Scope International Journal of Science, Humanities, Management and Technology. ISSN : 2455-068X
Vol.4 Issue 3 (2018) 37 - 43. Submitted 20/06/2018. Published 16/07/2018

UNICAST COMMUNICATION MANET'S USING GRAPH THEORY

Dr.R.Balakumar, M.Sc., M.Phil., Ph.D.,[1]
Assistant Professor, Department of Mathematics,
Prist University, Thanjavur

M.Manimehalai[2]
M.Phil Scholar, Department of Mathematics,
Prist University, Thanjavur

ABSTRACT

Mobile ad hoc networks (MANETs) are self-configuring networks of mobile nodes connected by wireless links. Each node within MANET operates as an end system and a router for all other nodes in the network. Due to the dynamic nature of MANETs, traditional fixed network routing protocols cannot be used. Based on that, new routing protocols have been introduced in MANETs. The purpose of this paper is to examine the current state-of-the-art of the existing unicast routing protocols for MANETs, and compare different approaches. For the purpose of this research, experiments are carried out in OPNET Modeler network simulator with the usage of reactive AODV and proactive OLSR unicast routing protocols. Data obtained in these experiments quantify and compare network performance, such as throughput, delay and network load. (PDF) Comparative analysis of unicast routing protocols in MANET networks. [accessed Jul 09 2018].

1. INTRODUCTION

A Mobile Ad hoc Network (MANET) is a dynamically changing infrastructure less and resource-constrained network of wireless nodes that may move arbitrarily, independent of each other. The transmission range of the wireless nodes is often limited, necessitating multi-hop routing to be a common phenomenon for communication between any two nodes in a MANET. Various routing protocols for unicast, multicast, multi-path and broadcast communication have been proposed for MANETs.

Most of the contemporary routing protocols proposed in the MANET literature adopt a Least Overhead Routing Approach (LORA) according to which a communication structure (route, tree or CDS) discovered through a global flooding procedure would be used as long as the communication structure exist, irrespective of the structure becoming sub-optimal since the time of its discovery in the MANET. We will also adopt a similar strategy and focus only on discovering a communication structure on a particular network graph taken as a snapshot during the functioning of the MANET. Such a graph snapshot would be hereafter referred to as a 'Static Graph' and a sequence of such static graphs over the duration of the MANET simulation session would be called a **'Mobile Graph'**.

Definition: 1.1

A graph G is an ordered triple $(V(G),E(G),\psi_G)$ consisting of

(i) a non-empty finite set V(G)

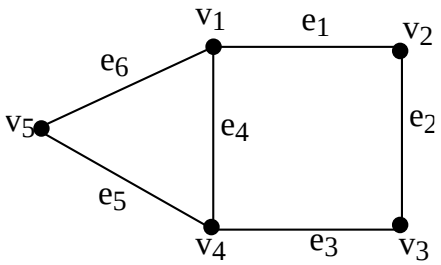
(ii) a finite set E(G) which is disjoint from V(G) and

(iii) an incidence function ψ_G that associates with each element of E(G) an unordered pair of elements of V(G).

The elements of V(G) are called vertices of G and the elements E(G) are called edges of G.

If e is an edge and $\psi(e)=(u,v)$ then we say that e is an edge joining u and v and the vertices u and v are called the ends of e.

Example :



$G = (V(G), E(G), \lambda\Psi_G)$

where

$V(G) = \{v_1,v_2,v_3,v_4,v_5\}$

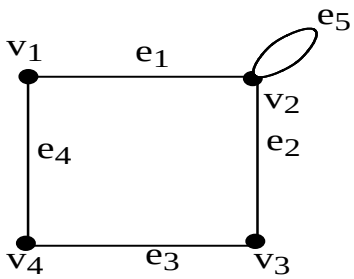
$E(G) = \{e_1, e_2, e_3, e_4, e_5, e_6\}$

$\Psi(e_1) = v_1v_2$, $\Psi(e_2) = v_2v_3$,…….

Definition: 1.2

An edge having the same vertex as both its end vertices is called a **self loop.**

Example :

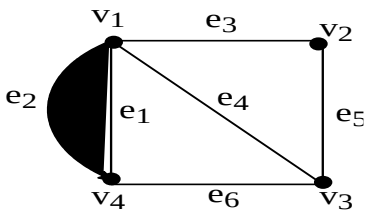


Here the edge e_5 having both its end vertices v_2

Definition: 1.3

A graph G with more than one edge associated with a given pair of vertices is called **parallel edges.**

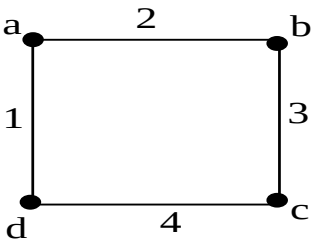
Example :



Here e_1 and e_2 are parallel edges

Definition: 1.4

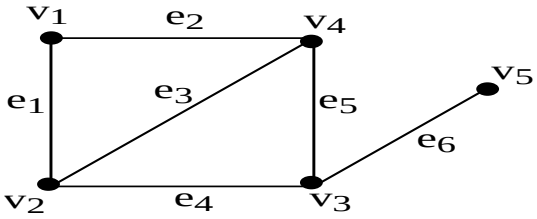
A graph that has neither self loop nor parallel edges is called a **simple graph. Example :**



Definition: 1.5

A vertex v_i is an end vertex of some edge e_j,v_i and e_j are said to be **incident with each other**.

Example :

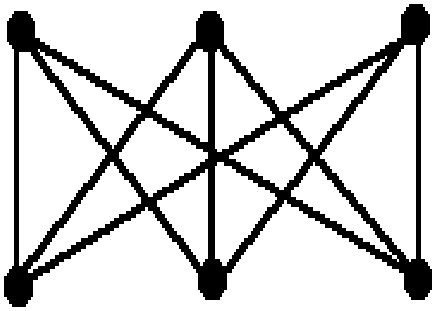


The edges e_4, e_5, e_6, are incident with v_3.

Bipartite graph

A graph is bipartite if its vertices can be partitioned into two disjoint subsets U and V such that each edge connects a vertex from U to one from V. A partite graph is a complete bipartite graph if every vertex in U is connected to every vertex in V. If U has n elements and V has m, then we denote the resulting complete bipartite graph by Kn,m

Example



Coloring

Painting all the verices of a graph with colors suchthat no two adjacent vertices have the same color is called proper coloring (or some times called coloring) of a graph.

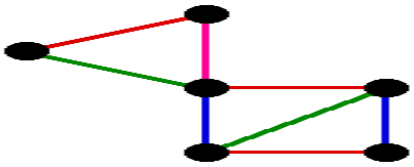
Graph coloring have commonly three types

- Edge coloring
- Vertex coloring
- Face coloring

Edge Colorig

Assignment of colors to the edges is called Edge Colorig (i.e., Each edge gets one color)

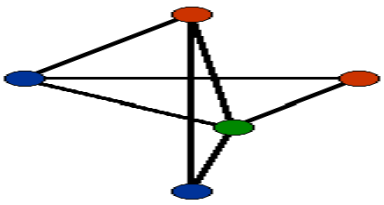
Example



Vertex coloring

Assignment of colors to the vertex is called **vertex coloring** or **node Colorig** (i.e., Each edge gets one color)

Example

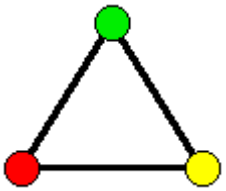


Chromatic number

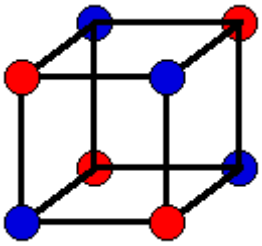
The vertex chromatic number or simply the chromatic number of a finite, loop-free graph G is the smallest positive number k such that G is k-colorable. The **chromatic number** of a graph G is usually denoted by (G).

Examples

The triangle graph has **chromatic number 3.**



The cube graph has **chromatic number 2**.



K-Edge colorable

A graph G is said to be **k-edge colorable** if its edges can be colored using up to k-colors such that no two edges with a vertex in common have the same color.

Example

Unicast Communication in MANETs

There are two broad classifications of unicast routing protocols: minimum-weight based routing and stability-based routing. Routing protocols under the minimum-weight category have been primarily designed to optimize the hop count of source-destination (s-d) routes. Some of the well-known minimum-hop based routing protocols include the **Dynamic Source Routing** (DSR) protocol [8] and the **Ad hoc On-demand Distance Vector** (AODV) routing protocol [16].

The **DSR** protocol is a source routing protocol that requires the entire route information to be included in the header of every data packet. However, because of this feature, intermediate nodes do not need to store up-to-date routing information in their routing tables. Route discovery is by means of the broadcast query-reply cycle. The Route Request **(RREQ)** packet reaching a node contains the list of intermediate nodes through which it has propagated from the source node. After receiving the first

RREQ packet, the destination node waits for a short time period for any more RREQ packets, then chooses a path with the minimum hop count and sends a Route Reply (RREP) along the selected path. Later, if any new RREQ is received through a path with hop count less than that of the selected path, another RREP would be sent on the latest minimum hop path discovered.

Graph Theory Algorithms for Unicast Communication in MANETs

In a graph theoretic context, we illustrate that the minimum-weight (minimum-hop) based routing protocols could be simulated by running the shortest-path **Dijkstra algorithm** [4] on a mobile graph (i.e. a sequence of static graphs).

We then illustrate that the **NVSP** and **FORP** protocols could be simulated by respectively solving the smallest bottleneck and the largest bottleneck path problems – each of which could be implemented as a slight variation of the shortest path Dijkstra algorithm. In addition, we also illustrate that the Prim's minimum spanning tree algorithm and its modification to compute the maximum spanning tree can be respectively used to determine the 'All Pairs Smallest Bottleneck Paths' and 'All Pairs Largest Bottleneck Paths' in a weighted network graph.

Shortest Path Problem

Given a weighted graph G = (V, E), where V is the set of vertices and E is the set of weighted edges, the shortest path problem is to determine a minimum-weight path between any two nodes (identified as source node s and destination node d) in the graph. The execution of the Dijkstra algorithm (pseudo code in Figure 1) on a weighted graph starting at the source node s results in a shortest path tree rooted at s. In other words, the Dijkstra algorithm will actually return the minimum-weight paths from the source vertex s to every other vertex in the weighted graph. If all the edge weights are 1, then the minimum-weight paths are nothing but minimum-hop paths.

Begin Algorithm Dijkstra-Shortest-Path (G, s)

1 **For** each vertex v Є V

2 weight [v] ← ∞ // an estimate of the minimum-
weight path from s to v

3 **End For**

4 weight [s] ← 0

5 S ← Φ // set of nodes for which we know the
minimum-weight path from s

6 Q ← V // set of nodes for which we know estimate of the minimum- weight path from s

7 **While** Q ≠ Φ

8 u ← EXTRACT-MIN (Q)

9 S ← S U {u}

10 **For** each vertex v such that (u, v) Є E

11 **If** weight [v] > weight [u] + weight (u, v) **then**

12 weight [v] ← weight [u] + weight (u, v)

13 Predecessor (v) ← u

14 **End If**

15 **End For**

16 **End While**

17 **End** Dijkstra-Shortest-Path

Data Structures used by the MaxD-CDS Algorithm and Breadth First Search

We use the following principal data structures for the MaxD-CDS algorithm:

(i) CDS-Node-List – includes all nodes that are members of the CDS

(ii) Covered-Nodes-List – includes all nodes that are in the CDS-Node-List and all nodes that are adjacent to at least one member of the CDS-Node-List.

Before we run the CDS formation algorithm, we make sure the underlying network graph is connected by running the Breadth First Search (BFS) algorithm [4]; because, if the underlying network graph is not connected, we would not be able to find a CDS that will cover all the nodes in the network. We run BFS, starting with an arbitrarily chosen node in the network graph. If we are able to visit all the vertices in the graph, then the corresponding network is said to be connected. If the graph is not connected, we simply continue with the static graph (snapshot of the network topology) collected at the next time instant and start with the BFS test. The pseudo code for BFS is shown in Figure 14. The run-time complexity of BFS is O(|V|+|E|).

Input: Graph G = (V, E)

Auxiliary Variables/Initialization:

Nodes-Explored = Φ, FIFO-Queue = Φ

Begin Algorithm BFS (G, s) root-node = randomly chosen vertex in V Nodes-Explored = Nodes-Explored U {root-node} FIFO-Queue = FIFO-Queue U {root-node} **while** (|FIFO-Queue| > 0) **do**

front-node u = Dequeue(FIFO-Queue) // extract the first node

for (every edge (u, v)) **do** // i.e. every neighbor v of node u

if (v Nodes-Explored) **then** Nodes-Explored = Nodes-Explored U {v}

FIFO-Queue = FIFO-Queue U {v} Parent (v) = u

end if end for

end while

if (| Nodes-Explored | = | V |) **then return** Connected Graph - true **else return** Connected Graph – false

end if

End Algorithm BFS

Conclusions

The high-level contribution of this paper is the idea of using traditional graph theory algorithms, which have been taught in academic institutions at undergraduate and graduate level, to simulate and study the behavior of the complex routing protocols for unicast, multicast, broadcast and multi-path communication in MANETs.

In the Section on Background work, we provided an exhaustive set of background information on the routing protocols that have been proposed for the above different communication problems. In the subsequent sections, we described one or more graph theoretic algorithms for studying each of these communication problems.

We chose the Dijkstra algorithm for shortest path routing as the core algorithm and meticulously modified it and/or adopted it to (i) find a solution for the largest bottleneck path and smallest bottleneck path problems, which could be used to determine a sequence of stable routes as well as to (ii) find a set of link-disjoint, node-disjoint or zone-disjoint routes for multi-path communication.

REFERENCES

[1] K. M. Alzoubi, P.-J Wan and O. Frieder, "Distributed Heuristics for Connected Dominating Set in Wireless Ad Hoc Networks," *IEEE / KICS Journal on Communication Networks*, Vol. 4, No. 1, pp. 22-29, 2002.

[2] S. Butenko, X. Cheng, D.-Z. Du and P. M. Paradlos, "On the Construction of Virtual Backbone for Ad Hoc Wireless Networks," *Cooperative Control: Models, Applications and Algorithms*, pp. 43-54, Kluwer Academic Publishers, 2002.

[3] S. Butenko, X. Cheng, C. Oliviera and P. M. Paradlos, "A New Heuristic for the Minimum Connected Dominating Set Problem on Ad Hoc Wireless Networks," *Recent Developments in Cooperative Control and Optimization*, pp. 61-73, Kluwer Academic Publishers, 2004.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. [17] T. Ozaki, J-B. Kim and T. Suda, "Bandwidth-Stein, "Introduction to Algorithms," 2nd Edition, MIT Press/ McGraw-Hill, September 2001.

[5] K. Fall and K. Varadhan, "ns notes and documentation," The VINT Project at LBL, Xerox PARC, UCB, and USC/ISI, http://www.isi.edu/nsnam/ns, August 2001.