CHAPTER 5
FINE TUNING OF PHEROMONE CONCENTRATION FOR MODIFIED TERMITE ALGORITHM (MTA)

Data transfer on the move is highly essential and is a basic need in MANETs. Mobility of nodes in MANETs affects the performance of routing protocols. Using greedy mechanism, the network tends to choose a single path every time which leads to congestion. Designing an effective load balancing algorithm is difficult due to dynamic topology of MANET. To address this problem, Modified Termite Algorithm (MTA) has been developed based on ant’s food foraging behavior.

The MTA developed by adopting efficient pheromone evaporation technique will address load balancing problems and will enhance the performance of the network in terms of throughput, and reduces end-to-end delay and routing overheads.

5.1 Termite hill building process

Termite’s hill building process involves, collecting the pebbles spread over an area and place them in one place. The cooperation between termites will lead to constructing of a huge hill (as big as 7 Feet). This is achieved by termites moving over the locally observed pheromone trails (laid by other termites). If no trails are found, they move randomly in any direction.

As shown in Figure 5.1, each termite may carry only one pebble at a time and move by observing the trail laid by other termites. If a pebble is found on the way, it just puts down the pebble carried by it and both pebbles are infused with pheromone which then evaporates [23].

Termite’s inclination towards pheromone trail (positive feedback) leads to large piles of pebbles forming a hill. At the same time, evaporation (negative feedback) declines the movement of termite towards the direction in which hill building is not appropriate or avoids small piles. It is essential that many individuals work together,
otherwise before the pebbles are added to the pile, evaporation of trail leads to a random walk by termites without forming any significant piles [23].

![Flowchart of Termite Hill building process](image)

**Fig 5.1** Flowchart of Termite Hill building process

Analogous to the above example, each node in the Mobile wireless network is a termite hill. As packets travel from the source to the destination, they leave pheromone at source and at the same time follows the pheromone to its destination through the MANET. The pheromone is laid between nodes’ links which are communicating with each other. As in the termite example, packets are inclined towards strong pheromone trails. Laying source pheromone trail on the path increases the likelihood of packets travelling on the same path.

### 5.2 Modified Termite Algorithm (MTA)

The Termite algorithm is a routing protocol for MANET. The packets in the network follow the principles of SI and result in an emergent routing behavior. Packets are not forwarded to the same neighbor from whom it was received (to avoid cycles or loops). It is similar to Ant Colony Base Routing Algorithm (ARA) [22, 23, and 24] in terms of routing data but varies with respect to route discovery and failure handling. It is inspired by termite hill building process.
Termites are biased towards the “pheromone” laid by other termites towards the destination hill. This is very similar to the way packets in MANETs are biased towards the optimal path to reach the destination. Information about the network will be carried by data packets which help in making probabilistic routing decision. This decision helps to find a maximum utility path in an emergent manner.

The algorithm takes source, destination as inputs and produces output as the route from “source” to the “destination” through which the data is to be routed. The process described in the above context diagram (Figure 5.2) can be viewed as a segregation of several sub-processes as shown in Figure 5.3.

Fig 5.2 Context Diagram - MANETs using MTA
Fig 5.3 Phases of MANETs using MTA

- Route Discovery (1.1) – Takes source and destination node as inputs and searches for paths from source to destination.
- Route Maintenance (1.2) – This maintains the best path discovered during Route Discovery sub-process
- Route Failure (1.3) – This detects any link failures and finds new path which will be maintained by the Route Maintenance sub-process

5.2.1 Routing in MANETs with MTA

Each node in the network maintains a pheromone table which keeps track of percentage of pheromone in its neighbor link. The table consists of rows representing the neighbors and columns representing destinations (except for node itself). The table size is dynamic and varies based on the destinations requested recently. An entry of node ‘n’ in the table is referenced by $P_{i,d}^n$ where $i \in N^n$ (set of neighbor nodes) is the neighbor and ‘d’ denotes the destination where $d \in D^n$ (set of destination nodes). $P_{i,d}^n$ is the percentage of pheromone at node “n” for the destination “d” from the
neighbor “i”. (as shown in Table 4.1). When a packet is received from an unknown source, a new entry for that node is created on the routing table. If the nodes are just destinations (and not neighbors) then only the columns are created.

![Diagram of node neighborhood]

**Fig 5.4** ‘S’ node neighborhood

For example as shown in Figure 5.4, Source ‘S’ considers N1, N2 and N3 as neighboring nodes since they are within the transmission range [34]. Once the neighboring nodes are known, then the routing table is updated as shown in Table 5.1 (route discovery also helps in updating the routing table). When node ‘S’ wants to communicate with destination ‘D’ it checks for the highest trail for destination ‘D’ and then forwards the packet.

**Table 5.1** Example Routing Table at node ‘S’

<table>
<thead>
<tr>
<th></th>
<th>N1</th>
<th>N2</th>
<th>N3</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1</td>
<td>0.18</td>
<td>0</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>N2</td>
<td>0.27</td>
<td>0.1</td>
<td>0.1</td>
<td>0.6</td>
</tr>
<tr>
<td>N3</td>
<td>0.01</td>
<td>0.2</td>
<td>0.1</td>
<td>0.3</td>
</tr>
</tbody>
</table>
5.2.1.1 Pheromone update/decay

Pheromone is updated only by data packets as compared to other ACO based algorithm in which all RREQ (forward ants), RREPLY (backward ants) and data packets update pheromone concentration. When packets arrive at a node ‘n’ from source ‘s’ and a previous hop ‘p’, the new pheromone concentration entry $P_{n,s}^p$ for the source of the packet is increased by constant pheromone value ‘$\gamma$’ as shown in Eqn (5.1).

$$P_{p,s}^n \leftarrow P_{p,s}^n + \gamma$$ \hspace{1cm} (5.1)

Pheromone decay is an important factor for efficient packet delivery. If pheromone decay is not handled properly, it may lead to stale entries in the routing table, leading to packet drops or leading to improper load in the network due to data packets travelling on the optimal path and as a result increasing the pheromone concentration. This makes other packets to travel blindly a path not knowing load on the link. The pheromone decay rate needs to be moderated in such a way that it is neither too quick nor too slow. The nominal pheromone decay interval is one second which is called as decay period.

$$\forall i \in N^n, \forall d \in D^n, P_{i,d}^n \leftarrow P_{i,d}^n \cdot e^{-\tau}$$ \hspace{1cm} (5.2)

If pheromone for a particular node decays fully (as it has not received any packets for a long time), the corresponding node’s row is removed from the routing table. However, column entry cannot be deleted directly if it is also a neighbor node. Pheromone bounds like max, min and initial pheromone values (prevents extreme difference in pheromone value) in the routing table play a major role in stagnation problems.
5.3 Efficient stagnation avoidance technique

Out of the many decay functions analyzed, the exponential decay technique in ACO is more applicable for dynamic networks like MANETs. The limitation of this technique is that the pheromone evaporates (uniformly) very quickly after some time leaving no traces for the data transmission. Hence, fresh route discovery is established to know the destination address. This causes more control overhead and reduces bandwidth efficiency. To deal with this problem, controlled exponential evaporation is adopted which fine tunes the rate of pheromone evaporation based on the stability factor ‘Δ’ of the node (as explained in the previous chapter #4) [1,2].

Each node in the network maintains two tables as shown in Figure 5.5
- Routing table - maintains entries of its neighbors and destinations nodes.
- Local traffic statistic table – maintains “node stability factor” for each of node’s neighboring nodes. This table helps in finding more stable nodes.

<table>
<thead>
<tr>
<th>Neighbor Node</th>
<th>Destination Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>D1</td>
</tr>
<tr>
<td>N1</td>
<td>0.18</td>
</tr>
<tr>
<td>N2</td>
<td>0.27</td>
</tr>
<tr>
<td>N3</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>………</td>
</tr>
<tr>
<td>L</td>
<td>PL1</td>
</tr>
</tbody>
</table>

**Fig 5.5** Data structure maintained at ‘S’ node
During packet forwarding process, each node calculates the ‘hello’ sent and ‘hello’ received by its neighboring nodes and calculates the stability ratio shown in Eqn (5.3).

\[ \text{Ratio} = \left( \frac{\text{Hello Replied}}{\text{Hello Sent}} \right) \times 100 \quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots (5.3) \]

**Table 5.2** Node Stability Ratio of Neighbor Nodes at Node ‘S’

<table>
<thead>
<tr>
<th>Neighbors</th>
<th>N1</th>
<th>N2</th>
<th>N3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ratio=Hello sent / Hello replied (%)</strong></td>
<td>55</td>
<td>78</td>
<td>24</td>
</tr>
</tbody>
</table>

With the help of the ratio calculated, evaporation is controlled for each neighboring node by fine-tuning the decay speed (like decaying fast for unstable nodes and decaying a little slower for stable nodes). Pheromone decay rate is controlled by adding an extra parameter ‘\( \Delta \)’ to the exponential decay factor.

\[
\text{Where } \Delta = \begin{cases} 
0.003 & \text{If Ratio is > 75 %} \\
0.002 & \text{If Ratio is } \leq 75 \text{ % and } <55\% \\
0.001 & \text{If Ratio is } \leq 55 \text{ %}
\end{cases}
\]

\[
P_{i,d}^n \leftarrow P_{i,d}^n \cdot e^{(\tau + \Delta)} \quad \ldots \ldots \ldots \ldots (5.4)
\]

(‘\( \Delta \)’ is tunable based on the application and the node mobility)

**Table 5.3** Decay Factor for different Stability Ratio

<table>
<thead>
<tr>
<th>Decay Factor ( e^{(\tau + \Delta)} )</th>
<th>If Ratio ( \geq 75 % )</th>
<th>If Ratio (&lt; 75 % \text{ and } \geq 55% )</th>
<th>If Ratio (&lt; 55 % )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( e^{-\tau + 0.003} )</td>
<td>( e^{-\tau + 0.002} )</td>
<td>( e^{-\tau + 0.001} )</td>
<td></td>
</tr>
</tbody>
</table>

A source ready to forward packet to the destination, takes packet forwarding equation’s help to find out the next hop neighbor. This formula (as shown in Eqn (5.5)) updates pheromone value for the destination ‘d’ on each outgoing link ‘i’. Random selection of next neighbor is performed with the help of packet forwarding equation. Extra care must be taken by not forwarding packet to the previous hop ‘p’.
\[ P_{i,d}^n = \frac{\left( P_{i,d}^n + K \right)^F}{\sum_{j \in S^{n-1}} \left( P_{j,d}^n + K \right)^F} \]  \hspace{1cm} (5.5)

Where \( F \) (pheromone sensitivity) is a global parameter which gives information of the state of paths to the destination at the time of route discovery phase and \( K \) (pheromone threshold) is a “significant” amount of pheromone. [23].

This technique helps MANETs to overcome stagnation problem (Through the usage of “pheromone sensitivity”). It also reduces the control overhead through efficient pheromone decay techniques leading to efficient packet delivery ratio.

**5.4 Results and discussion:**

Primary metrics considered for the Modified Termite Algorithm (MTA) are throughput, end-to-end delay and routing over head against packet sizes. A series of tests are conducted to check the feasibility of MTA as an effective mechanism for MANETs based on the community based mobility model. The behavior of success rate with the number of termites (nodes) has been studied. It is observed that only after 30 nodes, the algorithm shows a sharp increase in the success rate (considering 5 m/s mobility speed and pause time varying between 1-5 secs).

**Throughput variation with packet size**

Throughput variation with increase of packet sizes (between 1-6 Kb) has been implemented and studied. Figure 5.6 shows the throughput variation for different packet sizes for MTA as well as for standard Termite algorithm for 30 nodes.
The MTA based MANET shows continuous increase in throughput with the packet size, where as the throughput for Termite algorithm remains low. This suggests that the MTA algorithm implemented on a MANET with 30 nodes shows significant improvement in throughput ~ 26 % with increase of packet size as compared to Termite algorithm. Figure 5.7 shows the throughput variation for different packet sizes for MTA as well as for standard Termite algorithm for 50 nodes.

Fig 5.6 Throughput Vs Packet size Analysis for 30 nodes

Fig 5.7 Throughput Vs Packet size Analysis for 50 nodes
The MTA based MANET shows continuous increase in throughput with increase in packet size, whereas throughput for the Termite algorithm. This suggests that the MTA algorithm implemented on a MANET with 50 nodes indicates significant improvement in throughput ~4% with increase of packet size as compared to Termite algorithm.

It may be noted that the MTA algorithm exhibits throughput improvement of ~26% for lower number of (30) nodes as compared to higher number of (50) nodes. This may be attributed to the dynamic topology due to the mobility of nodes i.e. as the number of nodes increase the performance will decrease.

**End-end delay variation with packet size**

The MTA implemented on MANET has been used to determine the end–to-end (E2E) delay with increase of packet sizes (in the range of 1-6 Kb).

Figure 5.8 shows the end-to-end delay variations for different packet sizes for MTA as well as for standard Termite algorithm for 30 nodes.

![E2E Vs Packet Size](image)

**Fig 5.8 End-to-End Delay (ms) Vs Packet size Analysis for 30 nodes**

The Termite Algorithm shows continuous increase in end-to-end time delay with the packet size, whereas the end-to-end time delay for MTA shows no significant change.
The result suggests that the MTA shows significant reduction in the End to End time delay by ~21% compared to Termite algorithms.

Figure 5.9 shows the end-to-end delay variations for different packet sizes for MTA as well as for standard Termite algorithm for 50 nodes.

![E2E Vs Packet Size](image)

The Termite algorithm shows continuous increase in end-to-end time delay with packet size, whereas the end-to-end time delay for MTA do not show any significant variation in the end-to-end time delay with packet sizes. The results suggest that the MTA algorithm implemented on a MANET with 50 nodes indicate significant reduction in the end-to-end time delay ~22% compared to Termite algorithms. It may also be noted that the MTA algorithm exhibits significant reduction in the end-to-end time delay for lower number of (30) nodes as compared to higher number of (50) nodes.

**Routing Overhead variation with packet size**

The MTA implemented on MANET has also been used to determine the routing overhead changes with increase of packet sizes (in the range of 1-6 Kb). Routing
Overhead (ROH) is the ratio of number of control packets required for delivery of data.

Figure 5.10 shows the routing overhead variations for different packet sizes for MTA as well as for standard Termite algorithm for 30 nodes.

The Termite algorithm shows continuous increase in routing over heads with packet size, whereas routing over heads for MTA do not show any significant variation in the routing overhead with packet size. The result suggests that the MTA algorithm implemented on a MANET with 30 nodes indicates significant decrease in the routing over heads ~13% compared to Termite algorithms.

Figure 5.11 shows the routing overhead variations for different packet sizes for MTA as well as for standard Termite algorithm for 50 nodes.
The Termite algorithm shows continuous increase in routing over heads with the packet size; whereas the routing over heads for MTA suggests that the MTA algorithm implemented on a MANET with 50 nodes indicates very small decrease of the routing over heads ~36% compared to Termite algorithms.

It may also be noted that the MTA algorithm does not exhibit significant variation ~13% in the Routing over heads for 30 nodes as compared to 50 nodes. This may be attributed to the dynamic topology due to the mobility of nodes i.e as the number of nodes increases the performance will decrease.

MTA algorithms with efficient pheromone evaporation techniques applied to MANETs, indicated improvement in the performance of the network in terms of throughput, end-to-end delay and routing overheads.