

Tree-Based Double-Covered Broadcast for Wireless Ad Hoc Networks

Weisheng Si¹

¹ National ICT Australia Ltd. (NICTA)
Locked Bag 9031
Sydney, Australia
weisheng.si@nicta.com.au

Roksana Boreli^{1,2} Anirban Mahanti¹

² School of EE&T
Univ. of New South Wales
Sydney, Australia
roksana.boreli | anirban.mahanti
@nicta.com.au

Albert Y. Zomaya³

³ School of Information Technologies
Univ. of Sydney
Sydney, Australia
albert.zomaya@sydney.edu.au

ABSTRACT

The increasing use of smartphones with integrated Wi-Fi interface presents opportunities for broadcasting multimedia streams or contents using wireless ad hoc networks, particularly in crowd scenarios like stadiums or musical concerts. Since most of current broadcast protocols use 2-hop neighborhood information and hence involve long Hello packets, this paper proposes a tree-based double-covered broadcast protocol (TreeDCB), which uses fixed-length Hello packets and guarantees that each node is either a forwarding node or covered by at least two forwarding nodes (not including the children nodes of this node). TreeDCB uses the basic shortest path tree technique to decide the forwarding nodes. Meanwhile, it introduces the following two new mechanisms: (1) it selects parent nodes in the tree by examining which one has the greatest number of children, thus significantly reducing the number of parent nodes and (2) a leaf node will volunteer to do the forwarding if it hears no forwarding nodes other than its parent, thus ensuring double coverage for non-forwarding nodes. By *ns-2* simulation, we compare TreeDCB with the recent Double Covered Broadcast (DCB) protocol, which uses 2-hop neighborhood information, showing improvements in terms of the amount of control traffic, the number of forwarding nodes, the packet delivery ratio, and the packet path length.

Categories and Subject Descriptors

C.2 [Computer-communication networks]: Network protocols
—*wireless communication, routing protocols*

General Terms

Algorithms, Design, Performance, Experimentation

Keywords

Wireless ad hoc networks; multi-hop broadcast; shortest path tree; double coverage.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MobiWac'11, October 31- November 4, 2011, Miami, Florida, USA.

Copyright 2011 ACM 978-1-4503-0901-1/11/10...\$10.00.

1. INTRODUCTION

With widespread adoption of smartphones that have WiFi interfaces, wireless ad hoc communication is expected to play an important role in multimedia streaming and content sharing applications [2, 9]. A typical example scenario is the congregation of individuals with WiFi-enabled smartphones to form an ad hoc network in sports events or business events. A key component for the broadcast applications is a broadcast routing protocol. The straightforward blind flooding [22] approach, where each node other than the source rebroadcasts a packet if it receives this packet for the first time, results in the broadcast storm problem [24], in which packets experience severe collisions and congestion. Typical broadcast applications, therefore, looks for a routing protocol that reduces the number of forwarding nodes (the source nodes and the rebroadcasting nodes) in the network as well as achieves a high packet delivery ratio.

Many broadcast routing protocols have been proposed for wireless ad hoc networks [8-13, 15-18, 20-21, 26]. Since broadcasting along a tree topology is believed to be fragile in wireless environment, most of these protocols resort to flooding-like or connected dominating set (CDS) based approaches (to be detailed in Section II). However, CDS-based protocols require their nodes maintain k -hop ($k \geq 2$) neighborhood information. Consequently, a node needs to include the $(k-1)$ -hop information in its periodical Hello packets. Thus, in dense wireless networks such as crowds with WiFi-enabled smartphones, Hello packets become considerably long and consume excessive network capacity.

Motivated by the above observation, this paper proposes a tree-based double-covered broadcast (TreeDCB) protocol that only uses fixed length Hello packets and ensures that each node is either a forwarding node or covered by at least two forwarding nodes (not including the children nodes of this node). TreeDCB uses the basic shortest path tree technique to build a spanning tree with source node as the root among all network nodes and let the parent nodes do the forwarding and the non-parent nodes not. To reduce the number of parent nodes in the tree, when a node selects its parent among its upstream neighbors, TreeDCB selects the one that has the greatest number of children as the parent, which aims to cluster nodes with a common parent as much as possible. Meanwhile, to overcome the fragility of tree topology, a non-parent node volunteers to be a forwarding node if it hears no forwarding nodes from its upstream and sibling nodes other than its parent, thus ensuring that each non-forwarding node is covered by at least two forwarding nodes which are either its parent or a sibling node. Though the ideas of TreeDCB are simple, they suit the characteristics of dense wireless networks where (1) fixed-length Hello packets are desired, (2) the number of forwarding nodes

should be kept small, and (3) the broadcast redundancy can be highly utilized.

Extensive simulations using *ns-2* [14] show that TreeDCB achieves a low number of parent nodes, high packet delivery ratio, low control overhead, and short path lengths. We also compare TreeDCB with the Double Covered Broadcast (DCB) [13] protocol that utilizes 2-hop neighborhood information, demonstrating the improvements of TreeDCB over DCB.

The remainder of this paper is organized as follows. Section II presents a summary of the related work. Section III describes the details of TreeDCB, which is evaluated in Section IV. Finally, Section V concludes this paper.

2. RELATED WORK

Based on the information each node keeps, the existing broadcast protocols for wireless ad hoc networks can be classified into the following three categories [11]: (1) no neighbor information [22, 24]; (2) 1-hop neighbor information [9, 11, 20]; and (3) k -hop ($k \geq 2$) neighbor information [3, 10, 12-13, 15-18, 21, 26].

Within category (1), blind flooding [22] is shown to be suitable for highly dynamic networks [8], but results in the broadcast storm problem for less dynamic networks. In probabilistic flooding [24], a non-source node only rebroadcasts a packet with a probability P , thus reducing the number of broadcast transmissions. However, probabilistic flooding suffers from a low packet delivery ratio in many network scenarios.

Within category (2), Collaborative Streaming among Mobiles (COSMOS) [9] is based on the idea that a node will rebroadcast a packet if a large portion of its neighbors have not received that packet. Specifically, a node calculates the ratio of the number of its neighbors that will benefit from its rebroadcast of a packet versus the total number of its neighbors. If this ratio is larger than a threshold value (configurable), it rebroadcasts that packet, otherwise not. Though COSMOS uses constant length control packet, its forwarding node selection mechanism cannot guarantee that all nodes can be reached from a single source, so it may require multiple source nodes, while our protocol can build a tree to reach all the nodes from a single source. Protocols proposed in [11, 20] are also efficient due to the use of constant length control packets, but they exploit the location information of nodes, which is obtained from the GPS devices. Since the accuracy of GPS devices is limited and susceptible to environment [25], these location-based protocols are not suited to the wireless networks where the nodes can be very close to each other such as in a crowd.

Category (3) includes the largest number of protocols [3, 10, 12-13, 15-18, 21, 26], most of which are based on the concept of a connected dominating set (CDS) [6]. For a graph $G(V, E)$, a CDS is a subset of V , such that (1) the nodes in this subset are connected; (2) each node in V is either in this subset or connected to at least one node in this subset. The basic idea of these protocols is to find a CDS among all the nodes. By only letting the nodes in CDS broadcast, the number of forwarding nodes is reduced and a packet can still reach every node in the network. It is proved NP-complete to find the CDS that contains the minimum number of nodes [6]. Consequently, in this category of protocols, nodes use certain heuristics based on their k -hop ($k \geq 2$) information to compute a suboptimal CDS. In general, when a larger k is used, a smaller CDS can be computed. However, when the node density is high, the Hello packets become long, since this category of protocols needs to attach $(k-1)$ -hop ($k \geq 2$) information to the Hello packets. Besides, the computation time at a node to decide whether it is in the CDS also increases significantly, since all these protocols have

a computation complexity of at least $O(d^2)$ at a node [5], where d is the maximal node degree in the network.

Among the category (3) protocols, the Double Covered Broadcast (DCB) protocol [13] inspires our work, and we will compare our work with DCB in the evaluation section. In DCB, when a node v forwards a packet, it selects a subset of its 1-hop neighbors as the forwarding nodes based on the greedy algorithm for the Set Cover problem [4], with the constraints that (1) the selected forwarding nodes cover all the 2-hop neighbors of node v and (2) the 1-hop neighbors of node v are either selected as forwarding nodes or covered by at least two forwarding nodes (e.g., once by node v itself and once by one of the selected forwarding nodes). Then, node v attaches the IDs of the selected forwarding nodes to this packet and broadcast this packet. When a 1-hop neighbor of node v receives this packet and sees its ID in this packet, it knows that it is a forwarding node and continue the same broadcasting procedure as node v . Another feature of DCB is that a node tries to transmit a packet to all its selected forwarding nodes reliably. Specifically, a node v waits for a certain period to overhear the rebroadcasting from all its selected forwarding nodes. If node v fails to hear the rebroadcasting from one of them, it will retransmit the packet until all its forwarding nodes' rebroadcastings are heard or the maximum number of retries is reached.

The differences between our protocol and DCB are as follows. Firstly, in DCB, each node includes its 1-hop neighbor list to its Hello packets, so that each node can discover 2-hop neighborhood information, while in TreeDCB, each node only includes constant amount information in the Hello packets to build a tree topology. Secondly, in DCB, a node attaches the IDs of its selected forwarding nodes to every data packet it forwards, which increases the control overhead, while in TreeDCB, a node decides whether is a forwarding node by its own, adding no control overhead to a data packet. Thirdly, DCB is a reliable broadcast protocol, so a node in DCB retransmits a packet in case of reception failure, while TreeDCB only aims to achieve a high packet delivery ratio at a low cost, so a node in TreeDCB does not retransmit. Finally, in DCB, any node can be the source node, while in TreeDCB, a single source node is assumed to lower the control overhead.

3. PROTOCOL DESCRIPTION

TreeDCB works under the following assumptions: (1) there exists one source node in the network; (2) the wireless links are symmetrical: if node u is within the transmission range of node v , then node v is within the transmission range of node u . For the extension to multiple source nodes, please see our discussions in Subsection 3.5.

3.1 Roles of a node

A node in TreeDCB can be in one of the following three roles:

- *Parent node*: a node that has children in the current tree topology. A parent serves as a forwarding node.
- *Volunteer node*: a node that has no children in the current tree topology but volunteers to be a forwarding node.
- *Leaf node*: a node that has no children in the current tree topology and is not a volunteer.

3.2 Hello packets

The exchange of control messages in TreeDCB only involves periodical beaconing of Hello packets to its 1-hop neighbors at each node. A Hello packet has the following five fields with each

field costing four bytes (Figure 1). In the description below, we call the sending node of a Hello packet ‘the sender’.

- **SrcID**: the ID of the source node, which indicates the root of the tree to build.
- **SelfID**: the ID of the sender.
- **Cost**: the shortest path length from the source to the sender. For simplicity, we use the number of hops as the metric of Cost in this paper. In practice, any metric of positive value can be used by TreeDCB.
- **NumChildren**: if this field is larger than 0, it indicates the sender is a *parent node* and it gives the number of children that the sender has. If this field equals 0, it indicates the sender is a *leaf node*. If this field equals -1, it indicates the sender is a *volunteer node*.
- **ParentID**: The ID of the sender’s parent in the current tree.

| SrcID | SelfID | Cost | NumChildren | ParentID |
|-------|--------|------|-------------|----------|
|-------|--------|------|-------------|----------|

Figure 1. The format of the Hello packet

3.3 Updating fields in Hello packets

Upon receiving a Hello packet from a neighbor, a node executes the following algorithm to update the Cost, NumChildren, and ParentID fields in its own Hello packet.

Cost: The source node always sets its Cost value to 0. A non-source node initially set its Cost to 0xFFFFFFFF, a very large Cost to mimic an infinite value; after receiving a Hello packet, it sets its Cost to $\min\text{Cost} + 1$, where $\min\text{Cost}$ is the smallest Cost value in the Hello packets it has seen.

NumChildren: A node v determines this value by counting how many neighbors send Hello packets which carry the ID of node v in their ParentID fields. The children of a node are maintained in a list, and a child will be removed from this list if no Hello packet from this child is seen for a timeout period. If node v hears no children, it knows that it will be a *volunteer node* or a *leaf node*. For the algorithm deciding whether to be a volunteer node, please refer to the next subsection.

ParentID: A node selects its parent by checking which neighbor has the least Cost value contained in its Hello packets. If several neighbors’ Hello packets contain the same least Cost value, the tie is broken by first favoring a larger NumChildren value and then by a larger SelfID. If no Hello packet from the current parent is received for a timeout period, the current parent will expire, and a new parent will be selected from the current neighbors.

3.4 Deciding whether to be a forwarding node

In TreeDCB, each node maintains a Boolean variable named “b_forward” to indicate whether this node should serve as a forwarding node upon the reception of broadcast packets. This variable is calculated not at the time of receiving every broadcast packet, but at the time when the local information is changed (e.g., a new neighbor is discovered, the current parent is timed out, etc.), thus saving the computation overhead at a node. The algorithm run at a node v for calculating b_forward is given below, where the fields in node v ’s Hello packets is preceded by “v.”.

```

// If I am a parent node, then serve as a forwarding node.
if (v.NumChildren > 0) {
    b_forward = true;
    return;
}

// Below, decide whether to be a volunteer node.
n_parent = 0; // number of parent nodes being heard
max_id = -∞; // maximum ID of volunteers being heard
for each v’s neighbor w with w.Cost ≤ v.Cost {
    if (w.NumChildren > 0)
        n_parent = n_parent + 1;
    if (w.NumChildren == -1 and w.SelfID > max_id)
        max_id = w.SelfID;
}
if (n_parent ≥ 2 or max_id > v.SelfID) {
    b_forward = false;
    v.NumChildren = 0; // deciding v is a leaf node
}
else {
    b_forward = true;
    v.NumChildren = -1; // deciding v is a volunteer
}

```

Figure 2. The algorithm for calculating b_forward

This algorithm basically does the follows. If node v is a parent node, it serves as a forwarding node. When v is not a parent node, v decides whether to be a volunteer node by examining its neighbors with Costs less than or equal to v ’s. If there are more than one parent nodes among these neighbors, v knows that it is covered at least twice and will not volunteer. Otherwise, if there is another volunteer neighbor with its ID larger than v ’s, v will not volunteer either. Node v only volunteers when it hears no volunteer node with ID larger than v ’s, thus avoiding redundant volunteer nodes.

Note that for both algorithms of updating fields in Hello packets and calculating b_forward, the most time-consuming computation is one search of the neighbor list, so the computation complexity of TreeDCB at a node is $O(d)$, where d is the maximal node degree in the network. For the details of these two algorithms, please see our source codes, which are made available to the public at [19].

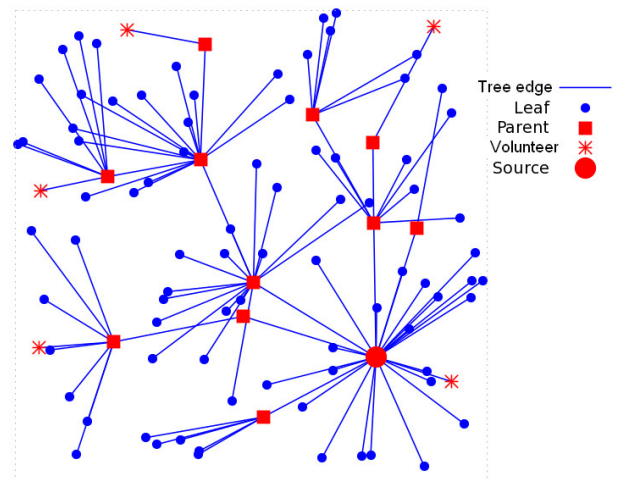


Figure 3. Parent, volunteer and leaf nodes obtained by TreeDCB in a 100-node network

To help understanding TreeDCB, the parent, volunteer, and leaf nodes obtained by TreeDCB in a 100-node network is shown in Figure 3, where the nodes are placed in a square area. From this figure, we see that TreeDCB only uses 12 parent nodes and 5 volunteer nodes to do the forwarding, and these 17 nodes cover each of the other 83 nodes at least twice.

3.5 Some notes on TreeDCB

Firstly, TreeDCB is based on a single source. This gives TreeDCB the advantages of low control overhead, but also brings the disadvantage that the established tree is only optimized for this single source in terms of path length. When additional sources exist, the parent and volunteer nodes computed according to the given source can still be used to forward packets for other sources, but some nodes may have long paths from certain sources. If minimizing the path length is a primary goal of applications, running a TreeDCB for each source is an alternative. Thus, TreeDCB is best suited to scenarios where a small number of sources exist, which is the case for a majority of broadcast applications.

Secondly, in building the tree, the time when a node finds out its Cost (number of hops) is affected by the Cost itself and the period of sending Hello packets. In our implementation, we apply the optimization technique that when a node sees a smaller Cost, it sends its Hello immediately instead of waiting until the next Hello period, thus the tree can be established much faster. Therefore, the convergence will not be a problem for TreeDCB.

Finally, node mobility does not have a strong impact on the stability of TreeDCB. Since TreeDCB is targeted at the crowd scenario where node mobility is in the walking speed level (about 1.4m/s) [23], a modestly small Hello period is enough to prevent instability. As to be shown in our simulations, with the Hello period set to 0.5 second, the performance of TreeDCB at the walking scenario is almost the same as the static scenario.

4. EVALUATION

We implemented TreeDCB in the latest version of *ns-2* (v2.34), which includes the new patches to the IEEE 802.11 MAC layer [14]. For comparison purpose, we also implemented DCB [13] in *ns-2*. Since TreeDCB does not have the reliable transmission part, we omit the reliable transmission part of DCB in our implementation. The source codes of our implementation are made available to the public at [19].

We evaluate TreeDCB and DCB on the following four metrics: *the ratio of forwarding nodes*, *the amount of control traffic*, *packet delivery ratio* and *the average packet path length*. Finally, particular to TreeDCB, we evaluate *the ratio of volunteer nodes* to show how many volunteer nodes are added to achieve double coverage for non-forwarding nodes.

4.1 Experiment setup

In our simulations, we use the IEEE 802.11 as the MAC layer protocol and the Two Ray Ground model as the physical layer propagation model. To simulate node mobility, we use the random waypoint (RWP) model [1]. In the RWP scenario generator “setdest” shipped with *ns-2*, a node selects a speed uniformly distributed between 0 and a maximal speed, and we change this maximal speed to adjust node mobility level. In all our experiments, we use a single source node to broadcast CBR traffic to all other nodes in the network.

We place nodes in a square area with a uniform distribution. The transmission range of all nodes is set to be 250m. With

transmission range fixed, to get different node densities (or different levels of node connectivity), we change the side length of the square area when the number of nodes is also fixed. For instance, with 100 nodes, to get a node density of 100 nodes/km², we set the side length of the square area to 1km; and to get a node density of 200 nodes/km², we set the side length to 0.707km.

To reflect the node connectivity at different node densities, we measured the average node degree of the networks formed by 100 nodes at the transmission range of 250m (see TABLE I). Later, we will use these node densities to conduct experiments.

TABLE I. Node Densities (per square km) and their corresponding Average Node Degrees

| Node density | Average node degree |
|--------------|---------------------|
| 50 | 8.49 |
| 100 | 15.6 |
| 150 | 22.06 |
| 200 | 28.55 |
| 250 | 33.73 |
| 300 | 38.41 |

The default values for our experimental parameters are summarized in TABLE II. In the evaluations presented below, if we do not mention the value of certain parameter explicitly, we use the value given in this table. Note that the 2.8m/s speed is used to simulate the crowd scenario, since the typical human walking speed is 5km/h (about 1.4m/s) [23]; and the 2 Mbps is chosen as the broadcast data rate, since the broadcast transmission must be conservative and use a rate in the basic rate set supported by all nodes in a BSS [7].

TABLE II. Experiment Parameters and Their Default Values

| | |
|---------------------------------|---------------------------|
| Number of node in the network | 100 |
| Transmission range | 250m |
| Maximal node speed in RWP model | 2.8m/s |
| Node Density | 100 nodes/km ² |
| Bandwidth | 2Mbps |
| CBR Source Packet Rate | 40 pkts/s |
| Data Packet Length | 200 bytes |
| Period of sending Hello packet | 0.5s |
| Simulation Time | 150s |
| Confidence Interval | 95% |
| Number of Trials | 50 |

In our experiments, we consider the impact of the following parameters: the mobility level, the node density, the number of nodes in the network, and the source packet rate. To evaluate the impact of certain parameter, we vary the value of this parameter and use the default values for all other parameters. For each set of parameter values, we conduct 50 experiments. In the figures presented below, we plot the 95% confidence interval in addition to the mean value of these 50 experiments. In all these figures, the difference between the confidence limit and the mean value is within 5% of the mean value, so we are confident with our experimental results.

4.2 Ratio of forwarding nodes

For a broadcast protocol, the ratio of forwarding nodes is defined as the number of forwarding nodes divided by the total number of nodes in the network. In this subsection, we present the experimental results on this ratio by varying the node density, mobility level, and node number.

Figure 4 gives the ratios of forwarding nodes at the node densities of 50, 100, 150, 200, 250, and 300 nodes/km². This figure shows that (1) for all node densities plotted, TreeDCB uses significantly less forwarding nodes than DCB; (2) for both TreeDCB and DCB, the ratios decrease with the growth of node density, since a smaller number of nodes are needed to do the forwarding when the node connectivity increases.

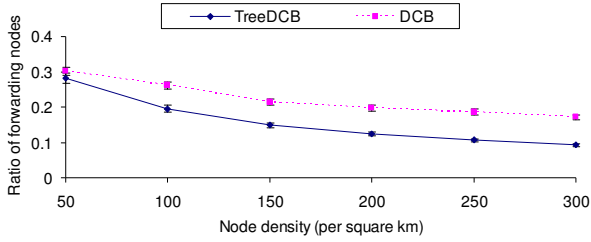


Figure 4. Ratio of forwarding nodes at different node densities

Figure 5 gives the ratios of forwarding nodes with different node numbers in the network at a fixed node density of 100 nodes/km². This figure mainly shows that (1) for all node numbers plotted, the ratio of TreeDCB is about 20% less than that of DCB; (2) for both TreeDCB and DCB, the ratios decrease slightly with increasing node numbers.

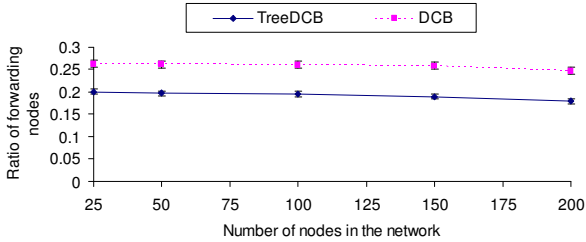


Figure 5. Ratio of forwarding nodes at different node numbers

Figure 6 gives the ratios of forwarding nodes when the nodes are static, and move at the maximal speeds of 2.8m/s, 20m/s, 40m/s, 80m/s, and 160m/s respectively. It can be observed that the ratios of both TreeDCB and DCB do not change significantly with the increase of mobility. This is because the nodes are randomly placed in a square area initially and the RWP model also moves the node randomly and independently. Thus, at any time during the node movement, the nodes remain randomly distributed in this square area.

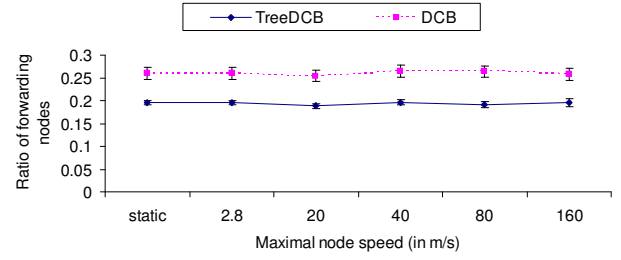


Figure 6. Ratio of forwarding nodes at different mobility levels

4.3 Amount of control traffic

The control traffic of TreeDCB only consists of the periodical Hello packets transmitted by every node. The control traffic of DCB includes the periodical Hello packets as well as the forwarding node list appended to every data packets. In our experiments, we set the Hello periods of both TreeDCB and DCB to be identical. For a Hello packet in both TreeDCB and DCB, the physical-layer and MAC-layer headers consist of the following: preamble (18 bytes), PLCP header (6 bytes), IEEE 802.11 header (30 bytes), and Frame Check Sequence (4 bytes), so the total header length is 58 bytes. For TreeDCB, the Hello payload has a fixed length of 20 bytes, so the total length of a Hello packet is 78 bytes. For DCB, since each node attaches its 1-hop neighbor list to its Hello packets, the payload has a length of $4n$, where n is the number of neighbors and each neighbor costs 4 bytes. Thus, a Hello packet in DCB has a variable length of $58+4n$ bytes.

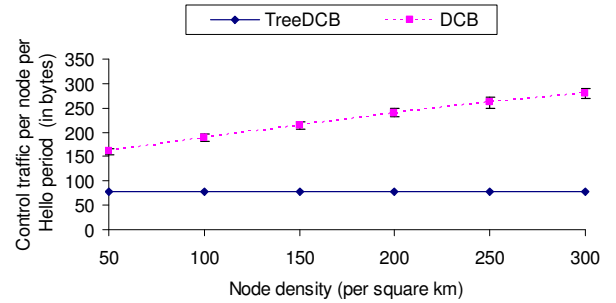


Figure 7. Amount of control traffic (in bytes) per node per Hello period at different node densities

Figure 7 gives the amounts of control traffic (in bytes) per node per Hello period at different node densities. This figure shows that (1) the amount of control traffic of TreeDCB does not change with the node density, since the Hello packets in TreeDCB have a fixed length; (2) the amount of control traffic of DCB increases quickly with the node density, because the average node degree increases quickly according to TABLE I and hence the lengths of both 1-hop neighbor list and forwarding node list increase.

4.4 Packet delivery ratio

The packet delivery ratio is defined as the ratio of the total number of packets received by all nodes versus the total number of packets to be received by all nodes. In this subsection, we present the experimental results on this ratio by varying the mobility level, node density, node number, and the source packet rate.

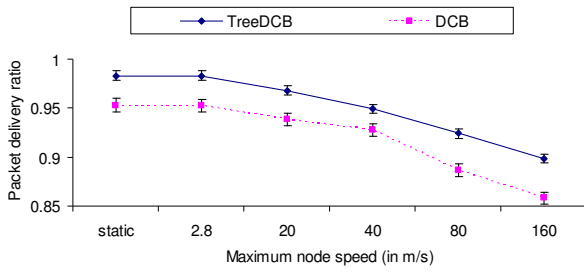


Figure 8. Packet delivery ratio at different mobility levels

Figure 8 plots the delivery ratios when the nodes are static and move at different maximum speeds, showing the follows. Firstly, at all mobility levels, the delivery ratio of TreeDCB is greater than that of DCB. This is because TreeDCB have much less control traffic than DCB. Secondly, the delivery ratios of both protocols at the low mobility level (2.8m/s) are approximately the same as those in the static scenario, and they only degrade significantly when the node speed is very large ($> 40\text{m/s}$). This reflects that the node mobility does not have a very strong impact to the broadcast packet delivery for both TreeDCB and DCB.

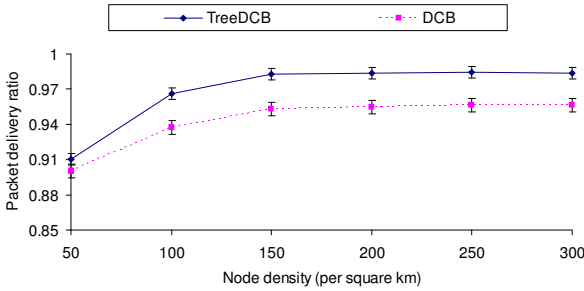


Figure 9. Packet delivery ratio at different node densities

Figure 9 plots the packet delivery ratios at different node densities, showing the follows. Firstly, TreeDCB remains performing better than DCB in all experimented node densities. Secondly, the delivery ratios of both protocols increase with the node density. This is because both protocols are effective in reducing the number of forwarding nodes (as shown in subsection 4.2) and there is not much broadcast redundancy in the network. Thus, when the node density grows, the broadcast redundancy also increases, which improves the delivery ratio.

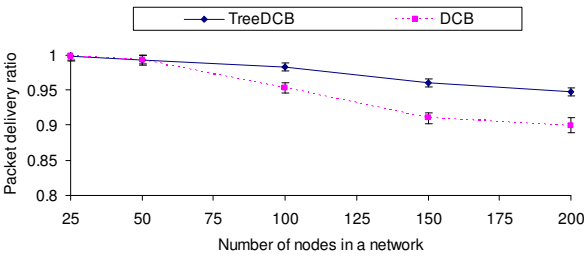


Figure 10. Packet delivery ratio with different node numbers

Figure 10 plots the packet delivery ratios with different number of nodes in the network at a fixed node density of $100\text{nodes}/\text{km}^2$. This figure shows that (1) for both protocols, the delivery ratios decrease with the growth of node number, since packets will

experience longer paths to reach the destination; (2) the delivery ratio of TreeDCB is greater than that of DCB.

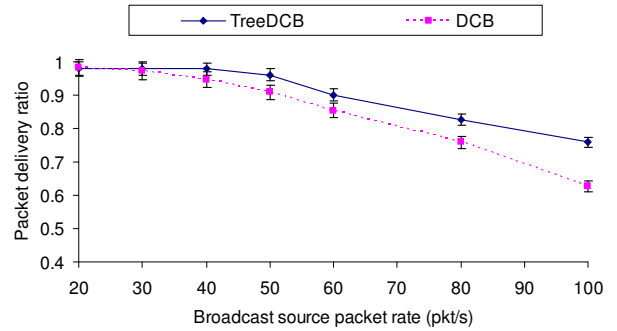


Figure 11. Packet delivery ratio at different src packet rates

Figure 11 plots the packet delivery ratios at different source packet rates. It can be observed that the delivery ratio of DCB drops more quickly than that of TreeDCB, because DCB has more control traffic than TreeDCB, thus incurring more collision and congestion.

4.5 Average path length of a packet

Since the path length of a packet is closely related to the end-to-end delay and the loss probability of this packet, we evaluate on this metric by measuring the average length of all the paths that a broadcast packet traverses to reach all the nodes in a network. In each of our experiments, we calculate the average value of all packets generated. In this subsection, we present results on this metric by varying the node density and the node number.

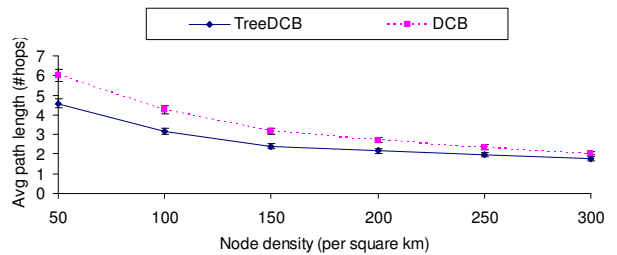


Figure 12. Average path length at different node densities

Figure 12 gives the average path lengths at different node densities for both TreeDCB and DCB. This figure mainly shows that a packet in TreeDCB generally experience a shorter path than a packet in DCB, since a packet in TreeDCB is transmitted along the shortest path tree, which is optimal in producing shortest paths.

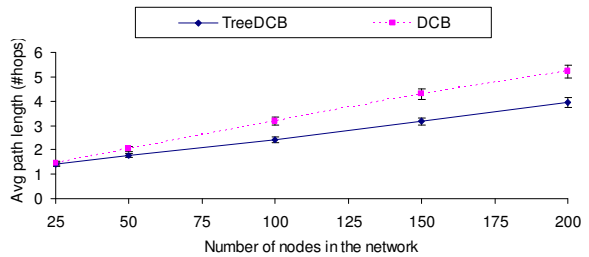


Figure 13. Average path length with different node numbers

Figure 13 gives the average path lengths with different number of nodes in the network at a fixed node density of 100 nodes/km². This figure mainly shows that for both protocols, the average path lengths increase with the node number, since more nodes in the network give rise to longer paths.

4.6 Ratio of volunteer nodes

In TreeDCB, when the parent nodes in established tree topology is not enough to ensure the double coverage of non-forwarding nodes, volunteer nodes are added to achieve this purpose. To reflect the importance of volunteer nodes in conducting forwarding, we define the ratio of volunteer nodes as the number of volunteer nodes divided by the total number of forwarding nodes in a network. In this subsection, we present the results on this ratio by varying the node density and the node number.

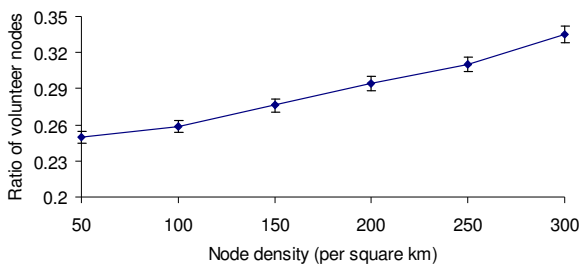


Figure 14. Ratio of volunteer nodes at different node densities

Figure 14 gives the ratios of volunteer nodes at the node densities of 50 - 300 nodes/km² respectively. It can be observed that the ratio increases with the node density, because when the node density increases, the number of parent nodes in the tree is greatly reduced and the number of volunteer nodes becomes more significant.

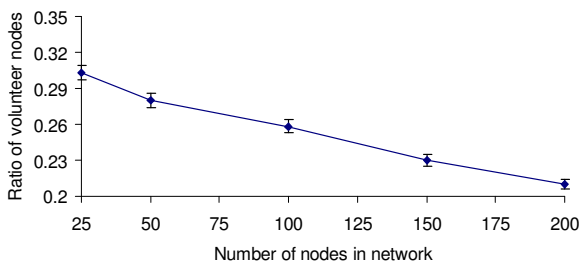


Figure 15. Ratio of volunteer nodes at different number of nodes

Figure 15 gives the ratios of volunteer nodes with different number of nodes in the network, showing that the ratio decreases with the node number. This is because the nodes near the border of the deployed square area have a higher demand for volunteer nodes to cover them doubly than the nodes in the middle. The previous 100-node network example given in Figure 3 actually illustrates this phenomenon. With the node number increasing, the proportion of the nodes near the border drops, so the ratio of volunteer nodes decreases.

5. CONCLUSIONS

Observing that a high-density wireless network desires a broadcast protocol with fixed-length Hello packets, we proposed TreeDCB, a tree-based double-covered broadcast protocol for wireless ad hoc networks such as a crowd of people with smartphones. TreeDCB uses fixed-length Hello packets and guarantees that each node v is either a forwarding node or covered by at least two forwarding nodes (not including the children nodes of node v). Using the basic technique of shortest path tree building, TreeDCB introduces two new techniques to improve the selection of forwarding nodes: (1) it selects parent nodes in the tree by examining which one has the greatest number of children, thus significantly reducing the number of parent nodes and (2) a leaf node will volunteer to be a forwarding node if it hears no forwarding nodes other than its parent, thus ensuring double coverage for non-forwarding nodes.

By *ns-2* simulation, we compare TreeDCB with DCB, a well-known double coverage protocol using 2-hop information. The experimental results show that TreeDCB outperforms DCB mainly due to its low control overhead in our experiment settings with node densities not low.

6. ACKNOWLEDGEMENT

This research work has been supported by funding from National ICT Australia (NICTA).

7. REFERENCES

- [1] F. Bai and A. Helmy, *Chapter 1: A Survey Of Mobility Models in Wireless Adhoc Networks*, Book: *Wireless ad hoc networks*: Kluwer Academic, 2006.
- [2] J. Cao and C. Williamson, "Towards Stadium-Scale Wireless Media Streaming", in *IEEE Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 33-42, 2006.
- [3] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)", *RFC 3626, Experimental*, 2003.
- [4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, 3rd ed. Cambridge, Boston, MA: MIT Press, 2009.
- [5] F. Dai and J. Wu, "Performance Analysis of Broadcast Protocols in Ad Hoc Networks Based on Self-Pruning", *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, vol. 15, 2004.
- [6] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets", *Algorithmica*, vol. 20, pp. 374-387, 1998.
- [7] IEEE, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", *IEEE Std 802.11-2007*, 2007.
- [8] J. Jetcheva, Y. Hu, D. Maltz, and D. Johnson, "A simple protocol for multicast and broadcast in mobile ad hoc networks", IETF, Internet Draft: draft-ietf-manet-simplembcast-01.txt, 2001.
- [9] M.-F. Leung and S.-H. G. Chan, "Broadcast-Based Peer-to-Peer Collaborative Video Streaming Among Mobiles", *IEEE Transactions on Broadcasting*, vol. 53, pp. 350 - 361, 2007.
- [10] H. Lim and C. Kim, "Flooding in wireless ad hoc networks", *Computer Communications, Elsevier*, pp. 353-363, 2001.
- [11] H. Liu, X. Jia, P.-J. Wan, X. Liu, and F. F. Yao, "A Distributed and Efficient Flooding Scheme Using 1-Hop Information in Mobile Ad Hoc Networks", *IEEE*

- Transactions on Parallel and Distributed Systems*, vol. 18, pp. 658-671, 2007.
- [12] W. Lou and J. Wu, "On reducing broadcast redundancy in ad hoc wireless networks", *IEEE Transactions on Mobile Computing*, vol. 1, pp. 111-123, 2002.
- [13] W. Lou and J. Wu, "Toward Broadcast Reliability in Mobile Ad Hoc Networks with Double Coverage", *IEEE TRANSACTIONS ON MOBILE COMPUTING*, vol. 6, 2007.
- [14] NS-2, *Network Simulator 2, Ver 2.34*, <http://www.isi.edu/nsnam/ns>, 2009.
- [15] W. Peng and X. Lu, "On the Reduction of Broadcast Redundancy in Mobile Ad Hoc Networks", in *ACM MobiHoc*, pp. 129-139, 2000.
- [16] W. Peng and X. Lu, "AHBP: An Efficient Broadcast Protocol for Mobile Ad Hoc Networks", *Journal of science and technology*, pp. 129-139, 2002.
- [17] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying for Flooding Broadcast Messages in Mobile Wireless Networks", in *35th Hawaii International Conf. System Sciences*, 2002.
- [18] M. Q. Rieck, S. Pai, and S. Dhar, "Distributed Routing Algorithms for Wireless Ad Hoc Networks Using d-Hop Connected Dominating Sets", in *Sixth Int'l Conf. High Performance Computing in Asia Pacific Region*, 2002.
- [19] W. Si, *Implementation of TreeDCB in ns-2*, Available: <http://sourceforge.net/projects/tdcb>, 2011.
- [20] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks", *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, pp. 14-25, 2002.
- [21] J. Sucec and I. Marsic, "An Efficient Distributed Network-Wide Broadcast Algorithm for Mobile Ad Hoc Networks", CAIP Technical Report 248, Rutgers University., 2000.
- [22] A. Tanenbaum, *Computer Networks (4th Edition)*: Prentice Hall, 2002.
- [23] TranSafety_Inc., "Study Compares Older and Younger Pedestrian Walking Speeds", *Road Engineering Journal, USA*, 1997.
- [24] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network", *Wireless Networks*, vol. 8, pp. 153-167, 2002.
- [25] M. G. Wing, A. Eklund, and L. D. Kellogg, "Consumer-Grade Global Positioning System (GPS) Accuracy and Reliability", *Journal of Forestry*, pp. 169-173, 2005.
- [26] J. Wu and F. Dai, "Broadcasting in Ad Hoc Networks Based on Self-Pruning", in *IEEE INFOCOM*, pp. 2240-2250, 2003.