Data Analytics and Artificial Intelligence
Vol: 1(3), 2021
REST Publisher
ISBN: 978-81-948459-4-2
Website: http://restpublisher.com/book-series/daai/

Exploring the Termite Algorithm: A Novel Adaptive Routing Technique for Efficient Communication in Mobile Ad Hoc Networks

Anto Ramya. S. I
St. Joseph's College of Arts and Science for Women, Hosur, Tamil Nadu, India.
Corresponding Author Email:

Abstract

A self-configuring network of mobile devices connected by wireless links is known as a mobile ad hoc network (MANET), sometimes known as a mobile mesh network. Ad Hoc Networks are not dependent on any established infrastructure, in contrast to Traditional Mobile Wireless Networks. Due to their complex nature and constant communication connectivity, studies on a variety of issues pertaining to mobile ad hoc networks are growing in popularity. In this study, a method for determining the path between nodes in a MANET is discussed. A new packet routing technique for communication networks is called Termite. Adaptive, distributed, and mobile agent-based algorithm Termite was motivated by the ant colony concept recently developed. This technique creates paths between pairs of nodes using a number of mobile agents sometimes known as artificial termites. The agents explore the network concurrently and exchange information to update the routing tables. Packet delays and throughput are two metrics that are used to evaluate its performance. When compared to previous algorithms, the algorithm's results demonstrate higher throughput. Thus, the termite algorithm is a promising choice for data routing in business networks.

Keywords: Mobile agents, Pheromone, Routing algorithm, Swarm intelligence and Termite.

1. Introduction

A self-configuring network of mobile hosts connected by wireless links is known as a mobile ad hoc network, and the network's topology is formed by their unification [1]. Convenience, mobility, productivity, deployment, and expandability are the benefits of ad hoc networks. The network topology fluctuates randomly as the nodes in the network change. Therefore, creating a spawn path between two nodes is challenging. This algorithm is distributed, adaptive, and swarm intelligence-inspired. Swarm intelligence (SI) encompasses the class of optimising algorithms known as ant algorithms [2][3]. Ants, which are network exploration agents, communicate to perform routing in the ant algorithm [4][5]. This approach builds a path between pairs of nodes by having a number of mobile agents update routing tables and exchange information.

2. Review of the Literature

Destination-sequenced distance vector routing, wireless routing protocol, ad hoc on-demand distance vector routing, and dynamic source routing protocol are a few of the routing algorithms used in ad hoc networks [6][7]. When compared to previous algorithms, the algorithm proposed in this research differs in the following ways:

- Because the aforementioned algorithms must send their routing tables to other nodes throughout the network, they all incur a significant overhead. They either move them using an event-based approach or a time-based one. Since the network does not require the transmission of routing tables, the termite does not present this issue.
- Because some of the algorithms in use today do not support numerous paths, load balancing is not possible in the event that the ideal path is extremely congested.
- In the event that the ideal path becomes clogged, the termite algorithm facilitates load balancing in addition to generating multiple paths. Special packets are also needed for the above methods in order to maintain the route. This is not the case with the termite algorithm, which is more scalable and stable than

the existing algorithms because of its innate nature, which is similar to that of biological insects. Instead, the termite algorithm relies on the data packets themselves to maintain the route.

Termite Algorithm:

A straightforward illustration of termites' behaviour in creating hills offers a compelling parallel to the workings of the termite routing algorithm. Imagine a level area sprinkled with pebbles and covered in termites. The termites want to use the pebbles to construct a hill. The following regulations govern a termite:

- It travels randomly but is biased in the direction of the locally detected pheromone gradient. A termite travels consistently and randomly in any direction in the absence of a pheromone.
- Only one stone may be carried at a time by each termite.
- A termite will pick up a stone if it comes across one while it is not carrying one.
- When a termite carrying a pebble comes across another, it will set the first pebble down. Pheromone will seep into the pebble in a specific quantity.

Every node in the network is a termite hill, similar to the one shown above. Data packets are carried by termites with pebbles, and control packets are carried by termites without pebbles. On the communication channels connecting nodes, pheromone is applied. Strong pheromone gradients are the preference of packets. The possibility that packets will take the opposite way to reach the source is increased when the source pheromone is laid along the same trial. This is a compliment. Pheromone decay is added as negative feedback to stop outdated routing solutions from being stored in the collective network memory [8].

a) Pheromone Table:

Every node has a table that records the quantity of pheromone present on every neighbouring link. The table can be shown as a matrix with the destination nodes listed across the top and the neighbour nodes listed down the side. Columns represent destinations, and rows represent neighbours. Pn, d, where n is the neighbour index and d is the destination index, references an entry in the pheromone table. Put differently, Pn,d represents the quantity of pheromone originating from node d connected to neighbour n.

b) Pheromone Update:

The pheromone for the packet's source is increased by a constant, γ, when the acknowledgement packet reaches a node. γ has a nominal value of one. Processing of packets is limited to those addressed to a node. If a node is the packet's intended recipient on the next hop, it is referred to as being addressed.

$$Prs=Prs+1 \qquad \text{----------(1)}$$

The pheromone update process when a packet from source s is delivered from previous hop r is described by equation 1.

c) Pheromone Decay:

Every value in the pheromone table is periodically subtracted by 20% of the initial pheromone to allow for pheromone deterioration. A low decay rate will cause the pheromone to break down gradually, whereas a high decay rate will swiftly deplete the amount that remains. The term "decay period" refers to the nominal one-second pheromone decay interval. The pheromone degradation is described by the following equation:

$$P_{n,d}=P_{n,d} - P_{n,d} *2 \qquad \text{----------(2)}$$

d) Routing:

An incoming packet with destination d is randomly routed upon arrival at node b, depending on the quantity of d's pheromone present on b's neighbour links. Never forward a packet to the same neighbour from whom it was received. The packet is dropped if node b only has one neighbour, which is the node from whom it was recently received.

e) Route Discovery:

A node sends route request (RREQ) packets when it is trying to figure out how to go to an unknown location. Up until it reaches the target node, a route request (RREQ) packet is sent over the network in a selected manner. A route request (RREQ) is dropped if it is unable to be forwarded. The greatest number of tries is used to complete the route request. The destination is considered to be unreachable if the RREQ packet is unable to deliver it. A route reply (RREP) packet is created and sent back to the requestor once the destination node receives the route request packet. The RREP message is constructed so that the requestor appears to be the packet's destination and the requested destination to be its source. The number of hops in the trail from the source to the destination determines how the route's reply packet updates the pheromone table. This kind of update prioritises load balancing and handles numerous pathways. The route request packet's trail content is used to route the reply packet. The requested node will be automatically found by intermediate nodes on the return path.

f) Route Maintenance:

Data packets are responsible for maintaining routes. The path will be lost due to the decaying process if the data packets are not sent through the found path. Given the high decay rate, this algorithm uses a reactive technique to find the path. This module is used to maintain the path, update the routing table, and extract information from packets. At the destination, the data is extracted and provided to the node. The optimal path for sending the data packet is determined by the intermediary node. The node with the largest pheromone for the destination entry in

the pheromone table determines the optimum path, and the decay route module is used to periodically decay the routing table in order to eliminate the bad solution from the router's memory.

g) Route Failure Handling:

The periodic transmission of "hello" packets will indicate if the link fails. The neighbour item for that node is removed from the pheromone table if the hello reply is not received in the allotted two seconds. Therefore, a fresher path is generated throughout the subsequent route request process. The destination field is eliminated from the pheromone table, indicating that the node has relocated, if every entry in the destination column reaches the minimum pheromone. As a result, each node will keep up the path for nodes that are actively engaged in routing.

3. Design

The network under consideration in this work has n nodes. Data packets must be sent from source node S to destination node D, where S ρ n and D ϵ n, respectively. Wireless or cable links can be used to connect the nodes. The number of hops is the optimisation metric, meaning that the distance with the fewest hops between S and D is the ideal one. Sending the hello packets establishes the network topology. In the event of a wireless node, these greeting packets are broadcast. The hello packets are received by the node located in the transmission region, which responds with a hello reply packet. The nodes that respond with a hello reply are initialised to value 1 in the pheromone table and added to the list of surrounding nodes. The hello packet's round-trip time is used to calculate the nearby link's cost. A record is made in the destination field when the packet arrives from the unidentified source. The routing table is implemented as a two-dimensional array of linked lists due to its dynamic nature. Header nodes, root nodes, and pheromone nodes are the three different kinds of nodes.

a) Header Node: Field name, next pheromone, and next header make up this string. The node's IP address is stored in name. The pheromone values of the node that the header node points to are then stored in Pheromone. In order to store the details of another node, the next header serves as a pointer to the next header node.

b) Root Node: Name: same as root node; contains time, row, and column. Time: The routing table's latest accessed time is stored in this time_t type variable. This field is used to ensure that the routing table ages properly. These are of the type struct header_node, Row and Column. Row is a pointer to the destination nodes, while column is a pointer to the neighbouring nodes.

c) Pheromone Node: The pheromone value is kept in this structure. Pheromone, row, and bottom is all present. Pheromone: The content of the pheromone value is stored in this field. To navigate the pheromone table, utilise the row and bottom pointers, which point to the pheromone node.

d) Routing Table Effective Decay: Because the routing table in the current application is created from an analogue of a termite pheromone trail that would eventually dissipate in the environment, it must periodically degrade. Periodic decay is replaced in the application by effective decay. In other words, effective deterioration occurs only when it is accessed. Due to this, the application no longer needs a second thread, and the synchronisation of this table amongst these threads has been interrupted. Timestamps are used to implement effective decay. The amount of time it needs to decay is determined by the difference between two access times.

e) Packets: Data is sent to the other nodes using packets. By raising the link's pheromone concentration, these packets are also utilised to maintain the route. Routine maintenance does not require any special packets.

- Data packets contain type, source address, destination address, previous hop, next hop, data length, data, and trail. These are the many packet types that are utilised.
- Type, source address, destination address, previous hop, future hop, and trail are all contained in the route request/reply/dataack.
- Type, source address, destination address, and timestamp are all included in the Hello/Hello reply.
    - Type: It is used for defining the type of the packet.
    - Type=0 hello packet type=1 hello reply packet
    - Type=2 route request packet (RREQ) type=3 route reply packet (RREPLY)
    - Type=4 Data Packet type=5 Data acknowledgement packet
    - sourceAddr: This variable is used to hold the source node's IP address, whereas destAddr is used to store the destination node's address.
    - prevHop: The source is stored in this field, while the destination is stored in nextHop.
    - msgId is used to remove duplicate packets. DataLen is used to hold the length of the data packets, data field stores the data content.
    - Trail: The path that the packet has travelled is stored in this character array-type field.
    - Timestamp: The hello packets' round trip time is determined by this field.

4. Modules

Three modules make up the termite algorithm for mobile ad hoc networks: route failure handling, route maintenance, and route discovery.

a) Route Discovery Module: The process of creating a route between a source and a destination is known as route discovery. This module operates as follows:

- Create an RREQ packet at the source node and send it across the neighbours in increasing cost value sequence.
- Any node that gets an RREQ does the following actions:
  if current_addr=dest_addr
  {
  Create the RREPLY packet, copy the RREQ packet's trail information, transmit the RREPLY packet to the node prevHop, which received the RREQ packet, and update the pheromone table based on the number of hops.
  }
  Else
  {
  Forward the RREQ packet to the neighbouring nodes by adding the current node address to the trail field.
  }
- At any node, when it receives RREPLY it does the following:
  If current_addr=dest_addr
  {
  After retrieving the packet from the data queue and inserting it into the regular queue, update the pheromone table based on the hop count.
  }
  Else
  {
  After updating the pheromone table based on the hop count, send the packet to the neighbour while adhering to the trail content.
  }

b) Route maintenance module: The path created during the discovery phase must be maintained by the route maintenance module.

• Upon receiving a data packet, each node performs the following actions.
If current_addr=dest_addr
{
Remove the data content, extract the data, set type=ack in the data packet, and send the acknowledgement packet to prev_id.
}
Else
{
Utilising the neighbour table, get the pheromone values for each link. Then, calculate the probability of each node in the neighbour table and route packets to the link with the highest probability.
}

- Whenever the table is accessed, it degrades. The destination entry should be removed from the routing database if the pheromone value for any destination is 0.

c) Route failure handling module: In the event that the current route malfunctions, this module is in charge of creating backup routes. At any node, the destination field is removed from the routing table and a new route request is made if an acknowledgement packet is not received within the 10-second delay period. The data packet maintains this new route request, which gets the updated path.

5. Conclusion

Termite routing algorithm, a novel adaptive algorithm technique for data networks based on mobile agents, was examined in this work. Because nodes in mobile ad hoc networks move around a lot, the topology of the network may change often. The packets take alternate paths found by the route discovery phase in case the topology changes and the best path changes. Because the method offers multipath routing, load balancing is encouraged. The dynamic nature of the algorithm can manage a node crash. Additionally, a time-out mechanism can be used

to deal with termite loss. Thus, it may be concluded that this algorithm's commercial implementation might be possible. One may even argue that it is utilised in extensive networks, like internet, as a future option.
The following features could be added to the built application to make it even better:

- To choose the best route, it can make use of several QoS (Quality of Service) characteristics.
- The audio and video application packets can be routed via it.
- By fragmenting at the sender side and defragmenting at the recipient side, longer packets can be transferred.

First of all, I am glad to thank THE LORD ALMIGHTY for giving me the fortitude in finishing this paper. I express my sincere gratitude to my parents for their continuous guidance, support and encouragement in all my endeavors. I would like to thank my colleagues for their constant support they provided throughout my preparation.

References

1. Andrew S. Tannenbaum, “Computer Networks”, 4th Edition, Prentice- Hall of India
2. E. Bonabeau, M. Dorigo and G. Theraulaz, Swarm intelligence: from natural to artificial systems, Oxford University Press, 1999.
3. T. White, “Swarm intelligence and problem solving in telecommunications”, Canadian Artificial Intelligence Magazine, Spring 1997.
4. G. Di Caro and M. Dorigo, “Mobile agents for adaptive routing”, Proc. 31st Hawaii International Conference on System Sciences, IEEE Computer Society Press, Los Alamitos, CA, pp. 74-83, 1998.
5. Schoonderwoerd R, Holland O, Bruten J, Rothkrantz L. “Ant based load balancing in telecommunications networks, Adaptive behaviour Hewlett-Packard Laboratories, Bristol- England, pp 162 – 207, 1996.
6. Nader F Mir, “Computer and Communication Networks”, Pearson Education, 2007.
7. Liang S, Zincir Heywood A N, Heywood M I, “The effect of Routing under local information using a Social insect Metaphor”, IEEE International Congress of Evolutionary Computation, pp, 1438 – 1443, May 2002.
8. Martin Roth and Stephen Wicker, “Termite: Emergent Ad-Hoc Networking”, Wireless Intelligent Systems Laboratory School of Electrical and Computer Engineering Cornell University Ithaca, New York 14850 USA.

