Chapter 7

Design and Development of Routing Algorithm for a Clustered Network

7.1 Introduction

Routing remains a challenging problem, particularly in mobile ad hoc networks which are considered to be less dynamic in our issue. This is due to the limited spectrum, user’s mobility and power constrains. There are several challenges: scalability, routing efficiency, ad hoc network of various levels of density and topology. For instance, scalability poses considerable challenges for ad hoc environment because it lacks the inherent hierarchy in the address structure. That is, in an ad hoc network, two neighboring nodes might have completely different addresses or/and identifiers. There are mainly two types of proposals specially designed for ad hoc routing to improve scalability: on-demand routing protocols \[77, 78\] and geographical routing schemes \[79, 80, 81\]. The on-demand routing does not require any prior-processing for route establishment, instead uses route request flooding to all nodes in the network in order to establish the route on demand. This often relies on the computation of the shortest path between
a source and a destination, and tends to work well for small or moderate size system with relatively stable routes. However, such scheme does not scale well due to the significant overheads in terms of both delay and flooding in large networks. The basic idea in geographical routing is to use a node’s location as
the address, and forward packets based on a predefined routing metric, usually the geographic distance. The greedy nature comes from the fact that such algorithms usually forward packets only based on the decrease of this metric in each step without considering complete topological information. The geographical routing achieves good scalability in that each node only needs to be aware of the neighbors’ location information, and does not rely on the flooding to exploit network topology. However, there is one serious limitation for geographical routing: the dead end problem, especially under low density environment or scenarios with obstacles or holes. The dead end problem is caused by the inherent greedy nature of the algorithm in that a packet may get stuck at a local optimal node that appears closer to the destination than any of its known neighbors under the pre-defined routing metric, which however, suffers the same dead end problem. In this chapter, we aim to design new routing protocols to solve the dead end problem without sacrificing routing efficiency, even for sparse ad hoc network with various topologies and obstacles. For routing efficiency, we seek for the shortest path route performance as that of the on-demand routing. To the best of our knowledge, we are among the first to achieve all these properties in one system with small overhead. A novel routing algorithm is proposed, utilizing a new virtual coordinate, called Hop ID. Each node maintains a Hop ID, a multi-dimensional coordinate, which are assigned based on its distance to all the landmarks nodes randomly selected from the network. With a predefined distance function, two nodes can calculate the ”distance” between them with respect to the landmarks. Based on this Hop ID metric, the routing algorithm performs greedy forwarding, similar to the geographic forwarding, i.e., a node forwards the packet to a neighbor which is ”nearest” to the destination.
in the Hop ID space. But in contrast to traditional geographical routing, such schemes effectively avoid the dead ends, even for very sparse network. Routing algorithm may consist of routing from source to its CH, from the CH to the CH of destination node, and from the later CH to the destination. Communication between CHs involves wireless communication which is an additional cost considered for an efficient shortest path switching from CH of source to CH of destination routing in a network with non-overlapping clustered network. To reduce the power consumption in CH nodes, the information about all CHs may be replicated to CHs of all the cluster of the network. Each node knows the content (i.e. the neighbor list of nodes) only for its own cluster. The sender may forward directly to the CHs of its own if single neighbor node exists. Otherwise randomly it has to select one of its neighbor nodes based on some conditions and from there to the CHs of destination and then to the destination. Moreover, the routing paths necessarily have to pass through at least one CH or at most two CHs, since the message can be routed toward the next cluster as soon as the destination is in another cluster through the CHs.

7.2 Related Works

Numerous routing protocols have been proposed in recent years. One of the most popular techniques for routing in communication networks is via distributed algorithms for finding shortest paths in weighted graphs [82], [83], [84], [85], [86], [87], [88]. These distributed algorithms differ in the way the routing Tables at each host are constructed, maintained and updated. The primary attributes for any routing protocol are:

- Simplicity: Simple protocols are preferred for implementation in operational networks [48].

- Loop-free: At any moment, the paths implied from the routing tables of
all hosts taken together should not have loops. Looping of data packets results in considerable overhead.

- Convergence characteristics: Time required to converge to new routes after a topology change should not be high. Quick convergence is possible by requiring the nodes to frequently broadcast the updates in the routing tables.

- Storage overhead: Memory overhead incurred due to the storage of the routing information should be low.

- Computational and transmission overhead: It is particularly important to limit these two in mobile wireless networks because the bandwidth of a wireless link is limited, and because mobile devices are typically low-power in order to be portable, and hence do not have the resources for many transmissions and lengthy computations.

Before presenting the Hop ID routing algorithm, the main motivations is first discussed and put them in the proper context with the related works. Routing is a recursive procedure to forward packets “closer” and “closer” to the destination. The most critical component in any routing algorithm is how to measure the “distance” between two nodes. This distance metric to a large degree determines the route performance, yet how to select this metric is non-trivial. Hop count or the shortest path distance is a natural candidate, since packets are forwarded on a hop-by-hop basis. But this poses considerable difficulty in ad hoc networks in that it incurs significant overhead to find and maintain the shortest path. On-demand routing algorithm [77], [78] and proactive routing protocol [89], [90] are typical examples using hop distance (i.e., the length in hops of the shortest path between a pair of nodes) as the routing metric. There have been other metrics proposed to measure the “distance” between two nodes such as geometric distance, last encountered time [91], and ID space distance [92]. Geographic routing uses geometric distance as the distance metric, and it is greedy
Routing Algorithm for a Clustered Network

in that each node forwards a packet to a neighbor with shorter distance to the
destination. Geographic routing does not incur explicit route discovery using
flooding; instead it only requires obtaining the position of the destination and
neighbors. Geographic routing in general consists of three parts:

1. Greedy routing algorithm
2. Dead ends resolution, and
3. Location service.

The existence of dead end is a well-known problem for geographic routing, in
which pure greedy algorithms hardly work in sparse networks or scenarios with
obstacles or holes. Many protocols, such as GPSR/GFG [79],[80] used face
routing technique to overcome dead end problem, but usually is at the expense
of much longer routing path. GOAFR+ [81] made an attempt in enhancing
face routing performance. In sparse networks, the fundamental problem in
geographic routing is that geometric distance can hardly reflect the true hop
distance between two nodes, thus often lead to dead end problem. Face routing
mitigates this problem at the cost of longer routing path. In fact, the routing
path can be several times longer than that of the shortest path length [81].
Another well-known limitation of geographic routing is that it requires GPS
or other location devices to obtain relatively precise location information. For
geographic routing, exact location might not be required and imprecise virtual
coordinates accordant to the network topology may perform better than the
real coordinates system. Under such motivation, recently Rao et al. made a
first attempt for geographic routing without location information [93]. They
proposed a virtual coordinate construction algorithm, which achieves comparab
performance with the real geometric coordinates in dense networks. It was
also shown in [93] that the virtual ordination has potential in the environment
with obstacles or holes, as virtual coordinates can better reflect the connectivity
Routing Algorithm for a Clustered Network

than real coordinates. But [93] performs badly in sparse network because its greedy success rate drops quickly and the dead end problem becomes more and more serious.

7.3 Efficient Routing With HOP ID

To design a scalable and efficient routing scheme for mobile ad hoc network, we observed that a pre-defined distance metric in geographical routing is the key to obtain scalability, in that it does not require any flooding or requires minimum flooding to explore the route discovery. On the other hand, the accuracy of the pre-defined distance metric representing the hop distance determines the route performance. In other words, if the greedy metric can more accurately reflect the hop distance, the route performance will be closer to that of the shortest path routing. This is precisely the problem in the existing geographical routing algorithms, where in sparse networks or scenario with obstacles or holes, the correlation between the geometric distance and hop distance subdues, thus it results in significantly more dead ends and unnecessarily longer route paths. To address these problems, in this Section, the Hop ID based routing is presented. Basically, to construct a multidimensional coordinates system, called Hop ID system, and use corresponding distance function to calculate the Hop ID distance between a pair of nodes. A node’s position, i.e. its Hop ID, is a vector, in which each dimension is the hop distance from the node to a pre-selected landmark node. Hop ID distance (vector) between two nodes is calculated from the relative hop distances to the set of landmarks. The result demonstrates that the Hop ID distance closely resembles the hop distance. In addition, Hop ID construction has a requirement from the density of clusters in the network, thus the routing protocol works well under both high and low density environments. Compared with existing proposals, in particular the virtual coordinates in [r9], the proposed Hop ID system obtains comparable performance under
Routing Algorithm for a Clustered Network
dense environment. Hop ID routing is also one type of geographic routing, thus it requires the careful design of the three parts identified in Section 5.4. Firstly, this needs a routing algorithm based on a pre-specified distance metric; to construct a multidimensional virtual coordinate, Hop ID system, which relies on the elected landmark nodes to compute the Hop ID distance between a pair of nodes. Secondly, formation of a non-overlapping cluster based on Hop ID. Third, classification of nodes inside the cluster based on Hop ID. Fourth, generating the Hierarchical code for cluster head for an efficient switching from one CH to another CH. Finally, path planning from source to destination based on the Hop IDs.

7.4 Efficient Path Planning by using LMUAC Routing scheme

To achieve an efficient path planning the following features are to be addressed and solved. The algorithms are developed and presented here to perform an efficient path planning.

- Path planning from Member node to CH of Source[FORWARDING].
- Switching between CHs at level 1 (w.r.t Source and Destination Hop ID)[SWITCHING].
- Path planning from CH of Destination to Destination node[ROUTE DISCOVERY].

\textbf{case(a) Single source to Multi Destination(via CHs)}

1. Covering entire cluster
2. Covering up to \(d\) Hop Destinations from \(CH_D\)
3. Covering from \(d\) Hop to \(e(CH_D)\)
4. Covering from $d$ Hop to $d$ Hop

5. Covering only $d$ Hop Destinations from $CH_D$

case (b) Single source to single destination (via CHs)

1. Position Based Routing

2. ACK/TTL based

The above addressed algorithm explores the following features

- Multi hop Network
- Distributed Routing Algorithm
- It is on demand routing protocol
- It helps in tracing the path from source to destination more effectively through clusterheads (wireless)

The purpose of designing and developing the routing algorithm is to effectively achieve path planning from source to destination.

7.5 Parameters:

- $R_{Start}$ is a parameter which tells the distance from the cluster head beyond which the nodes in cluster should receive and forward the data packets keeping a copy with them. i.e., $R_{Start} = d$ means only those nodes which are either at or beyond the distance $d$ from the cluster head should receive and forward the packets keeping a copy. This parameter is set at the cluster head of each cluster.

- $R_{Stop}$ is a parameter which tells the distance beyond which no node should receive the packets. All received nodes will keep a copy and forward.
• $F$ is a parameter which stores an integer value say $d$ and tells that nodes should receive and forward packets. This is done with out keeping a copy by every node at distance less than $d$.

• $PW$ - Path weight is a parameter which stores an integer value which is initially zero at every member node. Once the destination node $u_d$ is identified, assign a non zero integer called $k$, path weight, to $u_d$. (which is at distance $d$ from the cluster head ). Then find a node say $u_d - 1$ with $j^{th}$ Co-ord value ($d - 1$) in the neighbor Table of $u_d$. Assign the path weight $k$ to $u_d - 1$. Continue this procedure till we reach $u_0$ which is the cluster head. In this procedure, at any stage, the ties are broken at random.

• $D$ is the parameter which can have value either 0 or 1. If $D = 0$, then single cluster head communication. If $D = 1$, then distribute to all the cluster heads.

• Neighbor – Table : This Table is a matrix which is associated with the vertex $v$ belonging to cluster $C_j$. Neighbor Table list is a $2(K + 1)$ array where $K$ is degree of $v$ in $G$. In every row, in the $1^{st}$ column the unique clustering Hop Id of the corresponding vertex is stored and in the $2^{nd}$ column the $j^{th}$ Co-ord of that vertex is stored. In the $1^{st}$ row $1^{st}$ column source ID is stored and in $1^{st}$ row $2^{nd}$ column the $j^{th}$ Co-ord of vertex $v$ is stored. In the remaining $K$ rows ID of the neighboring vertices and then $j^{th}$ Co-ord are stored as follows. From top to bottom, those nodes having the same $j^{th}$ Co-ord as that of $v$ are stored in the beginning, those having $j^{th}$ Co-ord one less than that of $v$ are stored next and finally, those having $j^{th}$ Co-ord one more than that of $v$ are stored. Table 7.1 shows the neighbor list with the vertex.

Assumptions:

• A node is aware of all its neighbors at all times
Table 7.1: Neighbor Table

<table>
<thead>
<tr>
<th>Source ID of $v$</th>
<th>$j^{th}$ Co-ord of Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neighbor ID of $v$</td>
<td>same as $j^{th}$ Co-ord of Source</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Neighbor ID of $v$</td>
<td>$j^{th}$ Co-ord of Source-1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Neighbor ID of $v$</td>
<td>$j^{th}$ Co-ord of Source+1</td>
</tr>
</tbody>
</table>

- All packets transmitted over a link are received correctly and in proper sequence within a finite time.
- All control message are processed one at a time at the nodes in the order in which they occur.
- Communication between cluster heads is wireless (which is an additional cost).

7.6 Role of Member Node

- To know its own $N$-Tuple $v_i$.
- Maintain neighbor member list of that cluster.
- Maintain the $j^{th}$ co-ordinate of neighbor member list of that cluster $C_j$.
- If a member node say $v$ is a source initiated forward the data packet associated with the destination ID to lesser $j^{th}$ co-ord of the member say $u$ from the member Neighbor $-$ Table of $v$ by random selection.
- If received from the lower $j^{th}$ co-ord of the member node say $u$ then forward data packets associated with the destination ID to the higher $j^{th}$ co-ord of
Routing Algorithm for a Clustered Network

the member node say \( w \), from the member \( \text{Neighbor} \ - \ Table \ w \) list based on the path weight, \( R_{\text{stop}}, R_{\text{Start}}, F \) and \( \text{TTL} \).

- If received from the lower member node say \( u \), where \( R_{\text{stop}} \) value is \( > j^{th} \) co-ord of that member node \( v \) and \( R_{\text{Start}} \leq \text{the } j^{th} \) co-ord value of member node \( v \) then \( v \) will receive, keep a copy and forward, if \( F = \infty \).

- If received from the lower member node say \( u \) and \( R_{\text{stop}} \) value is \( = j^{th} \) co-ord of that member node say \( v \) and \( R_{\text{Start}} \leq \text{the } j^{th} \) co-ord value of the member node \( v \), then \( v \) will receive a packet, keep a copy and stop, if \( F = \infty \).

- If received from lower member node say \( u \) and if (\( F = \text{Val} = j^{th} \text{Co-ord of member node } v \)) \( \& N - \text{Tuple}_i == \text{destID} \) then \( v \) will receive a copy and stop else (no copy) simply forward. Set path weight of \( v \), \( PW = \text{Val} \neq \text{zero} \).

- If received from the higher member node say \( u \) and \( PW(u) \neq 0 \), then set \( PW(v) \neq \phi \), append this \( PW \text{val} \) to that member node \( v \) having the \( j^{th} \) Co-ord one lesser than from received member node and forward to the member node \( w \) having the \( j^{th} \) Co-ord value having one lesser than the received member node. (single source and single Destination).

- If received from the lower \( j^{th} \) Co-ord of the member node say \( u \) and if path weight \( PW \neq \phi \), then receive from the member node \( u \) having \( j^{th} \) Co-ord lesser than the \( j^{th} \) Co-ord say \( v \) then receive and forward to the member node say \( w \) having \( j^{th} \) greater than one with respect to the received member node and reset value of \( PW(v) = 0 \), (single source and single Destination).

- If received from the lower \( j^{th} \) co-ord of the member node say \( u \) and if \( \text{TTL} \neq \phi \) then decrement the value of \( \text{TTL} \leftarrow \text{TTL} - 1 \) forward to the
member node say \( v \) which is having the \( j^{th} \) Co-ord greater than present. i.e, \( j^{th} \) Co-ord \( \leftarrow j^{th} \) Co-ord +1. This procedure follows until \( TTL = 0 \) (Single Source and Single Destination)

### 7.7 Role of Cluster head

- To know its own ID, \( C_j, e(C_j) \), members list
- Maintain neighbor list
- Maintain the modulator and demodulator hardware of \( N \) Frequencies
- Maintain the \( j^{th} \) Co-ord of the neighbor Table
- Switching between \( CH_s \) if source and destination are in different cluster, by using Hierarchical code/pos value. Modulation \( val = pos \).
- Switch from source cluster head to destination cluster head if \( D = 0 \) (destination/destinations in single cluster assume \( j^{th} \) cluster ) the modulation \( val = j \), Demodulator \( val = j \) at \( j^{th} \) cluster.
- Switch from source cluster head to destinations cluster heads if \( D = 1 \) (communicate to all cluster heads) modulator \( val = g \), \( g \) =general, where all cluster head can demodulate that data.
- After Demodulation of data packets at cluster head level by \( val = g \) or \( val = j \) then forward to 1 hop member node from the neighbor Table.
- Set the parameters \( R - \text{Start}, R_{\text{Stop}}, D, F \) and \( TTL \).
- Receive data packets from the member node if initiated from the member node and modulate by \( val = pos \) or \( val = g \).
  - Single Source and Multi destination in the cluster
Routing Algorithm for a Clustered Network

1. Set parameters $R_{\text{Start}}$, $R_{\text{Stop}}$ and $F$ values
2. $R_{\text{Start}}$ can have the values $0, d$ and $\hat{d}$
3. $R_{\text{Stop}}$ can have values $d, e(C_j)$
4. $F$ can have value $\infty$ or $\text{val}$

   - Single source and Single Destination in the cluster
   1. $F = \text{val}$, just forward upto that value only.
   2. $R_{\text{Start}}$ and $R_{\text{Stop}} = \infty$
   3. For position based $F = \text{val}$ then compare the destination ID which is at $\text{val}$ hops, if equal then receive.
   4. Set $TTL$ value for $ACK$ based routing if destination is at $d$ distance then set the value of $TTL = d$.

7.8 LMUAC routing schemes

Let $G$ be a graph and $G = (V, E)$ and let $u, v \in V$. Let $u$ be a source and $v$ be a destination in vertex set $V$ of a network. Let $u$ be a vertex in a cluster $C_j$, $u \in C_j$ which is a member node having $j^{th}$ Co-ord non zero. The Land Marks for Unique Addressing and Clustering (LMUAC) routing algorithm is developed for path planning from source vertex $u$ to destination vertex $v$ which consists of three phases a) Forwarding, b) Switching and c) Route discovery. First and foremost in a network by using LMUAC Algorithm unique ID is generated for clustering and assigned to each and every node. Clustering Algorithm is executed to form non-overlapping clusters and the nodes are identified as a member node, cluster head and gateway node by next Algorithm 6.9.1 all these executions are carried out during off line conditions. The resultant is each and every node is now associated with unique clustering ID. Now our aim is to path planning from source $u$ to destination $v$. The first phase of the LMUAC routing
Routing Algorithm for a Clustered Network

algorithm is **forwarding** data packets from source \( u \) to the cluster head of that cluster \( C_j \). The second phase is **Switching** if the source and destination nodes are in different clusters through the cluster heads. The third and final phase of LMUAC routing is route discovery from CH to destination node, it may be either in the same cluster or in a different cluster. Each phase of the LMUAC routing is explained in detail.

7.8.1 FORWARDING

Let \( u \) be a vertex in \( C_j \) and if it is a source initiated then the data packet which is associated with the destination ID is forwarded from the source to the cluster head through intermediate nodes if the \( j^{th} \) Co-ord of the vertex \( u > 1 \). The source will verify from the neighbor Table of \( u \) and check for the neighbor having the \( j^{th} \) Co-ord \( \rightarrow (j^{th}\text{Co-ord of} u - 1) \) and forward to that member node. The ties are broken in random. This procedure will continue until the \( (j^{th}\text{Co-ord} y) = 0, y \in C_j \). If the \( j^{th} \) Co-ord of \( y = 0 \), then \( y \) is a cluster head. Stop forwarding the data packets associated with the destination ID once it has reached the cluster head this process is known as forwarding. which is as shown in the following Figures. The algorithm which 7.8.1.1 is developed to forward the data packets from source to cluster head.

7.8.1.1 Algorithm to forward the data packet from the member node to the Cluster head of that cluster \( C_j \)

---

1: **Step 0**: \[ \text{INPUT } ((\text{NTuple } v_i)_{N+1})_s, \text{which is a source member node} \]

2: \[ = ((Co - ordv_i)_{1}, (Co - ordv_i)_{2}, \ldots, (Co - ordv_i)_{N}, pos) \]

3: **Step 1**: \[ (Co - ordv_i)_{pos} \neq 0 \]

4: **Step 2**: \[ d = \text{is the number of hops to that CH from MN} \]

5: \[ \text{IF } ((pos v_i) == (pos v_j)) \text{and} (|(Co - ord v_i)_{pos}| = |(Co - ord v_i)_{pos} - 1|) \]

6: \[ \text{IF } ((Co - ord v_j) == 0) \]

7: \[ \text{EXIT} \]
Routing Algorithm for a Clustered Network

10: **Step 3:** Forward data packets from member node to Cluster head
11: \[ \text{FOR } 2 \leq j \leq d, \text{ } d \text{ is the number of hops from CH of that cluster} \]
12: \[ \text{IF } ((posv_j) == (posv_{\hat{j}}) \text{ and } (((C_o-ord v_j)_{pos}) = (((C_o-ord v_{\hat{j}})_{pos} - 1))) \]
13: \[ \{ \text{Forward the data packet from } v_j \rightarrow v_{\hat{j}} \}
14: \]
15: \[ j \leftarrow \hat{j}, \hat{j} \leftarrow k, \]
16: \[ \text{if}((C_o-ord v_j) == 0) \]
17: \[ \text{EXIT } \}
18: **Step 4:** Output
19: \[ \text{Path tracing from Member node to Cluster head} \]

![Diagram](image.png)

**Figure 7.1:** Forwarding

From the Figure 7.1 it is known that a source is a member node say \( u \) initiated which is at hop distance 3 hop from the cluster head \( C_j \). The \( j^{th} \) Co-ord of the source initiated is 3 which is not a cluster head, it will see its neighbor Table 7.2 and select a neighbor member node which is having the \( j^{th} \) Co-ord lesser than the source. There exist two member nodes having the \( j^{th} \) Co-ord lesser than one, so a tie exists. The tie is broken randomly and one of them is selected. Once the neighbor node is identified the data packets associated
with the destination ID is forwarded for the selected neighbor. This process continues until the data packet reaches the cluster head. When it reaches the cluster head the process of moving the data packets is stopped. The first phase of the algorithm called forwarding is completed. In the end the data packets along with the destination ID is available at the cluster head of that cluster $C_j$.

### 7.8.2 SWITCHING

It is the second stage of LMUAC Routing Algorithm that is used to switch between cluster heads if both source and destination are in different clusters. The communication between the cluster heads are carried out based on whether there is a single destination or multiple destination which is discussed in the next sections.

#### 7.8.2.1 Switching between Source Cluster Head to Destination Cluster head (Single Source and Single Destination)

Once the data packets are available at the cluster head of that cluster $C_j$, then the process starts to identify the destination ID from the received data packets. Whether the switching decision is to be carried out or not, is decided by either Hierarchical code of cluster heads or the $posvalue$ of source and destination ID. If the $posvalue$ of destination is the same as the value of cluster $C_j$ then no switching is carried out. If it is different then the switching is required from one cluster to another cluster. The major assumption is by considering the wireless
communication between the cluster heads. The wireless communication is carried out between the cluster heads, then all the cluster heads with all the cluster heads should be in the desired communication range to receive the modulated data and demodulate the data at the receiving cluster head. The modulation is carried out by the source cluster head and demodulation by the destination cluster head. If the switching is to be carried out to all the clusters then the data is to be modulated by the general frequency where all the cluster heads are capable of demodulating the general frequency. The cluster head is also capable of modulating the data with all the $pos$ value of cluster heads individually similarly demodulating them. This is a overhead cost for constructing this LMUAC to work efficiently. If $N$ clusters are there must be frequencies available for cluster to cluster communication, $N$ frequencies for communicating for the individual cluster heads and one unique frequency for communication to all the cluster heads. The switching can be performed by two ways. The algorithm 7.8.2.2 is developed to generate the hierarchical code.

1. By Hierarchical code of Cluster Heads Let $G(V, E)$ be a graph with $n$ vertices and $N$ clusters in a network. Let $u$ be a source vertex then the $(\text{Co-ord}_N - \text{Tuple } u)_{N+1} = \text{pos val}$. The pos val indicates the minimum value in the clustering unique ID. If pos val = $j$, then the minimum value is in $j^{th}$ position. If the minimum value is in $j^{th}$ position then $u \in C_j$. The $(\text{Co-ord}_N - \text{Tuple } u)_j \neq 0$, then vertex $u$ is not a cluster head. Let $v$ be a vertex $v \in C_j$. Where the $(\text{Co-ord}_N - \text{Tuple } v)_j = 0$, then $v$ is a cluster head. A unique special code called hierarchical cluster head code is generated for the cluster heads to perform switching. The hierarchical code of the cluster head with respect to $j^{th}$ cluster is in its unique code it has a combination of exactly one zero and $(N-1)$ 1s. It has zero in $j^{th}$ position and 1s in the remaining position. $HCH_j = \left\{ 1, 1, 1, \ldots, \underbrace{0}_{j^{th}-\text{position}}, \ldots, 1 \right\}$ $(Co - \text{ord} HCH_j)_j = 0$ and
Routing Algorithm for a Clustered Network

Let $x$ be a destination vertex $x \in C_k$ where $k \neq j$, then $x$ is in a different cluster the $(Co - ordN - Tuplex)_{N+1} = posval$. The $posval = k$, $j \neq k$ and $(Co - ordN - Tuplex)_k \neq 0$, then it is a member node of $C_k$. Let $y$ be a cluster head of cluster $C_k$, then the $(Co - ordN - Tupley)_k = 0$. The hierarchical code for cluster head $y$ is having zero in $k^{th}$ position only and 1s in the remaining position. $HCH_k = \{1, 1, 1, \ldots, 0, \ldots, 1\}$

and $(Co - ordHCH_k)_i = 0$ and $(Co - ordHCH_k)_i = 1, i \neq k, 1 \leq i \leq N$

If $u \in C_j$ and $y \in C_k$, then both source and destination are in different clusters. Switching is to be performed from $j^{th}$ cluster to $k^{th}$ cluster, the modulation frequency is $k$ from $CH_j$ and demodulation frequency at $c-k$ is frequency $k$. To know from which cluster it has to switch, consider the $HCH_j$ and $HCH_k$ hierarchical code and XOR them, check the values of 1s in the resultant. Switch from source to destination.

$HCH_j = \{1, 1, 1, \ldots, 0, \ldots, 1\}$

X or

$HCH_k = \{1, 1, 1, \ldots, 0, \ldots, 1\}$

The resultant is $= \{0, 0, 0, \ldots, 1, \ldots, 0, 0, 0\}$

Switch from $j^{th}$ cluster to $k^{th}$ cluster. The data packets along with the destination ID is switched from source to destination by using modulating frequency $k$, all the cluster heads are in the visible range.

2. By using source and destination pos value
Let \( x \in C_j \) and \( y \in C_k \), then the \textit{posval} of \( x \) is in the \( j^{th} \) position and the \textit{posval} of \( y \) is in \( k^{th} \) position then switch from source position to the destination position with modulating frequency (destination \textit{posval} ) and demodulating frequency at \( k^{th} \) cluster is (destination \textit{posval} ).

\[ \text{Let } x \in C_j \text{ and } y \in C_k, \text{ then the posval of } x \text{ is in the } j^{th} \text{ position and the posval of } y \text{ is in } k^{th} \text{ position then switch from source position to the destination position with modulating frequency (destination posval ) and demodulating frequency at } k^{th} \text{ cluster is (destination posval).} \]

### 7.8.2.2 Algorithm for switching between Clusterhead at level 1 (w.r.t source to destination)

1: Switching

2: \textbf{Step 0: Input}((NTuple \( v_i(N+1) \)), and((NTuple \( v_j(N+1) \))), \( i \neq j \)

3: \( 1 \leq j \leq N-1 \), if \( j=1 \), single destination, \( j \neq 1 \), multiple destination.

4: \textbf{Step 1:} \((\text{pos})_s \leftarrow (\text{Co - ordNTuple } v_i(N+1))\)

5: \((\text{pos})_{dj} \leftarrow (\text{Co - ordNTuple } v_j(N+1))\)

6: \textbf{Step 2: FOR} \( 1 \leq j \leq N-1 \)

7: IF\((\text{pos})_s == (\text{pos})_{dj}\)

8: \{switching=0, no switching\}

9: Else

10: \{SwitchingS \rightarrow D, Switching from (pos)s TO (pos)\( dj \}\}

11: EXIT

12: \textbf{Step 3: Output}

13: Switching from (pos)s TO (pos)\( dj \)

### 7.8.3 ROUTE DISCOVERY

This is the last stage of the LMUAC Routing algorithm. The decision of routing is decided by the applications which are discussed separately in the next sections. The Route discovery is entirely dependent upon the parameter settings which is listed in the following below Table 7.3 and 7.4. Based on this parameters we characterize the applications which can be applied to the route formation. The routing algorithm is developed for every applications. The step 3 in the general routing algorithm only varies. Let \( s \) be the source and \( N \) be the number of clusters in a given graph \( G \). Let \( s \in C_i \), where \( i = 1toN \), \( 1 \leq j \leq N, 1 \leq k \leq N \), let \( j \) be the destination cluster, switching is \( S \).

If the communication is to be carried out for all the clusters, then replace
Routing Algorithm for a Clustered Network

Table 7.3: Parameters setting for $C_j$ entire Cluster (Similarly can be Generalised for Multi Destination)

<table>
<thead>
<tr>
<th>Start</th>
<th>Top</th>
<th>$F$</th>
<th>$S$</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$d$</td>
<td>$\infty$</td>
<td>$j/g$</td>
<td>Distribute Data up to $d$ hops of the cluster $C_j$ (Level Communication)</td>
</tr>
<tr>
<td>0</td>
<td>$e(C_j)$</td>
<td>$\infty$</td>
<td>$j/g$</td>
<td>Distribute Data for entire member nodes</td>
</tr>
<tr>
<td>$d$</td>
<td>$e(C_j)$</td>
<td>$\infty$</td>
<td>$j/g$</td>
<td>Distribute Data from $d$ to the tail of the cluster</td>
</tr>
<tr>
<td>$d$</td>
<td>$\hat{d}$</td>
<td>$\infty$</td>
<td>$j/g$</td>
<td>Distribute Data from $d$ to $\hat{d}$</td>
</tr>
</tbody>
</table>

Table 7.4: Parameters setting for $C_j$ entire Cluster (Similarly can be Generalised) (Single/multi Destination at a given hop distance from cluster head of that cluster $c_j$)

<table>
<thead>
<tr>
<th>Start</th>
<th>Top</th>
<th>$F$</th>
<th>$S$</th>
<th>$PW$</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$d$</td>
<td>$j$</td>
<td>0</td>
<td>Forward data up to $d$ hop and Distribute Data for those member nodes which are at $d$ hops of the cluster $C_j$</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$d$</td>
<td>$j$</td>
<td>0</td>
<td>Path weight set if destination is matched and forward the ACK signal to cluster head by forwarding</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$j$</td>
<td>Non Zero</td>
<td>Forward the data where $Pw$ is set</td>
</tr>
</tbody>
</table>
the value of $S$ in the Table by $g$. If the communication is to be carried out only in cluster $j$ then retain only $j$ in the column of $S$.

### 7.8.3.1 Algorithm to distribute the packets from Cluster Head to all the member nodes of that same cluster-covering the entire cluster

Let there be a member node belonging to the cluster $C_j$. The application is to distribute the data packets available at the cluster head of the cluster $C_j$. The cluster head will check the Table and append the parameters to the data packet and distribute them to the neighbor nodes from the cluster head. If the value of either $R_{\text{Start}}$ or $R_{\text{Stop}}$ value is non zero and the value of $F$ is $\infty$, the data distribution will take place from the value of the $R_{\text{Start}}$ and end when $R_{\text{Stop}}$ is reached. When ever data packets move from node to node the value is increased and check only the $j^{th}$ Co-ord of the $N-Tuple$ of the visited nodes. The next visiting node will always have greater $j^{th}$ Co-ord of the $N-Tuple$ compared with current position of the node if so then only there is visit otherwise no visiting takes place. If the value of $R_{\text{Start}} = 0$ and $R_{\text{Stop}} = e(C_j)$, then covering the entire cluster takes place. Example communication for the entire Group.

The below algorithm is developed to cover the entire cluster. The flow diagram which is shown in Figure 7.2 cover the entire cluster.

1: Initialize $e(CH_j)$. Eccentricity of $j^{th}$ cluster with respect to $CH_j$
2: **Step 0:** Input $(N\text{-Tuple } v_i)_{N+1}$ of $CH_j$ and $MN_j$
3: \[= (\text{Co-ord } v_i)_1, (\text{Co-ord } v_i)_2, \ldots (\text{Co-ord } v_i)_N, \text{pos}\]
4: **Step 1:** Cluster head is in $j^{th}$ cluster
5: Then $(\text{Co-ord } v_i)._j = 0, (\text{Co-ord } v_i)._j \neq 0$
6: **Step 2:** Receive /Distribute data from CH to MN
7: \[p_k = d(v_i, v_i) = 1, \text{then there are neighbor nodes}\]
8: **FOR** $1 \leq i \leq k$
9: IF \[((\text{pos} v_i) == (\text{pos} v_i))\] and
10: \[(\text{Co-ord} v_i)_j = ((\text{Co-ord} v_i)_{j+1})\] and
11: \[(\text{Co-ord} v_i)_j \leq e(CH_j))\]
12: Distribute packets from CH to $v_i$ for some $k$
13: **Step 3: Receive /Distribute data from MN to MN**
14: FOR $1 \leq k \leq e(CH_j)$
15: IF (($posv_i == (posv_k)$) and
16: $(Co - ordv_k)_j = ((Co - ordv_i)_j + 1)$ and
17: $(Co - ordv_k)_j \leq e(CH_j)$)
18: Distribute packets from MN $v_i$ to MN $v_k$ for some $k$
19: EXIT
20: **Step 4: Output**
21: Distribution of packets for entire cluster

![cluster received.jpg](image)

**Figure 7.2: Data Flow from Node to Node in a Cluster**

### 7.8.3.2 Algorithm to distribute the packets for all the member nodes of that Clusters-covering up to $d$ hops

Let there be a member node belonging to the cluster $C_j$. The application is to distribute the data packets available at the cluster head of the cluster $C_j$ only up to $d$ hops. The cluster head will check the Table and append the parameters to the data packet and distribute them to the neighbor nodes from the cluster head. If the value of either $R_{Start}$ or $R_{Stop}$ value is non zero and the value of $F$ is $\infty$, the data distribution will take place from the value of the $R_{Start}$ and end when $R_{Stop}$ is reached. If the value of $R_{Start} = 0$ and $R_{Stop} = d$, then covering up to $d$ hop of that cluster takes place. Example: Categorized or Level
communication of that Group. The below Algorithm is replaced in step 3 of the algorithm covering the entire cluster which is developed to cover the level only from the cluster head.

13:Step 3: Receive /Distribute data from MN to MN
14: FOR 1 ≤ k ≤ e(CH)_j
15: IF (((posv_i) == (posv_k)) and
16: (Co - ordv_k)_j = ((Co - ordv_i)_j + 1)) and
17: ((Co - ordv_k)_j ≥ d))
18: Distribute packets from MN v_i to MN v_k for some k
19: EXIT

7.8.3.3 Algorithm to distribute the packets for all the member nodes of that Clusters- from d Hops to e(CH)_j hops

Let there be a member node belonging to the cluster C_j. The application is to distribute the data packets available at the cluster head of the cluster C_j from d hops to e(CH)_j hops. The cluster head will check the Table and append the parameters to the data packet and forward them to the neighbor nodes from the cluster head. If the value of either R_Start or R_Stop value is non zero and the value of F is d, the data is forwarded up to d hops and from d hops the data distribution will take place from the value of the R_Start and end when R_Stop is reached. If the value of R_Start = d and R_Stop = e(CH)_j, then covering from d hop of that cluster to e(CH)_j takes place. Example Categorized or Level l to e(CH)_j communication of that Group, where l > e(CH)_j. The below algorithm is replaced in step 3 of the algorithm for covering the entire cluster, which is developed to cover the categorized level.

13:Step 3: Receive /Distribute data from MN to MN
14: FOR 1 ≤ k ≤ e(CH)_j
15: IF (((posv_i) == (posv_k)) and
16: (Co - ordv_k)_j = ((Co - ordv_i)_j + 1)) and
17: ((Co - ordv_k)_j ≥ d))
18: and ((Co - ordv_k ≤ e(CH)_j))
Routing Algorithm for a Clustered Network

7.8.3.4 Algorithm to distribute the packets for all the member nodes of that cluster- from $d$ Hops to $d'$ Hops

Let there be a member node belonging to the cluster $C_j$. The application is to distribute the data packets available at the cluster head of the cluster $C_j$ from $d$ hops to $d'$ hops. The cluster head will check the Table and append the parameters to the data packet and forward them to the neighbor nodes from the cluster head. If the value of either $R_{\text{Start}}$ or $R_{\text{Stop}}$ value is non zero and the value of $F$ is $d$, the data is forwarded up to $d$ hops and from $d$ hops the data distribution will take place from the value of the $R_{\text{Start}}$ and end when $R_{\text{Stop}}$ is reached. If the value of $R_{\text{Start}} = d$ and $R_{\text{Stop}} = d$, then covering from $d$ hop of that cluster to $d'$ takes place. Example Categorized or Level $l$ to level $m$ communication of that Group, where $l > m$. The below algorithm is replaced in step 3 of the algorithm which is developed for covering the entire cluster which is developed as shown to cover $l$ to $m$ levels. The data flow is as shown in the Figure 7.3.

13: Step 3: Receive / Distribute data from MN to MN
14: FOR $1 \leq k \leq e(CH)_j$
15: IF $((\text{pos}v_i) == (\text{pos}v_{i,k}))$ and
16: $((\text{Co} - \text{ord}v_{i,k})_j = ((\text{Co} - \text{ord}v_i)_j + 1))$ and
17: $((\text{Co} - \text{ord}v_{i,k})_j \geq d)$
18: and $((\text{Co} - \text{ord}v_{i,k})_j \leq d)$
19: Distribute packets from MN $v_i$ to MN $v_{i,k}$ for some $k$
20: EXIT

7.8.4 Single source and single destination

Case (i) position based routing.

Let $x$ be a destination vertex in the cluster $C_j$ which is at $d$ hops from the cluster head $CH_j$. If single destination is in the cluster $C_j$. The route discovery in the cluster $C_j$ is as follows. To find an efficient route to the single destination
Routing Algorithm for a Clustered Network

Figure 7.3: Data received from d hop Node to \( d \) Node in a Cluster

from the cluster head three steps are to be carried out. They are as follows:

1. Forward packet up to \( d \) hops destination.

   The vertex \( u \) which is at \( d \) distance from the cluster head is known. By setting the parameter value \( F \) the request is forwarded up to the member nodes which are at \( d \) hops far away from the cluster head. All the member nodes which are within \( d \) hops from the cluster head will forward the query along with the destination ID to the next member nodes which are having the \( j^{th} \) Co-ord greater, this procedure is followed until it reaches \( d \) hop from the cluster head.

2. Find destination. The query available for all the member nodes which are at \( d \) hops from the cluster head of the cluster \( C_j \). These query carry the destination ID along with the data packets. Once these data packets reach the \( d \) hops it compares the \( N - T \)uple of destination and if it exists receive the data packets, otherwise stop. The below algorithm is developed is shown below. The flow diagram is as shown in the Figure 7.4.
7.8.4.1 Algorithm to distribute the packets for all the member nodes of that cluster at \( d \) Hops

This following algorithm is developed to distribute the data packets for all the member nodes of a cluster which are at \( d \) hops in the cluster. The flow diagram is also shown in the Figure 7.4.

13: Step 3: **Forward data from MN to MN**
14: \[ \text{FOR } 1 \leq k \leq d \]
15: \[ \text{IF } ((\text{posv}_i) == (\text{posv}_k)) \text{ and } ((\text{Co} - \text{ordv}_k)(j) = ((\text{Co} - \text{ordv}_i)(j) + 1)) \text{ and } ((\text{Co} - \text{ordv}_k)(j) \leq d)) \]
16: \[ \{ \text{Forward data packets to } v_k \text{ for some } \hat{k} \} \]
17: \[ \text{IF } (\text{Co} - \text{ordv}_k == d) \]
18: \[ \{ \text{Receive packets and break } \} \]
19: \[ \text{Distribute packets from MN } v_i \text{ to MN } v_k \text{ for some } \hat{k} \]
20: \[ \text{EXIT} \]

Figure 7.4: Data received only at \( d \) hop Node of that cluster
7.8.4.2 Algorithm to distribute the packets to a destination node of that cluster- at \(d\) Hops

This algorithm is developed based on TTL and path weight. The below algorithm is developed and the Flow diagram is as shown in Figure 7.5.

13: Step 3: Forward data from MN to MN
14: FOR \(1 \leq k \leq d\)
15: IF \(((posv_i) == (posv_k))\) and \((Co - ordv_k)_j = ((Co - ordv_i)_j + 1)\) and \((Co - ordv_k)_j \leq d)\)
16: { Forward data packets to \(v_k\) for some \(k\)}
17: IF \((Co - ordv_k == d)\) and \((NTuplev_i)_d == (NTuplev_k)\)
18: { Receive packets and break } 
19: ELSE 
20: { Discard and exit }
21: Forward packets from MN \(v_i\) to MN \(v_k\) for some \(k\)
22: EXIT

Figure 7.5: Based on path weight
7.9 Path Planning from Source to Destination

Consider the below Figure 7.6 for path planning from source to destination by LMUAC algorithm. We use the LMUAC routing scheme for path planning from source to destination. The discussion is with single source to single destination. Let $S$ be a source, the Hop ID of the source be $[1,7,6,4]$, where $S \in C_1$. The radius of the cluster $C_1$ is $r_1$ is 3. Let $D$ be the destination, the Hop ID of the destination be $[6,1,5,6]$, where $D \in C_2$. The radius of the cluster $C_2$ is $r_2$ is 3. The distance from the cluster head $L_1$ to $S$ is 1. The distance from the cluster head $L_2$ to $D$ is also 1 hop. The length of the source and destination in the non-clustered network is 7 hops. The first stage is to construct non-overlapping clusters. For path planning from source to destination the first stage is forwarding i.e., moving from source to the cluster head of that cluster. Once it has reached the cluster head stop forwarding. The length of the source to the cluster head of that cluster is 1 hop forward the data packet from source to cluster head. If source and destination is in the different clusters switching takes place. Here in this example S and D are in different cluster. The switching takes place from cluster head of $C_1$ to Cluster head of $C_2$. Route discovery takes place from cluster head to the destination based on the parameters setting.

\[
\text{SOURCE} \stackrel{\text{Forwarding}}{\rightarrow} \text{CH}_1 \stackrel{\text{Switching}}{\rightarrow} \text{CH}_2 \stackrel{\text{RouteDiscovery}}{\rightarrow} \text{DESTINATION}
\]

The path length for forwarding is 1 hop, the path length from cluster head of source to cluster head of destination is 1 hop, and the path length from the cluster head to destination is 1 hop. The total path length from source to destination by LMUAC algorithm is 3 hops. Which is an efficient one compared to the non-clustered network.
Routing Algorithm for a Clustered Network

Figure 7.6: Path planning from source to destination
7.10 Results and Discussion

In this section the comparative studies related to the Average path length, Maximum link utilization and Memory space requirement are discussed. The parameters are to be chosen carefully for the best results to overcome the drawbacks. The next sections will discuss the general comparative studies for an upper bound.

7.10.1 The upper bound for the average path length

If source and destinations are in different clusters then the communication will take place through the cluster heads. In LMUAC routing scheme the wireless communication between cluster heads is assumed. Let \( u \) be the source cluster head and \( v \) be the destination cluster head, then from \( u \) to \( v \) switching will take place either by using the hierarchical or pos value switching. They are like fully connected (mesh) i.e, the cluster head will broadcast the data packet along with the destination ID, which is modulated by using pos value or hierarchical code. The hop count between cluster head to cluster head is at most one hop through wireless communication.

Let \( D \) be the diameter of \( G \) with \( N \) clusters. Let \( C_1, C_2, C_3, \ldots, C_N \) be \( N \) clusters, then let \( n_1, n_2, n_3, \ldots, n_N \) be the number of elements in \( C_1, C_2, C_3, \ldots, C_N \) respectively, then maximum diameter in the cluster be \( d_i \), then \( n_1 - 1, n_2 - 1, n_3 - 1, \ldots, n_N - 1 \) is the maximum distance with \( C_1, C_2, C_3, \ldots, C_N \) respectively. Let \( x \) and \( y \) be the nodes in a network. Then

**Case (i):** Let \( x \) and \( y \) be two nodes in the network with \( x \)-a member node and \( y \) the cluster head in the same cluster \( C_j \). Then the path length between \( x \) and \( y \) is at most \( (n_j - 1) \), which is shown in Figure 7.7

\[
PL = (n_j - 1)
\] (7.1)
Case (ii): Let \( x \) and \( y \) be two nodes in the network with \( x \) a member node with cluster \( C_j \) and \( y \) a cluster head with cluster \( C_k \). Then the path length between \( x \) any \( y \) is at most \((n_j - 1) + 1\), which is shown in Figure 7.8.

\[
PL = (n_j - 1) + 1 \tag{7.2}
\]

\[
PL = n_j \tag{7.3}
\]

Case (iii): Let \( x \) and \( y \) be two nodes in the network with \( x \) and \( y \) in the same cluster say \( C_j \) with \( n_j \) nodes. Then the path length between \( x \) any \( y \), through the cluster heads, is at most \( 2(n_j - 1) \), which is shown in Figure 7.9.

\[
PL = 2(n_j - 1) \tag{7.4}
\]

Case (iv): Let \( x \) and \( y \) be two member nodes in the cluster \( C_i \) and \( C_j \) respectively.
Routing Algorithm for a Clustered Network

Figure 7.9: $x$ and $y$ in the same cluster say $C_j$

tively. Then the path length between $x$ any $y$ cannot exceed $((n_i - 1) + (n_j - 1) + 1)$. Which is shown in Figure 7.10

$$PL = (n_i + n_j - 1) \quad (7.5)$$

In this we calculate the average distance one has to travel to move from any

vertex to any other vertex in the clustered scenario. Where $N$ is the total distance in a network

$$N = \sum_{i=1}^{k} C_{2}^{n_i-1}2(n_i - 1) + \sum_{i=1}^{k} \sum_{j=1}^{k} (n_i - 1)(n_j - 1)(n_i + n_j - 1) \quad (7.6)$$
Routing Algorithm for a Clustered Network

$k$ is the distances between vertices, then $k$ is given by

$$k = n(n - 1)$$ (7.7)

Average path length (Upper Bound)=$\frac{N}{k}$

Example: Consider a network of $n$ nodes in a network. Let $D$ be the diameter of the graph $G$. Let $N$ be the number of clusters in a graph $G$ with diameter $d$. The average path length is calculated as follows by considering the below example. We show that $PL_{N,d} < PL_D$ should be chosen.

Let there be a network of 1000 nodes in a graph $G$, with diameter $D = 50$ of a network. The number of clusters in a network is 10 of various diameters varying within 40,35,30,25,20,15 and 10. We calculate the average path length by varying the number of clusters from 2 to 110 and compare the network performance with respect to $N$ and $d$ with $D$.

Average Path length for a given $N$ and $d$ in a clustered network be $PL_{N,d}$

Average Path length for a non clustered network be $PL_D$

$$PL_{N,d} = \frac{[(d - 2)(d - 1)N + ((d - 1)^2(2d - 1)(N - 1)N)]}{n(n - 1)}$$ (7.8)

$$PL_D = \frac{Dn(n - 1)}{n(n - 1)} = D$$ (7.9)

$$PL_{N,d} < PL_D$$ (7.10)

is always preferred for an average $PL$ in an upper bound range. Based on the following Table 7.10.1 the following Figures 7.11 and 7.12 shows that this gives us a chance to make a proper choice of path length with respect to $(N,d)$. If the $PL_{N,d}$is $> PL_D$, then the average path length is increased.

Choose

$$PL_{N,d} < PL_D$$ (7.11)
Routing Algorithm for a Clustered Network

Figure 7.11: Clustered with Average diameter is greater than D/2

Figure 7.12: Clustered with Average diameter is less than or equal to D/2
Table 7.5: The Average Path Length Upper bound for 1000 nodes and Diameter is 50 and cluster diameter varies 40,35,30,25,20,15,10 respectively

<table>
<thead>
<tr>
<th>Clusters</th>
<th>$PL_{N,d_1}$</th>
<th>$PL_{N,d_2}$</th>
<th>$PL_{N,d_3}$</th>
<th>$PL_{N,d_4}$</th>
<th>$PL_{N,d_5}$</th>
<th>$PL_{N,d_6}$</th>
<th>$PL_{N,d_7}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>11.27</td>
<td>7.46</td>
<td>4.63</td>
<td>2.62</td>
<td>1.30</td>
<td>0.52</td>
<td>0.14</td>
</tr>
<tr>
<td>12</td>
<td>16.37</td>
<td>10.84</td>
<td>6.73</td>
<td>3.81</td>
<td>1.89</td>
<td>0.76</td>
<td>0.20</td>
</tr>
<tr>
<td>14</td>
<td>22.42</td>
<td>14.86</td>
<td>9.22</td>
<td>5.22</td>
<td>2.59</td>
<td>1.04</td>
<td>0.27</td>
</tr>
<tr>
<td>16</td>
<td>29.43</td>
<td>19.50</td>
<td>12.10</td>
<td>6.85</td>
<td>3.40</td>
<td>1.36</td>
<td>0.36</td>
</tr>
<tr>
<td>20</td>
<td>46.28</td>
<td>30.66</td>
<td>19.03</td>
<td>10.78</td>
<td>5.35</td>
<td>2.14</td>
<td>0.57</td>
</tr>
<tr>
<td>22</td>
<td>56.14</td>
<td>37.19</td>
<td>23.08</td>
<td>13.08</td>
<td>6.49</td>
<td>2.59</td>
<td>0.69</td>
</tr>
<tr>
<td>26</td>
<td>78.70</td>
<td>52.14</td>
<td>32.35</td>
<td>18.33</td>
<td>9.09</td>
<td>3.64</td>
<td>0.97</td>
</tr>
<tr>
<td>30</td>
<td>105.05</td>
<td>69.60</td>
<td>43.19</td>
<td>24.48</td>
<td>12.14</td>
<td>4.86</td>
<td>1.29</td>
</tr>
<tr>
<td>35</td>
<td>143.35</td>
<td>94.97</td>
<td>58.93</td>
<td>33.40</td>
<td>16.57</td>
<td>6.63</td>
<td>1.76</td>
</tr>
<tr>
<td>40</td>
<td>187.57</td>
<td>124.28</td>
<td>77.11</td>
<td>43.70</td>
<td>21.68</td>
<td>8.67</td>
<td>2.30</td>
</tr>
<tr>
<td>45</td>
<td>237.74</td>
<td>157.52</td>
<td>97.74</td>
<td>55.39</td>
<td>27.48</td>
<td>10.99</td>
<td>2.92</td>
</tr>
<tr>
<td>50</td>
<td>293.85</td>
<td>194.69</td>
<td>120.80</td>
<td>68.47</td>
<td>33.97</td>
<td>13.59</td>
<td>3.61</td>
</tr>
<tr>
<td>55</td>
<td>355.89</td>
<td>235.80</td>
<td>146.31</td>
<td>82.93</td>
<td>41.14</td>
<td>16.46</td>
<td>4.37</td>
</tr>
<tr>
<td>60</td>
<td>423.87</td>
<td>280.84</td>
<td>174.26</td>
<td>98.77</td>
<td>49.00</td>
<td>19.60</td>
<td>5.21</td>
</tr>
<tr>
<td>65</td>
<td>497.79</td>
<td>329.82</td>
<td>204.65</td>
<td>115.99</td>
<td>57.55</td>
<td>23.02</td>
<td>6.11</td>
</tr>
<tr>
<td>70</td>
<td>577.65</td>
<td>382.73</td>
<td>237.48</td>
<td>134.60</td>
<td>66.78</td>
<td>26.71</td>
<td>7.09</td>
</tr>
<tr>
<td>75</td>
<td>663.44</td>
<td>439.58</td>
<td>272.76</td>
<td>154.59</td>
<td>76.70</td>
<td>30.68</td>
<td>8.15</td>
</tr>
<tr>
<td>80</td>
<td>755.17</td>
<td>500.35</td>
<td>310.47</td>
<td>175.97</td>
<td>87.30</td>
<td>34.92</td>
<td>9.28</td>
</tr>
<tr>
<td>85</td>
<td>852.84</td>
<td>565.07</td>
<td>350.63</td>
<td>198.73</td>
<td>98.60</td>
<td>39.44</td>
<td>10.48</td>
</tr>
<tr>
<td>90</td>
<td>956.45</td>
<td>633.72</td>
<td>393.22</td>
<td>222.88</td>
<td>110.58</td>
<td>44.23</td>
<td>11.75</td>
</tr>
<tr>
<td>95</td>
<td>1065.99</td>
<td>706.30</td>
<td>438.26</td>
<td>248.40</td>
<td>123.24</td>
<td>49.30</td>
<td>13.09</td>
</tr>
<tr>
<td>100</td>
<td>1181.48</td>
<td>782.82</td>
<td>485.74</td>
<td>275.32</td>
<td>136.59</td>
<td>54.64</td>
<td>14.51</td>
</tr>
<tr>
<td>105</td>
<td>1302.90</td>
<td>863.27</td>
<td>535.66</td>
<td>303.61</td>
<td>150.63</td>
<td>60.26</td>
<td>16.01</td>
</tr>
<tr>
<td>110</td>
<td>1430.26</td>
<td>947.65</td>
<td>588.03</td>
<td>333.29</td>
<td>165.36</td>
<td>66.15</td>
<td>17.57</td>
</tr>
<tr>
<td>120</td>
<td>1702.75</td>
<td>1128.23</td>
<td>700.08</td>
<td>396.80</td>
<td>196.87</td>
<td>78.75</td>
<td>20.92</td>
</tr>
<tr>
<td>130</td>
<td>1999.07</td>
<td>1324.54</td>
<td>821.89</td>
<td>465.85</td>
<td>231.13</td>
<td>92.46</td>
<td>24.56</td>
</tr>
</tbody>
</table>
7.10.2 Maximum Link Utilization (Upper Bound)

Consider a network of \( n \) nodes in a network. Let \( D \) be the diameter of the graph \( G \). Let \( N \) be the number of clusters in a graph \( G \) with diameter \( d_i \) and \( d_j \). Let \( d_i \) be the maximum diameter of one of the clusters \( C_i \) and \( d_j \) be the next maximum diameter of one of the clusters \( C_j \). The link utilization factor of the network is calculated as follows by considering the following examples.

Let there be a network of 1000 nodes in a graph \( G \) with diameter \( D \) of a network. The number of clusters in a network is 10 of various numbers of elements \( n_1, n_2, n_3, \ldots, n_{10} \). Calculate the average maximum link utilization with respect to non clustered, clustered and LMUAC Algorithm with respect to link utilization of the network, for the diameter \( d_1, d_2, d_3, \ldots, d_{10} \) etc.

Maximum Link Utilization for non clustered Network (Flooding) be \( ML_{nc} \).

Maximum Link Utilization for Clustered Network (Flooding within Clusters) be \( ML_C \).

Maximum Link Utilization for Clustered Network with LMUAC Algorithm be \( ML_{LMUAC} \).

\[
ML_{nc} = \frac{n(n-1)}{2} \tag{7.12}
\]

\[
ML_C = \left[ \left( \frac{n_i(n_i-1)}{2} \right) \ast \left( \frac{n_j(n_j-1)}{2} \right) \right]_{i\neq j} + 1 \tag{7.13}
\]

\[
ML_{LMUAC} = \left[ (d_i - 1) + (d_j - 1) + 1 \right]_{i\neq j} \tag{7.14}
\]

\[
ML_{LMUAC} = [2d_i - 2]_{i=j} \tag{7.15}
\]

\[
ML_{LMUAC(i=j)} < ML_{LMUAC(i\neq j)} < ML_c < ML_{nc} \tag{7.16}
\]

The following Table 7.6 shows the calculation with respect to maximum link utilization factor with the assumed values. The following Figures 7.13, 7.14, 7.15, 7.16 shows the maximum link utilization in non clustered network, clustered network, with 3 level hierarchical and with LMUAC Algorithm.
Routing Algorithm for a Clustered Network

Figure 7.13: Maximum Link Utilization in Non Clustered network

Figure 7.14: Maximum Link Utilization in Clustered network
Figure 7.15: Maximum Link Utilization in Non Clustered network, Clustered network and 3 level Hierarchical

Figure 7.16: Maximum Link Utilization for clustered LMUAC Algorithm
Table 7.6: Max Link utilization factor (upper bound) for 1000 nodes and Diameter is $D$

<table>
<thead>
<tr>
<th>$n$</th>
<th>$ML_{nc}$</th>
<th>$n_i = \begin{array}{c}\frac{n}{5} \ \frac{n}{3} \end{array}$</th>
<th>$n_j = \begin{array}{c}\frac{n}{5} \ \frac{n}{3} \end{array}$</th>
<th>$ML_c$ ((i \neq j))</th>
<th>$ML_{LMUAC}$ ((i \neq j))</th>
<th>$ML_{LMUAC_i}$</th>
<th>$ML_{LMUAC_j}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>1225</td>
<td>10</td>
<td>17</td>
<td>5876</td>
<td>26</td>
<td>18</td>
<td>32</td>
</tr>
<tr>
<td>75</td>
<td>2775</td>
<td>15</td>
<td>25</td>
<td>31501</td>
<td>39</td>
<td>28</td>
<td>48</td>
</tr>
<tr>
<td>100</td>
<td>4950</td>
<td>20</td>
<td>33</td>
<td>102390</td>
<td>52</td>
<td>38</td>
<td>67</td>
</tr>
<tr>
<td>125</td>
<td>7750</td>
<td>25</td>
<td>42</td>
<td>254168</td>
<td>66</td>
<td>48</td>
<td>82</td>
</tr>
<tr>
<td>150</td>
<td>11175</td>
<td>30</td>
<td>50</td>
<td>532876</td>
<td>79</td>
<td>58</td>
<td>98</td>
</tr>
<tr>
<td>175</td>
<td>15225</td>
<td>35</td>
<td>58</td>
<td>994974</td>
<td>92</td>
<td>68</td>
<td>115</td>
</tr>
<tr>
<td>200</td>
<td>19900</td>
<td>40</td>
<td>67</td>
<td>1707335</td>
<td>106</td>
<td>78</td>
<td>132</td>
</tr>
<tr>
<td>225</td>
<td>25200</td>
<td>45</td>
<td>75</td>
<td>2747251</td>
<td>119</td>
<td>88</td>
<td>148</td>
</tr>
<tr>
<td>250</td>
<td>31125</td>
<td>50</td>
<td>83</td>
<td>4202432</td>
<td>132</td>
<td>98</td>
<td>165</td>
</tr>
<tr>
<td>275</td>
<td>37675</td>
<td>55</td>
<td>92</td>
<td>6171001</td>
<td>146</td>
<td>108</td>
<td>182</td>
</tr>
<tr>
<td>300</td>
<td>44850</td>
<td>60</td>
<td>100</td>
<td>8761501</td>
<td>159</td>
<td>118</td>
<td>199</td>
</tr>
<tr>
<td>325</td>
<td>52650</td>
<td>65</td>
<td>108</td>
<td>12092890</td>
<td>172</td>
<td>128</td>
<td>215</td>
</tr>
<tr>
<td>350</td>
<td>61075</td>
<td>70</td>
<td>117</td>
<td>16294543</td>
<td>186</td>
<td>138</td>
<td>232</td>
</tr>
<tr>
<td>375</td>
<td>70125</td>
<td>75</td>
<td>125</td>
<td>21506251</td>
<td>199</td>
<td>148</td>
<td>248</td>
</tr>
<tr>
<td>400</td>
<td>79800</td>
<td>80</td>
<td>133</td>
<td>27878224</td>
<td>212</td>
<td>158</td>
<td>265</td>
</tr>
<tr>
<td>450</td>
<td>101025</td>
<td>90</td>
<td>150</td>
<td>44755876</td>
<td>239</td>
<td>178</td>
<td>298</td>
</tr>
<tr>
<td>500</td>
<td>124750</td>
<td>100</td>
<td>167</td>
<td>68337501</td>
<td>266</td>
<td>198</td>
<td>332</td>
</tr>
<tr>
<td>550</td>
<td>150975</td>
<td>110</td>
<td>183</td>
<td>100199765</td>
<td>292</td>
<td>218</td>
<td>365</td>
</tr>
<tr>
<td>600</td>
<td>179700</td>
<td>120</td>
<td>200</td>
<td>142086001</td>
<td>319</td>
<td>238</td>
<td>398</td>
</tr>
<tr>
<td>650</td>
<td>210925</td>
<td>130</td>
<td>217</td>
<td>195906210</td>
<td>346</td>
<td>258</td>
<td>432</td>
</tr>
<tr>
<td>700</td>
<td>244650</td>
<td>140</td>
<td>233</td>
<td>263737057</td>
<td>372</td>
<td>278</td>
<td>465</td>
</tr>
<tr>
<td>800</td>
<td>319600</td>
<td>160</td>
<td>267</td>
<td>450570668</td>
<td>426</td>
<td>318</td>
<td>532</td>
</tr>
<tr>
<td>900</td>
<td>404550</td>
<td>180</td>
<td>300</td>
<td>722533501</td>
<td>479</td>
<td>358</td>
<td>598</td>
</tr>
<tr>
<td>1000</td>
<td>499500</td>
<td>200</td>
<td>333</td>
<td>1102238890</td>
<td>532</td>
<td>398</td>
<td>665</td>
</tr>
</tbody>
</table>
7.10.3 Memory space requirement (Upper Bound)

Consider a network of 3 level Hierarchy. Let \( a_i \) be the area in a graph \( G \). Let there be \( n_i \) nodes. Let there be \( m \) such super areas and let there be \( k \) such super super areas with \( a - i > m > k \). The total memory space required for each and every node of non hierarchical, hierarchical and LMUAC Clustered network.

Let there be a network of 1000 nodes in a graph \( G \) with diameter \( D \) of a network. The number of clusters in a network is 10 they are \( a_1, a_2, a_3, \ldots, a_{10} \) of various diameters \( d_1, d_2, d_3, \ldots, d_{10} \) and the number of elements in each such area \( n_1, n_2, n_3, \ldots, n_{10} \). The maximum memory space required for the entire network with non hierarchical, hierarchical and LMUAC Clustered Network is calculated. Let there be 3 level hierarchy in splits up as 50 nodes in \( a_i \), \( m = 20 \) and \( k = 3 \).

Memory space requirement for non Hierarchical be \( MS_{nc} \).
Memory space requirement for Hierarchical Network with three level Hierarchy be \( MS_{H3L} \).
Memory space requirement for Clustered Network by LMUAC Algorithm be \( MS_{LMUAC} \).

\[
MS_{nc} = n(n - 1) \tag{7.17}
\]
\[
MS_{H3L} = \sum_{i=1}^{N} ((n_i - 1) + m + k)n_i \tag{7.18}
\]
\[
MS_{LMUAC} = \sum_{i=1}^{N} (2(k_i - 1)) + N \tag{7.19}
\]

Let \( n \) be the total number of vertices in \( G \). Let there be \( N \) Clusters = \( \{C_1, C_2, C_3, \ldots, C_N\} \). The number of elements in \( C_i = k_i \). In the worst case an element in \( C_i \) may be adjacent to \( (K_i - 1) \) nodes, so maximum number of memory space at each vertex in \( C_i = 2(k_i - 1) + 1 \). In the real scenario the maximum degree of a vertex is \( < (k_i - 1) \). Let \( \Delta_i \) be the maximum degree of a node in cluster \( C_i \). Then the maximum memory space required for a \( \Delta_i \) is...
Routing Algorithm for a Clustered Network

\[ MS_{LMUAC(\Delta_i)} \]

\[ MS_{LMUAC(\Delta_i)} < MS_{LMUAC} < MS_{HSL} < MS_{nC} \]  \hspace{1cm} (7.20)

The below following Figures 7.17, 7.18, 7.19 demonstrate the memory space requirement and the following Table 7.7 is considered.

**Figure 7.17:** Memory space in non Clustered network
Figure 7.18: Memory space requirement for clustered LMUAC Algorithm

Figure 7.19: Memory space requirement for non clustered, clustered and Hierarchical
Table 7.7: Memory space requirement (upper bound) for 1000 nodes and Diameter is $D$

| $n$  | $MSc$ | $n_i$ | $MSL_{H3L}$ | $n_i/2$ | $n_i/3$ | $n_i/4$ | $n_i/10$ | $MS(LMUAC) \frac{n_i}{2}$ | $MS(LMUAC) \frac{n_i}{3}$ | $MS(LMUAC) \frac{n_i}{4}$ | $MS(LMUAC) \frac{n_i}{10}$ |
|------|-------|-------|--------------|--------|--------|--------|----------|----------------|----------------|----------------|----------------|----------------|
| 50   | 2450  | 5     | 1350         | 3      | 2      | 1      | 1        | 200           | 117            | 75             | 0              |
| 75   | 5550  | 8     | 2213         | 4      | 3      | 2      | 1        | 488           | 300            | 206            | 38             |
| 100  | 9900  | 10    | 3200         | 5      | 3      | 3      | 1        | 900           | 566            | 400            | 100            |
| 125  | 15000 | 13    | 4313         | 6      | 4      | 3      | 1        | 1438          | 916            | 656            | 188            |
| 150  | 22350 | 15    | 5550         | 8      | 5      | 4      | 2        | 2100          | 1350           | 975            | 300            |
| 175  | 30450 | 18    | 6913         | 9      | 6      | 4      | 2        | 2888          | 1867           | 1357           | 438            |
| 200  | 39800 | 20    | 8400         | 10     | 7      | 5      | 2        | 3800          | 2467           | 1800           | 600            |
| 225  | 50400 | 23    | 10013        | 11     | 8      | 6      | 2        | 4838          | 3150           | 2307           | 788            |
| 250  | 62250 | 25    | 11750        | 13     | 8      | 6      | 3        | 6000          | 3917           | 2875           | 1000           |
| 275  | 75350 | 28    | 13613        | 14     | 9      | 7      | 3        | 7288          | 4767           | 3507           | 1238           |
| 300  | 89700 | 30    | 15600        | 15     | 10     | 8      | 3        | 8700          | 5700           | 4200           | 1500           |
| 325  | 105300| 33    | 17713        | 16     | 11     | 8      | 3        | 10238         | 6717           | 4957           | 1788           |
| 350  | 122150| 35    | 19950        | 18     | 12     | 9      | 4        | 11900         | 7817           | 5775           | 2100           |
| 400  | 159600| 40    | 24800        | 20     | 13     | 10     | 4        | 15600         | 10267          | 7600           | 2800           |
| 500  | 249500| 50    | 36000        | 25     | 17     | 13     | 5        | 24500         | 16167          | 12000          | 4500           |
| 600  | 359400| 60    | 49200        | 30     | 20     | 15     | 6        | 35400         | 23400          | 17400          | 6600           |
| 700  | 489500| 70    | 64000        | 35     | 23     | 18     | 7        | 48300         | 31967          | 23800          | 9100           |
| 800  | 639200| 80    | 81600        | 40     | 27     | 20     | 8        | 63200         | 41867          | 31200          | 12000          |
| 900  | 809100| 90    | 100800       | 45     | 30     | 23     | 9        | 80100         | 53100          | 39600          | 15300          |
| 1000 | 999000| 100   | 122000       | 50     | 33     | 25     | 10       | 99000         | 65667          | 49000          | 19000          |
Routing Algorithm for a Clustered Network

7.11 Conclusion

Routing is the main objective, to design routing algorithm clustering is performed based on unique ID. The developed LMUAC routing scheme has three basis parts. They are Forwarding, Switching and Route discovery. An Algorithm is developed for forwarding, which takes care of forwarding the data packets for source to cluster head of that cluster. Similarly we developed an algorithm to switch the data packets from source cluster head to destination cluster heads, if source and destination are in different clusters. The final phase of the algorithm is route discovery based on the application. The parameters are set and route discovery is performed. The upper bound for the path length, Maximum link utilization factor and Memory space requirement is calculated and we show that all the above perform better in the clustered network. The Path length (upper Bound) is best suited if the selection is carried if \( d < D \), Then \( PL_{N,d} < PL_D \). Similarly the Maximum link utilization is faster than the non clustered Network. Still further it is faster compared to the basic selection of diameter of the individual cluster. It is proved that \( ML_{LMUAC(i=j)} < ML_{LMUAC(i\neq j)} < ML_c < ML_{nc} \). comparison of the non clustered network, Hierarchy 3 level clustered and the clustered network is shown. It is proved that \( MS_{LMUAC(\Delta,i)} < MS_{LMUAC} < MS_{H3L} < MS_{nc} \) is better if the value of \( i \) is smaller it requires only a small amount of routing Table.
Table 7.8: Comparative study of Metric Dimension (LMUA) and Cluster Dimension (LMUAC)

<table>
<thead>
<tr>
<th>Concept</th>
<th>Resolving set, Doubly Resolving set</th>
<th>Strongly Resolving Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basis elements (Two)</td>
<td>$v_1$ and $v_2$ Metric basis</td>
<td>$v_1$ and $v_2$ Cluster basis</td>
</tr>
<tr>
<td>Dimension</td>
<td>$\beta(G)$</td>
<td>$\beta_c(G)$</td>
</tr>
<tr>
<td>Definition</td>
<td>Minimum cardinality of a subset $S$</td>
<td>It is a super set of metric basis with an additional property</td>
</tr>
<tr>
<td>Maximum degree of a basis elements(Two)</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Distance Between $v_1$ and $v_2$(Two)</td>
<td>odd/even</td>
<td>odd</td>
</tr>
<tr>
<td>Path $P_n$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cycle $C_n$</td>
<td>2</td>
<td>n=even-(2), n=odd-(3)</td>
</tr>
<tr>
<td>$P_m \times C_n$</td>
<td>$n=$odd-2, $n=$even-3</td>
<td>$n=$odd-3, $n=$even-4</td>
</tr>
<tr>
<td>$K_n$</td>
<td>$n-1$</td>
<td>$n$</td>
</tr>
<tr>
<td>Full Degree vertex(Two)</td>
<td>Not necessarily metric basis</td>
<td>Cluster basis</td>
</tr>
<tr>
<td>$K_{1,n}$</td>
<td>$n-1$</td>
<td>$n$</td>
</tr>
<tr>
<td>Max degree of a vertex(Two)</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Max (Two) $</td>
<td>V</td>
<td>$</td>
</tr>
<tr>
<td>Complexity(Two)</td>
<td>$\left[\frac{n(n-1)}{2}d^4\right]$</td>
<td>$\left[\frac{n(n-1)}{2}d^4\right]$</td>
</tr>
<tr>
<td>Find Intersection(Two)</td>
<td>$</td>
<td>V_{i_k} \cap V_{j_l}</td>
</tr>
</tbody>
</table>