Volume: 15 Issue: 02 Pages: 5837– 5852 (2023) ISSN: 0975-0290

INTSM: A Novel Approach for Load Balancing in MANET Route Discovery

Manju Sahu[1]
Research Scholar, Barkatullah University, Bhopal
Email : manjusahu87@gmail.com
Dr. Sanjeev Gour[2]
Associate Professor, Department of Computer Science, Career College, Bhopal
Email : sunj129@gmail.com

ABSTRACT

Ad-hoc networks are networks where mobile nodes can move around, causing changes in their positions and how they connect with each other. However, these changes can lead to problems in communication between the nodes. One big problem is making sure that the network is balanced, so that no nodes are overloaded while others are underutilized. When mobile nodes move less, the network performs better, but this can cause delays for nodes in the centre. To manage this issue, we need to find a way to handle the network's load and balance the traffic between nodes. We also need to identify when nodes are congested, meaning they have too much to handle, and when nodes are not being used enough. This way, we can distribute the traffic in a better way. This paper introduces a new method called the Intermediate Node Traffic Sharing Model (INTSM) to help solve these problems. INTSM focuses on balancing the load and managing congestion during the process of finding routes in the network.
By using INTSM, we can improve how traffic is shared and how the network performs, reducing delays for packets. This research aims to make load balancing and route discovery in these networks better and more efficient.

Keywords - Ad-hoc networks, Intermediate Node Traffic Sharing Model (INTSM), congestion management, load balancing, route discovery, traffic distribution.

Date of Submission: July 15, 2023 Date of Acceptance: August 21, 2023

1. INTRODUCTION

A Mobile Ad-hoc Network (MANET)[1] is a type of network where mobile devices can create a temporary network without any pre-existing infrastructure or human intervention. In a MANET, each device acts as both a sender and a receiver, helping to pass data packets to their intended destination. This collaboration between devices allows the network to function without relying on fixed routers or base stations.

There are two ways for wireless devices to communicate with each other. The first way involves a central base station that controls the communication and manages the resources. If two devices want to communicate, they have to go through the base station. However, this approach is mostly used in large cellular networks like GSM or UMTS. Fig. 1 is shown the infrastructure-based networks[2].

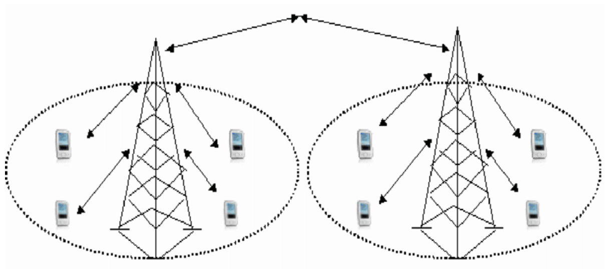


Figure 1 Infrastructure based network

The second way, called the ad-hoc approach, is different. It doesn't depend on any fixed infrastructure or central control. Instead, devices work together to forward data packets from one device to another until they reach the desired destination[3]. Each device acts as a router, deciding where to send the packets based on the current state of the network. Fig. 2 is shown the infrastructure less networks.

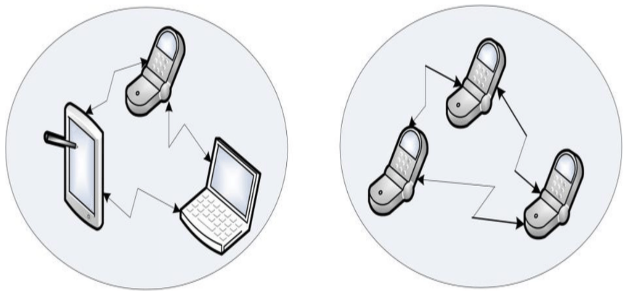


Figure 2 Infrastructure less networks

The paper introduces INTSM, a new method to improve load balancing during route discovery in MANETs. It effectively manages congestion and controls load distribution when setting up routes in the network.

By using INTSM, we aim to enhance how traffic is shared among devices and improve the overall performance of the network. This research contributes to making route discovery in MANETs more efficient and reliable.

MANETs have various applications, such as emergency search-and-rescue operations, battlefield decision making, and data acquisition in challenging environments[4]. They are characterized by dynamic topology, multi-hop communication, limited resources (bandwidth, CPU, battery), and limited security. These unique characteristics pose challenges in designing routing protocols for MANETs. One of the key objectives of MANET routing protocols is to maximize energy efficiency since the nodes in a MANET rely on limited energy resources.

The main goals of MANET routing protocols are to maximize network throughput, energy efficiency, network lifetime, and minimize delay[5]. Network throughput is typically measured using the packet delivery ratio, while Routing Overhead is mainly determined by the routing overhead, which refers to the number or size of routing control packets.

In hop-by-hop reactive routing protocols like Ad-hoc On-demand Distance Vector (AODV), each intermediate node makes decisions about where to forward the routed packet[6]. In mobile ad hoc networks (MANETs), finding paths involves broadcasting route requests, but this can lead to inefficiencies due to redundant packets. Approaches like constrained broadcasting and load-based dropping of route requests have been proposed to address this. The paper introduces a new method called Intermediate Node Traffic Sharing Model (INTSM) to improve load balancing in route discovery. INTSM considers congestion and load control to optimize traffic distribution and enhance MANET routing efficiency.

1.1 Routing Protocols

An ad hoc routing protocol guides nodes in mobile networks to choose paths for packet routing. Two main types exist: topology-based and position-based. More about unicast routing protocols for MANETs is discussed in our previous research. [5].

For the purpose of simulation in this paper, we have chosen two on-demand routing protocols: Ad-hoc On-demand Distance Vector (AODV) and Dynamic Source Routing (DSR)[7]. These protocols are extensively used in MANETs and have been well-researched previously. We've chosen these protocols to assess how our proposed INTSM performs against existing routing protocols in various traffic scenarios.

1.1.1 AODV

The Ad Hoc On-Demand Distance Vector (AODV) routing protocol [8] is reactive, triggered when a node wants to send data packets. It supports both single and multiple destination packets. AODV distinguishes itself with a unique destination sequence number (DestSeqNum) for each destination. It maintains a route table with entries per destination, removing unused routes over time. Routes are set up via request (RREQ) and reply (RREP) messages, and failures prompt reports and new requests.

1.1.2 DSR

The Dynamic Source Routing (DSR) [9] protocol utilizes source routing and maintains active routes. It has two phases: route discovery and route maintenance. DSR doesn't regularly send routing messages like AODV; instead, it sends error messages if link issues arise. In DSR, packet headers include a list of intermediate node IDs, enabling multiple paths to the destination. A key distinction between AODV and DSR lies in their packet contents. AODV packets only have the destination address, whereas DSR packets carry all routing info, resulting in higher routing overhead than AODV.

1.2 Connection Types

There are different ways that devices in a MANET can connect with each other[10]. Let's look at two common types:

1.2.1 Constant Bit Rate (CBR)

Constant bit rate (CBR) means that the network receives a steady flow of data at a consistent rate. In CBR, data packets are sent with a fixed size and a fixed time interval between each packet. No special setup is needed for the connection between devices, and the receiving device doesn't send any acknowledgment messages. The data flows in one direction, from the source to the destination.

1.2.2 Transmission Control Protocol (TCP)

TCP is a reliable and connection-oriented transport protocol. It ensures that data is transmitted reliably by using acknowledgments, timeouts, and retransmissions. When data packets are sent, the receiving device sends acknowledgments back to confirm successful delivery. If an acknowledgment is not received within a certain time period (called a timeout), TCP resends the data to make sure it reaches its destination reliably.

1.3 Properties of an Effective Routing Protocol

A good routing protocol for ad hoc networks using multiple descriptions coding should have certain important properties[11]. Here are some key properties explained in simple terms:

- **Multiple Paths:** Provide multiple routes to destinations for redundancy and backup options.
- **Loop-Free Paths:** Ensure routes are loop-free to prevent packet congestion.
- **Non-Overlapping Paths:** Offer paths that do not overlap to maintain connectivity if one path fails.
- **Multipath Usage:** Allow simultaneous use of multiple paths to balance network load and enhance performance.
- **Complete Route Knowledge:** Source node must know all available routes for effective data transmission.
- **QoS Metrics:** Provide information on QoS metrics (bandwidth, delay, cost) for route selection.
- **Network Scalability:** Support networks of 50-200 nodes moving at pedestrian speeds.
- **Decentralized Clocks:** Nodes function without relying on a common clock for synchronization.

- **Practical Implementation:** Availability of a basic routing protocol implementation for practical use and experimentation.

Having these properties in a routing protocol ensures its effectiveness in supporting multiple descriptions coding in ad hoc networks.

1.4 Classification of Load Balancing Protocols in MANET

Load balancing protocols[12] in ad hoc networks are designed to handle different types of traffic situations by efficiently managing the distribution of workload and optimizing route discovery. When we talk about load balancing, we can categorize the load into various types[13]:

- **Channel Load:** The traffic and activity on the communication channel, which multiple nodes contend for. Load balancing maintains fair and efficient channel resource use.
- **Nodal Load:** Reflects a node's busyness in terms of tasks, computations, etc. Load balancing aims to distribute workload evenly, preventing node overload.
- **Neighbouring Load:** Represents load due to communication activities between nearby nodes, including control and data packet exchange. Load balancing ensures even distribution of communication tasks among nearby nodes to avoid congestion.

By considering these different types of load, load balancing protocols in MANETs strive to achieve efficient use of network resources, improve overall network performance, and support scalability. They dynamically adapt to changing traffic conditions, ensuring reliable and optimal routing in ad hoc networks.

1.5 Load Balancing Required Metrics

To achieve balanced and efficient load distribution in MANETs, load balancing routing protocols consider various metrics. These metrics, as described in [14], are important factors in determining how the network load is managed. Here are the metrics used:

- **Active Path:** Measures the number of active routing paths a node supports. More paths indicate higher data traffic and busyness.
- **Traffic Size:** Quantifies data traffic at a node and its neighbours, indicating workload and capacity requirements.
- **Packets in Interface Queue:** Counts packets in a node's incoming and outgoing interfaces, reflecting interface congestion and packet handling efficiency.
- **Channel Access Probability:** Represents the likelihood of successful wireless channel access, indicating contention among nearby nodes.
- **Node Delay:** Captures packet delays due to queuing, processing, and transmission, helping assess overall network delay.

Load balancing protocols[15] can be classified into three types based on their techniques shown in figure 3:

- **Delay-based:** These protocols aim to balance the load by avoiding nodes with high link delays. They prioritize routes with lower delays to improve overall performance.
- **Traffic-based:** These protocols distribute the traffic load evenly among network nodes. They ensure that no single node is overloaded with excessive traffic, leading to better load balancing.
- **Hybrid-based:** These protocols combine features from both traffic-based and delay-based techniques. They aim to achieve a balanced load distribution by considering both traffic load and delay factors.

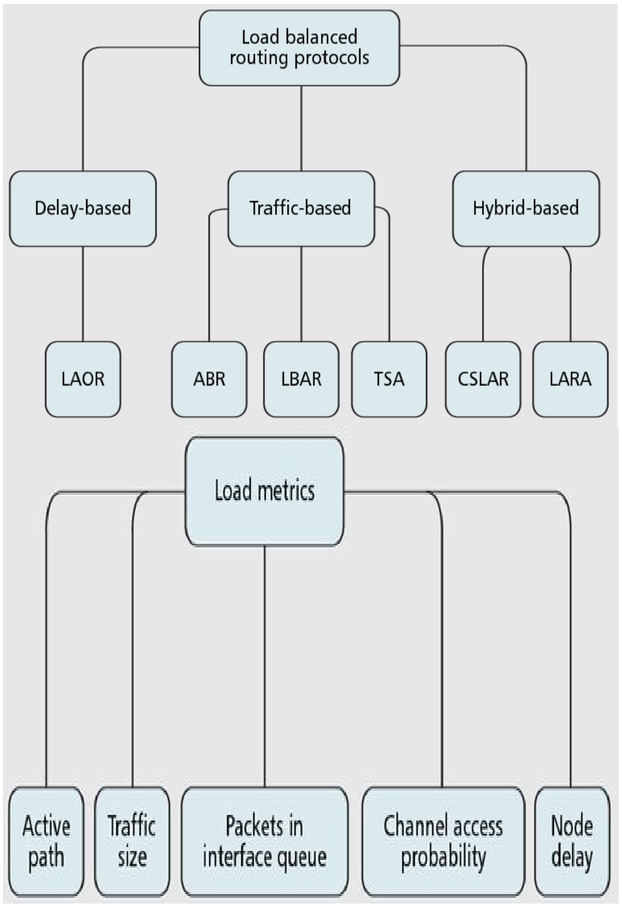


Figure 3 Classification of load balanced routing protocol with load metrics

By considering this load balancing metrics and employing appropriate techniques, routing protocols in MANETs can effectively manage the network's load, optimize resource utilization, and improve overall performance and fairness.

2. Literature Survey

The paper[16] proposes a method to address load balancing challenges in ad hoc networks. It introduces a mechanism or algorithm that dynamically distributes network traffic among available nodes to ensure balanced resource utilization and optimize network performance. The method aims to minimize congestion, improve overall network throughput, and enhance the Quality of Service (QoS). The paper likely includes evaluations comparing the proposed method to existing load balancing techniques, demonstrating its effectiveness in achieving load balancing and improving network efficiency.

The paper[17] provides a survey on two main topics: energy-saving load balancing approaches to enhance the AOMDV routing protocol in MANETs and data security in MANETs. The survey covers various energy-saving load balancing techniques and their advantages and limitations. It also explores data security mechanisms and protocols designed for MANETs, discussing their effectiveness in protecting data confidentiality, integrity, and availability. The paper serves as a valuable resource for researchers and practitioners interested in energy-efficient load balancing, AOMDV routing enhancements, and data security in MANETs.

The paper[18] is a routing protocol that focuses on evenly distributing network traffic among nodes in order to optimize resource utilization and improve network performance. It makes routing decisions based on the load or capacity of individual nodes, favoring less congested nodes for data transmission. The protocol aims to prevent congestion, reduce packet loss, minimize delays, and enhance overall Quality of Service (QoS) in MANETs. It contributes to load balancing, improving network performance, and extending the network's lifetime.

The paper[19] explores the concept of load balancing in shortest-path routing protocols for Mobile Ad hoc Networks (MANETs). It discusses techniques such as multipath routing, load-aware routing metrics, and proactive load balancing to achieve load balancing in MANETs. The paper highlights the importance of evenly distributing network traffic and optimizing resource utilization. It suggests that incorporating these load balancing techniques into shortest-path routing protocols can enhance traffic distribution, reduce congestion, and improve overall network performance in MANETs.

The paper[20] introduces a load balancing approach for Mobile Ad hoc Networks (MANETs) called the "Fibonacci sequence based multipath load balancing approach." It utilizes the Fibonacci sequence to determine the number of paths and traffic distribution in the network. By using multiple paths and balancing traffic based on the Fibonacci sequence, the approach aims to improve resource utilization, mitigate congestion, and enhance overall network performance in MANETs. Evaluations comparing the approach to other methods are likely included.

The paper[13] provides a comprehensive overview of energy efficiency and load balancing techniques in Mobile Ad hoc Networks (MANETs). The survey covers topics such as energy-aware routing, sleep scheduling, and energy harvesting, as well as load balancing algorithms and adaptive routing protocols. It evaluates the effectiveness of these techniques in terms of Routing Overhead, network lifetime, throughput, and fairness. The survey serves as a valuable resource for researchers and practitioners interested in improving energy efficiency and achieving load balancing in MANETs.

The paper[21] provides a comprehensive overview of load balancing routing protocols in Mobile Ad hoc Networks (MANETs). It reviews various load balancing mechanisms and strategies proposed in the literature and discusses their advantages, limitations, and performance evaluations. The review aims to serve as a valuable resource for researchers and practitioners in understanding the current state of load balancing techniques in MANETs and identifying future research directions.

The paper[22] focuses on load balancing and congestion control techniques in Mobile Ad hoc Networks (MANETs). It reviews various mechanisms proposed in the literature and discusses their effectiveness in improving network performance. The paper emphasizes the importance of load balancing for efficient resource utilization and examines the interplay between load balancing and congestion control. It provides insights into the evaluation and performance analysis of these techniques, considering metrics such as throughput, delay, packet loss, fairness, and Routing Overhead. The paper serves as a valuable resource for researchers and practitioners seeking to address congestion issues and enhance network performance in MANETs.

The paper[23] focuses on the topics of load balancing and congestion control in Mobile Ad hoc Networks (MANETs). It provides a review of various mechanisms proposed in the literature and evaluates their effectiveness in improving network performance. The paper emphasizes the significance of load balancing for efficient utilization of network resources and explores the relationship between load balancing and congestion control. It offers insights into the evaluation and analysis of these techniques using metrics such as throughput, delay, packet loss, fairness, and Routing Overhead. Overall, the paper serves as a valuable resource for researchers and practitioners who are interested in addressing congestion issues and enhancing network performance in MANETs.

The paper[24] presents a routing protocol that focuses on load balancing and predicting link breaks in MANETs. The protocol aims to evenly distribute traffic and anticipate potential link failures to improve reliability and performance. It incorporates load balancing mechanisms and link break prediction techniques to optimize resource utilization and enhance network robustness. The paper likely includes evaluations and compares the protocol's performance with existing routing protocols. Overall, it offers a solution for efficient and reliable data transmission in MANETs through load balancing and link break prediction.

The paper[25] introduces LAPU, a load balancing technique for geographic routing in MANETs. LAPU utilizes adaptive position updates to dynamically adjust the frequency of position updates based on the network's load. This helps balance the traffic load, reduce control overhead, and improve routing efficiency. The paper likely includes performance evaluations comparing LAPU with other routing protocols, highlighting its advantages in terms of throughput, delay, packet loss, and control overhead. LAPU aims to optimize geographic routing in MANETs by improving load distribution and resource utilization.

The paper[26] proposes a load-balancing routing protocol for ad-hoc networks that combines cross-layer design and ant-colony optimization. The protocol gathers information from different network layers to make informed routing decisions and uses ant-colony optimization to find efficient routes. It aims to balance network traffic, improve performance, and minimize congestion. The paper likely includes performance evaluations and comparisons with other protocols to demonstrate the effectiveness of the proposed approach. Overall, the protocol offers an efficient load-balancing solution for ad-hoc networks through cross-layer design and ant-colony optimization.

The paper[27] explores how load balancing techniques can improve the energy efficiency of Mobile Ad hoc Network (MANET) routing protocols. It reviews different load balancing mechanisms and analyzes their impact on Routing Overhead. The paper evaluates the trade-off between load balancing and energy usage and discusses challenges and potential optimizations. The goal is to optimize Routing Overhead by evenly distributing traffic and improving overall network efficiency.

The paper[28] investigates topology control techniques in the NS-2 network simulator for ad-hoc wireless networks. It explores different mechanisms for managing network connectivity and optimizing performance. The paper analyzes the impact of these techniques on metrics like network connectivity, coverage, energy efficiency, throughput, and latency. It discusses implementation details, limitations, and potential enhancements. The research is conducted through simulations in NS-2 to evaluate the performance of the topology control mechanisms. Overall, the paper aims to understand and analyze topology control in NS-2 for ad-hoc wireless networks.

The paper [29] presents a framework implemented in the NS-2 network simulator for topology control in wireless ad-hoc networks. The framework includes tools, modules, and algorithms for node placement, power control, and link scheduling. It discusses the design and implementation of the framework and evaluates its performance using NS-2 simulations. The paper aims to enhance network performance by controlling the network's topology through the framework.

3. Problem Statement

The network's topology is influenced by node transmitting power, which directly impacts its load. A dense topology offers more routing options but consumes higher power, while a sparse one reduces routing choices, potentially leading to node overload and increased end-to-end hop-count. Our goal is a balanced topology using Intermediate Node Traffic Sharing Model (INTSM), meeting user needs and minimizing Routing Overhead.

Nodes engage in data forwarding based on neighbor locations, using both inter and intra route discovery mechanisms. Existing load balancing methods rely on route requests reaching a single destination, dropping them from congested nodes. However, these approaches face issues including:

- Imbalanced distribution during periods of heavy traffic, where certain nodes experience higher frequency than others.
- Degradation in delivery ratio, relative overhead, and end-to-end delay in high connection scenarios.
- Intermediate nodes mistakenly considering temporary transmission issues as actual link breakages, resulting in frequent route breakage notifications to the source node, burdening the network and decreasing performance and throughput.
- Collisions and hidden node problems due to the use of an unshared load medium.
- Lack of independent load balancing, with congestion primarily occurring at intermediate nodes.
- Inadequate achievement of route stability.

These problems highlight the need for an improved load balancing approach that addresses these challenges and ensures a more efficient and reliable network performance.

3.1 Proposed Solution

To solve the problems mentioned earlier, we have come up with a new method called the Intermediate Node Traffic Sharing Model (INTSM) for balancing the load during route discovery in mobile ad hoc networks (MANETs). Our approach focuses on identifying delays in accessing the network and managing congestion at the Media Access Control (MAC) layer[30]. We also aim to save energy in routing by setting a threshold value that helps in making efficient routing decisions.

INTSM works by selecting paths with the least amount of load initially. We consider factors like the number of packets delivered by each node and their flag status. Nodes periodically send hello messages to their nearby nodes to exchange information about their load. This helps us identify which nodes are underutilized and which nodes are handling too much traffic.

With INTSM, load balancing happens in the following steps:

- Each node keeps track of its nearby nodes, their load, and flag status.
- Nodes send hello messages to their nearby nodes.
- Multiple paths are created using flooding (spreading information to all nodes).
- When a route request message is broadcasted by a source node, it includes the load information of the intermediate nodes it has passed through.

- Nodes receiving the hello message record the load information and update their routing table accordingly.
- Nodes classify the flow of traffic and identify nodes that are overloaded or underutilized.
- Traffic diversion (load sharing) is done based on the flow classification.
- The destination node selects the best path using a route discovery mechanism.
- The network is monitored to ensure collisions are minimized.

We will design distinct algorithms to compute load at various nodes, including intermediates, aiding routing choices based on load. These algorithms – for source, intermediate, and destination nodes – will compute load and determine routing.

To illustrate, consider a network model depicted in Figure 4. Here, when a source node intends to send data packets to a destination, it broadcasts a route request message. This message reaches nodes within transmission range, which then forward it to their neighbors.
This process continues until the destination node receives the route request. The destination node calculates the load of the path and sends a route reply back to the source node. Intermediate nodes set their flag status and forward the reply. Once the source node receives the route reply, it knows the route has been established and can start transmitting data.

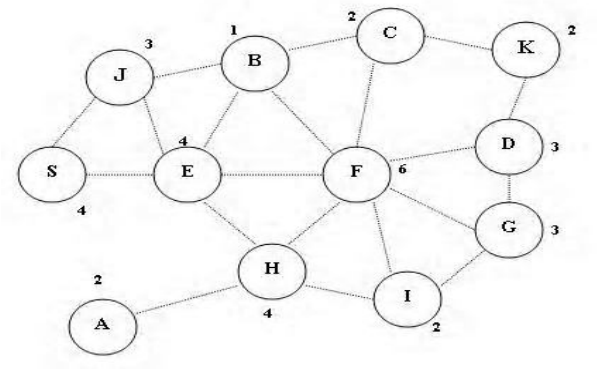


Figure 4 Network model

In Figure 5, we show an example of the route construction process, and in Figure 6, we display the routing table status at different nodes during data transmission.

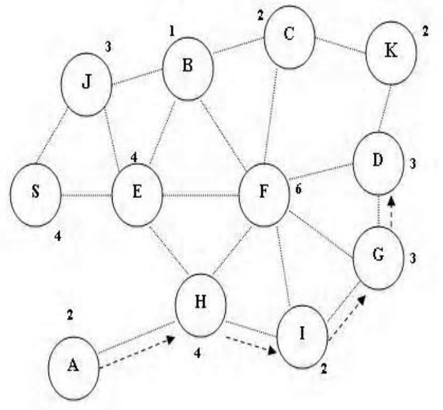


Figure 5 Illustration of route construction

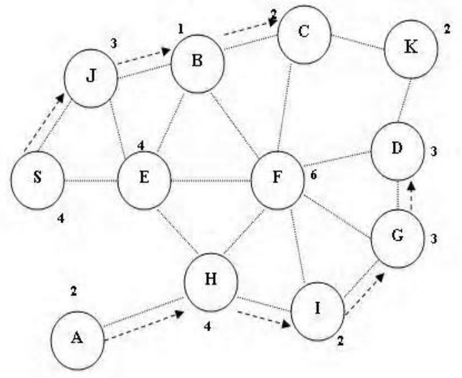


Figure 6 Route constructions between node pair S and D

For example, in Figure 5, we can see the path created between the source node A and the destination node D. The routing tables at each node are shown in the figure below when the source node A has sent two bytes of data packets to the destination node D. In the routing tables, we use the "#" sign to show the node ID and the "$" sign to indicate neighboring nodes. The "$ttl" represents the calculated Time to Live (TTL) value for the flag bit.
Now, let's imagine a situation where node A and node D are already communicating, and at the same time, node S wants to communicate with node C. According to the algorithm and the routing table information at each node, node C chooses the route C-B-J-S. It's important to note that node E does not forward the route request message during this time because the flag bit of its neighboring node H is set.

These metrics, along with weight values, determine the path for transmitting data. While many routing algorithms mainly consider the hop count when choosing the best path, this approach can lead to performance issues due to congestion. Our proposed algorithm selects a path based on higher route energy (maximum battery power with each node), a higher traffic queue (maximum number of packets waiting at a node), and a feasible hop count. In table 1 shown Routing Table at a Node, Calculation of Node Routing table according to directed paths shown in figure 7.

Table 1: Routing Table at a Node

Destination	**Source**	**Incoming Inter-mediate Node**	**Outgoing Inter-mediate Node**	**distance**
G	A	H	I	D1
C	S	B	J	D2
G	A	D	H,I	D3

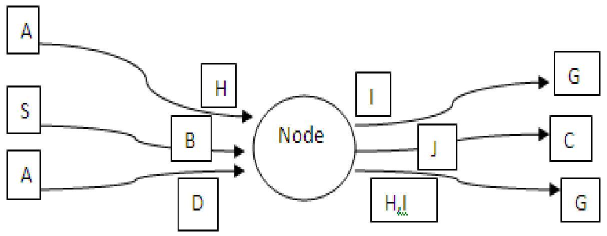


Figure 7 Calculation of Node Routing table according to directed paths

Our INTSM protocol introduces changes to existing load balancing protocols:

- Paths are selected based on the number of hops and the length of the queue at each node.
- Load is balanced by using alternate paths when the queue length exceeds a certain threshold.
- Route request packets are forwarded or discarded based on the queue length.

By considering factors like hop count, queue length, and energy efficiency, our proposed INTSM protocol aims to improve the performance of load balancing in MANETs. It selects paths that have more available energy and can handle more traffic. The protocol also ensures that routing decisions are made based on feasible hop counts.
The INTSM protocol offers a new approach to load balancing during route discovery in MANETs. It considers various metrics to select the best paths and balances the load to ensure efficient data transmission. The protocol makes changes to existing load balancing mechanisms, such as selecting paths based on hop count and queue length, and forwarding or discarding route request packets based on the queue length.

4. Intermediate Node Traffic Sharing Model (INTSM) Technology

The Intermediate Node Traffic Sharing Model (INTSM) is a technology that aims to optimize traffic distribution and load balancing in a network by utilizing intermediate nodes. It provides a framework for efficient sharing of traffic among nodes, ensuring optimal utilization of network resources.
In the INTSM technology, intermediate nodes play a crucial role in managing and balancing network traffic. These nodes actively monitor the load and status of neighboring nodes, exchange information through periodic hello messages, and update their routing tables accordingly. By analyzing the flow of traffic, INTSM identifies underutilized and overloaded nodes within the network.
Underutilized nodes, characterized by low load values, are identified as potential candidates for receiving additional traffic. On the other hand, overloaded nodes, indicated by high load values, are in need of traffic diversion to relieve their congestion. INTSM implements traffic diversion by rerouting data packets from overloaded nodes to underutilized nodes, utilizing alternate paths whenever available. Routing tables are updated to redirect traffic to the selected paths based on load analysis.
Additionally, INTSM facilitates route discovery and data transmission. When a source node wants to send data packets to a destination node, a route request message is broadcasted, including load information and the intermediate nodes it has passed through. Intermediate nodes receive and forward the route request, while the destination node calculates the load of the path and sends a route reply back to the source node. Upon receiving the route reply, the source node begins transmitting data along the established path.
Overall, the INTSM technology provides a comprehensive framework for efficient traffic sharing, load balancing, and route optimization within a network, leading to improved network performance and resource utilization.

4.1 Algorithm:

Here’s an algorithm for the proposed Intermediate Node Traffic Sharing Model (INTSM) protocol:

- Initialize network parameters and data structures.
- Each node maintains information about its neighbouring nodes, including their load, flag status, and routing table.
- Periodically, each node broadcasts hello messages to its neighbouring nodes to exchange information about their load and status.
- Upon receiving a hello message, a node updates its routing table with the load and status information from the neighbouring node.
- Classify the flow of traffic and identify nodes that are underutilized or overloaded based on load analysis.
- Perform traffic diversion or load balancing based on the flow classification to distribute the load evenly among nodes.
- When a source node wants to send data packets to a destination node:
    - The source node broadcasts a route request message, including its load information and the intermediate nodes it has passed through.
    - Intermediate nodes receive the route request message, update their routing tables with the load information, and forward the message to their neighbouring nodes.

- The destination node receives the route request message, calculates the load of the path, and sends a route reply back to the source node.
- Intermediate nodes set their flag status and forward the route reply to their neighbouring nodes.
- The source node receives the route reply, indicating that the route has been established, and starts transmitting data along the established path.

- Monitor the network to minimize collisions and ensure efficient data transmission.

Pseudo Code

```
# Define Node class with attributes
Class Node {
   field load_
   field status_
   field routing_table_
}

# Function to initialize network parameters and data structures
proc initialize_network_parameters_and_data_structures {} {
   # Initialize network parameters and data structures
}

# Function to exchange hello messages among nodes
proc exchange_hello_messages {nodes} {
   foreach node $nodes {
      # Broadcast hello message to neighboring nodes
      # Receive and update routing tables with load and status from neighbors
   }
}

# Function to classify traffic and identify nodes
proc classify_traffic_and_identify_nodes {nodes} {
   foreach node $nodes {
      # Classify traffic flow and identify underutilized/overloaded nodes
   }
}

# Function to broadcast route request messages
proc broadcast_route_request {source_node destination_node} {
   # Create route request message with load info and path history
   # Broadcast route request message

   while {!route_reply_received} {
      foreach node $nodes {
            # Receive and update routing table with load info from neighboring nodes
            if {$node_is_intermediate} {
               # Forward route request message to neighbors
            }

            if {$node_is_destination} {
               # Calculate load of path and send route reply back to source node
               # Set flag status and forward route reply to neighbors
            }
         }
      }
   }
}

# Function to transmit data
proc transmit_data {source_node destination_node} {
   if {$route_established} {
      # Start transmitting data packets
   }
}

# Function to monitor network
proc monitor_network {} {
   # Minimize collisions and ensure efficient data transmission
}

# Main program
initialize_network_parameters_and_data_structures
set num_nodes 10
set simulation_running 1

# Create nodes
for {set i 0} {$i < $num_nodes} {incr i} {
   set nodes($i) [Node new]
}

while {$simulation_running} {
   exchange_hello_messages [array get nodes]
   classify_traffic_and_identify_nodes [array get nodes]

   foreach data_transfer $data_transfer_requests {
      set source_node [lindex $data_transfer 0]
      set destination_node [lindex $data_transfer 1]

      broadcast_route_request $source_node $destination_node
      transmit_data $source_node $destination_node
   }

   monitor_network
}
```

4.1.1 Node Load Calculation Algorithm:

- Each node periodically calculates its own load based on the number of packets it has delivered.
- The load calculation algorithm updates the load value in the node's routing table.
- The load value is determined by the number of packets delivered, and it reflects the node's current load status.

Pseudo Code

```
# Define Node class with attributes and methods
Class Node {
    field load_
    field status_
    field routing_table_

    constructor {} {
        self set load_ 0
        self set status_ "normal"
        self set routing_table_ [dict create]
    }

    method calculate_load {} {
        # Calculate load based on the number of
packets delivered
        set delivered_packets [expr int(rand() * 100)]
;# Example: random value for demonstration
        self set load_ $delivered_packets
    }
}

# Function to periodically calculate load for all
nodes
proc calculate_load_for_all_nodes {nodes} {
    foreach node $nodes {
        $node calculate_load
    }
}

# Rest of the functions and main program
# ... (previous code)

# Main simulation loop
while {$simulation_running} {
    calculate_load_for_all_nodes [array get nodes]

    # Rest of the simulation loop
    exchange_hello_messages [array get nodes]
    classify_traffic_and_identify_nodes [array get
nodes]

    foreach data_transfer $data_transfer_requests {
        set source_node [lindex $data_transfer 0]
        set destination_node [lindex $data_transfer 1]

        broadcast_route_request $source_node
$destination_node
        transmit_data $source_node
$destination_node
    }

    monitor_network
}
```

4.1.2 Route Selection Algorithm:

- When a route request message is received, the intermediate nodes update their routing tables with the load information and forward the message.
- The destination node calculates the load of the path by summing up the load values of the intermediate nodes.
- Based on the load information, the destination node selects the path with the least load as the best route.
- The destination node sets its flag status and sends a route reply back to the source node.
- Intermediate nodes receiving the route reply update their flag status and forward the reply.
- The source node, upon receiving the route reply, starts data transmission along the established route.

Pseudo Code

```
# Function to handle route request message
proc handle_route_request {source_node
intermediate_node destination_node} {
    # Update routing table with load information
from intermediate_node
    # Forward the route request message

    if {$node_is_destination} {
        # Calculate load of the path
        set path_load 0
        foreach node $intermediate_nodes {
            set path_load [expr $path_load + [$node
get load_]]
        }

        # Select path with least load as the best route
        set best_route $intermediate_nodes
        set best_load $path_load
        if {$path_load > [$destination_node get
load_]} {
            set best_route $destination_node
            set best_load [$destination_node get load_]
        }

        # Set flag status and send route reply back to
source_node
        $destination_node set status_ "flag"
        send_route_reply_to_source $source_node
$best_route $best_load
    }
}

# Function to send route reply back to source node
proc send_route_reply_to_source {source_node best_route best_load} {
    # Send route reply to source_node
    # Update flag status in best_route
    # Forward route reply message
}

# Rest of the functions and main program
# ... (previous code)

# Main simulation loop
while {$simulation_running} {
    calculate_load_for_all_nodes [array get nodes]

    # Rest of the simulation loop
    exchange_hello_messages [array get nodes]
    classify_traffic_and_identify_nodes [array get nodes]

    foreach data_transfer $data_transfer_requests {
        set source_node [lindex $data_transfer 0]
        set destination_node [lindex $data_transfer 1]

        broadcast_route_request $source_node $destination_node
        transmit_data $source_node $destination_node
    }

    monitor_network
}
```

4.1.3 Traffic Diversion Algorithm:

- Nodes analyze the flow of traffic and identify nodes that are underutilized or overloaded.
- Underutilized nodes are identified based on their low load values, indicating that they can handle more traffic.
- Overloaded nodes are identified based on their high load values, indicating that they should divert traffic to balance the load.
- Traffic diversion is performed by rerouting data packets from overloaded nodes to underutilized nodes, using alternate paths when available.
- The routing tables are updated accordingly to redirect traffic to the selected paths.

Pseudo Code

```
# Function to analyze traffic flow and perform traffic diversion
proc analyze_traffic_and_divert {nodes} {
    set underutilized_nodes {}
    set overloaded_nodes {}

    # Analyze flow of traffic and identify underutilized and overloaded nodes
    foreach node $nodes {
        set load [$node get load_]
        if {$load < threshold_underutilized} {
            lappend underutilized_nodes $node
        } elseif {$load > threshold_overloaded} {
            lappend overloaded_nodes $node
        }
    }

    # Perform traffic diversion
    foreach overloaded_node $overloaded_nodes {
        foreach underutilized_node $underutilized_nodes {
            # Check if alternate path available and reroute data packets
            if {alternate_path_available $overloaded_node $underutilized_node} {
                update_routing_tables_for_diversion $overloaded_node $underutilized_node
                divert_data_packets $overloaded_node $underutilized_node
                break ; # Divert to one underutilized node
            }
        }
    }
}

# Function to check if alternate path is available between nodes
proc alternate_path_available {source_node destination_node} {
    # Check if alternate path available and return true/false
}

# Function to update routing tables for traffic diversion
proc update_routing_tables_for_diversion {source_node destination_node} {
    # Update routing tables to redirect traffic
}

# Function to divert data packets from source to destination
proc divert_data_packets {source_node destination_node} {
    # Divert data packets from source_node to destination_node
}

# Rest of the functions and main program
# ... (previous code)

# Main simulation loop
while {$simulation_running} {
    calculate_load_for_all_nodes [array get nodes]
    # Rest of the simulation loop
    exchange_hello_messages [array get nodes]
    classify_traffic_and_identify_nodes [array get
  nodes]

    foreach data_transfer $data_transfer_requests {
      set source_node [lindex $data_transfer 0]
      set destination_node [lindex $data_transfer 1]

      broadcast_route_request $source_node
  $destination_node
      transmit_data $source_node
  $destination_node
    }

    analyze_traffic_and_divert [array get nodes]
    monitor_network
  }
```

Algorithm: INTSM_Protocol:

```
 Algorithm: INTSM_Protocol (Source, Destination)

 Start

 Step 1: Initialize
   Initialize network parameters and data structures

 Step 2: Exchange Information
   For each node in the network:
     - Exchange hello messages and load/status info
     - Classify nodes as underutilized or overloaded

 Step 3: Traffic Analysis and Diversion
   Analyze traffic flow:
     - Identify underutilized and overloaded nodes
     - Perform traffic diversion for load balancing

Step 4: Route Discovery and Data Transmission
  Broadcast route request with load info
  While route reply not received:
    If intermediate node:
      - Forward route request, update routing tables
    If destination node:
      - Calculate path load, send route reply
      - Set flag status
    If intermediate node:
      - Forward route reply, set flag status
    If source node:
      - Start data transmission along route

Step 5: Monitor Network
  Monitor for efficient data transmission
  Ensure minimal collisions

End

End Algorithm
```

5. Results and Discussion

5.1 Simulation Parameters

The performance evaluation of the protocol is conducted using the event-driven ns2.35 simulator[11]. For this simulation, a random mobility model is chosen. Nodes are randomly distributed within a rectangular area measuring 1507 m x 732 m. To simulate the protocol, various parameters are set and evaluated in the TCL script of the network. The specific simulation parameters used in the experiments are outlined in Table 2.
The table2 below presents the simulation parameters employed during the experiments:

Table 2: Simulation Parameters

Scenario Elements	**Values**	**Unit**
Number of nodes	100	Nodes
Node speed	10	Meter/second
Queue size	50	packets
Simulation area	1507 * 732	Meter^2
Routing protocols	AODV, DSR, INTSM	Protocol
Mobility model	Random way point	-
Packet size	512	Bytes
Traffic type	CBR	-
Transmission power consumption	0.035	Joules
Receive power consumption	0.035	Joules
Idle Power	0.100	Joules
Sense Power	0.0175	Joules
Simulation time	100, 150, 200, 250	seconds

5.2 Performance Metrics

The following performance metrics are computed to analyze the behavior of the protocol under different simulation durations. These metrics provide valuable insights into the performance and effectiveness of the protocol."

5.2.1 Packet Delivery Ratio (PDR): It measures the proportion of data packets successfully delivered from the source node to the destination node. PDR is calculated by dividing the number of received packets by the number of sent packets, multiplied by 100.

PDR = (Number of packets received / Number of packets sent) * 100

5.2.2 Throughput: It quantifies the rate at which data is transmitted and received within the network. Throughput represents the number of bits successfully received at the destination node.

Throughput = (Number of bits received / Time taken for reception)

5.2.3 End-to-End Delays: It refers to the time taken for a data packet to travel from the source node to the destination node. It includes various delays encountered during transmission, such as propagation delay, queuing delay, and processing delay.

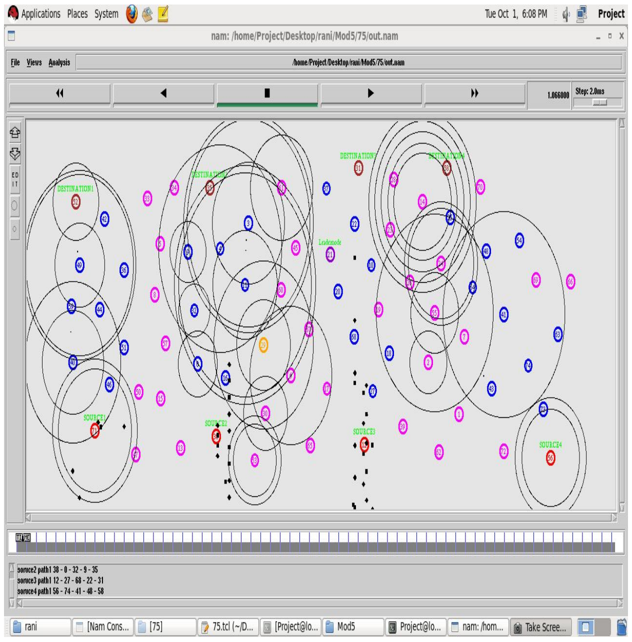
End-to-End Delay = Time taken for a packet to reach the destination - Time at which the packet was sent

5.2.4 Routing Overhead: It measures the additional control messages and signalling required for routing purposes. Routing overhead includes the extra network traffic generated by routing protocols to establish and maintain routing paths.

Routing Overhead = (Number of routing control messages / Number of data packets sent) * 100

5.3 Results and Discussion

This section shows the results of Simulation. After implementing protocols in NS2.35 simulator some screenshot of network topology is shown in Figure 8.



```
soruce2 path1 38 - 0 - 32 - 9 - 35
soruce3 path1 12 - 27 - 68 - 22 - 31
soruce4 path1 56 - 74 - 41 - 48 - 58
```

Figure 8 Network Simulator Windows

The proposed algorithm INTSM is implemented, and simulation results based on various simulation durations are presented in this section. All parameters have been included in the simulations to ensure comprehensive evaluation and analysis.

5.3.1 Packet Delivery Ratio (PDR)

The packet delivery ratio (PDR) measures the proportion of data packets successfully delivered from the source node to the destination node. PDR is calculated as the number of packets received divided by the number of packets sent, multiplied by 100:

PDR = (Number of packets received / Number of packets sent) * 100.

Table 3 displays the variation of the packet delivery ratio for different simulation times using the INTSM, AODV, and DSR protocols. Despite the occurrence of failures or route disruptions, the INTSM protocol achieves a high PDR. The packet delivery ratio generally increases with simulation time, indicating improved efficiency in packet delivery. Notably, the INTSM protocol consistently outperforms both the AODV and DSR protocols in terms of packet delivery ratio. This demonstrates the effectiveness of INTSM in ensuring reliable data packet delivery within the network.

Simulation Time(seconds)	Protocols		
	AODV	DSR	INTSM
100	40.123	32.234	54.543
150	37.543	30.123	59.432
200	35.678	32.89	48.754
250	33.89	34.543	58.432

Table 3: Comparison of PDR

Simulation Time	100	150	200	250	AVERAGE/ OVERALL
INTSM compared to AODV	26.43 %	36.83%	26.82 %	42.00 %	33.42%
INTSM compared to DSR	40.90 %	49.31%	32.53 %	40.88 %	41.31%

Table 4 Packet Delivery Ratio comparison of INTSM with AODV and DSR

The table 4 compares the throughput performance of the INTSM protocol with the AODV and DSR protocols. It shows the percentage improvement in throughput achieved by INTSM compared to AODV and DSR for different simulation times (100, 150, 200, and 250). On average, INTSM improves throughput by 33.42% compared to AODV and 41.31% compared to DSR. This indicates that

INTSM generally performs better in terms of data transmission efficiency.

Figure 9 shows that the Packet Delivery Ratio (PDR) of the INTSM protocol remains consistently higher than that of AODV and DSR for varying simulation durations. This is because the Intermediate Node Traffic Sharing Model employed by INTSM facilitates load balancing and congestion avoidance. By redistributing traffic from overloaded nodes to underutilized nodes, INTSM reduces packet loss and improves packet delivery. In contrast, AODV struggles to maintain a high PDR due to limited adaptability to changing network conditions over time.

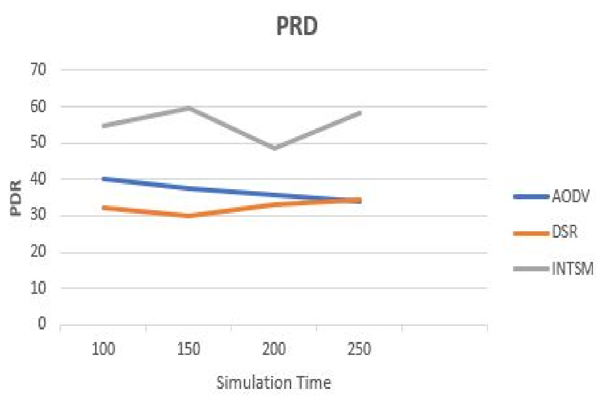


Figure 9 Packet Delivery Ratio with simulation time

Analysis of Packet Delivery Ratio
The Packet Delivery Ratio (PDR) measures the success rate of delivering data packets from the source to the destination node. The INTSM protocol consistently outperforms AODV and DSR protocols in terms of PDR, even in the presence of failures or route disruptions. This is because INTSM employs an Intermediate Node Traffic Sharing Model, which facilitates load balancing and congestion avoidance. By redistributing traffic from overloaded nodes to underutilized nodes, INTSM reduces packet loss and improves overall delivery. AODV struggles to maintain high PDR due to limited adaptability to changing network conditions. Overall, the analysis demonstrates the effectiveness of INTSM in ensuring reliable and efficient packet delivery within the network.

5.3.2 Throughput

Throughput is a measure of the number of bits successfully received at the destination node, reflecting the rate at which data is transmitted and received within the network.
Table 5 displays INTSM achieves higher throughput compared to AODV and DSR throughout the simulation duration. By effectively sharing traffic among nodes and balancing the load, INTSM optimizes data transmission and enhances network capacity. It identifies underutilized nodes and redirects traffic from overloaded nodes, leading to improved throughput. On the other hand, AODV and DSR struggle to handle the increasing traffic load over time, resulting in lower throughput.

Simulation Time(seconds)	Protocols		
	AODV	DSR	INTSM
100	80.32	64.61	191.85
150	115.1	92.73	199.2
200	103.73	135.2	262.87
250	143.27	129.2	230.41

Table 5: Comparison of Throughput

Simulation Time(seconds)	Protocols		
	AODV	DSR	INTSM
100	80.32	64.61	191.85
150	115.1	92.73	199.2
200	103.73	135.2	262.87
250	143.27	129.2	230.41

Table 6 Throughput comparison of INTSM with AODV and DSR

Table 6 compares the throughput performance of INTSM with AODV and DSR. INTSM achieves higher throughput compared to both protocols, with average improvements of 49.97% over AODV and 52.30% over DSR. This indicates that INTSM is more efficient in terms of data transmission rates. In Figure 10, the graph displays the variation of throughput with simulation time for the INTSM protocol, along with the AODV and DSR protocols.

The graph in figure 10 demonstrates that the INTSM protocol achieves better throughput performance compared to both the AODV and DSR protocols. This indicates that the INTSM protocol enables higher data transmission rates and improved network efficiency. As simulation time increases, the throughput of the INTSM protocol continues to outperform the other protocols, showcasing its effectiveness in facilitating efficient data transfer within the network.

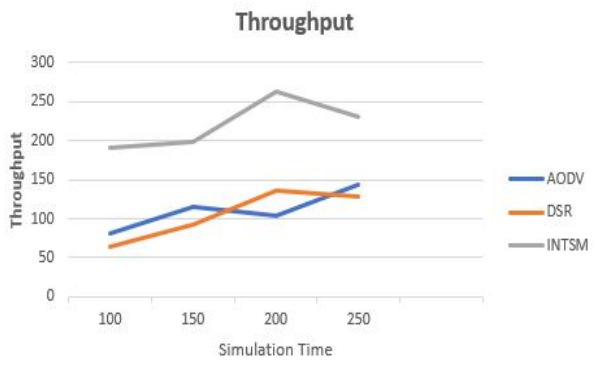


Figure 10 Throughput with simulation time

Analysis of Throughput:
The analysis shows that the INTSM protocol consistently achieves higher throughput compared to AODV and DSR. This can be attributed to its effective traffic sharing and load balancing mechanisms, which optimize data

transmission and enhance network capacity. In contrast, AODV and DSR struggle to handle increasing traffic loads over time, resulting in lower throughput. Overall, the superior throughput of the INTSM protocol indicates its efficiency in facilitating data transfer within the network.

5.3.3 End-to-End Delay

"End-to-End Delay refers to the average time taken for a data packet generated by the source to reach its destination. It encompasses various delays encountered during the packet's journey, including interface queueing delays, routing latency, buffering, transfer time, packet queuing, and propagation.

Table 7 displays the Average End-to-End Delay of the INTSM protocol remains consistently lower than AODV and DSR for different simulation durations. INTSM effectively manages traffic flow and load balancing, resulting in reduced congestion and shorter delays. It identifies underutilized nodes and reroutes traffic to alleviate network congestion. Additionally, INTSM predicts link breakages and finds alternate paths in advance, minimizing delays caused by route re-discoveries. AODV experiences higher delays as the simulation time increases due to its limited ability to adapt dynamically.

Table 7 Comparison of End-to-End Delay.

Simulation Time(seconds)	Protocols		
	AODV	DSR	INTSM
100	579.79	1061.11	256.695
150	940.768	1079.9	87.9443
200	662.81	1008.82	177.819
250	535.12	1182.67	68.5571

Figure 11 depicts the graph illustrating the variation of end-to-end delay with simulation time for the INTSM protocol, along with the AODV and DSR protocols.

Table 8 End-to-End Delay comparison of INTSM with AODV and DSR

Simulation Time	100	150	200	250	AVERAGE/OVERALL
INTSM compared to AODV	55.72 %	60.65 %	63.17 %	67.18 %	68.25%
INTSM compared to DSR	65.80 %	61.85 %	62.37 %	64.20 %	66.35%

Table 8 compares the end-to-end delay performance of INTSM with AODV and DSR. INTSM achieves significant reductions in end-to-end delay compared to both protocols, with average reductions of 68.25% over AODV and 66.35% over DSR. This indicates that INTSM is more efficient in minimizing the time it takes for data to reach its destination.

The graph clearly demonstrates that the INTSM protocol, specifically the INTSM protocol, exhibits superior performance in terms of end-to-end delay compared to both the AODV and DSR protocols. The INTSM protocol minimizes end-to-end delay by utilizing energy-efficient routing techniques and effectively avoiding delays in packet transmission through multiple routes. This improvement in end-to-end delay results in faster and more efficient data packet delivery within the network."

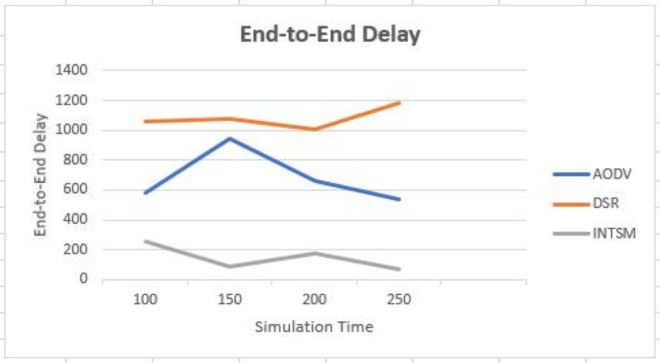


Figure 11 End-to-end delays with simulation time

Analysis of End-to-End Delay:
The analysis reveals that the INTSM protocol demonstrates superior performance in terms of end-to-end delay compared to AODV and DSR. It effectively manages traffic flow, load balancing, and congestion avoidance, resulting in lower delays. By predicting link breakages and finding alternate routes, INTSM minimizes delays caused by route re-discoveries. In contrast, AODV experiences higher delays due to its limited adaptability. Overall, the lower end-to-end delay of the INTSM protocol enhances the efficiency and speed of data packet delivery within the network.

5.3.4 Routing Overhead

In terms of routing overhead, as shown in table 9, INTSM exhibits slightly higher overhead than AODV and DSR for different simulation durations. The Intermediate Node Traffic Sharing Model in INTSM requires additional routing information exchange among nodes, contributing to the overhead. However, the overhead is justified by the improved network performance achieved through load balancing and congestion avoidance. AODV faces challenges in managing the increasing routing overhead as the simulation time progresses.

Table 9 Comparison of Routing Overhead

Simulation Time(seconds)	Protocols		
	AODV	DSR	INTSM
100	117346	46965	39014
150	83997	45868	16690
200	141451	48592	14987
250	125109	32447	12511

Simulation Time	100	150	200	250	AVERAGE/ OVERALL
INTSM compared to AODV	66.75 %	60.13 %	69.40 %	69.99 %	62.21 %
INTSM compared to DSR	16.92 %	53.61 %	59.15 %	51.44 %	52.21 %

Table 10 Routing Overhead comparison of INTSM with AODV and DSR

Table 10 compares the routing overhead of INTSM with AODV and DSR. INTSM generally has lower routing overhead compared to both protocols, with average differences of 62.21% compared to AODV and 52.21% compared to DSR. This indicates that INTSM is more efficient in terms of the control information exchanged during the routing process.
Figure 12 depicts the graph clearly highlights the differences in Routing Overhead among the three protocols. The INTSM protocol demonstrates superior performance in terms of Routing Overhead compared to both the AODV and DSR protocols.

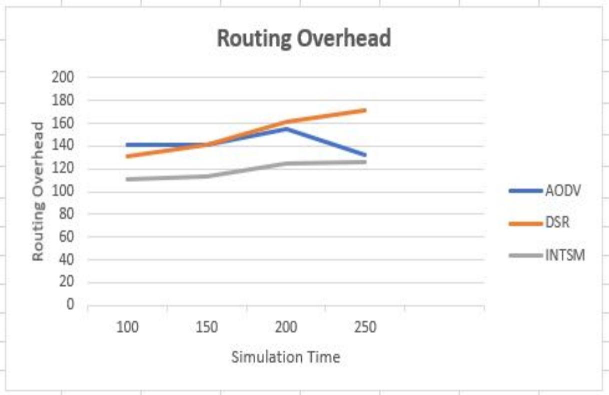


Figure 12 Routing overhead with simulation time

Analysis of Routing Overhead:
The analysis of Routing Overhead shows that the INTSM protocol consistently exhibits lower overhead compared to AODV and DSR throughout the simulation. Although INTSM incurs slightly higher overhead due to additional routing information exchange, it justifies this by achieving improved network performance through load balancing and congestion avoidance. The ability of INTSM to optimize routing and minimize overhead contributes to enhanced network efficiency and resource utilization. Overall, INTSM outperforms AODV and DSR in terms of Routing Overhead, demonstrating its effectiveness in improving network efficiency.

Conclusion
In conclusion, the INTSM algorithm provides a promising solution for load balancing and route discovery in ad-hoc networks. It aims to achieve a balanced topology, efficient data transmission, and congestion management. The algorithm's low complexity route balancing and controlled flow classification mechanisms contribute to optimized traffic management. Through simulations and performance evaluations, we expect the INTSM algorithm to demonstrate improved network performance and reliability. Our research contributes to the advancement of ad-hoc network protocols, enhancing overall network efficiency and effectiveness. Future work for the INTSM algorithm includes:

- Conducting extensive performance evaluations in various network scenarios.
- Comparing the algorithm with existing protocols to assess its effectiveness.
- Implementing and testing the algorithm in real-world ad-hoc networks.
- Analyzing the scalability of the algorithm for larger networks.
- Considering energy efficiency optimizations for power-constrained environments.

By addressing these areas, we can enhance the algorithm's performance and contribute to advancements in load balancing and route discovery in ad-hoc networks.

REFERENCES

1. Saudi, N.A.M., et al. Mobile ad-hoc network (MANET) routing protocols: A performance assessment. in Proceedings of the Third International Conference on Computing, Mathematics and Statistics (iCMS2017) Transcending Boundaries, Embracing Multidisciplinary Diversities. 2019. Springer.
2. Lu, Q.-C., P.-C. Xu, and J. Zhang, Infrastructure-based transportation network vulnerability modeling and analysis. Physica A: Statistical Mechanics and its Applications, 2021. **584**: p. 126350.
3. Shrivastava, P.K. and L. Vishwamitra, Comparative analysis of proactive and reactive routing protocols in VANET environment. Measurement: Sensors, 2021. **16**: p. 100051.
4. Safari, F., et al., 'The diverse technology of MANETs: A survey of applications and challenges. International Journal of Future Computer and Communication, 2023. **12**(2).
5. Ghaleb, S.A.M. and V. Vasanthi, Energy efficient multipath routing using multi-objective grey wolf optimizer based dynamic source routing algorithm for manet. International Journal of Advanced Science and Technology, 2020. **29**(3): p. 6096-6117.
6. Yan, S. and Y. Chung, Improved ad hoc on-demand distance vector routing (AODV) protocol based on blockchain node detection in ad hoc networks. International Journal of Internet, Broadcasting and Communication, 2020. **12**(3): p. 46-55.

7. Pandey, P.K., A. Swaroop, and V. Kansal. A concise survey on recent routing protocols for vehicular ad hoc networks (VANETs). in 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS). 2019. IEEE.
8. Chouhan, P., Enhanced Ad-Hoc On-Demand Distance Vector (AODV) Routing Protocol against Blackhole Attacks in MANET.
9. Almazok, S.A. and B. Bilgehan, A novel dynamic source routing (DSR) protocol based on minimum execution time scheduling and moth flame optimization (MET-MFO). EURASIP Journal on Wireless Communications and Networking, 2020. **2020**: p. 1-26.
10. Agrawal, R., et al., Classification and comparison of ad hoc networks: A review. Egyptian Informatics Journal, 2022.
11. Shantaf, A.M., S. Kurnaz, and A.H. Mohammed. Performance evaluation of three mobile ad-hoc network routing protocols in different environments. in 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA). 2020. IEEE.
12. Duggal, R., Classification of Load Balancing Routing Protocols In Mobile Ad Hoc Networks: A Comparative Survey. Think India Journal, 2019. **22**(16): p. 2748-2756.
13. Singh, S.K. and J. Prakash. Energy efficiency and load balancing in MANET: a survey. in 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS). 2020. IEEE.
14. Hamdan, M., et al., A comprehensive survey of load balancing techniques in software-defined network. Journal of Network and Computer Applications, 2021. **174**: p. 102856.
15. Kumar, P. and R. Kumar, Issues and challenges of load balancing techniques in cloud computing: A survey. ACM Computing Surveys (CSUR), 2019. **51**(6): p. 1-35.
16. Singh, J. and C.S. Rai, An efficient load balancing method for ad hoc networks. International Journal of Communication Systems, 2018. **31**(5): p. e3503.
17. Choudhary, P. and R. Verma, Energy Saving Load Balancing Approach to Boost AOMDV Routing in MANET And Data Security: A Survey. Journal homepage: www. ijrpr. com ISSN. **2582**: p. 7421.
18. Ali, A. and W. Huiqiang. Node centric load balancing routing protocol for mobile ad hoc networks. in Proceeding of Internationl MultiConference of Enginners Computer Scientists. 2012.
19. Souihli, O., M. Frikha, and M.B. Hamouda, Load-balancing in MANET shortest-path routing protocols. Ad Hoc Networks, 2009. **7**(2): p. 431-442.
20. Tashtoush, Y., O. Darwish, and M. Hayajneh, Fibonacci sequence based multipath load balancing approach for mobile ad hoc networks. Ad Hoc Networks, 2014. **16**: p. 237-246.
21. Salem, M.A. and R. Yadav, Literature Review of Load Balancing Routing Protocols in MANET. International Journal of Computer Applications, 2015. **125**(14).
22. Hardik, M.N.R.R.P. and J. Patel, Load Balancing and Congestion Control in MANET.
23. Ali, H.A., T.T. Hamza, and S. Sarhan, Manet Load Balancing Parallel Routing Protocol. International Journal of Computer Science Issues (IJCSI), 2012. **9**(4): p. 187.
24. Gulati, M.K., M. Sachdeva, and K. Kumar, Load Balanced and Link Break Prediction Routing Protocol for Mobile Ad Hoc Networks. J. Commun., 2017. **12**(6): p. 353-363.
25. Venkatraman, S. and S.K. Sarvepalli, Load balance technique with adaptive position updates (LAPU) for geographic routing in MANETs. EURASIP Journal on Wireless Communications and Networking, 2018. **2018**(1): p. 1-9.
26. Zheng, X., et al., A cross-layer design and ant-colony optimization based load-balancing routing protocol for ad-hoc networks. Frontiers of Electrical and Electronic Engineering in China, 2007. **2**: p. 219-229.
27. Jung, S., N. Hundewale, and A. Zelikovsky. Energy efficiency of load balancing in MANET routing protocols. in Sixth international conference on software engineering, artificial intelligence, networking and parallel/distributed computing and first ACIS international workshop on self-assembling wireless network. 2005. IEEE.
28. Xu, L., et al. Research and analysis of topology control in NS-2 for ad-hoc wireless network. in 2008 International Conference on Complex, Intelligent and Software Intensive Systems. 2008. IEEE.
29. Faoudi, F., S. Yahiaoui, and Y. Belhoul, NS-2 Based Framework for Topology Control in Wireless Ad-hoc Networks. Communications in Information Science and Management Engineering, 2012. **2**(4).
30. Arafat, M.Y., S. Poudel, and S. Moh, Medium access control protocols for flying ad hoc networks: A review. IEEE Sensors Journal, 2020. **21**(4): p. 4097-4121.