Fairness Improvement and Efficient Rerouting in Mobile Ad Hoc Networks

Norihiro Ohata, Yongbing Zhang
Department of Social Systems and Management
Graduate School of Systems and Information Engineering
University of Tsukuba, Japan
e-mail: yb.zhang@sk.tsukuba.ac.jp

Yusheng Ji
Institute of Informatics, Japan
e-mail: kei@nii.ac.jp

Xuemin (Sherman) Shen
Department of Electrical and Computer Engineering,
University of Waterloo, Canada
e-mail: xshen@bbcr.uwaterloo.ca

Abstract— Mobile ad hoc networking allows nodes to form temporary networks and communicate with each other possibly via multiple hops. By using a special node called the gateway, an ad hoc network can be connected to the Internet so that packets generated in the ad hoc network can be relayed to the Internet and vice versa. However, a problem may arise in which the network bandwidth is not used fairly among the nodes. That is, nodes near the gateway may overuse the bandwidth while nodes far away from the gateway scarcely share the bandwidth. Furthermore, if the destination node is located in the same ad hoc domain as the source, existing route selection methods result in high overhead or long routes. Addressing these problems, we propose an efficient tree construction algorithm that constructs a tree structure rooted at the gateway in order to improve the fairness of the bandwidth usage. We also devise a rerouting algorithm that dynamically finds new and better routes for the internal traffic within the same ad hoc domain.

Keywords—fairness; rerouting; multi-hop wireless networks; flow balance; mobile IP.

I. INTRODUCTION

Wireless communication has generated an explosive growth in the number of laptop computers and personal digital assistants (PDAs); this growth has been accompanied by advancement in computer and wireless networks. Mobile IP has been proposed for wireless networks [1, 2] and enables mobile computers, called mobile nodes (MNs), to roam between different wireless networks but still maintain connectivity to the Internet. A special node called the gateway has been introduced to connect MNs to the Internet and relays the packets generated by MNs to the Internet and vice versa [3, 4]. However, some nodes may not reach the gateway directly due to radio transmission limitations. One solution to this problem is to group MNs to form an ad hoc network and allow MNs to send packets to the gateway via multiple intermediate nodes. To establish a transmission path from a MN to the gateway, a tree structure is usually used by taking the gateway as the root and taking MNs as the internal and leaf nodes.

Fairness is one of the most important performance indices to show how well MNs share the network's scarce resources and implement their performance objectives. The current wireless standards, e.g., IEEE 802.11 [5], are used to ensure medium access control communication (MAC) layer fairness [6]. However, MAC layer cannot achieve good network or higher layer fairness. Some research has focused on the fairness problem at the network or higher layer by serving the original traffic and the relayed traffic using distinct queues [7, 8]. Authors in [7] proposed several schemes that provide various fairness by using distinct MAC layer and network layer queues. The problem of the above approach [7] is that each node needs to maintain a priority queue list for various traffic requests and has to determine the number of queues it should hold. Additionally, there is not an efficient tree construction method to construct a tree structure in order to realize good fairness. A tree generated using a shortest path algorithm may not lead to good fairness. This is because a node with more offspring than others has to relay the traffic of its offspring as well as its own traffic, so each traffic flow can share only a small percentage of the whole bandwidth.

When the destination is located in the Internet while the source is located in the ad hoc domain, the source simply sends packets to the gateway and then the gateway relays the packets to the destination. If the destination is also located in the same ad hoc domain as the source, the route selection from the source to the destination can be determined by using either a tree structure approach with the root at the gateway or a flooding route discovery approach similar to those in ad hoc networks [9, 10]. The former approach yields a small route discovery overhead but probably a long path, resulting in a long transmission delay. The latter approach can find the best route, but a flooding message has to be broadcast to all the nodes in the network, resulting in a large route discovery overhead.

In this paper, we firstly propose an efficient tree construction algorithm that constructs a tree structure rooted at the gateway in order to improve the fairness of the bandwidth usage for each traffic flow. We then devise a rerouting algorithm with low overhead and short path length that dynamically finds new and better routes for the traffic whose source and destination are located in the same ad hoc domain. We use simulation to evaluate the performance of the proposed approaches and compare them with existing approaches.
II. PROBLEM FORMULATION

The assumptions used in this paper, such as agent advertisement broadcast and registration mechanisms, are basically the same as those in NEWLANA-RD [7], except that in our algorithm each node in an ad hoc network is capable of accessing the Internet if the network graph is connected. When a node communicates with its partner outside of its own ad hoc domain, it transfers the packets first to the gateway of its domain, probably via multiple hops. The gateway acts as the interface point between an ad hoc network and the Internet; it supports both the ad hoc and the Internet routing protocols. The gateway maintains the information of all the nodes in its ad hoc domain.

When a node has packets to send, it has two options to choose: to check whether the destination is located in the same network or to send all the packets to the gateway. In the former choice, the source uses an appropriate ad hoc routing algorithm to locate the destination and determine the route to it. It then transfers the packets directly to the destination if it is located in the same ad hoc domain, or to the gateway if it is not. In the latter choice, the source sends all the packets to the gateway. The gateway checks the location of the destination and relays the packets to the destination; the gateway uses the Internet routing protocol if the destination is on the Internet and uses the same ad hoc routing protocol if the destination is located in the same ad hoc domain.

We consider only one ad hoc network even though multiple ad hoc networks may be connecting to the Internet independently. We assume that there is only one gateway in an ad hoc network. The network graph is denoted by $G=(N, E)$, where the sets of nodes and links are denoted by $N$ and $E$, respectively. A tree structure, denoted by $T$, is constructed rooted at the gateway and only the gateway knows the route to each node on the tree. Each node on the tree has the information of its offspring and the nodes along the route from itself to the gateway. Each node also has the exact information of its neighboring nodes. We introduce two load metrics for constructing the tree structure. One is the node load, denoted by $\rho$, which indicates the number of nodes on the sub-tree rooted at node $i$, denoted by $T_i$, and the other is the path load, denoted by $\pi$, which indicates the sum of the node loads along the route from node $i$ to the gateway. The set of links along the route from node $i$ to the gateway is denoted by $\Pi_i$. Then, we have

$$\rho_i = |T_i|,$$  \hspace{1cm} (1)

$$\pi_i = \sum_{k \in \Pi_i} \rho_k,$$  \hspace{1cm} (2)

where $|T_i|$ denotes the number of nodes on sub-tree $T_i$ including node $i$.

An example network is shown in Figure 1, where the solid thick and the dotted lines indicate the links on the tree and not on the tree, respectively. The node and the path loads of nodes 1, 4, 5, 6, and 7 are shown in the rectangular boxes. The node load of node 1 is 5 because the tree rooted at node 1 has a sub-tree with 4 internal and leaf nodes. Furthermore, the path load of node 1 is also 5 because node 1 is connected to the root directly. Similarly, the node load of node 4 is 2. However, the path load of node 4 is 7 because $\pi_4 = \rho_1 + \rho_3 = 7$.

III. PROPOSED APPROACH

In this section, we first describe the tree construction algorithm that constructs a balanced tree structure based on the path load. Then, we describe a rerouting algorithm for data transmission done inside an ad hoc network.

A. Flow Balanced Tree Algorithm

MAC layer medium control protocols [5, 11] cannot guarantee flow fairness at the network layer because they deal only with the fairness of resource (channel) utilization between nodes. Generally, the number of traffic flows passing through a node differs from node to node, so some nodes may serve many traffic flows while some other nodes may have few traffic flows to serve. In this case, a node with many passing flows can only provide each flow with very limited bandwidth.

The flow balanced tree algorithm proposed in this paper takes into account the number of flows passing through a node and attempts to construct a balanced tree structure in the sense that all nodes with the same depth have the same path load. That is, when node $i$ receives the route discovery messages from its neighbors, denoted by the set $N_i$, it determines its parent node $n_p$ to join as follows.

$$n_p = \arg \min_{j \in N_i} \{\pi_j\}.$$  \hspace{1cm} (3)

For example, in the example network shown in Figure 1, node 8 receives three route discovery messages from its neighboring nodes 5, 6, and 7, and knows that the routes via nodes 5 and 6 are the shortest and they both have the same number of hops to the gateway. Node 8 can choose to join the route via either node 5 or 6 if using a common shortest path algorithm. However, when using the tree construction algorithm proposed in this paper, node 8 will select node 6 because the path load of node 6 is lighter than that of node 5.
The flow balanced tree algorithm shown in Figure 2 can be implemented similarly to NEWLANA-RD [7] except the mechanism to determine a node's parent. To construct the tree structure rooted at the gateway, the gateway broadcasts an agent advertisement (route discovery) message. When a node receives the agent advertisement message, the node selects the route with the lowest hop count to the gateway as its route to the gateway. If there are multiple routes to the gateway, it selects the route with the smallest path load. After updating the hop count, the gateway broadcasts the agent advertisement message again.

![Figure 2. Tree construction algorithm.](image)

B. Rerouting Algorithm

Most route discovery approaches in ad hoc networks are based on a flooding mechanism like dynamic source routing (DSR) [9], where a route request (RREQ) message is broadcast by the source toward the destination to find the best route. The overhead of route discovery is inevitably large since the RREQ message has to be sent to all the nodes in the network. On the other hand, in the rerouting algorithm we propose, the tree structure with the root at the gateway is used and packets generated at a source are firstly sent to the gateway. If the destination is located outside of the packets' source domain, the gateway relays them to the Internet by using an Internet routing protocol. On the other hand, if the packets' destination is located in the same ad hoc domain as their source, they are relayed by the gateway to the destination along the path from the gateway to the destination. The path passing through the gateway from the source to the destination surely may not be the best one. The rerouting algorithm we propose attempts to find dynamically a better path than the currently-used one between the source and the destination by using only the information of the current path and its neighboring nodes along the path, reducing the route adaptation overhead. Route adaptation approaches based on the flooding route discovery [10] can find the best path adaptively but their overhead is significantly large.

![Figure 3. Rerouting example.](image)

In the rerouting algorithm we propose, three kinds of messages are used: INFO messages sent by the destination and the nodes along the current route to the source to communicate the information of their neighboring nodes, UPDATE messages sent by the source to the destination and the nodes along the new route to communicate route updates, and FIN messages sent by the source to the destination to communicate the end of the rerouting process. The rerouting algorithm works as follows. When the destination receives a first data packet or an UPDATE message from a source in the same ad hoc domain, it sends an INFO message back to the source along the current route. The information of its neighboring nodes is included in the message. The previous node along the route also appends the information of its neighboring and forwards the message to the next node on the route. This procedure continues until the message reaches the source. The source determines a better route to the destination, if any, when it receives the INFO message and sends an UPDATE message to the destination. If the source cannot find any better route, it sends a FIN message to the nodes along the new route and the destination to end the
rerouting process. We assume that the routing information is valid within a pre-defined period and will be discarded when it expires.

Figure 3 shows a rerouting example where the thick solid lines denote the links on the tree structure rooted at the gateway and the dotted lines denote the links not on the tree. The route with the arrows from the source $s$ to the destination $d$ denotes the current route used for data transmission. Initially, as shown in Figure 3(a), the packets from the source are sent to the gateway and then relayed to the destination. After receiving the first packet or the UPDATE message, the destination and the nodes along the current route send the information of their one-hop neighboring nodes to the source. The source then determines whether to use a new route to the destination. The new route between the source $s$ and the destination $d$ shown in Figure 3(a) after rerouting is shown in Figure 3(b). The thin solid lines in the figure denote the links the source knows after obtaining the INFO message.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our proposed approach by simulation. It is assumed that the network service area is $1000 \text{ m} \times 1000 \text{ m}$ and nodes are randomly and uniformly allocated in the service area. The gateway is allocated at position $(500, 500)$. The transmission range of each node including the gateway, denoted by $R$, is the same for all nodes. It is assumed that every node has the exact information of its one-hop neighboring nodes. The traffic demand is based on constant bit rate (CBR) and the medium access control mechanism is based on IEEE 802.11b. The bandwidth of each wireless link is set to be 11 Mbps and the packet size is set to be 1500 bytes. Furthermore, the sizes of route discovery message and IP address are set to be 128 and 4 bytes, respectively.

To evaluate the fairness of different approaches, we define the following fairness indices for the per-flow throughput, which means the traffic flow between a source-destination pair, and for the power consumption.

$$f = \frac{\left( \sum_{i=1}^{n} x_i \right)^2}{n \sum_{i=1}^{n} x_i^2}, \quad (1 \leq i \leq n), \quad (4)$$

where $x_i$ denotes the per-flow throughput or the power consumption and $n$ denotes the total number of flows passing through in the network. From Eq. (4), we know that $0 < f \leq 1$, and the higher the fairness the larger the value of $f$. When all nodes achieve equal fairness, the value of $f$ is 1.

A. Comparison of Throughput and Power Consumption

Figures 4 and 5 show the fairness indices of per-flow throughput and power consumption in the flow balanced tree algorithm proposed in this paper, denoted by B-TREE in the figures, and NEWLANA [3] when the numbers of nodes are 50 and 100, respectively. In NEWLANA we used a shortest path routing algorithm based on the hop count to construct the tree structure. In the simulation experiments, the length of each simulation run was 30 s and the results shown in the figures are the average values obtained from 1000 simulation runs. The confidence interval is less than 5% of the sample mean shown in the figures.

Figure 4 shows that when the packet interarrival time is long, i.e., the offered load is less than the network capacity, there is no large difference in per-flow throughput performance due to tree structures. However, when the offered load becomes large, i.e., the traffic load is increased to exceed the network capacity, the B-TREE algorithm outperforms NEWLANA. This is because in B-TREE the total traffic flows passing through each node to the gateway are taken into account when constructing the tree structure. The figure also shows that the superiority of B-TREE becomes more obvious when the number of nodes in the network increases. Figure 5 shows that the B-TREE algorithm outperforms the NEWLANA algorithm in terms of power consumption over all ranges of offered loads. Furthermore, when the number of nodes is large, the advantage of B-TREE over NEWLANA becomes more noticeable.
B. Comparison of Overhead

As described in Section III.B, in previous research when the destination is located in the same ad hoc domain as the source, the source activates an ad hoc routing protocol to find the route to the destination or just sends packets to the gateway and asks the gateway to relay the packets to the destination. In the former method, the route discovery message is recursively broadcast, i.e., a flooding message is sent to all the nodes in the network. In the latter method, route discovery is not necessary. Each packet is sent from the source to the destination along the tree rooted at the gateway, so the route length and the transmission delay may be long. Hereafter, for the sake of simplicity we call this method the simple tree method. In the simulation experiments, we evaluated our proposed approach in comparison with these two methods using two performance measures, the route length and the route search overhead. The route length denotes the route length by the number of hops between a source-destination pair and the route search overhead denotes the total number of messages generated for route discovery in the network. The simulation results were obtained from 10000 simulation runs each with 100 randomly distributed nodes and 100 randomly selected source-destination pairs.

Figure 6 shows the route length using the rerouting algorithm proposed in this paper compared with those using the flooding route discovery and the simple tree method for various transmission ranges, i.e., \( R = 250 \text{ m}, 200 \text{ m}, \) and \( 150 \text{ m} \). The route length using the flooding method is the shortest because the source knows the exact information of all the nodes. The rerouting algorithm leads to a route length close to the shortest one. The route length obtained by using the simple tree method is much longer than those obtained using the other two methods. This figure demonstrates that by rerouting we can have much shorter paths. For example, when \( R = 250 \text{ m} \), the route length obtained using the rerouting algorithm is 20% shorter than that using the simple tree method. Figure 6 also shows that the transmission range \( R \) has no significant impact on the performance.

Figure 7 shows the route discovery overhead when using the rerouting algorithm proposed in this paper and the flooding approach for various transmission ranges. The route discovery overhead of our proposed approach is much smaller than that of the flooding approach even though the overheads of both approaches increase when the number of nodes becomes large. The figure also shows that the route discovery overhead in the rerouting algorithm increases gradually when the number of nodes becomes large, while in the flooding approach the overhead increases very sharply. In the rerouting approach, the source needs to collect only the information of the neighboring nodes of the nodes along the route from the source to the destination, so the route discovery overhead depends not on the total size of the network but on the route length from the source to the destination. In contrast, in the flooding approach the route discovery message must be delivered to each node in the network, so when the size of the network becomes large the number of route discovery messages increase explosively.

V. CONCLUSION

We proposed an efficient flow balanced tree approach for connecting mobile ad hoc networks to the Internet and devised a novel rerouting algorithm for data transmission when the source and the destination are both located in the same ad hoc network. The proposed flow balanced tree approach improves the fairness index dramatically when the size of the network is large and the offered traffic load is moderate or high. Furthermore, the proposed rerouting algorithm yields a much shorter path length for a source-destination pair than the simple tree approach by itself and does so with a much lower route discovery overhead than does the flooding routing approach. We verified the advantages of our proposed approach using simulation.

REFERENCES


Figure 6. Path length in number of hops.

Figure 7. Route discovery overhead.