Master of Science in Computer Science and Engineering

An Efficient Multicast Routing Protocol to Minimize Multipoint Relays in MANET

by

Md. Zahid Hassan

ID: 18MCSE045F

This thesis is submitted in partial fulfillment of the requirement for the degree of
Master of Science in Computer Science and Engineering

Department of Computer Science and Engineering
Chittagong University of Engineering and Technology
Chattogram-4349, Bangladesh.
March, 2024

CERTIFICATION

The thesis titled **"An Efficient Multicast Routing Protocol to Minimize Multipoint Relays in MANET"** submitted by **Md. Zahid Hassan**, Roll No. **18MCSE045F**, Session **2018-2019** has been accepted as satisfactory in partial fulfillment of the requirement for the degree of **Master of Science in Computer Science and Engineering** on **19/03/2024**.

BOARD OF EXAMINERS

1. _____

Dr. Asaduzzaman — Chairman (Supervisor)
Professor
Department of Computer Science and Engineering
Chittagong University of Engineering & Technology (CUET)

2. _____

Dr. Abu Hasnat Mohammad Ashfak Habib — Member (Ex-officio)
Professor and Head
Department of Computer Science and Engineering
Chittagong University of Engineering & Technology (CUET)

3. _____

Dr. Mohammad Shamsul Arefin — Member
Professor
Department of Computer Science and Engineering
Chittagong University of Engineering & Technology (CUET)

4. _____

Dr. Mahfuzulhoq Chowdhury — Member
Associate Professor
Department of Computer Science and Engineering
Chittagong University of Engineering & Technology (CUET)

5. _____

Dr. Md. Mamun-Or-Rashid — Member (External)
Professor
Department of Computer Science and Engineering
University of Dhaka (DU)

CANDIDATE'S DECLARATION

It is hereby declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.

Signature of the Candidate

Md. Zahid Hassan
ID: 18MCSE045F

Dedicated to,

my beloved parents,

Md. Asaduzzaman Sarkar *and* ***Mst. Zinna Khatun***

my respected uncle,

Md. Mizanur Rahman *and*

my lovely and energetic wife,

Dr. Rukaiya Shultana (Rinty)

for their love, endless support, encouragement and sacrifices.

Acknowledgment

I am overwhelmed to get this opportunity to express my gratitude to those whose constant support and encouragement led me to the completion of this thesis. The satisfaction that accompanies the successful completion of this thesis would be incomplete without the mention of the people whose ceaseless cooperation made it possible. First and foremost, I feel immense proud to express my heartfelt respect and deepest sense of gratitude to my supervisor, Prof. Dr. Asaduzzaman for his continuous guidelines, constructive criticism, inspiration, and overall supervision of this research work. I am ever grateful to the honorable board members for their valuable suggestions and helpful feedbacks. I am also incredibly thankful to all the faculty members and the staff of department of CSE, CUET for their support. Many of my colleagues and friends deserve special thanks for their assistance and motivation.

Finally and always, I am grateful to the Almighty Allah for giving me the strength and patience to complete this thesis work.

Abstract

Reducing control packets, especially in proactive routing protocols, needed to establish routes can lower network overhead in Mobile Ad-hoc Networks (*MANETs*). Optimized Link State Routing (*OLSR*) is a proactive routing protocol renowned for its widespread culmination in *MANET*. In *OLSR*, each Multi-point Relay (*MPR*) node propagates Topology Control (*TC*) messages across the network to advertise neighbor information. The number of *MPR* nodes and hence *TC* messages are significant contributors to increased network overhead; however, *OLSR* counteracts or controls the *TC* messages by reducing the number of *MPR* nodes. In this study, we propose an efficient *MPR* node selection mechanism to reduce the *TC* message volume leading to a minimized routing overhead. Each node selects the lowest cost node from its first hop neighbors as the *MPR* node for any destination. The same *MPR* node can be selected for multiple destinations if it costs the lowest for each destination node. The selection technique is realized by modifying only the default *OLSR TC* and *Hello* messages. The proof-of-concept implementation in the *NS3* simulator reveals that the proposed methodology reduces the routing overhead by selecting around 55%, 28% and 49% (on average) fewer *MPR* nodes compared to the traditional *OLSR*, *SSTB* and *M-OLSR* protocol respectively, without negotiating packet delivery ratio, throughput and delay.

Keywords: *MANET, OLSR, TC Messages, Routing Overhead, MPR*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

MANET [1, 4] is a variant of ad-hoc networks where nodes are mobile and decentralized in type, and packet routing does not need any pre-established centralized infrastructure. Nodes in *MANET* communicate in a peer-to-peer fashion using single- or multi-hop pathways. A node acts as a host and intermediary device to forward or route packets for other devices, and any node can join or leave the network anytime. *MANET* is autonomous, self-configurable, and highly adaptive, and the distinct features make it ideal for realization in scenarios where an infrastructure network is absent or failed, or establishment is challenging or impossible, for example, military applications [5], forest fire surveillance [6], search and rescue operations [7], disaster recovery and rescue operations [8], etc.

1.1 Overview of *MANET* architecture

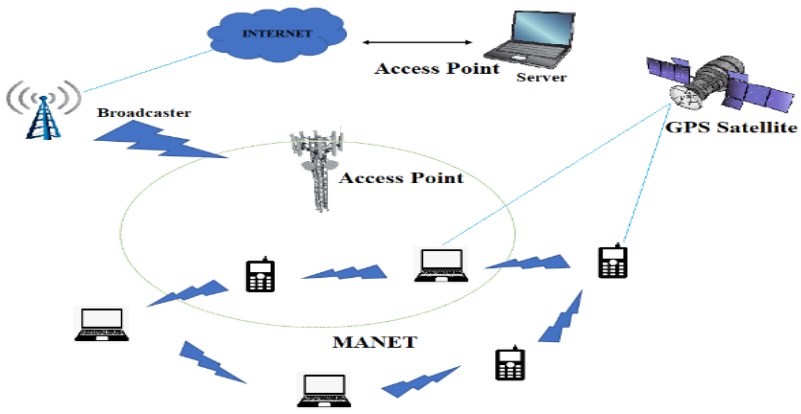


Figure 1.1: *MANET* architecture [1].

Figure 1.1 depicts the *MANET* architecture. Enabling technologies, network-

ing, application and middleware comprise the three primary levels. The layer allowing technologies can be further subdivided into Body Area Networks (*BAN*), Personal Area Networks (*PAN*), and Wireless Local Area Networks (*WLAN*) based on the coverage area. Different layers used in *MANET* are presented in Figure 1.2.

With the help of wireless local area networks, or *WLAN*s, many buildings can be connected to a single network over a 500-meter radius. However, *PAN* communications have a maximum range of 10 meters. The most important networking protocol characteristics require a self-configured, dynamic, secure, peer-to-peer environment, which calls for a redesign of *MANET* architecture. The original intent of networking protocols is to provide a one-hop transmission service. Notable benefits can be derived from *WLAN*, Bluetooth, *IEEE 802.11*, and *WiMAX*, especially in areas like environmental monitoring, emergency services, and disaster recovery. Ad-hoc mobile frameworks that have recently been developed rely on each individual application to handle all necessary services rather than using middleware techniques. Different types of *MANET*s are briefly discussed bellow:

- **Vehicular ad hoc networks (*VANET*s):** Vehicles function as the network's moving nodes in the context of *MANET*. Small inter-contact periods between hosts, fast vehicle speeds, fluctuating vehicle densities, short-range communications, and real-time data exchange requirements are more characteristics that set *VANET*s apart.

- **Flying ad hoc networks (*FANET*s):** By supporting various network types via satellite or other mobile devices, or by transferring data flow from landing devices to a remote server, *FANET*s can function independently.

- **Internet-based mobile ad hoc networks (*IMANET*s):** *IMANET*s are compatible with TCP/UDP, IPv4, and IPv6 and uses the proper protocols to route data between mobile nodes on the network tiers.

- **Intelligent vehicular ad hoc networks (*INVANET*s):** Intelligent *VANET*s use *WiMAX* (*IEEE 802.16*) and *WiFi* (*IEEE 802.11 p*) to facilitate quick and effective dynamic vehicle-to-vehicle communication.

1.1.1 Characteristics of *MANET*s

The portability and flexibility of its autonomous mobile nodes significantly influence packet routing in *MANET*. Alongside, the routing protocols [1] are responsible for delivering packets and maintaining the paths between communicating nodes. In addition, some connected and constantly changing facts, such as

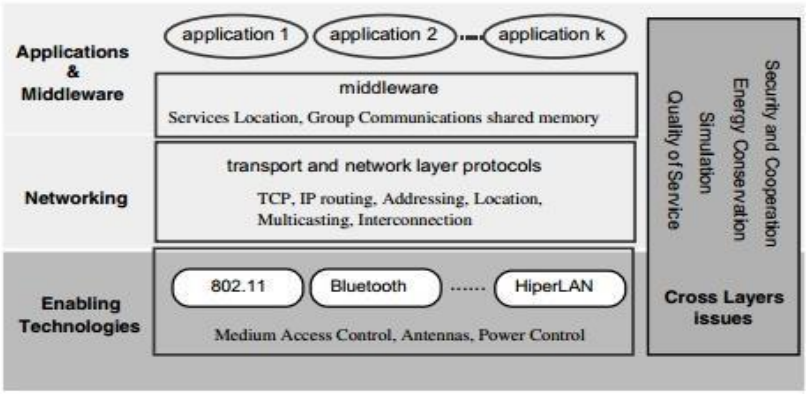


Figure 1.2: Layers in *MANET* [1].

network topology, link quality, bandwidth, residual node energy [9], and more, render the routing protocol a vibrant research area in *MANET* [10, 11]. Some characteristics have been summarised bellow:

- Operates independently of established or centralized infrastructure.
- *MANET* implementations include both fixed or local Wi-Fi networks and large networks, such tactical and sensor networks with thousands of nodes.
- Radio interfaces, which operate on several frequency bands and have various transmission and receiving capabilities, may be present on one or more nodes.
- Every node in a *MANET* serves as both an end device and a router.
- Data packet transmission to other nodes requires a substantial amount of energy.

1.1.2 Routing Protocols

Routing protocols in *MANET* can broadly be categorized into proactive, reactive, and hybrid routing protocols [12]. The proactive routing protocols or table-driven protocols, for example, Optimized Link State Routing (*OLSR*) and (Destination Sequenced Distance Vector (*DSDV*) frequently exchange routing or control packets to establish and maintain paths between nodes. In contrast, on-demand protocols or reactive routing protocols, for instance, Dynamic Source Routing (*DSR*) and Adhoc On-Demand Distance Vector (*AODV*), build a route only when solicited. However, there are routing methods that blend reactive and

proactive routing techniques, such as Zone Routing Protocol (*ZRP*) as well as Temporally Ordered Routing Algorithm (*TORA*).

- **Proactive Routing Protocols:** Table-driven protocols are another name for proactive routing methods. Every node has a routing table that contains every detail of the network topology. Although this feature is helpful for datagram traffic, it also gains a lot of power and advertises other traffic. Routing tables are periodically updated in response to changes in network topology. For larger networks, proactive protocols are not the best option because maintaining records of every node in the routing table would take a lot of time.

- **Reactive Routing Protocols:** Because there are no preset routes that are saved in reactive routing protocols, these protocols are often referred to as on-demand routing protocols. As an alternative, routes are built only as needed. When a path to a destination is unavailable, the cache is searched to see if there are any alternative paths; if not, a new route needs to be found.

- **Hybrid Routing Protocols:** A new generation of *MANET* technologies is represented by hybrid routing protocols. Both proactive and reactive routing algorithmic characteristics are present in these systems. They may be more scalable than protocols that are only reactive or constructive. Additionally, in order to address bottleneck issues, hybrid routing protocols have produced a new class of nodes and included additional characteristics including the capacity to identify individual failure spots. To do this, a set number of nodes' data transmission is allowed only in the event that the preferred route is unavailable.

1.1.3 Optimized Link State Routing (*OLSR*) Protocol

OLSR [3] is one of the most popular wireless routing protocols exhibiting comparatively better performance in *MANET*, and the classical link state routing mechanism is optimized to develop *OLSR*. Being a proactive protocol, *OLSR* guarantees prior route availability every time. The prior route availability enables it to outperform its counterpart benchmarks in terms of packet delivery ratio (*PDR*), throughput, and end-to-end delay [13, 14, 15, 16]. However, the table-driven characteristics cause *OLSR* to experience a higher routing overhead than those counterparts. Thus, the performance enhancement *OLSR* has become a highly debated research topic. This research chooses to address and improve

the *OLSR* routing overhead issue without sacrificing other performance issues, for instance, *PDR*, throughput, and delay.

Nodes in *MANET* can establish and maintain required routes through a regular or periodic exchange of *Hello* and *TC* messages. However, the rise in *TC* messages, especially in dense networks, could lead to message collisions, traffic congestion, and increased energy use, which are potential reasons for performance degradation. *OLSR* controls or optimizes the *TC* message broadcasting by permitting only the selected *MPR* nodes to forward *TC* messages. A single *TC* packet dispensed by an *MPR* node may encapsulate two or more *TC* messages, which aids in lowering the routing overhead and the likelihood of packet collision from different nodes. Thus, reducing the *MPR* set can reduce the number of *TC* messages.

The traditional *MPR* selection algorithm is unsuitable for keeping the *MPR* set small as it selects more *MPR* nodes needed to cover all possible 2-hop neighbors, resulting in many *MPR* nodes being selected for a comparatively dense network. A few heuristic solutions for selecting the best *MPR* are proposed in the literature; however, the schemes are sophisticated, challenging to use, and consume additional resources. Therefore, this work proposes an improved *MPR* selection technique covering only one-hop neighbors and effectively decreasing the number of control packets without sacrificing other performance metrics. The proposed strategy considers Euclidean distance while selecting the *MPR* and modifies the default control messages to achieve the objective.

Characteristics of *OLSR*: For mobile ad hoc networks, *OLSR* is an improvement over the traditional link state protocol. Some basic features of *OLSR* are explained bellow:

Reduced Control Traffic: By re-transmitting control messages utilizing only a subset of nodes, known as *MPR*s, *OLSR* reduces the overhead caused by flooding of control traffic. The amount of re-transmissions needed to flood a message to every node in the network is greatly decreased using this strategy. Second, *OLSR* can produce shortest path routes with just a partial link state flooding. It is necessary for all nodes that have been designated as *MPR*s to disclose their links to their *MPR* selectors as the minimum set of link state information. If there is more topological information, it Can be used, for example, for redundancy.

Greater Coverage: Through the reduction of the maximum time interval for periodic control message transmission, *OLSR* can optimize the reactivity to topological changes. Additionally, as *OLSR* consistently keeps routes to all destinations in the network, traffic patterns where a sizable portion of nodes

communicate with another sizable portion of nodes and where the [source, destination] pairs fluctuate over time are advantageous for the protocol. Because *MPR* optimization functions effectively in big and dense networks, this protocol is especially well-suited for them. In comparison to the traditional link state technique, more optimization is possible with larger and denser networks.

Discrete Architecture: Because it is designed to operate entirely remotely, *OLSR* is independent of any central authority. Reliable transmission of control messages is not required by the protocol. Every node transmits control messages on a regular basis, and as a result, can tolerate losing part of these messages. In radio networks, these kinds of losses are common because of transmission issues like collisions.

Remaining Up-to-date: The scheduled delivery of messages is not necessary for *OLSR*. A sequence number is included in every control message, and it is increased with each communication. Consequently, a control message's receiver can, if necessary, quickly determine which data is more recent, even if messages have been rearranged while being transmitted.

Flexibility: Protocol extensions like multi-cast routing and sleep mode operation are supported by *OLSR*. The protocol may be expanded with these changes without affecting compatibility with previous iterations. *IP* packet format does not need to be altered for *OLSR* to work. Since the protocol simply communicates with routing table management, any *IP* stack that already exists can be utilized exactly as is.

1.1.4 Assumptions Related to *MPR*s

By minimizing redundant re-transmissions in the same region, multi-point relays aim to reduce the overhead of flooding messages in the network. Every node within the network chooses a group of nodes within its symmetric 1-hop neighborhood that have the ability to relay its messages again. The *MPR* set of a given node is the collection of chosen neighbor nodes. While they receive and process broadcast messages from the sender node, neighbors not included in the *MPR* set do not re-transmit the broadcast messages.

Every node chooses its *MPR* set from a group of its neighbors that are 1-hop symmetric. This set is chosen to encompass all symmetric strict 2-hop nodes within the given radio range. The basic *MPR* selection strategy has been represented through Algorithm 1.

Every node keeps records of the group of neighbors that have recognized it as *MPR*. This group is referred to as a node's *MPR* selector set. A node gets this

data from the neighboring nodes' periodical *Hello* messages.

When a broadcast message, from any of node's *MPR* selectors, spreads over the whole network, it is expected to re-transmitted by that node. The selector nodes convey this set in their *Hello* messages, and it may change over time.

1.1.5 Protocol Functioning

This section describes briefly about the core functionality of *OLSR* protocol following *RFC 3626* provisions.

Standard Packet Format: The transport mechanism for all *OLSR* control traffic consists of an optimal flooding mechanism and a universal specification of the packet format, explained in Figure 4.13.

Link Establishment: By periodically sending out *Hello* messages across the interfaces used to assess connectivity, hence, link sensing is achieved. Each interface generates its own *Hello* message, which contains the details needed to establish a link.

Neighbor Identification: A node may determine the neighbor set directly from the data shared during link sensing in a network with a single interface node. To map interface addresses to main addresses in a network with many interface nodes, more data is needed and hence, exchanging of multiple interface declaration (*MID*) messages is required.

Selection of Multi-point Relays: A node must choose a subset of its neighbors, in order for a broadcast message to be received by all nodes two hops away, when it is re-transmitted by these chosen neighbors. This is known as *MPR* selection. For every interface, the *MPR* set of a node is calculated so that it meets this requirement. Through the regular exchange of *Hello* messages, the data needed to complete this calculation is obtained.

Diffusion of TC Messages: The goal of disseminating control messages is to give every node in the network enough link-state data to enable route computation.

Route Establishment: Routing tables for each node can be generated based on link-state information obtained from periodic message exchanges and the nodes' interface settings.

1.1.6 *OLSR* Terminology

The terms used in the paper are as follows:

- **Node:** A *MANET* router that carries out the protocol outlined in this paper for optimized link state routing.

- ***OLSR* Interface:** One of the network devices running *OLSR* and taking part in a *MANET*. An individual *IP* address is assigned to each of the several *OLSR* interfaces that a node may have.
- **Non *OLSR* interface:** Network device running *OLSR* that is not a part of a *MANET*. It is possible to introduce routing data into the *OLSR* routing domain from these interfaces.
- **Single *OLSR* interface node:** A node participating in an *OLSR* routing domain with a single *OLSR* interface.
- **Multiple *OLSR* interface node:** An *OLSR* routing domain is occupied by a node with numerous *OLSR* interfaces.
- **Main Address:** One node's main address serves as the "originator address" for all messages this node emits and is utilized in *OLSR* control traffic. This address corresponds to one of the node's *OLSR* interfaces. The primary address of an individual *OLSR* interface node must correspond to that of its single *OLSR* interface. One of the *OLSR* interface addresses on a multiple *OLSR* interface node must be defined as the "main address".
- **2-hop Neighbor:** A neighbor's heard node.
- **Strict 2-hop Neighbor:** A 2-hop neighbor of the node that is neither the node itself nor its neighbor; additionally, it is a neighbor of a neighbor of the node that has a willingness that differs from $WILL\_NEVER$.
- **Link:** A pair of *OLSR* interfaces that can hear each other is called a link. When one of a node's interfaces is connected to an interface of another node, the two nodes are said to be linked.
- **Symmetric Link:** A valid two-way link between two *OLSR* interfaces.
- **Asymmetric Link:** A single-direction verified link between two *OLSR* interfaces.
- **Symmetric 1-hop Neighborhood:** The collection of nodes that have at least one symmetric link to every given node, is known as its symmetric 1-hop neighborhood.
- **Symmetric 2-hop Neighborhood:** The collection of nodes with a symmetric link to the symmetric 1-hop neighborhood of X, except X itself, is known as the symmetric 2-hop neighborhood of X.

- **Symmetric Strict 2-hop Neighborhood:** The set of nodes that, aside from X and its neighbors, have a symmetric link to some symmetric 1-hop neighbor of X with willingness distinct from $WILL\_NEVER$ is known as the symmetric strict 2-hop neighborhood of X.
- **Multi-point Relay:** A node that is chosen by node X, its 1-hop neighbor, to "re-transmit" all broadcast messages that it gets from X, given that the message is unique and has a time to live field larger than one.
- **Multi-point Relay Selector:** A node that designates node X, its 1-hop neighbor, as its multi-point relay will be referred to as node X's multi-point relay selector.

1.1.7 Repositories Used

Every node gathers network awareness by exchanging *OLSR* control messages with other nodes. This section explains how this data is stored.

Multiple Interface Association: "Interface Association Tuples" consisted of I_iface_addr, I_main_addr, and I_time, are kept track of for every destination in the network. A node's interface address is I_iface_addr, and its main address is I_main_addr. The time at which this tuple expires and needs to be deleted is indicated by I_time. The "Interface Association Set" is the collection of $Interface\_Association\_Tuples$ in a node.

Link Set: $L\_local\_iface\_addr$, $L\_neighbor\_iface\_addr$, L_SYM_time, L_ASYM_time, and L_time define "Link Tuples" that are recorded by a node. The interface address of the local node is denoted by $L\_local\_iface\_addr$.

L_SYM_time, defines the time until which the link is considered symmetric while L_ASYM_time indicates the time until which the neighbor interface is considered heard; and L_time specifies the time at which this record expires and needs to be removed. $L\_neighbor\_iface\_addr$ is the interface address of the neighbor node. The link is deemed lost when both L_SYM_time and L_ASYM_time pass their expiration dates.

Neighbor Set: A node keeps track of a collection of "neighbor tuples" that describe its neighbors: $N\_neighbor\_main\_addr$, N_status, and $N\_willingness$. The primary address of a neighbor is $N\_neighbor\_main\_addr$, and N_status indicates if the node is the node's readiness to carry traffic on behalf of other nodes is indicated by $N\_willingness$, an integer between 0 and 7, rather than NOT_SYM or SYM.

2-hop Neighbor Set: The defined "2-hop tuples" ($N\_neighbor\_main\_addr$, N_2hop_addr, N_time) that a node records describe symmetric links between

its neighbors as well as the 2-hop symmetric neighborhood. N_2hop_addr is the main address of a neighbor with a symmetric link to $N\_neighbor\_main\_addr$; N_time is the time at which the tuple expires and must be deleted; and $N\_neighbor\_main\_addr$ is the main address of a neighbor. The set of 2-hop tuples in a node is called the "2-hop Neighbor Set".

***MPR* Set:** A node keeps a list of neighbors that have been chosen for maximum proximity. The *MPR* Set contains a list of their primary addresses.

***MPR* Selector Set:** A set of *MPR*-selector tuples (MS_main_addr, MS_time) that describe the neighbors that have chosen this node as an *MPR* are recorded by a node. The main address of a node that has been designated as *MPR* is MS_main_addr. The time when the tuple expires and needs to be deleted is specified by MS_time. The "*MPR* Selector Set" refers to the collection of *MPR*-selector tuples in a node.

Topology Set: Every network node keeps track of the network's topology. Routing table computations use this data, which is collected from *TC*-messages. Consequently, at least one "Topology Pair" (T_dest_addr, T_last_addr, T_seq, T_time) is stored for every destination in the network. A node with the main address T_dest_addr can be reached in a single hop from the node with the main address T_last_addr. T_last_addr is typically an *MPR* of T_dest_addr. The sequence number T_seq and the time T_time indicate when this tuple expires and needs to be deleted. The "Topology Set" refers to the collection of Topology Tuples within a node.

1.2 System Model and Assumptions

This section commences by briefly picturing the working procedure of the classical *OLSR* algorithm. The problems identified in the *MPR* selection process of the default algorithm tend to introduce a more efficient strategy for *MPR* selection.

1.2.1 Network Topology

OLSR enables proactive routing to determine the best path by spreading various types of control messages such as *Hello*, *TC*, *MID*, and *HNA*. The *MANET* nodes exchange neighbor and routing information through the control messages. The nodes utilize the control packets to build and keep the topology information in their routing tables. The network topology in Figure 1.3 illustrates the proposed *MPR* selection technique where data from a sender finds the best paths to the given destinations.

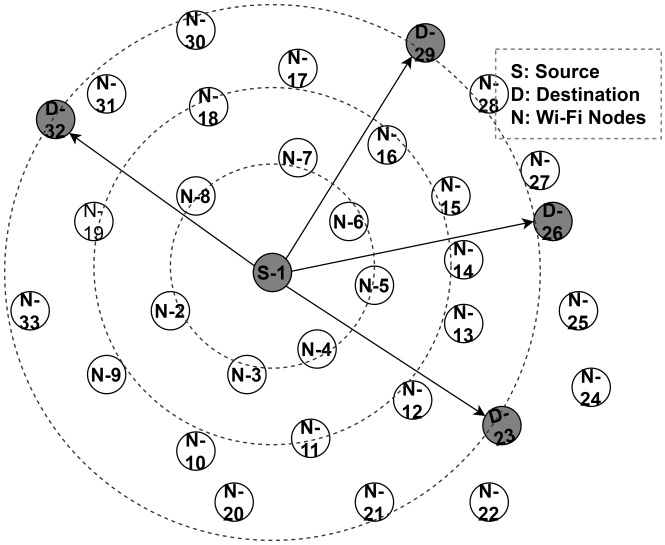


Figure 1.3: Network topology.

1.2.2 Existng *MPR* selection strategy used by *OLSR*

(*MPRs*) nodes are vital to reduce the dissemination of control messages (*TCs*). The classical *MPR* selection algorithm is heuristic in manner [3] where a node (u) needs to maintain its one-hop and two-hop neighbor sets, denoted as $N(u)$ and $N2(u)$, respectively. $N2(u)$ includes nodes reachable by the members of one-hop neighbors $N(u)$, and whose willingness is not $WILL\_NEVER$. Each node maintains the "willingness" parameter, an integer value that ranges from 0 to 7, indicating its eagerness to forward traffic on behalf of other nodes. Any node not interested in forwarding traffic for other nodes, such as because of resource limitations, is indicated by $WILL\_NEVER(0)$. $WILL\_ALWAYS(7)$ denotes that a node is always ready to carry traffic on behalf of other nodes, for instance, because resources are adequate. By default, every node has the willingness set to $WILL\_DEFAULT(3)$. When any node y is a member of $N(u)$, its degree is denoted as $D(y)$. $D(y)$ defines the number of symmetric neighbors of node y, omitting any other nodes that are also members of $N(u)$, and the node u doing the computation.

The detailed classical *MPR* selection algorithm has been given in Algorithm 1.

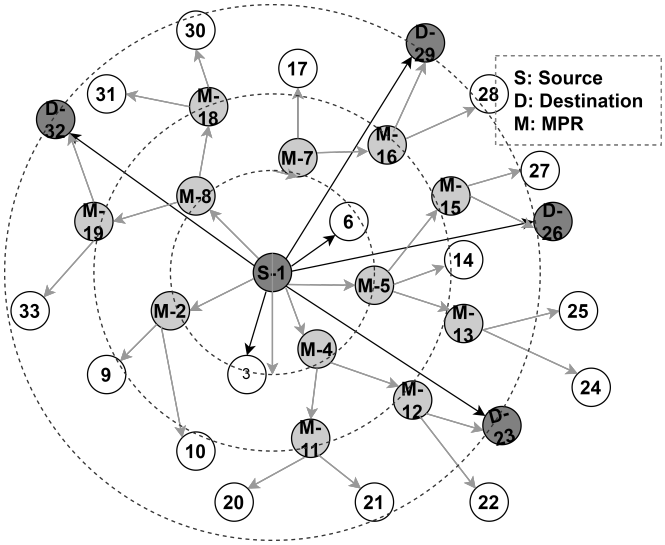
1.3 Motivation

The classical *MPR* selection algorithm explained in section 1.2.2 results in many *MPR* nodes being selected for TC dissemination. Here, all the two-hop neighbors need to be covered by *MPR* nodes. However, the proposed methodology selects

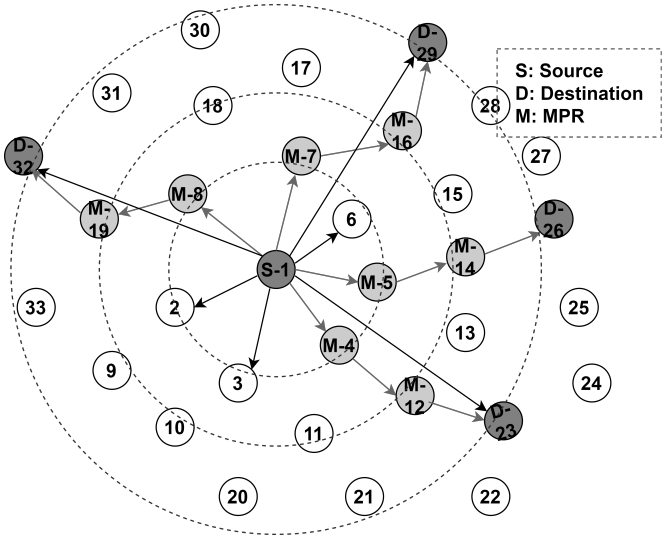
Algorithm 1: Classical *MPR* selection strategy defined in *RFC 3626* [3]

1: Start with $MPR(u) \leftarrow N(u)$ where willingness of $y \in N(u)$ is $WILL\_ALWAYS$
2: Compute $D(y)$ for all $y \in N(u)$
3: **for** Each $y \in N(u)$ **do**
4: **if** y is the only node to reach some $w \in N2(u)$ **then**
5: Add y to $MPR(u)$ and Remove w from $N2(u)$
6: **end if**
7: **end for**
8: **while** $N2(u)$ remains not empty **do**
9: **if** Only $y \in N(u)$ has highest reachability and willingness for some $w \in N2(u)$ **then**
10: Add y to $MPR(u)$ and Remove w from $N2(u)$
11: **if** More $y \in N(u)$ with same reachability and willingness **then**
12: Find $y \in N(u)$ where $D(y)$ is maximum
13: Add y to $MPR(u)$ and Remove w from $N2(u)$
14: **end if**
15: **end if**
16: **end while**
17: Integrate $MPR(u)$ for all interfaces of u

only those nodes as *MPR* needed to obtain optimal paths toward the destinations. The *MPR* and route selection scenarios of the classical and proposed algorithms are pictorially presented in Figure 1.4. A node y in the proposed technique uses a heuristic function to select *MPR* nodes from its one-hop neighbor set, $N(u)$, explained in 3.2. Each node selects the lowest cost node from its $N(u)$ neighbor set as the *MPR* node for a particular destination node. The same *MPR* node can be selected for multiple destinations if it costs the lowest for each destination node. Only nodes that reside along the optimal path are selected as *MPR* nodes in this process. Therefore, the number of *MPR* nodes can be drastically reduced by pruning unnecessary or sub-optimal paths toward the destinations. If n and $|MPR(y)|$ represent the number of sinks and *MPR* nodes of y, respectively, then $|MPR(y)| \leq n$ for each node, y. In contrast, in classical *OLSR*, $|MPR(y)| \propto N2(u)$. Thus, the number of *MPR* nodes selected in the proposed strategy is not dependent on the $N2(u)$ set, rather it leans on the number of sinks resulting in a smaller-sized *MPR* set.



(a) Existing *OLSR*.



(b) Proposed *OLSR*.

Figure 1.4: *MPR* selection scenario of both existing and proposed *OLSR*.

1.4 Challenges

Despite its enormous popularity, *MANET*s' special qualities provide a number of difficulties that must be properly taken into account in order to predict significant commercial installations. But these difficulties also provide room for creative routing approaches.

- **Nodes' Mobility:** Because nodes can switch at random, the network architecture, which typically consists of several hops, can also change abruptly and spontaneously. It can also include both one-way and two-way connections.

- **Communication:** Routing packets between nodes is challenging with

*MANET*s because of the dynamic topology. Most protocols make advantage of reactive routing. Considering the Multi Casting Tree is no longer static due to the unpredictable mobility of the nodes on the network, multicast routing presents additional challenge. Multiple hop routes have the potential to be more intricate than just one hop of interaction between nodes.

- **Updating routes:** In order to enable an ideal route selection to be automatically supported, this involves defining and notifying the network and the nodes about recently transferred nodes that need dynamic updating.
- **Quality of Service (*QoS*):** Delivering various service quality levels in a continuously changing environment is a difficult task. It is challenging to offer the services offered to a device with specified assurances because of the stochastic nature of *MANET*s. Resource reservation must be used to establish a *QoS* adaptable protocol to accommodate multimedia facilities.
- **Routing Overhead:** As *OLSR* is a proactive routing protocol, it needs to disseminate a large number of control packets, hence, routing overhead arises.

1.5 Contributions

The key contributions of this paper can be summarized as follows:

- The size of *MPR* set is reduced since only the lowest cost node/s in the first-hop neighbor is considered as the *MPR* node/s.
- The same *MPR* node can be used for multiple destinations if it is the lowest-cost node for each destination.
- Only the default control messages are extended to realize the proposed strategy.
- The proposed strategy is contrasted against the default *OLSR* in terms of *PDR*, throughput, delay, and overhead, by varying the number of nodes and pause time.

1.6 Objectives with Specific Aims and Outcomes

With minimal network traffic, the goal of this work is to route packets to certain multi-destinations in mobile ad hoc networks. To get the intended outcomes,

certain modifications have been introduced to the conventional *OLSR* algorithm. The specific objective of our work is summarized below:

- To reduce the total number of *MPR* nodes in the network in order to lessen unnecessary packet flooding.
- To disseminate packets across destinations while simultaneously choosing the best routes for each of them.
- To avoid all non-optimal paths in the network.
- To expeditiously facilitate routing in large-scale networks.
- To reduce control packets without degrading the routing performance.

Possible outcomes from this research are listed below:

- A heuristically cost function has been introduced for selecting best routes.
- *MPR* set has been reduced as much as possible without degrading the route quality.
- To reduce routing overhead, network traffic has been minimized.

1.7 Organization of the Report

The remaining part of this paper is organized as follows. Chapter 2, reviews the related literature of different optimizations in existing *OLSR*. Chapter 3.2 presents the working methodology related to the minimization of *TC* message dissemination as well as *MPR* selection. Chapter 4 demonstrates the simulation results, and finally, chapter 5 concludes the paper.

Chapter 2

Literature Review

Several different sorts of research have been done in the last few decades to enhance the *OLSR* protocol's functionality on *MANET* networks. For enhancing the performance, researchers have focused more attentions in *MPR* selection strategy to reduce routing overhead in the network. Being a proactive protocol, *OLSR* maintains route quality and experience lower latency than their reactive counterparts, such as *DSR* and *AODV*, as routing information is available anytime. However, the proactive protocols show deteriorated performance regarding routing overhead [17].

2.1 Related Work

This section explores past efforts that made similar contributions to several *OLSR* routing schemes in ad-hoc networks. The major contributions to the *MPR* selection process were concentrated on a few critical enhancements to routing performance, such as end-to-end latency, control overhead, throughput, energy efficiency, security concerns, etc. Major contributions to the selection of the *MPR* set are categorized in this section.

2.1.1 *MPR* Selection for Enhancing Performance Metrics

This section enlisted some contributions on *MPR* selection, considering routing overhead, end-to-end delay and throughput.

Many studies have been conducted on the original *OLSR* for *MPR*s selection, which has demonstrated good local properties. However, this does not provide information regarding the qualities of the global set of *MPR*s. The authors of this paper [18], introduce a new process of choosing *MPR* nodes, named *M-OLSR*, by giving higher priority to nodes that are more stable in terms of energy and

mobility. The objective of this approach is to improve overall network performance by incorporating a mobility metric into the traditional *MPR* selection procedure. Based on the mobility degree captured or the node with the largest residual energy, this protocol gives priority to less mobile candidate *MPR* nodes. The drawback of this strategy is that, depending on the flow of motion around the node, the parameter (coefficient of flow) must be fixed between three values (0.25, 0.5, and 0.75). *M-OLSR* does not, however, adequately reduce the routing overhead.

In [19], Maccari et al proposed a new strategy called "Selector Set Tie Breaker" (*SSTB*) for minimizing the global *MPR* set (the union of all the *MPR* sets). Prior to implementing the initial tie-break [20], they include an additional step that essentially favors the node with the greatest number of selectors among *MPR*s and the node that is already an *MPR* for another node. However, this mechanism reduces the number of *MPR* set compared to original *OLSR*, without considering other performance metrics.

To pick reliable multi-point relays with appropriate residual batteries and high-quality links for interconnections, an appropriate multi-point relay selection algorithm has been presented in [2]. In the selecting phase, the algorithm additionally considers the nodes' degree of willingness, reach-ability, and relative mobility. Based on the optimal values of willingness, trustworthiness, residual battery capacity, link quality, mobility factor, reach-ability, and degree, the suggested suitable *MPR* selection (*SMS*) algorithm (Figure 2.1) chooses the most acceptable neighbors for the *MPR* set.

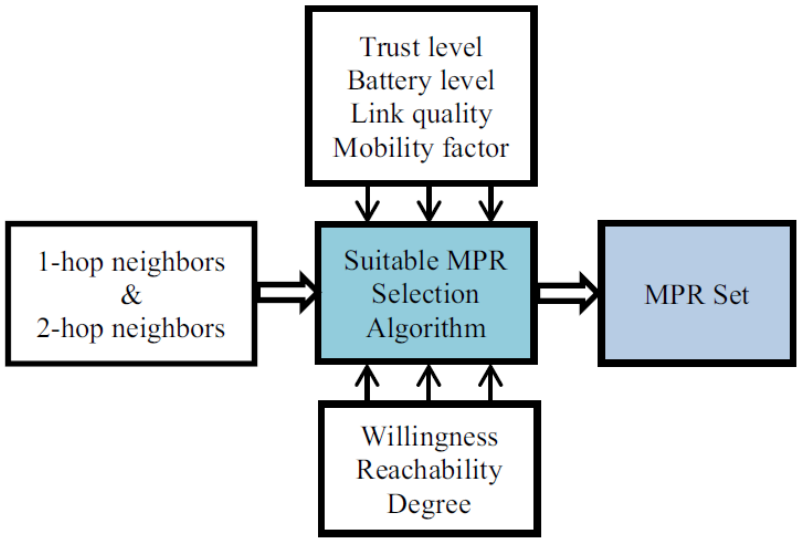


Figure 2.1: Flexible *MPR* selection mechanism [2].

The algorithm's possible limitations in real-world deployments include its re-

liance on the physical layer, its inability to obtain information on some nodes, and its ability to mask the trust parameters.

S.Dong et al, in [21], presents a new *MPR* selection mechanism based on set operation. It is possible to successfully remove invalid superfluous nodes by combining set operations and cyclic operations. This approach is more efficient and has less data overhead than the conventional *OLSR* protocol, while still achieving the same result.

Although this approach increases the success probability of data transfer, it also marginally increases delay and increases control traffic.

By reducing the transmission's hop count, the proposed *MMPR* approach [22] focuses on making effective use of bandwidth. The primary goal of the suggested protocol (Figure 2.2) is controlled and efficient transmission with efficient channel utilization. The protocol continues to identify the two-hop neighbor nodes covered by the selector nodes after choosing the first *MPR* set. In order to determine which nodes will cover the maximum number of two-hop neighbor nodes, the remaining one-hop neighbor nodes—which are not included in the *MPR* selection set—are analyzed. This procedure is continued until a minimal set of relay nodes (*MMPR*s) covers every two-hop neighbor node. *OLSR* with *MMPR* outperforms *AODV* in throughput, demonstrating the efficacy of the suggested methodology. However, the *MMPR* technique may result in issues such as hidden terminal issues, transmission failures, and accidents.

AOLSR, a protocol proposed by P Kumar et al. [23], offers greater *MPR* selection criteria optimization. Less overhead is accomplished by placing the *MPR* node on either the left or right side of the sender node (Figure 2.3), depending on where the destination node is located. Though this protocol works well in terms of packet delivery ratio and throughput, routing overhead increases for different node speed.

In [24], the proposed mechanism extends the visibility to a three hop node (Figure 2.4) when two nodes that are one hop apart have the same degree and reach-ability. The scheme's main goal is to provide nodes with additional options for choosing the optimal *MPR* set, even in situations when certain conditions such as the same reach-ability and degree exist. Therefore, the quantity of *TC* messages transmitted may be lower if the chosen *MPR* nodes have a higher number i.e. absorbed degree of neighboring nodes that absorb *TC* messages. A node that does not broadcast the *TC* messages it receives from its neighbor—a candidate for *MPR*—is one that absorbs those messages. However, considering higher degrees of reach-ability for *MPR* selection needs to maintain more information, that may slower the node processing time.

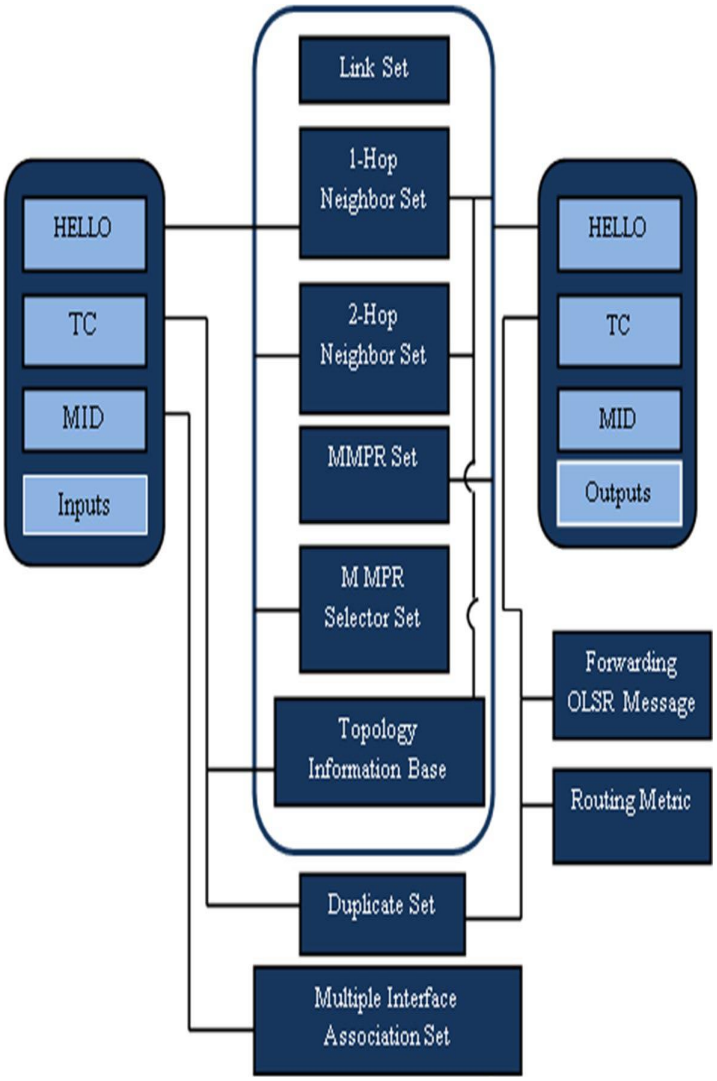


Figure 2.2: Architecture of *OLSR* protocol with refined *MMPR*.

S. Sharma et al [25] proposed a position-based *OLSR* for ad-hoc networks in order to minimize the control overhead. By modifying the *OLSR* protocol, this article fixes the problems with the neighbor prediction scheme. Each node can determine whether or not its neighbor is within radio range because the neighbor's position may be anticipated at any time. As a result, nodes in *P-OLSR* only consider radio-range neighbors while calculating *MPRs*. The neighbor prediction method presumes that nodes move in the same direction and at the same speed, which may not be possible for all situations.

D Zhang et al [26] proposed a quantum-genetic-based modified *OLSR* protocol to reduce the redundant information in *MANET*. According to an improved

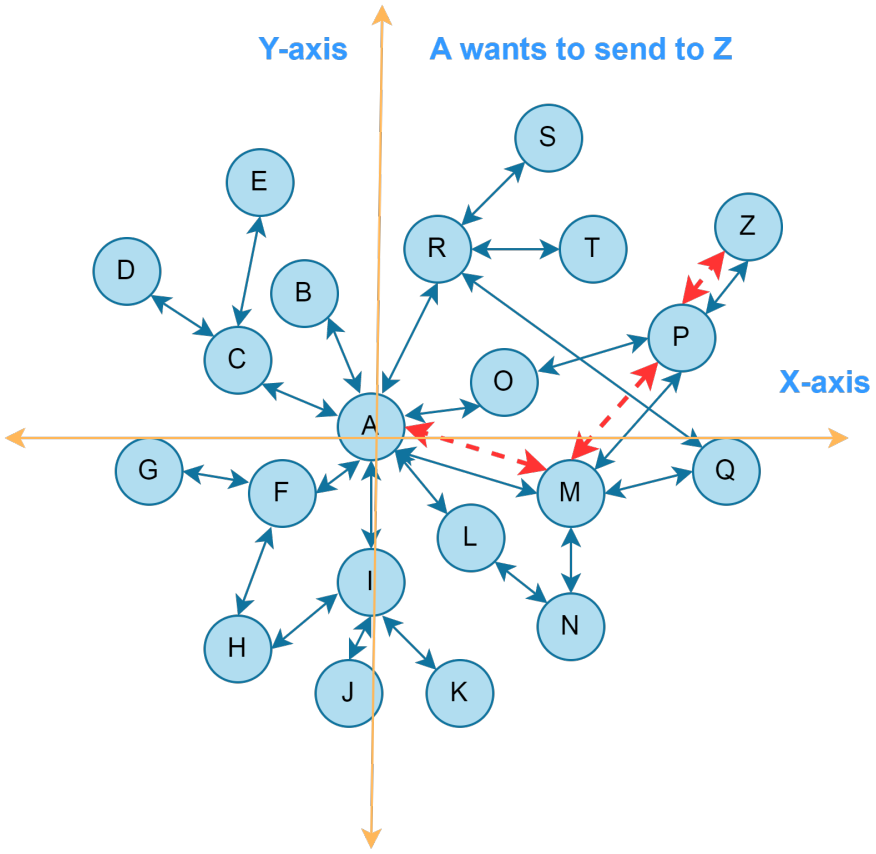


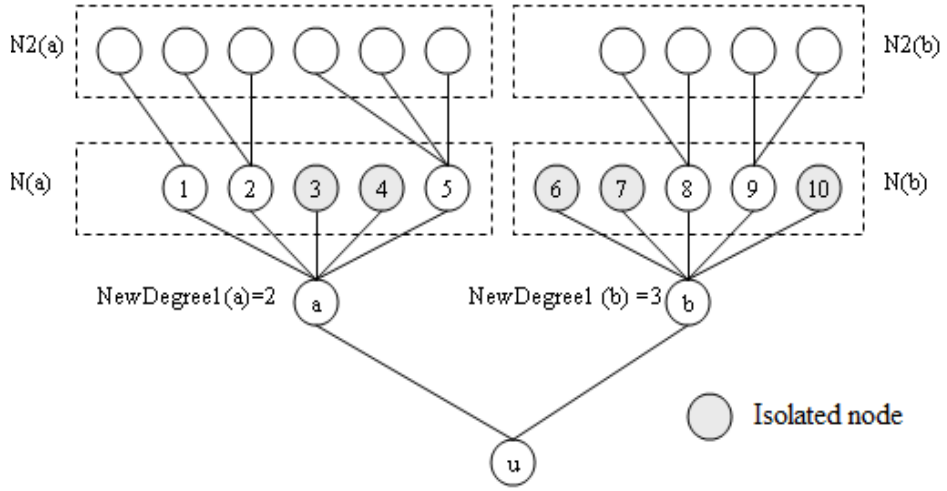
Figure 2.3: *MPR* selection in *AOLSR*.

version of the quantum genetic algorithm, they introduced a new *MPR* selection scheme in which a newly designed Q-Learning technique has been adopted, and nodes are encoded by the quantum gene bit. A heuristic node fitness rule has been followed to select a small *MPR* set for each node. In this paper, network control overhead drastically increases with network size.

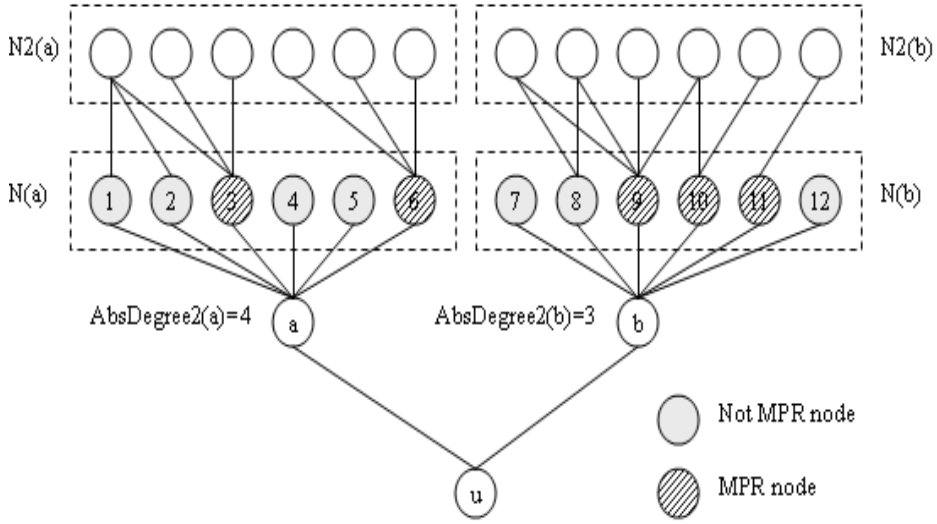
In [27], P Kumari et al proposed a swarm based hybrid *ACO PSO* meta-heuristic (*HAPM*) routing protocol to ensure routing in large and dynamic ad hoc network. To increase *QoS* restrictions and reduce *QoS* data dropping, this protocol combines *ACO*, *PSO*, and a dynamic queue mechanism. Although this protocol works well in large scale dynamic environment, routing overhead has not been reduced up to the mark.

An ant colony based improved routing protocol has been suggested by Y Sun et al in [28].This paper introduces a new route updating rule that, by optimizing the heuristic function as well as considering the path length of communicating base stations, distance from the sink nodes, direction of transmission and remaining energy as a whole, yields a higher average residual energy level.

Z Yihui ei al, in [29], proposed Node-Status Self-Sensing *OLSR* (*N3S-OLSR*)



(a) Computing the Absorbed Degree based on isolated neighbor's nodes.



(b) Computing the absorbed degree based *NotMPR* neighbor's nodes.

Figure 2.4: Example of computing the absorbed degree.

routing protocol to calculate *MPR* sets by combining the information of self-sensing with hop count. Since the node residual energy information has been added, nodes with lower remaining energy can be selected as *MPR* nodes according to this protocol. However, the proposed protocol degrades its performance related to packet delivery ratio and overhead in high-speed mobility scenario.

The researchers have made a number of enhancements to increase the performance of *OLSR* protocol considering routing overhead [24, 30, 31, 32], energy consumption [33], network lifetime [34], packet delivery ratio [35, 36], throughput [37, 38] etc. Some other researches have been summarized in Table 2.1.

2.1.2 Energy Efficient *MPR* Selection

In order to facilitate energy-efficient routing, this section presents various prior *MPR* selection mechanisms.

To provide the best routing for a safe and energy-efficient *FANET*, [43] suggested the whale optimization algorithm based optimized link state routing (*WOA-OLSR*). Using both of single or multi-key encryption and *WOA* algorithm, *WOA-OLSR* enables both security and energy efficiency over flying ad hoc network, though, time complexity arises to $O(n^4)$.

In response to the growing concerns over security and energy efficiency, [44] presents a combination energy-saving scheduling and secure routing algorithm (Figure 2.5) for important event reporting. To optimize routing security while minimizing power consumption in the up-link, a joint power allocation and secure routing technique (*JPASR*) (Figure 2.5a) has been proposed. The backbone nodes are chosen to broadcast messages in the down-link using an energy-first multi-point relays set selection mechanism (*EFMSS*) (Figure 2.5b), and they are woken up using the same level-by-level sleeping scheduling technique used for the up-link transmission. Nodes are woken up layer by layer as the sink node sends the control message to the entire network via the *MPR* set selected by *EFMSS* during the 2k-th duty cycle.

In [45], Jabbar et al extends the standard *MP-OLSRv2* protocol and suggests a novel energy and mobility-aware multi-point relay (*EMA-MPR*) selection method to boost route stability, lengthen node lifetime, and enhance *QoS*. In order to ensure the optimal number of nodes added to the *MPR* set, it alters the willingness setting of the conventional *MPR* selection process. It demonstrates good performances in terms of packet delivery ratio, delay, and throughput in different speeds; however, comparison based on different network sizes has not been taken into consideration. The suggested *EMA-MPR* selection mechanism chooses the *MPR* set among the most stable nodes in terms of energy reserves and mobility.

Some other energy efficient relay selection techniques are explained in [46, 47, 48] with outstanding performance enhancement.

Table 2.1: Related Work.

Author name and reference	***Contribution***	***Limitation***	***Simulator***
S Dong et al [21], (2021)	Reduces flooding of control packets by combining cyclic and set operations	Delay has been increased slightly	OPNET
NM Al-Kharasani et al [39], (2020)	The *CACA* algorithm is added to the *MPR* scheme to enhance its capacity for maintaining long-lived routes	Degrades performance when number of node is high	NS2
M Al Mojamed et al [40], (2019)	Discovered a new *MPR* selection strategy of *OLSR* protocol for lightweight *MANET*	Optimizes *OLSR* without considering routing overhead	OMNeT++
M Usha et al [41], (2019)	Enhances *OLSR* combining *GSA-PSO* and cognitive radio technique	Routing overhead hasn't been considered	NS2
IKE Purnama et al [42], (2018)	Presented a new *MPR* selection mechanism based on min-max algorithm	Decreases *PDR* & throughput with increasing node number	NS2

2.1.3 Secured *MPR* Selection Techniques

Being a proactive protocol, *OLSR* allows the *MPR* nodes to optimize the influx of control messages into the network. This privilege can be exploited by malicious nodes to disseminate false topological information—a tactic known as an *MPR* attack. The attack has an impact on the *MANET*'s overall performance. Researchers have designed several *MPR* selection mechanisms, considering these security issues. This section enlisted some of the works done in earlier.

For the Quality-of-Service Optimized Link State Routing (*QoS-OLSR*) protocol in urban *VANET*s, a blockchain-enabled Stackelberg game model is put out in [49]. According to the game model, nodes are categorized as leaders (relays) and followers (other nodes) according to reputation and quality of service metrics that are shared via protocol messages. In order to improve the off-chain implemented *QoS-OLSR* protocol for *VANET*, which allows for trusted relay selection, end-to-end incentive payments for relays, and verification of nodes' exchanged reputations, blockchain is incorporated. Figure 2.6 shows the interactions that take place in the suggested Stackelberg game model with blockchain support.

To enhance the identification of malicious nodes in VANETs, in [50], the authors have suggested a trusted routing method based on fuzzy logic and blockchain technology. A routing protocol called *FT-OLSR* enhances the security of communications within the *VANET*. In *FT-OLSR*, the process of finding black holes is carried out reciprocally at the vehicle level, wherein vehicles that stop sending *HELLO* and *TC* signals are identified and communication links are confirmed. Messages from the black hole node are ignored and not processed once it is recognized, which means that these nodes are no longer eligible to be elected as *MPR*s. By enhancing collaboration between *VANET* components in a dynamic setting with constrained resources, blockchain technology has been utilized to isolate threatening vehicles identified by *FT-OLSR* (Figure 2.7) and do away with laborious computations.

In [51], the authors proposed a new *MPR* selection mechanism, to enable *OLSR* performing against single black hole attack. A single black hole attack, which lowers routing protocol performance in mobile ad hoc networks by deleting any routing packets, is one of the most prevalent types of routing attacks nowadays. In order for its 1-hop neighbors to choose the malicious node as the *MPR* node based on the transparency employed by the *MPR* selection process, the malicious node in the *OLSR* routing protocol distributes fictitious *HELLO* messages around the network.

The purpose of this paper is to authenticate the existence of each 2-hop neighbor declared by sending an ACK_HELLO routing control message when each

2-hop neighbor successfully receives the forwarded HELLO message with *TTL* equal to 1. Prior to initiating the *MPR* computation, a new algorithm for the *MPR* selection process has been introduced, that removes all potential points of failure utilized in the single black hole attack, thereby isolating the attacker as an *MPR* node.

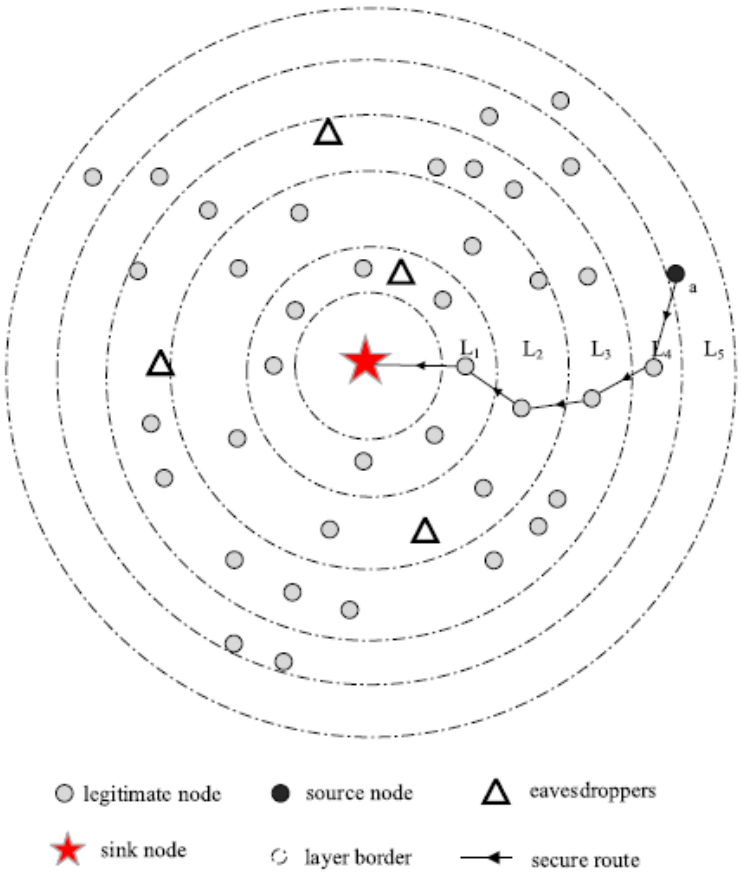
Other security-related tasks, like, reducing the number of lost connections [52], fostering confidence in wireless communication [53], preventing hostile nodes [54], encrypting control messages [55], and so forth, improve the security of *OLSR*.

2.2 Research Gap

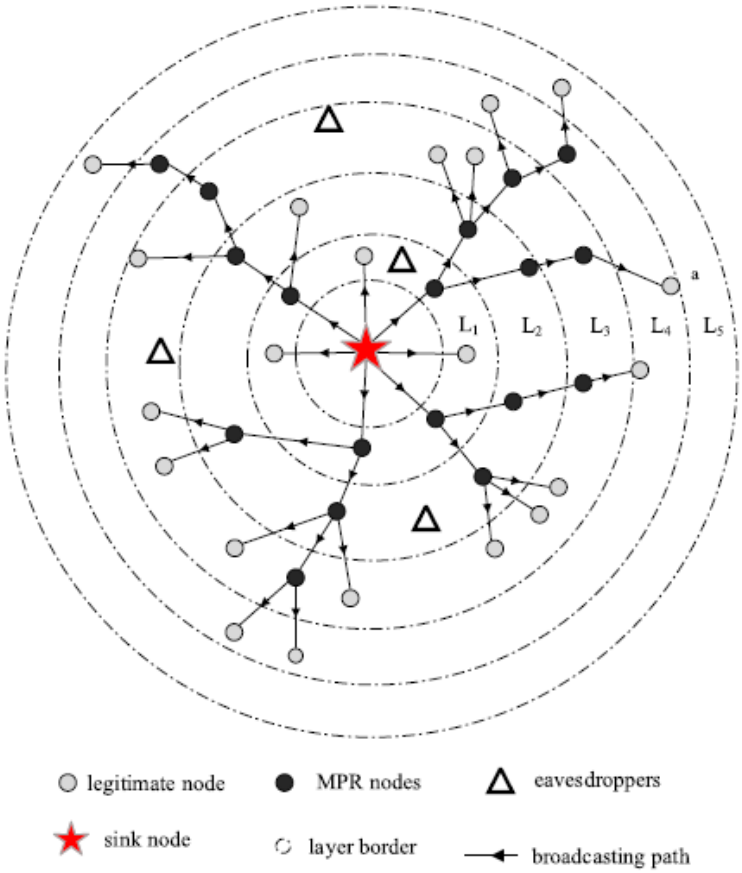
The majority of previously referenced works for improving the earlier *MPR* selection strategy defined in standard *OLSR* protocol, which increases the number of chosen *MPR* nodes as well as introduce more complexities. Large *MPR* sets, however, will disseminate more control packets, which will raise the network's routing overhead. Furthermore, the volume of the chosen *MPR* set, total *TC* propagation, *TC* size, total sent control messages, the size of the total sent control messages, and other performance metrics were not entirely compared in the majority of the previously mentioned research. Therefore, the key research questions we are investigating in this work are-,

- How to deliver a message to certain multiple destinations keeping a reduced *MPR* set as well as less processing time?
- How can we maintain only required optimal links except maintaining all possible links for efficient routing?

To overcome these issues, we have applied heuristic concepts in *MPR* selection process which has been able to choose less number of *MPR* nodes and less *TC* message propagation compared to standard *OLSR* as well as *SSTB* protocol without degrading other performances.



(a) Uplink: event detection and report in the (2k-1)-th duty cycle.



(b) Downlink: notification broadcast in the 2k-th duty cycle.

Figure 2.5: *MPR* selection of *JPASR* and *EFMSS*.

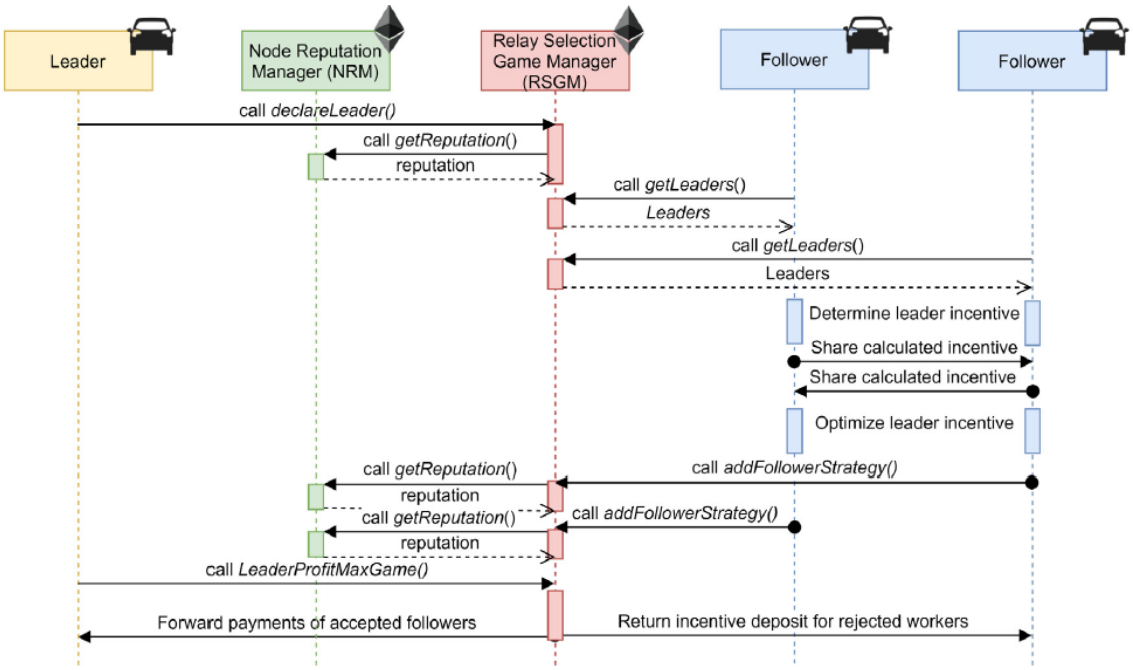


Figure 2.6: Time sequence diagram of blockchain enabled Stackelberg game model.

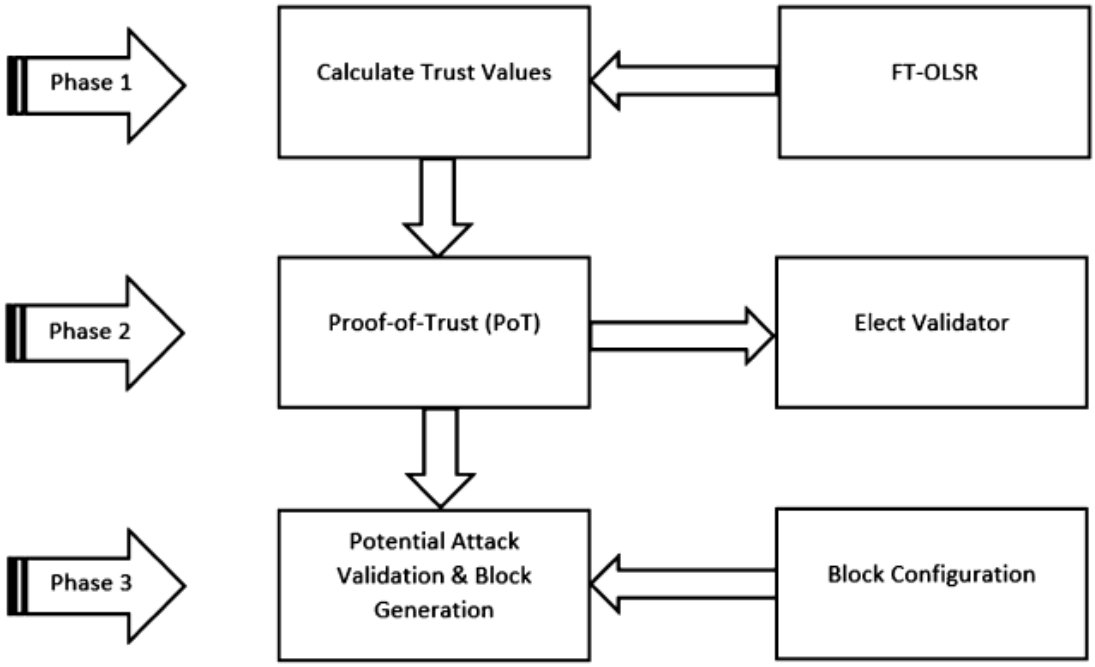


Figure 2.7: System design of blockchain based *FT-OLSR*.

Chapter 3

Methodology

3.1 Problem Formulation

3.2 Proposed Architecture

This section introduces the needed modifications of *Hello* and *TC* messages to execute the proposed technique. The modifications and the *MPR* selection strategy collectively aid in lowering the number of *MPR* nodes to diffuse fewer *TC* messages.

3.2.1 Extended *Hello* Message Format

As nodes' locations are at the heart of the proposed *MPR* selection process, every node must know its neighbors' and destination nodes' locations. In this study, each node is assumed to equipped with a *GPS* receiver to obtain its location information; longitude and latitude positions. A node maintains and shares its neighbors' and destination locations by broadcasting periodic *Hello* Messages. A new table, named *Dest_Table* (Figure 3.3), is introduced to maintain the destinations' location information. In addition, the default neighbor table (Figure 3.1) is extended by adding two fields to store neighbors' location and node costs. Figure 3.2 exhibits the proposed *Hello* message to accommodate the location information.

Location (X) and *Location* (Y) represent the longitude (X) and latitude (Y) co-ordinates, respectively of the sender node. A node retrieves its neighbors' location information once a Hello is received. The $NodeCost$ field is used to share the link cost established for each neighbor node. $NodeCost$ is calculated using Eq. 3.3 as explained in section 3.2.9. $IsDest$ represents a Boolean value that determines whether the *Hello* message's sender is a destination. $DestMsgSize$

contains the size of *Dest_Table* of the sender node. This field helps a receiver node to store sender's *Dest_Table* related information. The information of each tuple in *Dest_Table* is shared through *DestinationLocation* (X), *DestinationLocation* (Y), and *DestinationInterfaceAddress* fields, respectively. the rest of the fields are similar to the original *Hello* message format.

***Hello* Message Overhead:** The proposed *Hello* message format (Figure 3.2), discussed in section 3.2.1, contains some additional fields for sharing location related information. From this figure, it shows that *Hello* is expanded by 24 bytes more than the classical format. Although *Hello* size increases, this problem has been counteracted by disseminating less number of total sent messages (*Hello* and *TC*) compared to the other protocols, presented in Figure 4.4.

Neighbor Main Address (32 bit)	Status (2 bit)	Willingness (8 bit)	Node Cost (16 bit)	Neighbor Location (32 bit)

Figure 3.1: Extended neighbor table format.

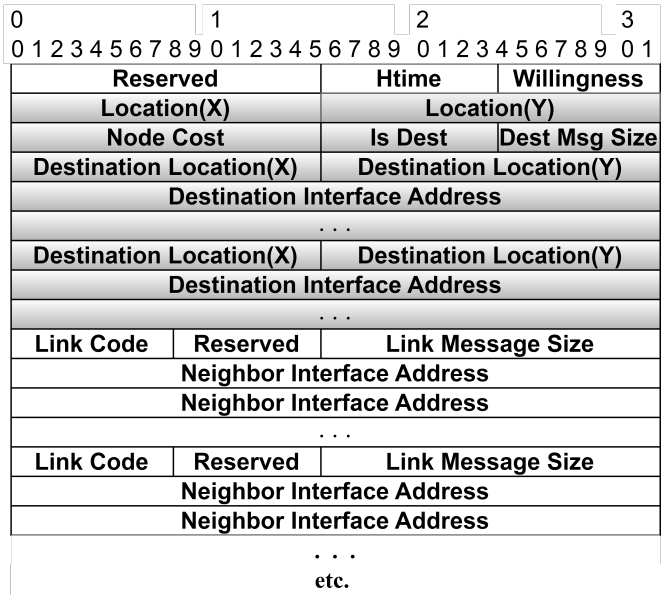


Figure 3.2: Extended *Hello* message format.

3.2.2 Proposed Table Formats

The *MPR* selection technique is realized by each node maintaining three new tables named *Dest_Table*, *MPR_Table*, and *Cost_Table*. The tables' purposes are described in the following sections.

3.2.3 *Dest_Table* Format

Dest_Table stores information related to the specified destinations, as represented in Figure 3.3. A *Hello* message uses the table's information to broadcast destination-related information. Also, the table is used for *MPR* calculation. *IPv4* address and location collected via the exchanges of *Hello* messages. When a node receives a *Hello* message, it first determines whether the sender is a destination node by inspecting the $IsDest$ field of the *Hello* message. If the sender is the destination node, it updates its *Dest_Table* with the destination address and location. The node later shares the destination information by broadcasting *Hello* messages to its neighbors. The process continues, and each node is informed about the destinations once the network converges.

Destination Node Address (32 bit)	**Destination Node Location** (32 bit)

Figure 3.3: *Dest_Table* format.

3.2.4 *MPR_Table* Format

This table consists of five fields as represented in Figure 3.4. A node's $MPRSelector\ Address$ field stores the *IPv4* address of the node that has selected it as the *MPR*. $DestinaitonAddress$ and $DestinationLocation$ fields refer to the information of a destination node for which this node has been selected as *MPR*. $Cost$ and $NodeCost$ fields store the total cost (Eq. 3.1) and link cost (Eq. 3.3) between the selector node and the node itself. The cost calculation process is given in section 3.2.9. A node may update its MPR_Table once it receives a TC message from its neighbors. Since the source node cannot be selected as *MPR* node, its $MPRSelectorAddress$ field always contains $NULL$ value, or equivalently "0.0.0.0". Initially, the source nodes create a separate tuple in their *MPR* table with $MPRSelectorAddress$ = "0.0.0.0". MPR_Table enables a node to know if it is a MPR node or not. This table, also, decides TC generation. A node runs Algorithm 4 in association with Eq. 3.1 to select the next *MPR* node using the table entries.

MPR Selector Address (32 bit)	**Destination Address** (32 bit)	**Cost** (8 bit)	**Node Cost** (From Source) (8 bit)	**Destination Location** (32 bit)

Figure 3.4: MPR_Table format.

3.2.5 *Cost_Table* Format

This table (Figure 3.5) stores the next selected *MPR* information. For example, if a node b is selected as an *MPR* by node a for a particular destination node c, then $DestNodeAddress$ and $NextNodeAddress$ fields are populated by c and b, respectively. *Cost* field stores the cost-related information (Eq. 3.1) for selecting the next *MPR* node. A node updates its $Cost\_Table$ utilizing the information stored in $Neighbor\_Table$ and MPR_Table using Algorithm 4.

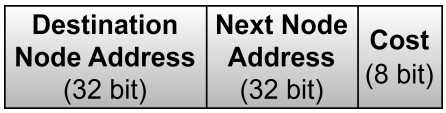


Figure 3.5: $Cost\_Table$ format.

3.2.6 Extended *TC* Message Format

This section introduces the modified *TC* message as given in Figure 3.6. Only the *MPR* nodes generate *TC* messages containing the information stored in $Cost\_Table$, MPR_Table and $Neighbor\_Table$. A node shares its selected *MPR* set with its neighbor nodes through TC. A neighbor node receiving the TC message updates its MPR_Table if its *IPv4* address is piggybacked in this message. TC modification or extension increases its size; however, the demerit is counteracted by reducing the number of MPR nodes (and hence TC messages). The sender node shares its own *IPv4* address, and the *MPR* set through $MPRSelectorNodeAddress$ and $MPRNodeAddress$ fields, respectively.

$DestinationNodeAddress$ contains the address of the destination node for which *MPR* has been selected. $NodeCost$ field contains the cost between the sender node and the selected *MPR* node, and *Cost* field contains the total cost to select an *MPR*.

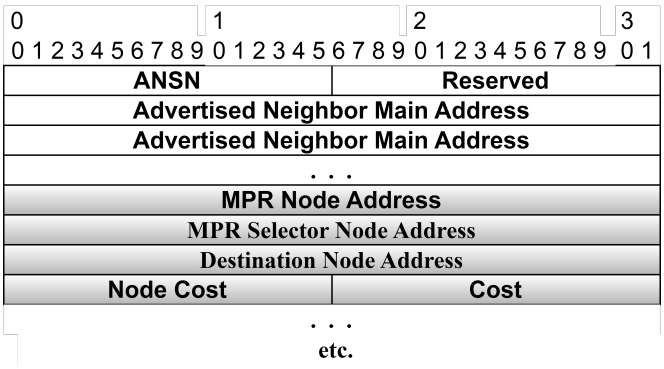


Figure 3.6: Extended *TC* message format.

3.2.7 *TC* forwarding Technique

TC message can only be generated and circulated by the selected *MPR* nodes and the source node. This can be implemented by checking the size of MPR_Table i.e. $|MPR\_Table|$ for each node. If $|MPR\_Table| \neq \varnothing$, only then it can send *TC* messages to its neighbors. A node can be identified as an *MPR* only if $|MPR\_Table| \neq \varnothing$ and $MPRSelectorAddress \neq$ "0.0.0.0". *TC* messages are generated on basis of the information stored in $Neighbo\_Table$, MPR_Table, and $Cost\_Table$. The detailed *TC* message generation technique has been explained in Algorithm 2.

This approach states that a node y checks its $|MPR\_Table|$ to generate the *TC* messages. For each tuple i of node y's $Cost\_Table$, the values of $NextNodeAddress$, $DestinationNodeAddress$ and $Cost$ fields are shared, respectively, through the $MPRNodeAddress$, $DestinationNodeAddress$ and $Cost$ fields of the generated *TC*. $MPRSelectorNodeAddress$ field of *TC* contains the main address of node y and $NodeCost$ represents the link cost. The remaining fields contain information following *RFC 3626* [3]. Other fields of *TC* messages contain information according to the basic *OLSR*.

3.2.8 *TC* processing Technique

Upon receipt of a *TC* message, a node processes it only if its *IPv4* address is listed in the $MPRNodeAddress$ field of the message. If the receiver node finds itself as listed, then it confirms itself to be an *MPR* node selected by the *TC* sending node and starts to process *TC* and updates its MPR_Table. Algorithm 3 shows the processing technique of the received *TC* message to update MPR_Table.

For each row i of the received *TC*, a new tuple j is inserted into the node y's MPR_Table. $MPRSelectorAddress$, $DestinationAddress$, $Cost$, $NodeCost$ fields of each tuple j in MPR_Table of y stores the received information carried by $MPRSelectorNodeAddress$, $DestinationNodeAddress$, $Cost$, $NodeCost$ fields, respectively, of each i of the received *TC*. $DestinationLocation$ field of tuple j updates from node y's $Dest\_Table$. The remaining information is processed according to the basic *TC* message processing technique stated in [3].

3.2.9 Proposed Cost Function

The proposed *MPR* selection technique, illustrated in Algorithm 4, is based on the heuristic cost function presented in Eq. 3.1. For example, if j is selected as the next *MPR* of i for a particular destination k, then the cost for selecting

j is the sum of the residual cost between j and k and node cost between i and j. It is assumed that the cost is directly proportional to Euclidean distance; the cost increases as the distance between two nodes increases. Euclidean distance between any two nodes is calculated as:

$$Cost^{j} = NodeCost^{i,j} + ResidualCost^{j,k} \tag{3.1}$$

$$D\left(p,q\right) = \sqrt{\left(q_x - p_x\right)^2 + \left(q_y - p_y\right)^2} \tag{3.2}$$

$$NodeCost^{i,j} = \frac{D\left(i,j\right)}{\alpha^{j}} \tag{3.3}$$

$$\alpha^{j} = 2 * w^{j} + 1, w^{j} = willingness^{j} \tag{3.4}$$

$$ResidualCost^{j,k} = \frac{D\left(j,k\right)}{\beta} \tag{3.5}$$

$$Cost_{NextMPR^{i}} = \min_{\forall j \in N(i)} Cost^{j} \tag{3.6}$$

In Eq. 3.3, node cost represents the cost between any two 1-hop neighbor nodes. Node cost is directly proportional to the distance between these two nodes and inversely proportional to the willingness factor, α, of the reaching node. α is a function of willingness (Eq. 3.4) of the neighbor node to forward a TC message. According to Eq. 3.3, if the willingness of neighbor node increases, node cost decreases, i.e., the possibility of being selected as MPR increases. On the other hand, node cost is high for a higher distance leading to a lesser possibility in MPR selection.

Residual Cost (Eq. 3.5) between the 1-hop neighbor node (j) and the destination node (k) is directly proportional to the Euclidean distance and inversely proportional to a normalization factor, β. If $D\left(j,k\right)$ increases, it means that, node j is far away from destination k. This results in a lesser possibility to select j as an MPR node for i. The normalization factor β depends on the nodes' transmission power and network area. In this study, β is determined heuristically.

Finally, the cost of the selected next MPR node of i is calculated using Eq. 3.6. Here, $N\left(i\right)$ represents all 1-hop neighbors of node i. From all the symmetric 1-hop neighbors of i, the selected next MPR is j, if the cost to reach j is lowest.

3.2.10 *MPR* Calculation Technique

A node calculates *MPR* periodically after each $TC\_Interval$ stated in the classical *OLSR*. Initially, the *MPR* is calculated according to the heuristic Algorithm 4, a node finds its next *MPR* set and updates $Cost\_Table$ to store *MPR* information as follows. A node finds its next *MPR* node based on a heuristic function stated in Eq. 3.1. The cost calculation process for selecting next *MPR* node follows Eq. 3.6.

If i and j represents each tuple of node y's MPR_Table and $Neighbor\_Table$ respectively, then it needs to find the lowest cost node j for each tuple i. Node y finds total cost for reaching each destination, stored in its MPR_Table, through each 1-hop neighbor j and finds the lowest cost neighbor j for each destination using Eq. 3.6. If y finds the lowest cost node p, from its all the 1-hop neighbors j, for a destination node q, then it considers node p as next *MPR* node for q. node y updates its $Cost\_Table$'s $NextNodeAddress$ and $DestinationNodeddress$ fields with p and q respectively. $Cost$ field contains the lowest cost for selecting p as next *MPR* node.

The basic working procedure is represented in Figure 3.7. The neighbor table gets updated via continuous exchange of *Hello* messages. Each node can store neighbor information, including neighbor location and destination information, through exchanging *Hello* messages. Each node gets the information of the available destinations in the network as explained in section 3.2.3.

MPR_Table of a node is updated using the information piggybacked in the TC message. A node runs Algorithm 4 to find out the next *MPR* nodes based on the $Neighbor\_Table$ and MPR_Table tables. Each node with $|MPR\_Table| \neq \varnothing$ sends this *MPR*-related information to its neighbors using TC message. After receiving a TC message, a node can update its MPR_Table only if it is listed in this message.

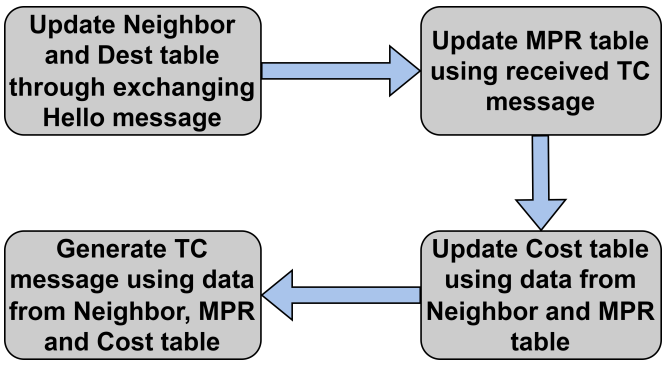


Figure 3.7: The basic working process for calculating *MPR*.

Algorithm 2: *TC* message (*TC_Msg*) generation.

```
1: if |MPR_Table| ≠ ∅ then
2:   for i = 1, 2, .... do
3:     #i represents each tuple in Cost_Table.
4:     MPRNodeAddress (TC_Msg) ← NextNodeAddress^i (Cost_Table)
5:     MPRSelectorNodeAddress (TC_Msg) ← SenderNodeAddress
6:     DestinationNodeAddress (TC_Msg) ←
       DestinationNodeAddress^i (Cost_Table)
7:     Cost (TC_Msg) ← Cost^i (Cost_Table)
8:     for j = 1, 2, .... do
9:       #j represents each tuple in MPR_Table.
10:      if DestinationNodeAddress^i (Cost_Table) =
         DestinationNodeAddress^j (MPR_Table) then
11:        for k = 1, 2, .... do
12:          #k represents each tuple in Neighbor_Table.
13:          if NextNodeAddress^i (Cost_Table) =
             NeighborMainAddress^k (Neighbor_Table) then
14:            NodeCost (TC_Msg) ←
               NodeCost^j (MPR_Table) + NodeCost^k (Neighbor_Table)
15:            break
16:          end if
17:        end for
18:        break
19:      end if
20:    end for
21:  end for
22: end if
```

3.2.11 Analysis of Time and Space Complexity

Analysis of time and space complexity of the proposed algorithms have been given bellow:

- ***TC* Generation Technique:** The time complexity of this algorithm is $O(n^3)$, where n is the number of tuples in the *Cost_Table*. This is because there are three nested loops in the algorithm, each iterating over the tuples in the *Cost_Table*, *MPR_Table*, and *Neighbor_Table* respectively.

  The space complexity of this algorithm is $O(1)$ because it does not require any additional space that grows with the input size. The algorithm only uses a constant amount of space to store variables and does not create any data structures that grow with the input size.

- ***TC* Processing Technique:** The time complexity of this algorithm is $O(n^2)$, where n is the number of tuples in the *TC* message. This is because

Algorithm 3: *TC* message (*TC_Msg*) processing.

```
1: for i = 1, 2, ....  do
2:    #i represents each tuple in TC message.
3:    if ReceiverNodeAddress = MPRNodeAddress^i (TC_Msg) then
4:       MPRSelectorAddress (MPR_Table) ← SenderNodeAddress
5:       DestinationAddress (MPR_Table) ←
         DestinationNodeAddress^i (TC_Msg)
6:       Cost (MPR_Table) ← Cost^i (TC_Msg)
7:       NodeCost (MPR_Table) ← NodeCost^i (TC_Msg)
8:       for j = 1, 2, ....  do
9:          #j represents each tuple in Dest_Table.
10:         if DestinationNodeAddress^i (TC_Msg) =
            DestinationNodeAddress^j (Dest_Table) then
11:            DestinationLocation (MPR_Table) ←
               DestinationNodeLocation^j (Dest_Table)
12:            break
13:         end if
14:      end for
15:      break
16:   end if
17: end for
```

Algorithm 4: Next *MPR* node Calculation.

```
1: for i = 1, 2, ....  do
2:    #i represents each tuple in MPR_Table.
3:    for j = 1, 2, ....  do
4:       #j represents each tuple in Neighbor_Table.
5:       Calculate the Cost of each j node according to Eq. 3.1 and find out the
         minimum cost node k using Eq. 3.6
6:    end for
7:    Insert the tuple of Cost_Table as below: Step 8-10
8:    DestinationNodeAddress (Cost_Table) =
      DestinationAddress^i (MPR_Table)
9:    NextNodeAddress (Cost_Table) =
      NeighborMainAddress^j (Neighbor_Table) which has been selected as k
10:   Cost (Cost_Table) =The Calculated Cost for reaching this node k
11: end for
```

there are two nested loops in the algorithm - one loop iterating over the tuples in the *TC* message (lines 1-17) and another loop iterating over the tuples in the *Dest_Table* (lines 8-14). As a result, the algorithm's time complexity is quadratic.

The space complexity of this algorithm is $O(1)$ because it does not use any additional data structures that grow with the input size. The algorithm

only uses a constant amount of memory to store variables and perform computations, regardless of the input size.

- **Next *MPR* Calculation Technique:** The time complexity of this algorithm is $O(n^2)$, where n is the number of tuples in the MPR_Table. This is because there are two nested loops, one iterating over the tuples in the MPR_Table (outer loop) and the other iterating over the tuples in the $Neighbor\_Table$ (inner loop).

  The space complexity of this algorithm is $O(n)$, where n is the number of tuples in the MPR_Table. This is because the $Cost\_Table$ is created to store the calculated costs for each node, and it will have a size proportional to the number of tuples in the MPR_Table.

  Overall, the time complexity is $O(n^2)$ and the space complexity is $O(n)$ for this algorithm.

3.3 Mathematical Synopsis of the Proposed Methodology

This section includes an experimental calculation that demonstrates the entire methodology in action. Assume that, in Figure 1.3 D-23, D-26, D-29 and D-32 are destination nodes and S-1 is source node for a certain time. Data packet needs to be propagated towards these certain destinations.

3.3.1 Node 1's Processing Techniques

As node 1 is assumed as source node for visualizing a demo calculation, no other nodes can select it as *MPR* node. For convenient calculation, channel costs of all of its neighbors are assumed as 1. In practical environment, this channel cost calculation follows Eq. 3.3.

Through exchanging *Hello* messages every node gets its surrounding neighborhood information. Table 3.1 depicts node 1's neighbors' information for a certain period of time. This table's entire contents is merely an assumption meant to illustrate computation methods. Node 1 gets its neighbor's location information through *Hello* messages (Figure 3.2).

Table 3.2 demonstrates MPR_Table of node 1. Since node 1 is assumed as a source node for a particular time period, its $MPRSelectorAddress$ field is $NULL$. That means no other nodes select it as their next *MPR* node for this time period. Through exchanging *Hello*, $DestinationAddress$ and $DestinationLocation$

Table 3.1: Node 1's neighbor table.

Neighbor Main Address	Status	Willingness	Node Cost	Neighbor Location
2	SYM	3	1	$(x1, y1)$
3	SYM	3	1	$(x2, y2)$
4	SYM	3	1	$(x3, y3)$
5	SYM	3	1	$(x4, y4)$
6	SYM	3	1	$(x5, y5)$
7	SYM	3	1	$(x6, y6)$
8	SYM	3	1	$(x7, y7)$

Table 3.2: Node 1's MPR_Table.

MPR Selector Address	Destination Address	Cost	Node Cost (From Source)	Destination Location
$NULL$	D-23	0	0	$(x1, y1)$
$NULL$	D-26	0	0	$(x2, y2)$
$NULL$	D-29	0	0	$(x3, y3)$
$NULL$	D-32	0	0	$(x4, y4)$

fields are filled with all of the certain destinations and their perspective locations respectively at that time interval. $Cost$ and $NodeCost$ fields are filled with *zero* values, as, being source node.

Node 1's $Cost\_Table$ is depicted by Table 3.3. For each destination node, node 1 chooses the minimum cost node from its one hope neighbor set following Eq. 3.6 and Algorithm 4 using data stored in both of $Neighbor\_Table$ and MPR_Table. In this scenario, it shows that, M-4 is the minimum cost node among node 1's one hop neighbors for destination D-23 according to Algorithm 4. Thus, node 1 includes M-4 as its MPR node for D-23. Similarly, M-5, M-7 and M-8 have been included in node 1's MPR set for destinations D-26, D-29 and D-32 respectively. Hence, node 1 selects its Next MPR nodes from one hop neighbors.

Node S-1 needs to share its MPR set information to all of its one hop neighbors. TC message provides the opportunities for sharing this MPR related infor-

Table 3.3: Node 1's *Cost_Table*.

Destination Node Address	Next Node Address	Cost
D-23	M-4	1+5=6
D-26	M-5	1+4=5
D-29	M-7	1+6=7
D-32	M-8	1+5=6

Table 3.4: Generated *TC* messages from Node 1.

MPR Node Address	*MPR* Selector Node Address	Destination Node Address	Node Cost	Cost
M-4	S-1	D-23	0+1=1	6
M-5	S-1	D-26	0+1=1	5
M-7	S-1	D-29	0+1=1	7
M-8	S-1	D-32	0+1=1	6

mation. Each one hop neighbor of node 1, receives this *TC* message and checks whether their *ID*s are included in the received *TC* message or not. A node from one hop neighbor set can only process the received *TC* message according to Algorithm 3, when its *ID* is listed in the received *TC* message and confirms itself as node 1's *MPR* node. Thus, each of node 1's *MPR*s can update their MPR_Table after receiving and processing of received *TC* message.

Table 3.4 demonstrates node 1's circulated *TC* message. All the fields of this message are filled according to Algorithm 2 using data set stored in Table 3.1, Table 3.2 and Table 3.3. Only M-4, M-5, M-7 and M-8 from one hope neighbor of node 1, can process node 1's circulated *TC* message, as, their *ID*s are enlisted.

3.3.2 Node 4's Processing Techniques

As node 4's *ID* is enlisted in the received *TC* message circulated by node 1, it can process it according to Algorithm 3 and update its MPR_Table with *TC* information.

From this example, it shows that, node 4 updates its MPR_Table (Table 3.6) according to the received *TC* message. A new tuple is inserted through processing of the received *TC*. $MPRSelectorAddress$ filed is filled with S-1 ac-

Table 3.5: Node 4's neighbor table.

Neighbor Main Address	Status	Willingness	Node Cost	Neighbor Location
1	SYM	3	1	$(x1, y1)$
3	SYM	3	1	$(x2, y2)$
11	SYM	3	1	$(x3, y3)$
12	SYM	3	1	$(x4, y4)$
13	SYM	3	1	$(x5, y5)$
5	SYM	3	1	$(x6, y6)$

Table 3.6: Node 4's MPR_Table.

MPR Selector Address	Destination Address	Cost	Node Cost (From Source)	Destination Location
S-1	D-23	6	1	$(x1, y1)$

cording to Algorithm 3, means that, S-1 node selects node 4 as its *MPR* node. *DestinationAddress* is updated with D-23, means that, node 4 has been selected as *MPR* for destination D-23. *Cost* field depicts the total cost for establishing a route from source node, S-1, to node 4 and it updates with 6 according to the received *TC*'s *Cost* field.

$Neighbor\_Table$ (Table 3.5) of node 4 updates with the neighborhood information through exchanging *Hello* messages, according to the previous manner. *Status* field depicts the neighbor' link status. *Willingness* field is filled with the default value. *NodeCost* is assumed as 1 in this example for convenience, which depicts the total channel cost from node 4 to its neighbor node. In real environment, it updates according to Eq. 3.3. In this scenario (Figure 1.3), it shows that, node 1, 3, 11, 12, 13 and 5 are enlisted as node 4's symmetric one hop neighbor. From these neighbors, node 4 selects the most cost efficient one according to Algorithm 4 and updates its $Cost\_Table$ accordingly.

Table 3.7 depicts node 4's $Cost\_Table$. From the above scenario, it shows that, among all one hop neighbors of node 4, M-12 is more cost efficient or least cost node. Hence, Node 4 selects M-12 as its next *MPR* node for D-23 and updates its *NextNodeAddress* field. *Cost* field updates as a summation of node

Table 3.7: Node 4's *Cost_Table*.

Destination Node Address	Next Node Address	Cost
D-23	M-12	1+1+RC(M-12,D-23)=4

Table 3.8: Generated *TC* messages from Node 4.

MPR Node Address	*MPR* Selector Node Address	Destination Node Address	Node Cost	Cost
M-12	M-4	D-23	1+1=2	4

cost from S-1 to M-4 (Table 3.6), node cost from M-4 to M-12 (Table 3.5) and residual cost from M-12 to D-23 (Eq. 3.5).

Now, Node 4 follows the same process as previous, to circulate its *MPR* related information. Table 3.8 depicts a demo calculation according to the example stated in this book. All the fields of node 4's *TC* message updates according to Algorithm 2 using dataset stored in Table 3.5, Table 3.6 and Table 3.7. *NodeCost* field depicts the total channel cost from S-1 to M-12 which is the summation of *NodeCost* (Table 3.6), as channel cost from S-1 to M-4 and *NodeCost* (Table 3.5), as channel cost from M-4 to node 12.

3.3.3 Node 12's Processing Techniques

Node 12 receives the circulated *TC* message from node 4, as node 12 is a one of neighbor of node 4. The first step after receiving *TC* message from node 4 is to check whether node 12 is listed in that message or not. As, node 4 selects node 12 as its next *MPR*, node 12 can update its *MPR_Table* (Table 3.10) according to Algorithm 3. *MPRSelector* field is updated with M-4 according to *TC* message's *MPRSelectorNodeAddress* field. *DestinatioAddress* field is updated with D-12 according to *TC*'s *DestinationNodeAddress* field. *Cost* field is updated with 4 according to *TC*'s *Cost* field. *NodeCost* field depicts the total channel cost from S-1 to M-12 and is updated as 2 according to *TC*'s *NodeCost* field.

Node 12's *Neighbor_Table* (Table 3.9) updates according to the basic *Hello* processing techniques [3] as well as the same process stated for the other nodes' *Neighbor_Table*. In this example scenario, Node 4, 13, 25, 11, 24 and 23 are enlisted as one hop neighbors for node 12 through exchanging periodic *Hello*.

Table 3.9: Node 12's neighbor table.

Neighbor Main Address	Status	Willingness	Node Cost	Neighbor Location
4	SYM	3	1	$(x1, y1)$
13	SYM	3	1	$(x2, y2)$
25	SYM	3	1	$(x3, y3)$
11	SYM	3	1	$(x4, y4)$
24	SYM	3	1	$(x5, y5)$
23	SYM	3	1	$(x6, y6)$

Table 3.10: Node 12's MPR_Table.

MPR Selector Address	Destination Address	Cost	Node Cost (From Source)	Destination Location
M-4	D-23	4	2	$(x1, y1)$

Table 3.11 illustrates node 12's $Cost\_Table$. Following the same MPR selection process stated above, node 12 selects its next MPR node. In this scenario, D-23 is at the one hop distance of node 12. Thus, destination node is found as the one hop neighbor node and hence, a complete route from source S-1 to destination D-23 is established. As, destination has been found, M-12 doesn't circulate further TC messages for this time period.

The whole process continues for other selected MPR nodes (M-5, M-7 and M-8) of S-1. Hence, for 4 destinations in this example, 4 distinct optimal routes (S-1 → M-4 → M-12 → D-23, S-1 → M-5 → M-14 → D-26, S-1 → M-7 → M-16 → D-29 and S-1 → M-8 → M-19 → D-32) are established at the concurrent time as Figure 1.4b.

Table 3.11: Node 12's $Cost\_Table$.

Destination Node Address	Next Node Address	Cost
D-23	D-23	1+2+RC(D-23,D-23)=3

Chapter 4

Experimental Results and Evaluation

NS3 is a famous discrete event simulator that is widely adopted for implementing *MANET* and *OLSR*. The default *OLSR* installed in *NS3* (*NS-3.30*) [16] has been modified for implementing our improved *OLSR* and comparing its performance with the default *OLSR*. The metrics such as routing overhead, end-to-end delay, *PDR*, and throughput are considered for performance evaluation and comparison.

4.1 Simulation Parameters

Simulation experiments have been conducted using *NS3* (version 3.30) network simulator to validate our proposed *MPR* selection technique. Then, we compared the obtained results with standard *OLSR*. All simulation parameters have been summarized in Table 4.1.

4.2 Evaluation Criteria

Our proposed methodology has been executed and validated using *NS3* network simulator analyzing the following performance metrics:

- **Number of selected *MPR* nodes:** The total number of Multi-point Relay nodes selected in the network.
- **Number of *TC* messages:** The total number of Topology Control messages flooded in the network.
- **Number of total messages:** The total number of *TC* and *Hello* messages flooded in the network.

Table 4.1: Simulation Parameters.

Platform used	Ubuntu-18.04
Type of network	*MANET*
Simulator used	NS-3.30
Simulation time	120 s
Total area	500*500 sq. m.
Number of nodes	50, 60, 70. 80, 90, 100
Transmit power	7.5 dBm
Transmission Range	250 m
Mobility model	Random waypoint
Type of MAC	IEEE 802.11b
Transport layer	*UDP*
Total packet size	64 bytes
Pause Time	1, 5, 10 s
Stream index	0-9
Speed	4 m/s
Data rate	2048 bps
β	5

- **Size of *TC* messages:** Total size of *TC* messages flooded in the network.
- **Size of total messages:** Total size of disseminated *TC* and *Hello* messages.
- **Packet delivery ratio (*PDR*):** The overall ratio between the total number of successfully delivered packets and the total number of sent packets.
- **Throughput (*TH*):** The volume of data transported between the source and the destination.
- **End-to-end delay:** Time that is required by a packet to send from a source and received by a destination.
- **Routing overhead:** Total number of sent *TC* messages has been considered as routing overhead for this experiment.

4.3 Simulation Results

Experiment results presented in this paper are taken as the average values after running the simulator 10 times for each scenario.

Figure 4.1 demonstrates the comparison of the total number of selected *MPR* nodes among classical *OLSR*, proposed efficient *OLSR* and *SSTB*. Experiment results show that, the selection of *MPR* nodes increases with increasing number of nodes, as, more nodes are needed to establish routes towards destinations. However, among all available nodes in the network, only a few nodes are selected as *MPRs* using our methodology. As, our proposed approach selects *MPR* from neighbor nodes using a heuristic cost function, only the nodes having less cost can be elected as *MPR*s for the particular destination nodes. Thus, all the optimal paths, established using the cost function stated in Eq. 3.1, towards each destination node, are composed of these selected *MPR* nodes. Consequently, all the necessary routes, needed for data forwarding, are being established with these less number of selected *MPR* nodes. This scenario validates the thought that our proposed *MPR* selection technique outperforms the classical *OLSR*, *SSTB* and *M-OLSR* protocol in terms of 55% (on average), 28% (on average) and 49% (on average) less *MPR* selection respectively which causes less overhead or less propagation of *TC* messages.

Figure 4.2 illustrates the total number of *TC* messages sent according to a different number of nodes both for standard *OLSR* and the proposed efficient *OLSR*. This result shows that *TC* dissemination increases according to the increasing node number for both protocols. Because, if number of node increases,

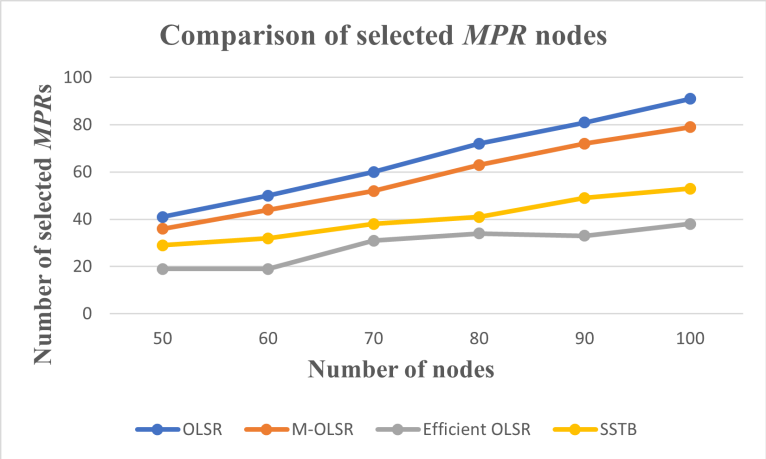


Figure 4.1: Total selected *MPR* nodes.

it causes a rise in *MPR* selection. So, more *TC* messages are required to share network topology information. Moreover, the proposed method reduces the total *TC* dissemination for all cases. This is because, our proposed *OLSR* protocol selects less number of *MPR* nodes which absorb unnecessary *TC* flooding in the network. Consequently, our proposed protocol achieves up to 75% and 68% less *TC* propagation compared to the standard *OLSR* and *M-OLSR* protocol respectively.

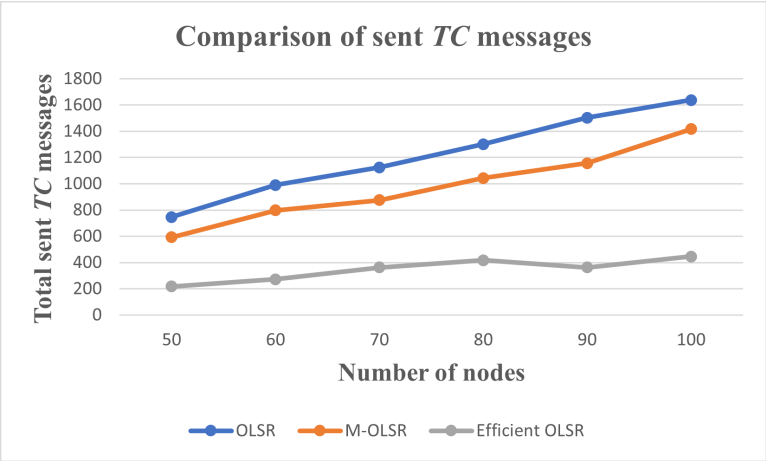


Figure 4.2: Total sent *TC* messages.

Fewer *TC* dissemination also causes a reduction in the total size of the sent *TC* messages. This reduction in *TC* size is illustrated by Figure 4.3. As the number of *MPR* nodes are reduced using the proposed protocol, it causes a reduction in the total number of flooded *TC* messages as well as *TC* size resulting less routing overhead.

Figure 4.4 shows the comparison of total sent messages (*Hello* and *TC*) in the network. As, network density increases with higher number of nodes, number of

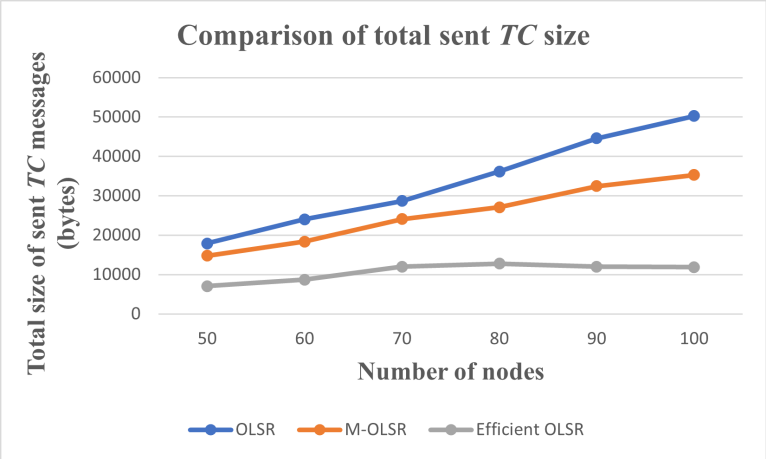


Figure 4.3: Total size of sent *TC* messages.

sending messages also increases for establishing necessary routes. However, the experiment results show that our methodology produces up to 16% and 11% fewer messages than standard *OLSR* and *M-OLSR* respectively. Only *Hello* and *TC* messages are taken under consideration in calculating total messages for their significant impacts on routing overhead. As, the *Hello* message format is extended by our approach without contributing in the number of *Hello* message dissemination, the total size of flooded messages is also increased. But the increased size can be ignored as the number of flooded messages are reduced. This reduction in the total number of message dissemination causes less processing time as well as lower overhead.

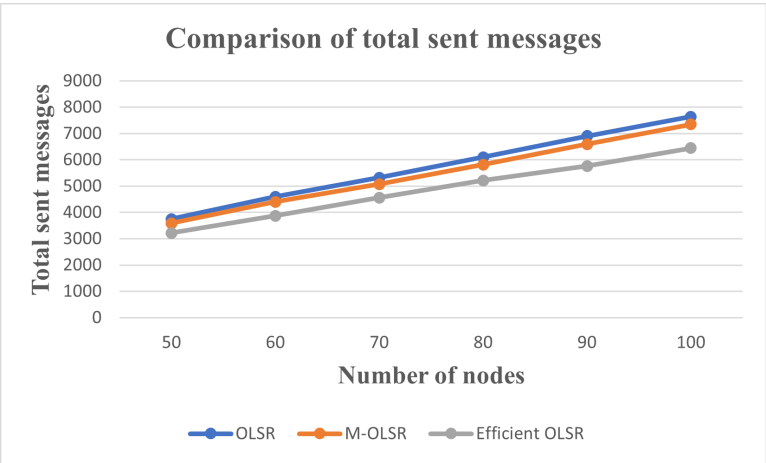


Figure 4.4: Total sent messages.

Our proposed efficient *OLSR* doesn't degrade its performance compared to classical *OLSR* and *M-OLSR* in terms of packet delivery ratio, throughput, and end-to-end delay. The packet delivery ratio of proposed *OLSR*, standard *OLSR* and *M-OLSR* is compared against the different numbers of nodes and pause time

in Figure 4.5 and Figure 4.6, respectively. These results show that, packet delivery ratio is being increased slightly, in terms of node number and pause time, using the proposed protocol. As, all optimal routes are composed of selected *MPR* nodes, this scenario causes an increase in packet delivery ratio. However, for large number of nodes, some *MPR* nodes may be selected wrongly i.e., these wrongly selected *MPR* nodes may not contribute in establishing optimal paths. This scenario degrades packet delivery ratio slightly for large scale networks. Moreover, it is a very difficult task to reduce packet delivery ratio in wireless networks.

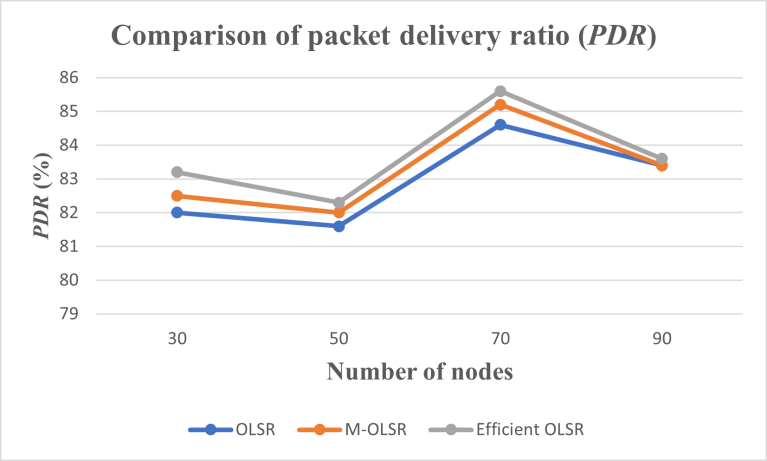


Figure 4.5: Packet delivery ratio as a function of node number.

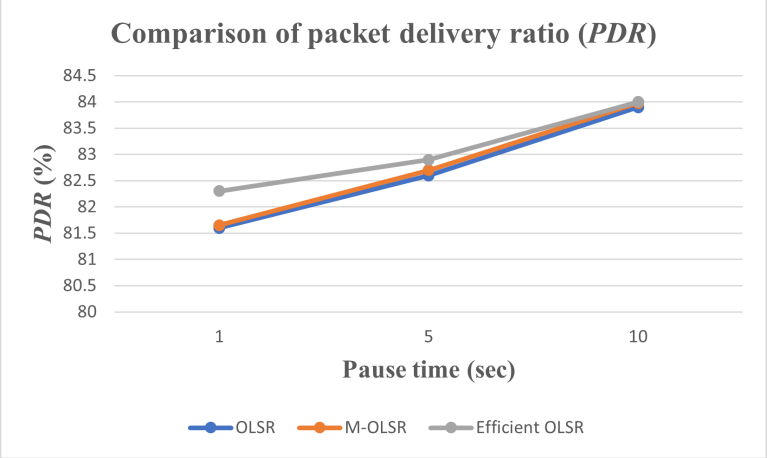


Figure 4.6: Packet delivery ratio as a function of pause time.

On the other hand, packet delivery ratio increases with increasing pause time (Figure 4.6). Because, if pause time increases, the possibility of link breaking reduces, that, supports establishing optimal paths and increases packet delivery ratio.

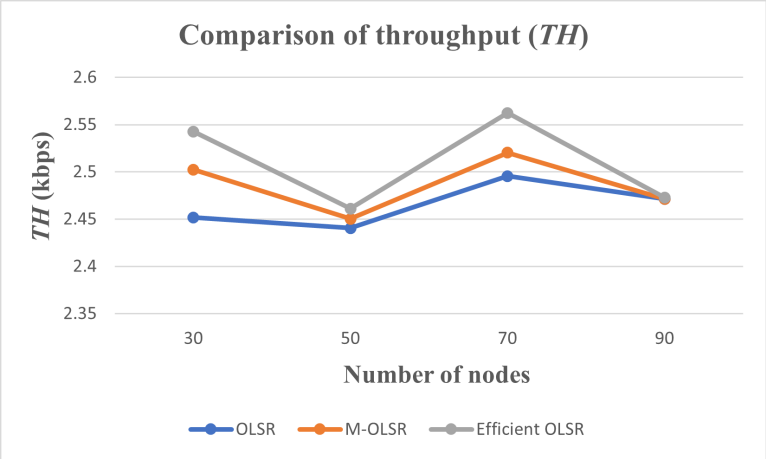


Figure 4.7: Throughput as a function of node number.

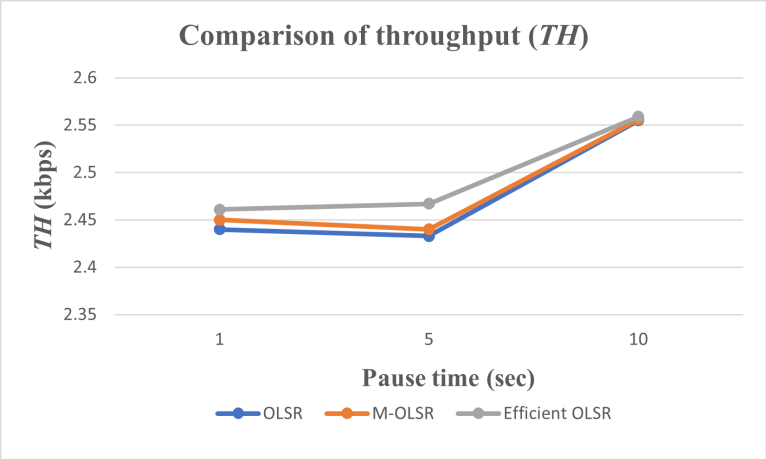


Figure 4.8: Throughput as function of pause time.

Figure 4.7 and Figure 4.8 demonstrate the performance of the proposed *OLSR*, classical *OLSR* and *M-OLSR* in terms of throughput. These results depict that, throughput is being increased slightly in terms of both node number and pause time. From Figure 4.8, it shows that, pause time creates more impacts on increasing throughput. Because, more stable links are established when pause time increases.

End-to-end delay is also compared in terms of node number in Figure 4.9. Delay increases with increasing node number, as, the possibility of false *MPR* selection also increases. This causes establishing non-optimal routes which increases end-to-end delay for data transmission.

4.4 Constant Values

The values of the constants used to describe the protocol are listed in this section.

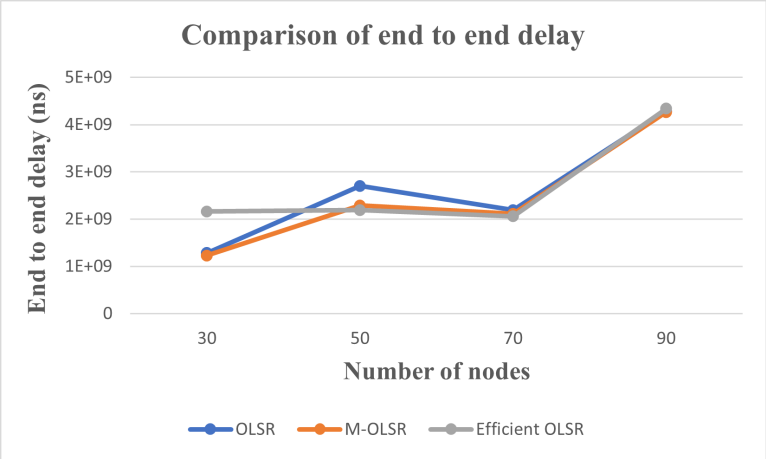


Figure 4.9: Delay as function of node number.

4.4.1 Emission Intervals and Holding times

The "validity time" computation ("Vtime" and "Htime" fields in message headers) uses a scaling factor called C. The "validity time" advertisement is made so that nodes within a network can still completely cooperate even while their emission intervals vary and are individually adjustable. C = 1/16 seconds (0.0625 seconds). The points listed bellow make the protocol functioning:

- There must always be a difference between the advertised holding time and the advertised information's refresh interval (Table 4.2). Furthermore, it is advised to maintain the established relationship between the hold time and the interval in order to account for appropriate packet loss.
- The recommended value for the constant C should be used. Interoperability can only be reached if C is the same across all nodes.
- It is possible to choose the emission intervals and the stated holding time individually for each node.

4.4.2 Assumed Constants for Link, Neighbor and Message Type

Willingness: A node's willingness may alter dynamically in response to changing circumstances. Willingness indicates its desire to forward traffic on behalf of other nodes and can be adjusted to any integer value between 0 and 7.

Nodes will have a willingness of $WILL\_DEFAULT$ by default. A node marked with $WILL\_NEVER$ indicates that it does not want to transport traffic for other nodes, maybe because of resource limitations (such as poor battery life).

Table 4.2: Values for constants [3].

Emission Intervals	
$HELLO\_INTERVAL$	2 Sec.
$REFRESH\_INTERVAL$	2 Sec.
$TC\_INTERVAL$	5 Sec.
$MID\_INTERVAL$	$TC\_INTERVAL$
$HNA\_INTERVAL$	$TC\_INTERVAL$
Holding Time	
$NEIGHB\_HOLD_TIME$	$3xREFRESH\_INTERVAL$
TOP_HOLD_TIME	$3xTC\_INTERVAL$
DUP_HOLD_TIME	30 Sec.
MID_HOLD_TIME	$3xMID\_INTERVAL$
HNA_HOLD_TIME	$3xHNA\_INTERVAL$

$WILL\_ALWAYS$ denotes that a node should always be chosen to transport traffic for other nodes, for instance because resources (such as high capacity interfaces with other nodes and a permanent power supply) are abundant.

Link Type: A local interface and a remote interface together define a "link". Each neighbor node, or more precisely, the link to each neighbor, has an associated state that is either "symmetric" or "asymmetric" for link sensing. "Symmetric" means that the bidirectional characteristics of the link to that neighbor node has been confirmed, meaning that data can be transmitted in both directions. "Asymmetric" means that the node has received *Hello* messages from the neighboring node, indicating that communication with that node is possible. However, it is not established that this node is likewise capable of receiving messages, meaning that communication with the neighboring node is uncertain. Table 4.4.2 enlisted the necessary constants, assumed for working of our proposed protocol perfectly. *OLSR* recognizes the following link types:

- $UNSPEC\_LINK$: No particular details provided regarding the links.
- $ASYM\_LINK$: This denotes an asymmetric link, meaning that the neighbor interface is "heard".
- SYM_LINK: This denotes symmetry between the links and the interface.
- $LOST\_LINK$: Links that have been lost are represented.

It is assumed that the link code has two distinct fields, each consisting of two bits, assuming the value is less than or equal to 15. From the value of a link code, neighbor as well as link type can be understood easily. Figure 4.10 represents the basic format of a link code.

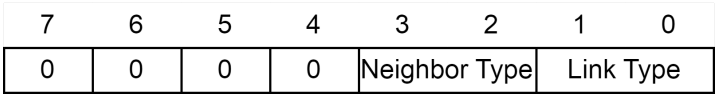


Figure 4.10: Link Code format.

Neighbor Type: The following three types of neighbors in *OLSR* are:

- SYM_NEIGH: This denotes that at least one symmetrical link exists between this node and its neighbors.
- MPR_NEIGH: The neighbors have been chosen as *MPR* by the sender as well as have at least one symmetrical link.
- NOT_NEIGH: This means that the nodes are not symmetric neighbors yet or are not neighbors at all.

Messages: *OLSR* requires the following messages:

- *Hello:* This message (Figure 3.2) is utilized in order to neighborhood detection, *MPR* selection signaling, link sensing and to allow for future extensions.
- *MID:* The exchange of numerous Interface Declaration (*MID*) messages defines the link between *OLSR* interface addresses and main addresses for numerous *OLSR* interface nodes. All nodes having multiple interfaces are required to broadcast information about their interface configuration to other nodes in the network on a regular basis by sending out *MID* messages in large quantities. The basic *MID* message format has been represented in Figure 4.11.
- *TC:* The propagation of *TC* (Figure 3.6) messages is responsible for both route construction and the distribution of topology information throughout the network. Only the selected *MPR* nodes can propagate this type of messages to minimize the control traffic across the network.
- *HNA:* The *TC* message and the *HNA* message are similar in that, their senders both declare "reachability" to other host(s). This makes the *HNA* message a "generalized version" of the *TC* message. Figure 4.12 represents the basic *HNA* format.

Table 4.3: Values assumed for Links, Neighbor and Message types [3].

Link Types	
$UNSPEC\_LINK$	0
$ASYM\_LINK$	1
SYM_LINK	2
$LOST\_LINK$	3
Neighbor Types	
NOT_NEIGH	0
SYM_NEIGH	1
MPR_NEIGH	2
Message Types	
$HELLO\_MESSAGE$	1
$TC\_MESSAGE$	2
$MID\_MESSAGE$	3
$HNA\_MESSAGE$	4
Link Hysteresis	
$HYST\_THRESHOLD\_HIGH$	0.8
$HYST\_THRESHOLD\_LOW$	0.3
$HYST\_SCALING$	0.5
Willingness	
$WILL\_NEVER$	0
$WILL\_LOW$	1
$WILL\_DEFAULT$	3
$WILL\_HIGH$	6
$WILL\_ALWAYS$	7

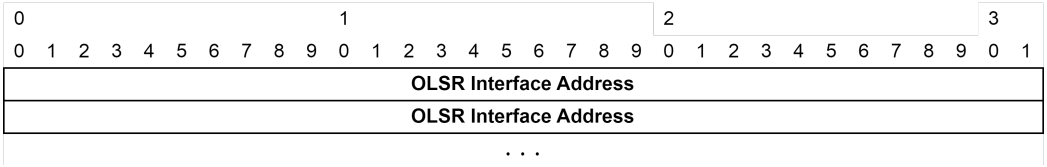


Figure 4.11: *MID* message format.

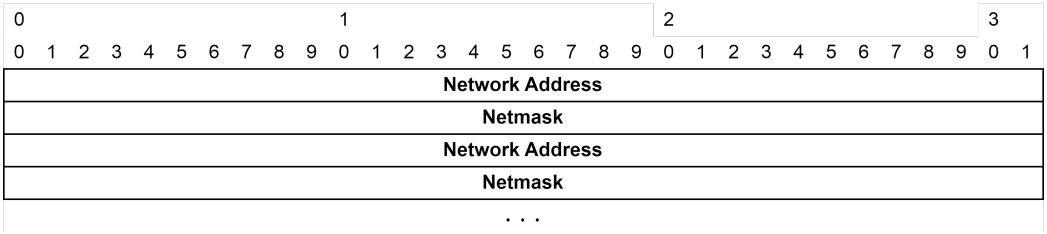


Figure 4.12: *HNA* message format.

4.5 Basic Packet Format (*RFC 3626*) Assumed for *OLSR*

For all protocol-related data, *OLSR* uses a single packet format for connectivity. This will allow for more protocol expansion without compromising backward compatibility. This facilitates the seamless integration of various "types" of data into a solitary transmission, enabling an implementation to maximize its use of the network's maximum frame size. For network transmission, these packets are included into *UDP* datagrams. *IPv4* addresses are displayed in the current article.

A message or messages are contained within each packet. Nodes can appropriately receive and, if necessary, re-transmit messages of an unknown nature since they all use the same header format. Messages can be flooded across the whole network or restricted to nodes that are a certain diameter (measured in hops) from the message's source. Therefore, sending a message to a node's neighborhood is just a particular kind of flooding. Duplicate re-transmissions during flooding any control message will be reduced both locally (each node keeps a duplicate set to prevent broadcasting the same *OLSR* control message twice) and globally by using *MPR*s, as will be covered in following sections.

In addition, a node can read a message's header to find out how far away the message's originator is (measured in hops). Sometimes, this feature could come in handy where the distance to the originator determines, for example, the time information from received control messages stored in a node.

Protocol Type: *UDP* is used in *OLSR* packet communication.

Port Number: *IANA* has designated port 698 for the sole use of the *OLSR* protocol.

Main Address: The main address of a node with a single interface needs to be set to that interface's main address.

Basic Packet Format: Any packet in *OLSR* has the following basic structure (ignoring the *IP* and *UDP* headers) (Figure 4.13):

- **Packet Length:** Represents the total length of packet and measured in bytes.
- **Packet Sequence Number:** Every time an *OLSR* packet is transmitted, the Packet Sequence Number (*PSN*) has to be increased by one. Each interface has a unique packet sequence number that is kept track of, allowing packets sent across the interface to be listed consecutively.
- **Basic Rules:** The *IP* header of a packet contains information about the *IP* address of the interface the packet was sent over. In the event that the packet is empty (that is, its length equals or less than the size of its header), it must be discreetly discarded. This suggests that for *IPv4* addresses, packets with a length of less than 16 must be silently deleted. The above two fields represent packet header.

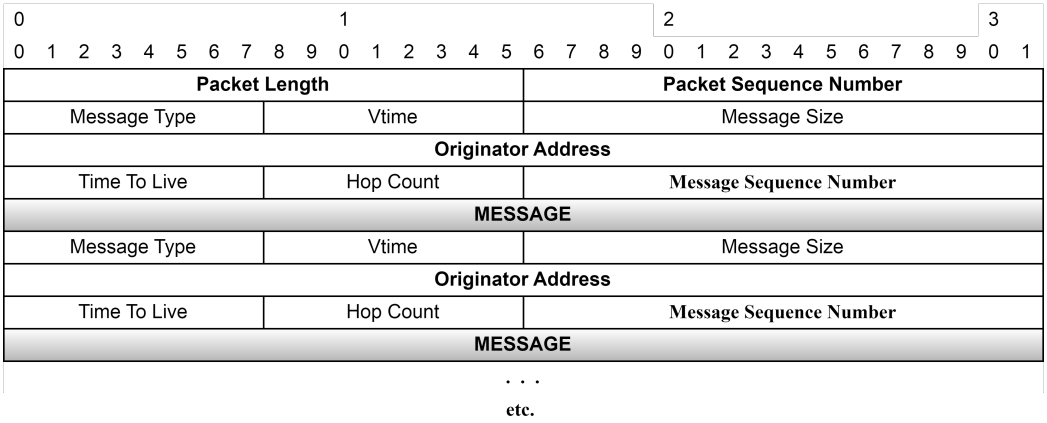


Figure 4.13: Basic packet format of *OLSR*.

Fields included in message header are explained bellow:

- **Message Type:** This parameter specifies the kind of message that can be found in the "MESSAGE" section. For messages in this document and any future expansions, message types between 0 and 127 are reserved.
- **Vtime:** This section specifies the amount of time that, barring the receipt of a more current update to the information, a node must regard the data in the message as valid.

- **Message Size:** This indicates the message's size in bytes, measured from the start of the "Message Type" field to the start of the subsequent "Message Type" field.
- **Originator Address:** The main address of the node that first generated this message is contained in this field.
- **Time To Live:** The maximum number of hops a message will travel through is contained in this parameter. One must subtract 1 from the Time To Live before a message is re-transmitted. A node must not, under any circumstances, re-transmit a message that it receives with a Time To Live of 0 or 1. A node wouldn't typically get a message with a *TTL* of zero. Therefore, the message originator can control the flooding radius by setting this field.
- **Hop Count:** The number of hops a message has made is contained in this field. The Hop Count must be increased by 1 prior to a message being re-transmitted. This is initially set to $'0'$ by the message's creator.
- **Message Sequence Number:** The "originator" node will give each message a distinct identification number when it is being generated. This number is entered into the message's Sequence Number field. Message sequence numbers are used to prevent any node from re-transmitting a particular message more than once.

4.6 Basic Forwarding Techniques

The basic packet forwarding technique follows the *RFC 3626* rules [3]. According to this rules, some key points are listed bellow:

- The forwarding algorithm must silently stop here (and the message must not be sent) if it is not determined that the sender interface address of the message is in the symmetric 1-hop neighborhood of the node.
- Re-transmission of the message is required if the sender interface address is an interface address of an *MPR* selector on this node and if the message's time to live exceeds 1.
- The message must be taken into consideration for forwarding in accordance with the message type standards if the node supports the message type.

It should be mentioned that receiving and sending messages are two distinct processes that are subject to various regulations. Forwarding is the act of sending the same message again to other network nodes, whereas processing is the use of the message content.

It will be feasible to expand the protocol by adding new message types while preserving compatibility with earlier implementations by defining a set of message types that all *OLSR* implementations must understand. The following message types must be sent in order for *OLSR* to function properly:

- Link sensing, neighbor discovery, and *MPR* signaling are all carried out using *Hello* messages.
- Topology declaration (advertisement of link statuses) is carried out by *TC* messages.
- Announcing the existence of numerous interfaces on a node is done by *MID* messages.

4.7 Packet Processing Techniques

A node looks at each of the "message headers" after receiving a basic packet. The node can decide what happens to the message based on the value entered in the "Message Type" field. A node could get the same message more than once. As a result, every node keeps a Duplicate Set in order to prevent processing messages that have already been received and performed. *RFC 3626* rules applied in this context. The key considerations related to processing a packet are summarized bellow:

- The packet needs to be silently discarded if it is empty (that is, if its length is equal to or less than the size of its header).
- This suggests that packets with a length of less than 16 for *IPv4* addresses must be discreetly rejected.
- The message must be discreetly dropped if its time to live is less than or equal to "0" (zero), or if it was transmitted by the receiving node (that is, if the message's originator address is the receiving node's primary address).
- If the message's "Message Type" is implemented by the node, the message must be handled in accordance with the message type's specifications.

4.8 Defining Jitter

Neighboring nodes may seek to transmit control traffic simultaneously due to synchronization in their emission of control traffic, which can occur for a variety of reasons. This could result in collisions and message loss, possibly including the loss of many consecutive messages of the same type, depending on the underlying link-layer's characteristics.

In order to prevent such synchronizations, a node ought to introduce some jitter into the message generation interval according to *RFC 3626* rules. Each time a message is generated, the jitter needs to have a random value. Jitter is a random value in the interval [*0, MAXJITTER*], and the actual message interval is calculated by subtracting jitter from $MESSAGE\_INTERVAL$.

In order to minimize the amount of packet transmissions, the node may choose to piggyback additional messages when it transmits a control message. There's a minimum number of control messages required. If it is advantageous for a particular deployment, a node may transmit control messages more frequently.

Chapter 5

Conclusion and Future Recommendations

This paper proposes an improved *MPR* selection strategy for *OLSR* protocol to enhance its performance in terms of network overhead in *MANET*. The major contribution is to reduce the number of selected *MPR* nodes, which disseminates fewer *TC* messages without affecting the other performance matrices.

The proposed method is efficient in terms of less selected relay nodes without degrading the other performance metrics. Less volume of *MPR* nodes cause the *OLSR* experiencing lower routing overhead as shown in the experiment result section.

The proposed *MPR* selection strategy requires additional repositories and header extensions of *Hello* and *TC* messages. The technique works according to a Euclidean distance and willingness-based heuristic function.

The experiment results show that routing overhead is reduced by 75% and 68% (as maximum) compared to the classical *OLSR* and *M-OLSR* protocols respectively. The proposed *MPR* selection technique also outperforms the standard *OLSR*, *M-OLSR* and *SSTB* protocols by selecting 58% (as maximum), 49% (on average) and 28% less *MPR* set respectively. Our proposed *MPR* selection strategy also shows good performance compared to standard *OLSR* and *M-OLSR* protocols in terms of packet delivery ratio, throughput and delay.

5.1 Limitations

Research related to *MPR* selection strategy in *MANET* can have numerous contributions. Enhancing performance in terms of control overhead, end-to-end delay, throughput, energy efficiency, and security is the focus of the majority of frequent contributions. In this thesis, we have tried to focus on the minimiza-

tion of routing overheads as well as reduction of *MPR* set. For this purpose, we have introduced a new heuristic based cost function for relay selection process. Willingness and Euclidean distance have been considered to formulate the cost function. As, our research considers the node position for calculating cost function there exists limitations related to node speed. For high speed nodes, the routing table will be updated frequently, thus, the number of lost links may be increased.

5.2 Future Recommendations

As the cost function is vital to the proposed *MPR* selection technique, in the future, the normalization and willingness factors and hence the cost function will be determined considering network area, node speed, and transmission power.

As, *OLSR* is a proactive routing protocol, it needs to update the routing tables on a regular time basis to store the most recent information. Link loss ratio increases with higher node speed, as next *MPR* calculation depends on the node positions. So, we'll take node speed into account in the future when we optimize the cost function.

We ignored to address security-related issues related to *MPR* selection in our thesis. The *OLSR* protocol is susceptible to a variety of threats since it operates in a wireless environment. Thus, we will concentrate on improving security in the *MPR* selection process in the future.

5.3 List of Publications

1. Z. Hassan and Asaduzzaman, "Advanced Recursive Best-First Search (RBFS) based Routing Protocol for Multi-hop and Multi-Channel Cognitive Wireless Mesh Networks", *Journal of Computer Science and Technology Studies*, vol. 6, no. 1, pp. 1-10, 2024.

2. Z. Hassan, S. M. A. Iqbal and Asaduzzaman, "An Efficient OLSR Routing Protocol to Minimize Multipoint Relays in MANET", *International Journal of Intelligent Engineering and Systems*, vol. 17, no. 2, pp. 489-500, 2024.

Bibliography

[1] D. Ramphull, A. Mungur, S. Armoogum, and S. Pudaruth, "A review of mobile ad hoc network (manet) protocols and their applications," in *2021 5th international conference on intelligent computing and control systems (ICICCS)*, pp. 204–211, IEEE, 2021.

[2] T. K. Saini and S. C. Sharma, "Flexible multipoint relay selection for suitable route in mobile ad hoc networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 8173–8186, 2021.

[3] T. Clausen and P. Jacquet, "Optimized link state routing protocol (olsr)," tech. rep., 2003.

[4] M. Sharma, M. Singh, K. Walia, and K. Kaur, "A comprehensive study of performance parameters for manet, vanet and fanet," in *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pp. 0643–0646, IEEE, 2019.

[5] S. Lee, J. Youn, and B. C. Jung, "A cooperative phase-steering technique in spectrum sharing-based military mobile ad hoc networks," *ICT Express*, vol. 6, no. 2, pp. 83–86, 2020.

[6] F. T. Al-Dhief, N. Sabri, S. Fouad, N. A. Latiff, and M. A. A. Albader, "A review of forest fire surveillance technologies: Mobile ad-hoc network routing protocols perspective," *Journal of King Saud University-Computer and Information Sciences*, vol. 31, no. 2, pp. 135–146, 2019.

[7] M. H. Hassan, S. A. Mostafa, H. Mahdin, A. Mustapha, A. A. Ramli, M. H. Hassan, and M. A. Jubair, "Mobile ad-hoc network routing protocols of time-critical events for search and rescue missions," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 192–199, 2021.

[8] U. Aliyu, H. Takruri, M. Hope, and A. G. Halilu, "Ds-olsr–disaster scenario optimized link state routing protocol," in *2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, pp. 1–6, IEEE, 2020.

[9] S. A. H. Belkhira, S. Boukli-Hacene, P. Lorenz, M. Belkheir, M. Gilg, and A. Zerroug, "A new mechanism for mpr selection in mobile ad hoc and sensor wireless networks," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2020.

[10] V. K. Quy, V. H. Nam, D. M. Linh, N. T. Ban, and N. D. Han, "A survey of qos-aware routing protocols for the manet-wsn convergence scenarios in iot networks," *Wireless Personal Communications*, vol. 120, no. 1, pp. 49–62, 2021.

[11] K. Raghavendra Rao and B. Jagadesh, "Routing protocols: A survey," in *Proceedings of the International Conference on Computer Vision, High Performance Computing, Smart Devices and Networks*, pp. 85–94, Springer, 2022.

[12] P. Shah and T. Kasbe, "A review on specification evaluation of broadcasting routing protocols in vanet," *Computer Science Review*, vol. 41, p. 100418, 2021.

[13] A. Kurniawan, P. Kristalina, and M. Z. S. Hadi, "Performance analysis of routing protocols aodv, olsr and dsdv on manet using ns3," in *2020 International Electronics Symposium (IES)*, pp. 199–206, IEEE, 2020.

[14] N. Gupta, A. Jain, K. S. Vaisla, A. Kumar, and R. Kumar, "Performance analysis of dsdv and olsr wireless sensor network routing protocols using fpga hardware and machine learning," *Multimedia Tools and Applications*, vol. 80, no. 14, pp. 22301–22319, 2021.

[15] B. A. Mohamed, B. A. Abdelhakim, S. Sohaib, F. Hassan, *et al.*, "Comparative study on the density and velocity of aodv and dsdv protocols using omnet++," in *International Conference on Networking, Intelligent Systems and Security*, pp. 176–189, Springer, 2023.

[16] R. R. Chandan, B. S. Kushwaha, and P. K. Mishra, "Performance evaluation of aodv, dsdv, olsr routing protocols using ns-3 simulator.," *International Journal of Computer Network & Information Security*, vol. 10, no. 7, 2018.

[17] S. Mohapatra and P. Kanungo, "Performance analysis of aodv, dsr, olsr and dsdv routing protocols using ns2 simulator," *Procedia Engineering*, vol. 30, pp. 69–76, 2012.

[18] Y. Hamzaoui, M. Chekour, and M. Amnai, "Enhanced mpr selection algorithm based on metrics for the olsr routing protocol," in *2023 6th International Conference on Advanced Communication Technologies and Networking (CommNet)*, pp. 1–6, IEEE, 2023.

[19] L. Maccari, M. Maischberger, and R. L. Cigno, "Where have all the mprs gone? on the optimal selection of multi-point relays," *Ad Hoc Networks*, vol. 77, pp. 69–83, 2018.

[20] L. Maccari and R. L. Cigno, "How to reduce and stabilize mpr sets in olsr networks," in *2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 373–380, IEEE, 2012.

[21] S. Dong and H. Zhang, "An mpr set selection algorithm based on set operation," in *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, vol. 5, pp. 5–8, IEEE, 2021.

[22] M. Usha and B. Ramakrishnan, "A robust architecture of the olsr protocol for channel utilization and optimized transmission using minimal multi point relay selection in vanet," *Wireless Personal Communications*, vol. 109, pp. 271–295, 2019.

[23] P. Kumar and S. Verma, "Implementation of modified olsr protocol in aanets for udp and tcp environment [j/ol]," *Journal of King Saud University-Computer and Information Sciences (S1319-1587).(2019-08-02)[2020-04-10]. https://doi.org/10.1016/j. jksuci*, vol. 9, 2019.

[24] A. Boushaba, A. Benabbou, R. Benabbou, A. Zahi, and M. Oumsis, "Multi-point relay selection strategies to reduce topology control traffic for olsr protocol in manets," *Journal of Network and Computer Applications*, vol. 53, pp. 91–102, 2015.

[25] S. Sharma, "P-olsr: Position-based optimized link state routing for mobile ad hoc networks," in *2009 IEEE 34th Conference on Local Computer Networks*, pp. 237–240, IEEE, 2009.

[26] D.-g. Zhang, Y.-y. Cui, and T. Zhang, "New quantum-genetic based olsr protocol (qg-olsr) for mobile ad hoc network," *Applied Soft Computing*, vol. 80, pp. 285–296, 2019.

[27] P. Kumari and S. K. Sahana, "Swarm based hybrid aco-pso meta-heuristic (hapm) for qos multicast routing optimization in manets," *Wireless Personal Communications*, vol. 123, no. 2, pp. 1145–1167, 2022.

[28] Y. Sun, W. Dong, and Y. Chen, "An improved routing algorithm based on ant colony optimization in wireless sensor networks," *IEEE communications Letters*, vol. 21, no. 6, pp. 1317–1320, 2017.

[29] Z. Yihui, Q. Xirong, W. Wendong, G. Xiangyang, and M. Jian, "N3s-olsr: Node-status self-sensing optimized link-state routing protocols for manet," in *2010 International Conference on Communications and Mobile Computing*, vol. 3, pp. 288–292, IEEE, 2010.

[30] Y. Xue, H. Jiang, and H. Hu, "Optimization on olsr protocol for lower routing overhead," in *International Conference on Rough Sets and Knowledge Technology*, pp. 723–730, Springer, 2008.

[31] A. B. Paul and S. Nandi, "Modified optimized link state routing (m-olsr) for wireless mesh networks," in *2008 International Conference on Information Technology*, pp. 147–152, IEEE, 2008.

[32] Z. Xiang and J. Li, "Multi point relay set selection algorithm based on fruit fly-inspired optimizers," in *International Conference on Algorithms, High Performance Computing, and Artificial Intelligence (AHPCAI 2023)*, vol. 12941, pp. 120–128, SPIE, 2023.

[33] A. Hawbani, X. Wang, A. Abudukelimu, H. Kuhlani, Y. Al-Sharabi, A. Qarariyah, and A. Ghannami, "Zone probabilistic routing for wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 728–741, 2018.

[34] G. Yin, S. Wang, and X. Liu, "Power aware routing protocol based on olsr in ad hoc network," in *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*, pp. 437–440, IEEE, 2012.

[35] Y. Mostafaei and S. Pashazadeh, "An improved olsr routing protocol for reducing packet loss ratio in ad-hoc networks," in *2016 Eighth international conference on information and knowledge technology (IKT)*, pp. 12–17, IEEE, 2016.

[36] H. Amraoui, A. Habbani, and A. Hajami, "Mobility quantification for multipoint relays selection algorithm in mobile ad hoc networks," in *2016 5th International Conference on Multimedia Computing and Systems (ICMCS)*, pp. 278–283, IEEE, 2016.

[37] A. Abdellaoui, J. Elmhamdi, and H. Berradi, "Multipoint relay selection through estimated spatial relation in smart city environments," in *2018 International Conference on Advanced Communication Technologies and Networking (CommNet)*, pp. 1–10, IEEE, 2018.

[38] H. Abualola, H. Otrok, H. Barada, M. Al-Qutayri, and Y. Al-Hammadi, "Matching game theoretical model for stable relay selection in a uav-assisted internet of vehicles," *Vehicular Communications*, vol. 27, p. 100290, 2021.

[39] N. M. Al-Kharasani, Z. A. Zukarnain, S. K. Subramaniam, and Z. M. Hanapi, "An adaptive relay selection scheme for enhancing network stability in vanets," *IEEE Access*, vol. 8, pp. 128757–128765, 2020.

[40] M. al Mojamed and M. Kolberg, "Olsr optimisation for lightweight manet internet integration," in *2019 12th International Conference on Information & Communication Technology and System (ICTS)*, pp. 141–145, IEEE, 2019.

[41] M. Usha and B. Ramakrishnan, "An enhanced mpr olsr protocol for efficient node selection process in cognitive radio based vanet," *Wireless Personal Communications*, vol. 106, no. 2, pp. 763–787, 2019.

[42] I. Purnama, E. Setijadi, M. Purnomo, *et al.*, "Minmax algorithm for mpr selection in improving the performance of olsr protocol on manet," in *2018 International Conference on Intelligent Autonomous Systems (ICoIAS)*, pp. 136–140, IEEE, 2018.

[43] M. Namdev, S. Goyal, and R. Agarwal, "An optimized communication scheme for energy efficient and secure flying ad-hoc network (fanet)," *Wireless Personal Communications*, vol. 120, no. 2, pp. 1291–1312, 2021.

[44] W. Feng, F. Wang, D. Xu, Y. Yao, X. Xu, X. Jiang, and M. Zhao, "Joint energy-saving scheduling and secure routing for critical event reporting in wireless sensor networks," *IEEE Access*, vol. 8, pp. 53281–53292, 2020.

[45] W. A. Jabbar, M. Ismail, R. Nordin, and R. M. Ramli, "Ema-mpr: Energy and mobility-aware multi-point relay selection mechanism for multipath olsrv2," in *2017 IEEE 13th Malaysia international conference on communications (MICC)*, pp. 1–6, IEEE, 2017.

[46] S. Li, F. Wang, J. Gaber, and Y. Zhou, "An optimal relay number selection algorithm for balancing multiple performance in flying ad hoc networks," *IEEE Access*, vol. 8, pp. 225884–225901, 2020.

[47] M. Hayati, H. Kalbkhani, and M. G. Shayesteh, "Energy-efficient relay selection and power allocation for multi-source multicast network-coded d2d communications," *AEU-International Journal of Electronics and Communications*, vol. 128, p. 153522, 2021.

[48] R. Jain and I. Kashyap, "Energy-based improved mpr selection in olsr routing protocol," in *Data Management, Analytics and Innovation: Proceedings of ICDMAI 2019, Volume 1*, pp. 583–599, Springer, 2020.

[49] M. Kadadha and H. Otrok, "A blockchain-enabled relay selection for qos-olsr in urban vanet: A stackelberg game model," *Ad Hoc Networks*, vol. 117, p. 102502, 2021.

[50] Y. Inedjaren, M. Maachaoui, B. Zeddini, and J.-P. Barbot, "Blockchain-based distributed management system for trust in vanet," *Vehicular Communications*, vol. 30, p. 100350, 2021.

[51] A. Nabou, M. D. Laanaoui, and M. Ouzzif, "New mpr computation for securing olsr routing protocol against single black hole attack," *Wireless Personal Communications*, vol. 117, pp. 525–544, 2021.

[52] S. Jayaraman, U. Mohanakrishnan, and A. Ramakrishnan, "A trusted water fall model for efficient data transmission in vanet," *Wireless Personal Communications*, vol. 120, no. 1, pp. 821–848, 2021.

[53] X. Jian, P. Leng, Y. Wang, M. Alrashoud, and M. S. Hossain, "Blockchain-empowered trusted networking for unmanned aerial vehicles in the b5g era," *IEEE Network*, vol. 35, no. 1, pp. 72–77, 2021.

[54] S. G. Qureshi and S. K. Shandilya, "Novel fuzzy based crow search optimization algorithm for secure node-to-node data transmission in wsn," *Wireless personal communications*, pp. 1–21, 2021.

[55] P. Yellanki and M. P. Narasimham, "Secure routing protocol for vanets using ecc," in *2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, pp. 1–5, IEEE, 2020.