

Distributed Location-Aware Transmission for Ad Hoc Networks

Bey-Ling Su
Dept. of Engineering Science
National Cheng Kung University
Tainan, Taiwan 700
Phone: +886-6-390-1177
su12@mail.ncku.gov.tw

Ray Yueh-Ming Huang
Dept. of Engineering Science
National Cheng Kung University
Tainan, Taiwan 700
Phone: +886-6-275-7575 Ex. 63336
raymond@mail.ncku.edu.tw

Fenglien Lee
National Center for
High-Performance Computing
Hsin-Chu, Taiwan 300
Phone: +886-3-577-6085
flee@nchc.org.tw

Abstract

In this paper, we present an efficient single-flooding algorithm for collecting the node location information in a wireless ad hoc network. Then we use these nodes information, as an undirected graph, and a modified Dijkstra algorithm to find the shortest packet transmission paths. Using these paths, we describe how to perform unicast, multicast and broadcast in the network easily and feasibly. Finally, we discuss the transmission handling for node reorganization.

Keywords: Wireless Ad Hoc Network, Flooding, Dijkstra Shortest-Path Algorithm, Unicast, Multicast, Broadcast.

1. Introduction

One of the most vibrant and active "new" fields today is that of ad hoc networks. Within the past few years, though, the field has seen a rapid expansion of visibility and work due to the proliferation of inexpensive, widely available wireless devices and the network community's interest in mobile computing.

An ad hoc network is a (possibly mobile) collection of communications devices (nodes) that wish to communicate, but have no fixed infrastructure available, and have no predetermined organization of available links. Individual nodes are responsible for dynamically discovering which other nodes they can directly communicate with. A key assumption is that not all nodes can directly communicate with each other, so nodes are required to relay packets on behalf of other nodes in order to deliver data across the network. A significant feature of ad hoc networks is that rapid changes in connectivity and link characteristics are introduced due to node mobility and power control practices. Ad hoc networks can be built around any wireless technology, including infrared and radio frequency (RF).

Ad hoc networks are suited for use in situations where infrastructure is either not available, not trusted, or should not be relied on

emergency. A few examples include: military soldiers in the field; sensors scattered throughout a city for biological detection; an infrastructure-less network of notebook computers in a conference or campus setting; the forestry or lumber industry; rare animal tracking; space exploration; undersea operations; and temporary offices such as campaign headquarters[1].

In this paper, we assume the location of each node is aware of its geographical position that can be easily obtained by equipping a node with Global Positioning System (GPS) receiver. Since nodes in an ad hoc network lack of a fixed infrastructure, every node in the network is responsible for disseminating its GPS measures to all the other nodes. Traditionally, this is obtained by flooding the network with a packet containing its GPS measures [2].

In this paper, at first we present an efficient single-flooding algorithm for a root node, which can be any node initially, to collect all other node locations in an ad hoc network instead of using the traditional multi-flooding approach. Then we use these GPS information to build a location table for all nodes and a distance table for each pair of nodes. Each entry of the location table contains the (x,y) coordinates and the id of a node. Initially, the root node uses the node information collected by the above single-flooding strategy, as an undirected graph, to develop a modified Dijkstra algorithm to find the shortest paths for packet transmissions in the network.

Next, the root node disseminates, by these shortest paths, the location and distance tables to all other nodes. Afterward, each node owns the position and distance tables. Consequently, each node may use the modified Dijkstra algorithm to find the shortest paths to all other nodes as well.

Later, we use these paths to describe how to perform unicast, multicast, and broadcast in the network effectively. Finally, we discuss the transmission handling situations for node reorganization in the network. In conclusion, we show our contributions in this paper and point out the open problems in this popular research area.

2. Node Location Collection by Single-Flooding

Assume there are n nodes in a wireless ad hoc network, if the location of each node is located by a GPS initially. For the traditional localization strategy, at first, node 1, can be any node, floods its location information to all other $n-1$ nodes, then each of the other $n-1$ nodes needs to flood its location information to other $n-1$ nodes. Therefore, it needs to execute n times of flooding to let each node to get the location information of all n nodes in the network. This inefficient method is not only time-consuming, but also create the traffic jam problem in the network. In our algorithm below, we only use flooding once and run the modified Dijkstra algorithm to get the same result as the traditional way. Obviously, our algorithm will relieve the above two problems dramatically.

Algorithm 1: Single-Flooding

Step 1: Arbitrarily select a node as the root-node in the network. Then the root node floods an RREQ to request the ids and GPS positions of all other $n-1$ nodes

Step 2: Each of the $n-1$ nodes follows the request path in Step 1 to send its id and GPS location back to the root node

Step 3: Until the root node receives the ids and GPS locations from all the other $n-1$ nodes, it builds a location table and creates a distance table for all n nodes

Step 4: Call the modified Dijkstra algorithm below to find the shortest path to each of other $n-1$ nodes.

Step 5: Follow each shortest path to dispatch the location and distance tables to other $n-1$ nodes.

End (* of Single-Flooding *)

In the above algorithm, the root-node floods its location information to other $n-1$ nodes in step 1 only. This is why we called it "single-flooding".

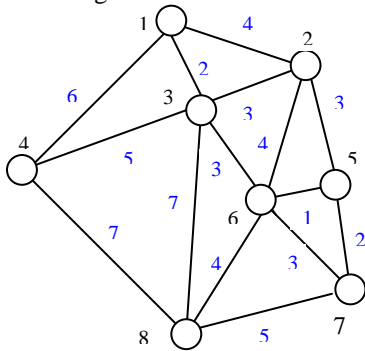


Figure 1: A Sample Ad Hoc Network

Node id	Location
1	X1, Y1
2	X2, Y2
3	X3, Y3
4	X4, Y4
5	X5, Y5
6	X6, Y6
7	X7, Y7
8	X8, Y8

Table 1: Location Table

	1	2	3	4	5	6	7	8
1								
2	4							
3	2	3						
4	6	8	5					
5	6	3	3	8				
6	5	4	3	6	1			
7	7	5	5	8	2	3		
8	8	7	6	7	4	5	5	

Table 2: Distance Table

Assume the transmission radius of each node is 5. If the distance between any two nodes greater than 5, then set its value to ∞ . The above distance table can be modified to the following table.

	1	2	3	4	5	6	7	8
1								
2	4							
3	2	3						
4			5					
5		3	3					
6	5	4	3		1			
7		5	5		2	3		
8					4	5	5	

Table 3: Modified Distance Table

As described in Step 5 above, the root node will dispatch the location and distance tables, e.g. Table 1 and Table 3, to other $n-1$ nodes.

3. The Modified Dijkstra Algorithm

So far, each node owns the location table and distance table. If any node wants to unicast, multicast or broadcast packets efficiently, then it has to execute this modified Dijkstra algorithm

on-demand to generate the shortest path to each of other n-1 nodes.

In the following algorithm, we assume
n: the total node number in the ad hoc network.
v: the source node.
k: the transmission radius in the network.
s(i): the visiting flag of node i. If it is not visited then s(i)=0 else s(i)=1.
cost(v,i) : the virtual distance between node v and node i.
dist(w) : the shortest path from source node to node w.

Algorithm2:

Modified Dijkstra Algm (v, cost, dist, n, k)

Output: A table of the shortest paths from source node v to all other nodes.

Begin

Step1: For i=1 to n do
 s(i)=0 ;
Step2: If cost (v, i) <= k
 then dist (i) <- cost (v, i)
 else dist(i) <- (* greater than the transmission radius k *)
 endif
Step3: s(v) <- 1 ; dist(v) <- 0 ; num <- 2 ;
Step4: While num < n do
Step5: Choose u: dist(u)= Min{dist(w)} at s(w)=0
Step6: s(u) <- 1 ; num <- num + 1
Step7: For all w with s(w) =0 do
Step8: If dist(w) <= k
 Then dist(w) <- Min{dist(w), dist(u) + cost (u,w)}
 End (* For all *)
End (* While *)
End (* Algorithm2 *)

Note, if dist(w) is within the transmission radius k and there is an intermediate node u such that dist(u) plus cost (u,w) is longer than dist(w), then substitute dist(w) for (dist(u) + cost(u,w)).

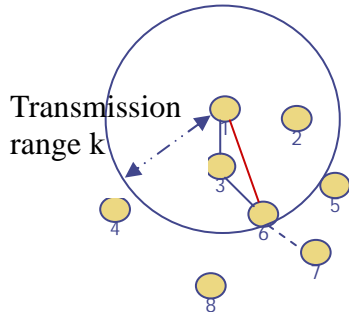


Fig. 2: The Shortest Path Example (Node 1 transmits the packages to node 7, by using the modified Dijkstra algorithm, will take the

furthest path from node 1 to node 6 rather than from node 1 to node 3 to reduce hops).

The shortest path means that a node transmits packages to another node will choose the furthest node within its transmission range to transfer packages and skip other intermediate nodes (see Fig. 2). The source node will go through the least-hop path to reach the destination node. As a result, the whole path will eliminate unnecessary hops and package passes in the network.

4. Unicast, Multicast and Broadcast

After a node executed the modified Dijkstra algorithm, it will get the shortest paths to all other n-1 nodes (see Table 4 below).

Destination node id	shortest path
Node 2	1->2
Node 3	1>3
Node 4	1->3->4
Node 5	1->3->6->5
Node6	1->3->6
Node 7	1->3->6->7
Node 8	1->3->6->8

Table 4: The shortest path table for node 1 (assume node 1 is the source node)

A node can simultaneous unicast, multicast and broadcast to other n-1 nodes efficiently by using the shortest path table. For unicast, a node, such as node 1, can choose any node in the network as the destination node. Then it will follow the shortest path to that node to transmit packets. For broadcast, a node, such as node 1, can follow the shortest paths to all other n-1 nodes to perform broadcasting.

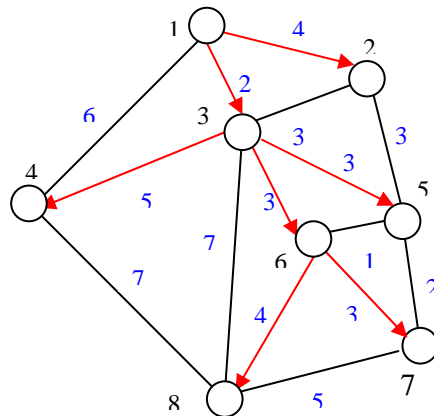


Fig. 3: Broadcast: Assume node 1 sends packets to all other n-1 nodes according to the following shortest-path table after execute the Modified

Dijkstra Algorithm.

For multicast, a node can choose shortest paths for the multicast nodes from its shortest path table. Assume the multicast nodes are 4, 6, 7 and 8, then we may retrieve shortest paths to these nodes as shown in Table 5 below.

Multicast node	shortest path
Node 4	1->3->4
Node 6	1->3->6
Node 7	1->3->6->7
Node 8	1->3->6->8

Table 5: Multicast table for node 1

For broadcast and multicast, our approach will eliminate any repeat paths during packet transmission. For example, in Table 5, it repeats path from node 1 to node 3 for four times, and from node 3 to node 6 for three times. The repeated paths will be integrated into a single path. This means, at first, node 1 sends packets only one time to node 3. Then, node 3 multicasts the packets to nodes 4 and 6. Finally, node 6 multicasts the packets to nodes 7 and 8, respectively. Obviously, we skip the repeated paths and make the packet transmission efficiently

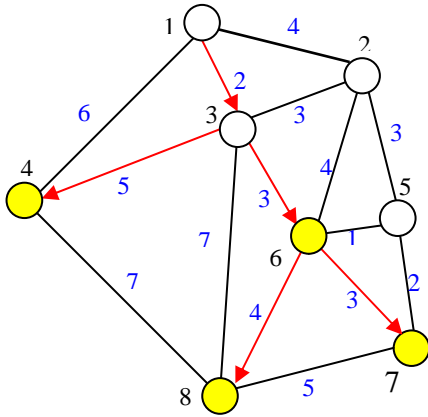


Fig. 4: For multicast, we take the repetition paths only once. (Assume the source node is node 1)

5. Transmission Handling for Node Reorganization

If a node adds to or removes from the ad hoc network, it should inform all other nodes. Each node has an added-list and a removed-list for handling the node changes in the network. The added-list (or the Removed List) is a link list for storing the added (or removed) nodes in the network. After a node executes the modified Dijkstra algorithm, the added-list (or removed-list) will be reset to null.

If one node added in, it sends RREQ to its neighbors within the transmission range for requesting the location table, the distance table and the two lists. After getting the above information, the added node executes the modified Dijkstra algorithm to send its position to other n-1 nodes by using the shortest-path table.

Once a node received the added node information, it will append the node's info to the added-list. It is not necessary to update the location table, distance table and shortest-path table immediately. Only when it will send packets to a node in the added list, it needs to update location table, distance table by using the information of both added-list and removed-list. Then it executes the modified Dijkstra algorithm as well. Otherwise, it uses its previous shortest-path table.

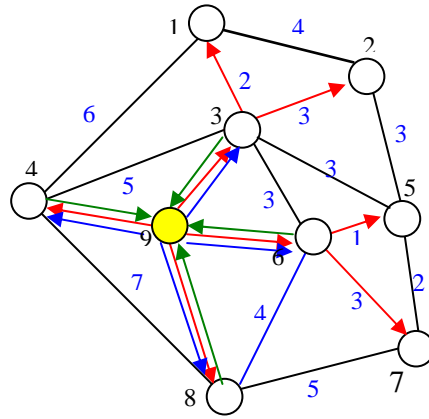


Fig.5 Blue paths: added node 9 send RREQ to its neighbors; Green paths: reply from its neighbors for sending information of tables and lists; Red paths: notify to all other n-1 nodes for adding the network.

If one node removed, it will inform other nodes by using the shortest-path table. Once the other node received the information of the removed node. It will append the information to the removed node list. It is not necessary to update the location table, distance table and shortest-path table immediately. When any node sends packets to other nodes, it will check if its transmission path including any removed nodes or not. If yes, it needs to execute the modified Dijkstra algorithm. Otherwise, it uses its previous shortest-path table.

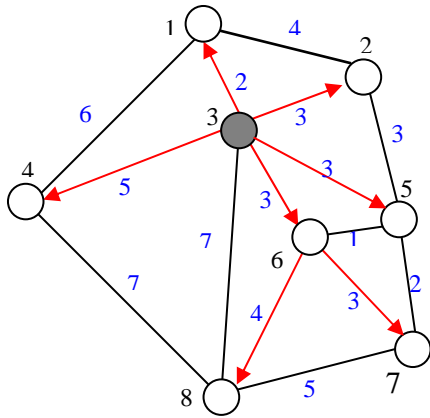


Fig. 6: Node 3 removed, will inform other nodes by using the shortest-path table.

6. Conclusion

In this paper, for a wireless ad-hoc network with n nodes, we developed a Single-Flooding algorithm and a modified Dijkstra algorithm to distribute the id and location of each node to all other $n-1$ nodes efficiently. With these information in each node, we also described how to use the shortest path, which were generated by the modified Dijkstra algorithm, from a source node to each destination node to unicast, multicast and broadcast packets in the network effectively. At last, we also described how to handle nodes adding and removing efficiently by means of the added-list and removed-list concepts, respectively. The contributions of this paper are as follows.

- (1) Single flooding. The main advantage in this paper is single flooding. Our algorithm uses only one flooding for collecting node locations at the beginning. Afterward all other $n-1$ nodes have the position table and the distance table by executing the modified Dijkstra algorithm. Traditional methodology has to flood n times to get the same result.
- (2) Using less hops to transmit packets in the network. Our paradigm builds a shortest path table for each node. Hence, a node transfers packets through the longest distance node in the transmission range, the whole path has less hops and saving transmission time.
- (3) Efficient unicast, multicast and broadcast. In our approach, each node could simultaneously execute unicast, multicast and broadcast according to the shortest path table.
- (4) Handling moving-nodes easily. If a node moves (added in or left out), it will notify all

other $n-1$ nodes by using the shortest-path table instead of using flooding. Those nodes received the notice will update its added-list or removed-list.

The only concern in our paper is that it will take much time to run the modified Dijkstra algorithm if the network has a huge amount of nodes. Besides, we need the GPS location information for each node initially.

7. References

- [1]“A Brief Overview of Ad Hoc Networks Challenges and Directions”, Ram Ramanathan and Jason Redi, BbnTechnologies.
- [2]“On-Demand Location Aware Multicast (OLAM) for Ad Hoc Networks”, Stefano Basagni, Imrich Chlamtac, Violet Syrotiuk, and Rodeen Talebi, U of Texas at Dallas, 2000
- [3]“Position-Based Multicast Routing for Mobile Ad-Hoc Networks”, Martin Mauve, Holger Fubler, Jorg Widmer, Thomas Lang, 2003.