects of node movement or disconnection. After a link failure occurs, DSDV responds by generating triggered updates, which immediately broadcast the invalidation of affected routes using infinite metrics and incremented sequence numbers. At the same time, periodic updates continue to propagate through the network, ensuring that all nodes receive refreshed routing information. As these updates spread, each node discards stale routes and reconstructs its routing table to reflect the new topology.

The next stage is routing table convergence analysis, where the simulation monitors how long it takes for all nodes to stabilise their routing tables and achieve a consistent, loop-free view of the network. Once convergence is reached, the network resumes normal operation, but the intermediate period highlights the temporary disruptions caused by link breaks. Finally, performance metrics such as convergence time, routing overhead, packet delivery ratio, and packet loss are collected and analysed. These results demonstrate DSDV's ability to handle link failures effectively, while also exposing its limitations in terms of delayed convergence and vulnerability to malicious disruptions such as false link failures.

Chapter 5

Working Code

```
import random
import time
import networkx as nx
import matplotlib.pyplot as plt

# ------------------------------
# DSDV Node Representation
# ------------------------------
class Node:
    def __init__(self, node_id):
        self.id = node_id
        self.routing_table = {}

    def initialize_table(self, graph):
        """Initialize routing table with direct neighbors"""
        for n in graph.nodes():
            if n == self.id:
                self.routing_table[n] = (self.id, 0, 0)
            elif graph.has_edge(self.id, n):
                self.routing_table[n] = (n, 1, 0)
            else:
                self.routing_table[n] = (None, float("inf"), -1)

    def update_table(self, neighbor_table, neighbor_id):
        """DSDV Update rule"""
        updated = False
        for dest, (nh, hc, seq) in neighbor_table.items():
            if dest == self.id:
                continue
```

```python
            current = self.routing_table.get(dest, (None, float("inf"), -1))
            if seq > current[2] or (seq == current[2] and hc + 1 < current[1]):
                self.routing_table[dest] = (neighbor_id, hc + 1, seq)
                updated = True
        return updated

    def invalidate_route(self, dest):
        """Mark route invalid on link failure"""
        nh, hc, seq = self.routing_table.get(dest, (None, float("inf"), -1))
        self.routing_table[dest] = (None, float("inf"), seq + 1)


# ------------------------------
# DSDV Simulation
# ------------------------------
class DSDVSimulation:
    def __init__(self, num_nodes=5):
        self.graph = nx.Graph()
        self.nodes = {i: Node(i) for i in range(num_nodes)}
        self.graph.add_nodes_from(self.nodes.keys())
        self.control_messages = 0

        # Randomly connect nodes
        for i in range(num_nodes):
            for j in range(i + 1, num_nodes):
                if random.random() < 0.6:
                    self.graph.add_edge(i, j)

        # Initialize routing tables
        for node in self.nodes.values():
            node.initialize_table(self.graph)

    def broadcast_updates(self):
        converged = False
        while not converged:
            converged = True
            for i, node in self.nodes.items():
                for neighbor in list(self.graph.neighbors(i)):
                    updated = self.nodes[neighbor].update_table(node.routing_table, i)
                    self.control_messages += 1
                    if updated:
                        converged = False

    def introduce_link_failure(self, u, v):
        if self.graph.has_edge(u, v):
            self.graph.remove_edge(u, v)
            self.nodes[u].invalidate_route(v)
            self.nodes[v].invalidate_route(u)

    def simulate_packet_delivery(self, num_packets=30):
        delivered = 0
        total_delay = 0
        for _ in range(num_packets):
            src, dst = random.sample(list(self.nodes.keys()), 2)
            route = self.get_route(src, dst)
            if route:
                delivered += 1
                total_delay += len(route)
        pdr = delivered / num_packets
        avg_delay = total_delay / delivered if delivered > 0 else float("inf")
        return pdr, avg_delay

    def get_route(self, src, dst):
        path = [src]
        current = src
        visited = set()
        while current != dst and current not in visited:
            visited.add(current)
            next_hop, hc, seq = self.nodes[current].routing_table.get(dst, (None, float("inf"), -1))
            if next_hop is None or hc == float("inf"):
                return None
            path.append(next_hop)
            current = next_hop
        return path if current == dst else None

    def run(self, fail_link=(0, 1)):
        # Initial convergence
        self.broadcast_updates()
        pdr_before, delay_before = self.simulate_packet_delivery()

        # Introduce failure
        start = time.time()
        self.introduce_link_failure(*fail_link)
        self.broadcast_updates()
        end = time.time()

        convergence_time = end - start
        pdr_after, delay_after = self.simulate_packet_delivery()

        return {
            "pdr": pdr_after,
            "delay": delay_after,
            "overhead": self.control_messages,
            "convergence": convergence_time,
        }


# -------------------------------
# Run Multiple Experiments & Plot
# -------------------------------
if __name__ == "__main__":
    node_counts = range(4, 11)  # vary nodes from 4 to 10
    results = {"nodes": [], "pdr": [], "delay": [], "overhead": [], "
        convergence": []}

    for n in node_counts:
        sim = DSDVSimulation(num_nodes=n)
        metrics = sim.run()
        results["nodes"].append(n)
        results["pdr"].append(metrics["pdr"])
        results["delay"].append(metrics["delay"])
        results["overhead"].append(metrics["overhead"])
        results["convergence"].append(metrics["convergence"])
        print(f"Nodes={n}: {metrics}")

    # Plot graphs
    plt.figure()
    plt.plot(results["nodes"], results["pdr"], marker="o")
    plt.xlabel("Number of Nodes")
    plt.ylabel("Packet Delivery Ratio (PDR)")
    plt.title("PDR vs Number of Nodes")
    plt.savefig("PDR_vs_Nodes.png")
    plt.grid(True)
    plt.show()

    plt.figure()
    plt.plot(results["nodes"], results["delay"], marker="o", color="orange"
        )
    plt.xlabel("Number of Nodes")
    plt.ylabel("Average End-to-End Delay")
    plt.title("Delay vs Number of Nodes")
    plt.savefig("Delay_vs_Nodes.png")
    plt.grid(True)
    plt.show()

    plt.figure()
    plt.plot(results["nodes"], results["overhead"], marker="o", color="red"
        )
    plt.xlabel("Number of Nodes")
    plt.ylabel("Routing Overhead (Control Messages)")
    plt.title("Overhead vs Number of Nodes")
    plt.savefig("Overhead_vs_Nodes.png")
    plt.grid(True)
    plt.show()

    plt.figure()
    plt.plot(results["nodes"], results["convergence"], marker="o", color="
        green")
    plt.xlabel("Number of Nodes")
    plt.ylabel("Convergence Time (seconds)")
    plt.title("Convergence Time vs Number of Nodes")
    plt.savefig("Convergence_vs_Nodes.png")
    plt.grid(True)
    plt.show()
```

Listing 5.1: DSDV Simulation Code

Chapter 6

Results and Discussion

This chapter presents the performance evaluation of the Destination Sequenced Distance Vector (DSDV) routing protocol using a Python based simulation. The number of nodes in the network was varied from 4 to 21, and the performance of DSDV was analyzed in terms of the following metrics:

1. Packet Delivery Ratio (PDR)
2. Average End-to-End Delay
3. Routing Overhead
4. Convergence Time

The purpose of this analysis is to study how DSDV behaves under different network sizes and to understand its strengths and limitations as a proactive routing protocol.

Packet Delivery Ratio (PDR) vs Number of Nodes

The **Packet Delivery Ratio (PDR)** is defined as the ratio of successfully delivered data packets to the total number of packets sent in the network. It indicates the reliability and efficiency of the routing protocol. Mathematically, it is expressed as:

$$PDR = \frac{\text{Packets Delivered}}{\text{Packets Sent}}$$

In the initial stages with fewer nodes, the PDR is generally high because the network topology is simple, and stable routes can be easily maintained. As the number of nodes increases, the PDR tends to fluctuate due to increased routing overhead, link failures, and random topologies affecting route stability. Although variations occur, the DSDV protocol maintains a reasonably high packet delivery ratio overall. This shows that DSDV can effectively deliver packets in moderately sized networks, even in the presence of link changes. DSDV ensures reliable packet delivery for small and medium-sized networks. The minor fluctuations in PDR are mainly due to random connectivity and mobility effects during simulation.

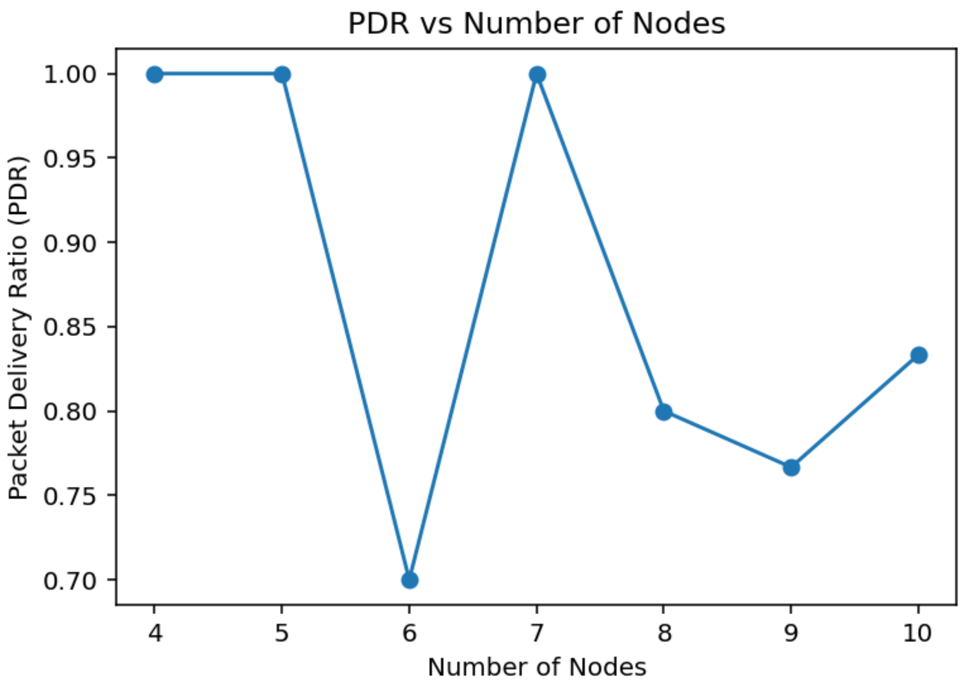

Figure 6.1: PDR vs Nodes

Average End-to-End Delay vs Number of Nodes

The **Average End-to-End Delay** represents the average time taken by a data packet to travel from the source node to the destination node. It includes all possible delays such as transmission, propagation, and queuing delays. The average delay is computed using:

$$\text{Average Delay} = \frac{\text{Total Transmission Time}}{\text{Packets Delivered}}$$

For smaller network sizes, the delay is low because routes are shorter and congestion is minimal. As the network grows in size, the average delay increases due to longer routes, higher hop counts, and frequent topology updates. Occasional variations may be observed depending on the random connectivity among nodes. The delay increases with the number of nodes, indicating that DSDV is better suited for small or moderately sized networks where the routing tables can converge quickly and paths remain stable.

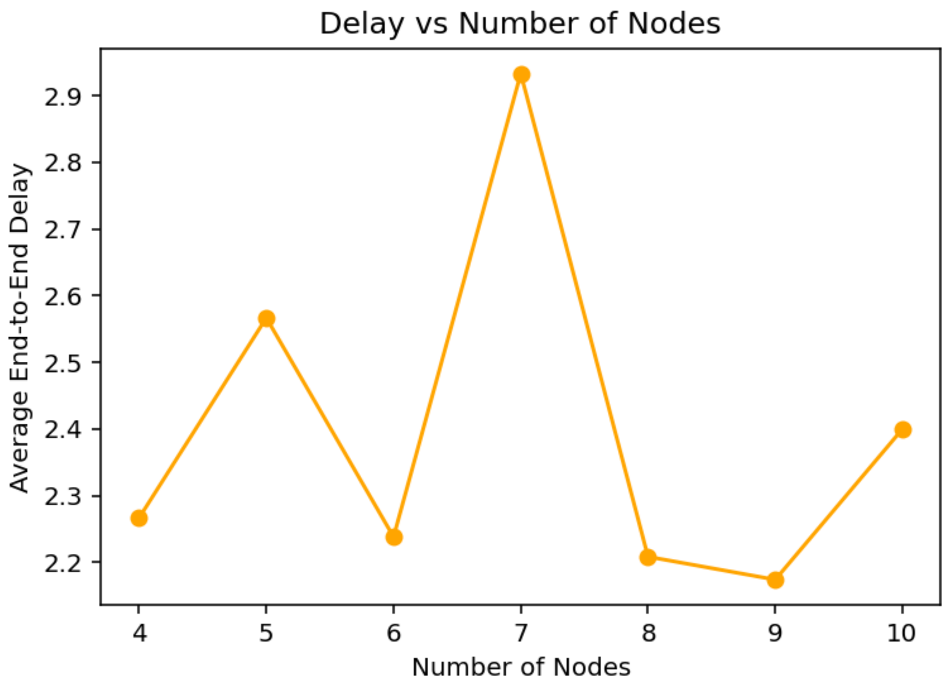

Figure 6.2: Delay vs Nodes

Routing Overhead vs Number of Nodes

Routing overhead measures the total number of control messages exchanged among nodes for maintaining and updating routing tables. In DSDV, since every node periodically broadcasts its entire routing table to its neighbors, the control overhead increases significantly with the number of nodes.

As the network size increases, more control packets are generated, leading to higher bandwidth consumption and potential congestion. This is one of the major drawbacks of proactive routing protocols like DSDV, as updates occur even when no data packets are being transmitted. The simulation results show a steady increase in routing overhead as the number of nodes increases. This confirms that DSDV trades bandwidth efficiency for route availability, making it less suitable for large or highly dynamic networks.

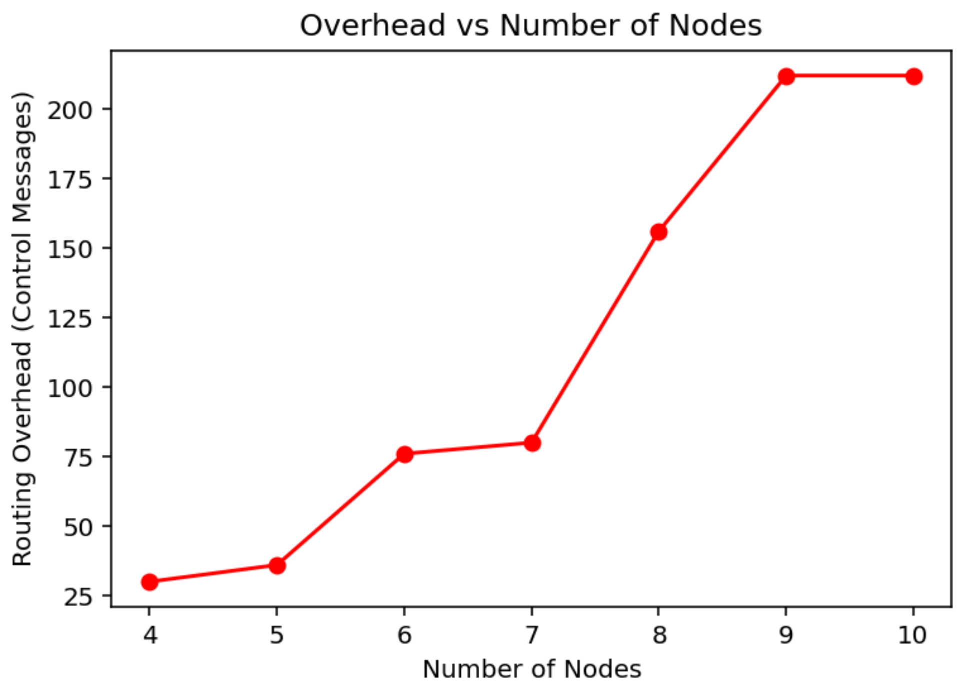

Figure 6.3: Overhead vs Nodes

Convergence Time vs Number of Nodes

Convergence time refers to the time required for the network to stabilize after a topology change, such as a link failure. During this period, nodes exchange routing updates until all routing tables are consistent.

For smaller networks, convergence time is very low since updates spread quickly among a few nodes. As the network size increases, the convergence time generally increases because more updates are required for all nodes to synchronize their routing information. However, the graph may remain flat for some node counts, especially when random topologies result in similar update propagation times.

Inference: DSDV achieves fast convergence for small and medium networks, but the time taken increases gradually with network size. This behavior highlights the scalability limitation of DSDV in large and dynamic environments.

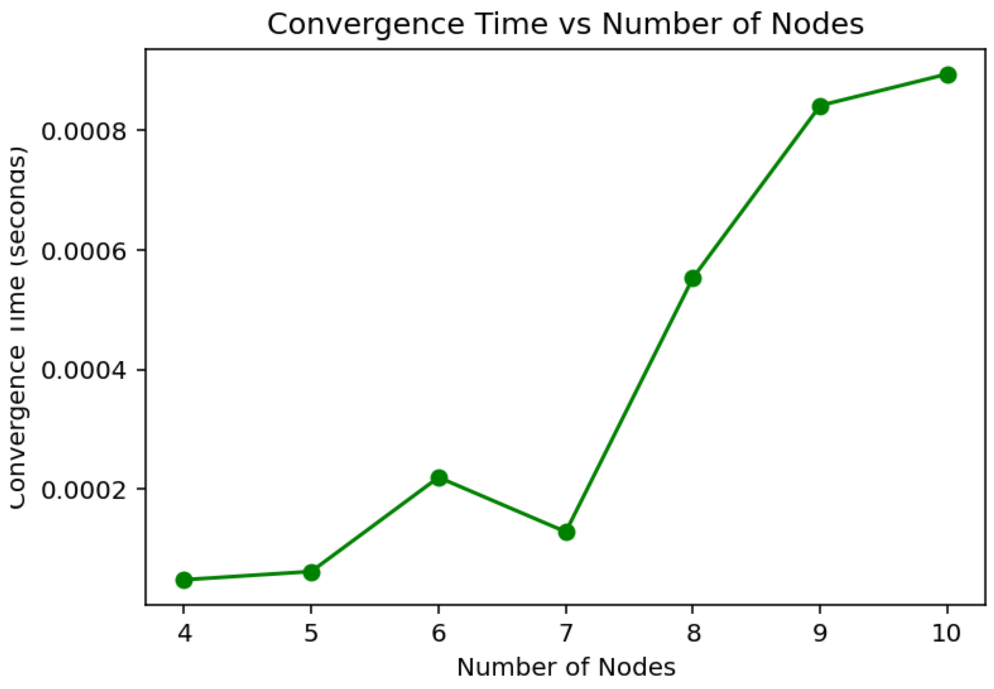

Figure 6.4: Convergence vs Nodes

Discussion

Table 6.1 summarizes the performance behavior of DSDV with respect to the number of nodes and different performance metrics.

Metric	**Trend with Increasing Nodes**	**Observation**
Packet Delivery Ratio (PDR)	Slightly decreases / fluctuates	Occasional packet drops due to link changes
Average Delay	Increases	More hops and congestion in larger topologies
Routing Overhead	Increases sharply	Frequent updates and broadcasts
Convergence Time	Gradually increases	More updates required for synchronization

Table 6.1: Performance summary of DSDV protocol

The simulation results demonstrate that the DSDV routing protocol performs effectively in small and medium-sized ad hoc networks. It provides high packet delivery ratios, low delay, and quick convergence. However, as the network grows larger, routing overhead and convergence time increase due to the protocol's proactive nature.

Hence, DSDV is well-suited for moderately sized, stable networks where route freshness and reliability are more critical than bandwidth efficiency.

Chapter 7

Conclusion

This project focused on the simulation and performance analysis of the Destination Sequenced Distance Vector (DSDV) routing protocol in a Mobile Ad-hoc Network (MANET) environment. The implementation utilized Python programming within the Spyder IDE, leveraging libraries such as NetworkX and Matplotlib to model the network topology and visualize the results. The objective was to evaluate DSDV's performance in terms of key metrics such as Packet Delivery Ratio (PDR), end-to-end delay, routing overhead, and convergence time, particularly under scenarios involving topology changes and link failures. The simulation results demonstrated that DSDV, being a proactive routing protocol, maintains up-to-date routing information for all nodes, thereby ensuring quick route availability. This proactive approach results in low latency in packet delivery for relatively stable networks. However, the simulation also revealed that frequent topology changes, such as those caused by node mobility or link failures, significantly impact the routing overhead and convergence time. This is due to the periodic updates required to maintain routing tables, which consume network resources and may increase delays under high mobility scenarios. Through this project, the strengths and limitations of DSDV were clearly identified. While DSDV offers reliable and loop-free routes in static or low-mobility environments, it is less efficient for highly dynamic networks due to its constant control message exchanges. The simulation has provided valuable insights into how routing performance is affected by network parameters such as node density and mobility patterns. These findings are useful for researchers and engineers designing MANET systems for real-time applications such as disaster recovery, vehicular networks, and military communications.

In conclusion, the project successfully demonstrated the viability of DSDV as a routing protocol for certain MANET environments while highlighting areas for improvement. Future work could involve comparing DSDV with other routing protocols such as AODV and DSR under similar conditions, implementing hybrid routing approaches, and exploring optimization techniques to reduce routing overhead while improving adaptability to frequent topology changes. Such enhancements would make DSDV and similar proactive protocols more robust for a wider range of dynamic network scenarios.

References

[1] M. G. K. Alabdullah, B. M. Atiyah, K. S. Khalaf, and S. H. Yadgar, "Analysis and simulation of three manet routing protocols: A research on aodv, dsr & dsdv characteristics and their performance evaluation," *Periodicals of Engineering and Natural Sciences (PEN)*, vol. 7, no. 3, pp. 1228–1238, 2019.

[2] A. Dwivedi, S. Kushwaha, and O. Vyas, "Performance of routing protocols for mobile adhoc and wireless sensor networks: A comparative study," *International Journal of Recent Trends in Engineering*, vol. 2, no. 4, p. 101, 2009.

[3] A. P. Patil, N. Sambaturu, and K. Chunhaviriyakul, "Convergence time evaluation of algorithms in manets," *arXiv preprint arXiv:0910.1475*, 2009.

[4] K. U. R. Khan, R. U. Zaman, A. V. Reddy, K. A. Reddy, and T. S. Harsha, "An efficient dsdv routing protocol for wireless mobile ad hoc networks and its performance comparison," in *2008 Second UKSIM European Symposium on Computer Modeling and Simulation*. IEEE, 2008, pp. 506–511.

[5] J. Lu, B. Zhang, G. Han, J. Wang, and W. Dou, "A new improvement on dsdv," in *2011 7th International Conference on Wireless Communications, Networking and Mobile Computing*. IEEE, 2011, pp. 1–4.

[6] M. Naseem and C. Kumar, "Edsdv: Efficient dsdv routing protocol for manet," in *2013 IEEE International Conference on Computational Intelligence and Computing Research*. IEEE, 2013, pp. 1–4.

[7] R. Singh and N. Singh, "Performance assessment of dsdv and aodv routing protocols in mobile adhoc networks with focus on node density and routing overhead," in *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*. IEEE, 2020, pp. 298–303.

[8] Y. Feng, W. Liu, and S. Hu, “Evaluating topology based routing protocols with different network node speeds,” in *2024 IEEE 24th International Conference on Communication Technology (ICCT)*. IEEE, 2024, pp. 39–43.

[9] M. A. Taha, A. H. Al-fatlawi, and H. A. Rasool, “Performance evaluation of aodv, dsr and dsdv routing protocols in manet networks,” *Journal of University of Babylon for Pure and Applied Sciences*, vol. 26, no. 4, pp. 308–318, 2018.

[10] S. Vemuri and S. Mirkar, “A performance comparison of manet routing protocols,” in *2021 Innovations in Power and Advanced Computing Technologies (i-PACT)*. IEEE, 2021, pp. 1–5.