MULTIPLE TREE VIDEO MULTICAST OVER WIRELESS AD HOC NETWORKS[1]

Avideh Zakhor and Wei Wei

Department of Electrical Engineering and Computer Sciences
University of California at Berkeley, CA 94720, USA

ABSTRACT

In this paper, we propose a multiple tree multicast streaming scheme for video applications over wireless ad hoc networks. Specifically, we propose a multiple tree construction protocol, which builds two nearly disjoint trees simultaneously in a distributed way. Simulation shows that video quality of our proposed scheme to be superior to that of single tree multicast, even though they have similar control overhead and forwarding efficiency.

1. INTRODUCTION

Multicast is an essential technology for many applications, such as group video conferencing, and results in bandwidth savings as compared to multiple unicast sessions. Due to the inherent broadcast nature of wireless networks, multicast over wireless ad hoc networks can be potentially more efficient than over wired networks. However there are many challenges for supporting video multicast over wireless ad hoc networks. Mobility of nodes, time-varying nature of the wireless channel, and congestion all make video multicast unreliable.

Path diversity has been shown to improve video streaming quality for wired and wireless networks in both unicast and multicast scenarios [1-4]. Most recently, we introduced a serial scheme for constructing multiple multicast trees in wireless ad hoc networks, and showed its superior video streaming performance over single tree multicast[3]. In this paper, we extend this work to a parallel multiple tree construction scheme in order to reduce the time and control overhead it takes to construct the trees. In this scheme, trees are constructed in parallel and in a distributed fashion. We characterize the performance of our proposed scheme in terms of video quality, routing overhead, and construction delay.

This paper is organized as follows. In Section 2, we review the framework of multiple tree multicast streaming. We present the proposed parallel multicast routing protocol in Section 3. We verify performance of the proposed protocol in Section 4.

[1]This work was supported by AFOSR contract F49620-00-1-0327.

2. MULTIPLE TREE VIDEO MULTICAST FRAMEWORK

A multiple tree video multicast system consists of two parts: a multicast routing protocol to construct multiple trees, and a scheme to distribute video packets into different trees[2-4]. It is possible to employ Multiple Description Coded (MDC) video to form multiple video streams, and transmit different video streams through different trees[2-4]. Similar to other path diversity schemes, the basic idea is that in the event packets in one tree are not received on time or are lost, the overall video quality at the receiver degrades in a graceful manner.

Multicasting MDC video was first discussed in CoopNet [2] in the context of peer-to-peer networks to prevent web servers from being overwhelmed by large amount of requests. In [4], the authors propose a genetic algorithm based solution for multiple tree multicast streaming. In addition, there has been a large body of work in the area of multicast routing in wireless ad hoc networks [5-8]. The Independent-Tree Ad Hoc Multicast Routing (ITAMAR) creates multiple edge disjoint or nearly disjoint multicast trees in a centralized way[7].

In this paper, we propose a parallel multiple tree scheme which constructs the trees in a distributed fashion. There are two advantages to our technique as compared to ITAMAR[7]: first, it is distributed, and not centralized, and hence does not need to know network topology in advance; second, our overhead is $O(n)$, rather than $O(n^2)$, where n is the number of nodes.

During multicast operation, the application layer protocol sets a tree-flag in each packet's header to determine to which tree the packet should be forwarded. The multiple tree multicast protocol forwards the packet in different trees according to the tree-flag. We also apply the *tree flood* approach [5] to achieve extra diversity gain without consuming extra network resources.

3. PARALLEL MULTIPLE NEARLY-DISJOINT TREES MULTICAST (PARALLEL MNTMR)

We have three main design goals in mind for the Parallel MNTMR:

• *Low routing overhead and construction delay*: The routing overhead and construction delay of Parallel MNTMR should be similar to that of a typical single tree multicast protocol.

• *High tree connectivity*: if a receiver is connected to the sender, it should be able to connect to both trees.

• *Near disjointness*: The ratio of the number of shared nodes of two trees to the number of nodes of the smaller tree should be minimized.

3.1. Overview

In a general single-tree multicast protocol, e.g. ODMRP [6], when a multicast source has packets to send, it triggers a multicast tree construction process by flooding a Join-Query (JQ) message to the network. Upon receiving the JQ message, each receiver unicasts back a Join-Reply (JR) message to the sender to construct the multicast tree. In Parallel MNTMR, we apply similar JQ and JR processes to construct two nearly-disjoint trees simultaneously.

The basic idea behind parallel tree construction is to first classify all the nodes randomly into one of two categories: group 0 or group 1. We define a *pure JQ message* as a JQ message whose route only consists of nodes in the same group, and a *mixed JQ message* as a JQ message whose route consists of nodes in both groups. The protocol uses a technique based on delay timer to select and forward *pure JQ messages* with a priority over *mixed JQ messages*. As will be seen shortly, this improves the disjointness of the constructed trees in the JR process.

We also propose an *upstream node selection rule* so that nodes close to each other tend to select the same upstream node for the same tree, thereby avoiding nodes of the other tree. This rule improves the disjointness of two trees, and forwarding efficiency of the multicast protocol.

3.2. Conditions and Rules

In order to construct two trees with both high tree connectivity and low tree similarity, Parallel MNTMR applies the following conditions and rules at each node to control the flow of JQ and JR messages. Without loss of generality, we assume the current node a is in group x, where x is 0 or 1. For brevity, we call a JOIN-QUERY message with a group-x node as the last hop, a group-x JQ message.

• *JQ message storing condition*: In order to obtain two loop-free trees in the JR process, each node only stores JQ messages satisfying the *storing condition* into its *JQ Message Cache*. A JQ message satisfies the *storing condition*, either if it is the first received JQ message, or if the following two conditions are satisfied: (a) the number of hops it travelled is no larger than that of the first received JQ message at node a plus one, and (b) the JQ message has not been forwarded by node a.

• *JQ message forwarding condition*: A JQ message satisfies the *forwarding condition*, if the following two conditions hold true: (a) node a has not forwarded a JQ message in this JOIN-QUERY round, and (b) the message's last hop is the sender or a group-x node. The *forwarding condition* results in *pure group-x JQ messages* to be selected and forwarded with a priority over *mixed JQ messages*, thus helping the protocol to construct trees that are as disjoint as possible.

• *Upstream node selection rule*: The objective of the *upstream node selection rule* is to maximize the disjointness of two trees. Let JQM_a denote the set of all the messages in the *JQ Message Cache* of node a. If there exist both group-0 and group-1 JQ messages in JQM_a, node a selects last hops of the earliest received group-0 and group-1 JQ messages as upstream nodes for tree-0 and tree-1 respectively. Otherwise, we assume all the JQ messages in JQM_a are group-y JQ messages. In this case, if $|JQM_a| > 1$, node a selects last hops of the earliest and the second earliest received JQ messages as upstream nodes for tree-y and tree-$(1-y)$ respectively; otherwise if there is only one message in JQM_a, the last hop of the only JQ message is selected as upstream nodes for both tree-0 and tree-1.

3.3. Detailed Double Nearly-Disjoint Tree Construction

When a multicast source has packets to send, it triggers a multicast tree construction process by broadcasting a Join-Query (JQ) message to its neighbors. When a node receives a group-y JQ message, if the message satisfies the *storing condition*, the node stores it into the *JQ Message Cache* for later usage in the JR process, otherwise the message is simply discarded. If the message also satisfies the *forwarding condition*, the current node forwards the JQ message to its neighbors immediately; otherwise if the JQ message is the earliest received JQ message in the current Join Query round, the node sets a JQ-delay timer. When the JQ-delay timer expires, if the node has not forwarded a JQ message in this JQ round, it forwards the earliest received JQ message at that time. The JQ-delay scheme encourages *pure JQ messages* to be selected and forwarded with a priority over *mixed JQ messages* in the distributed tree construction process.

When a receiver receives a group-y JQ Message, if the message is a *pure JQ message*, and the node has not initiated a JOIN-REPLY (JR) message in this JQ round for tree-y, it selects the last hop of this JQ message as its upstream node for tree-y, and unicasts a JR message to the sender via the selected upstream node. All nodes, receiving and forwarding the JR message for tree-y, become middle nodes of tree-y. The receiver also sets a timer upon receiving the earliest JQ message. When the timer expires, for each tree for which it has not already initiated a JR message, the receiver selects an upstream node according to the *upstream node selection rule* and unicasts a JR message to the sender via the selected upstream node to construct that tree. In the end, we obtain one tree mainly consisting of group-0 nodes and another mainly consisting of group-1 nodes.

3.4. Discussion

In this section, we argue that Parallel MNTMR achieves our three design goals. Firstly, the Parallel MNTMR builds two trees simultaneously, and each node forwards the JQ mes-

sage at most once in one JOIN-QUERY round. Therefore the routing overhead and the construction delay is similar to that of a typical single tree multicast routing protocol. Secondly, as long as a receiver is connected to the sender, the protocol requires it to send JR messages for both trees; therefore the tree connectivity is the same as that of a single tree protocol. Thirdly, regarding the disjointness of the two trees constructed by MNTMR, we have the following claim:

Claim 1: Given any two nodes N_a and N_b, which are middle nodes for tree-0 and tree-1 respectively, let JQ_a and JQ_b denote node sets of last hops of JQ messages stored in the JQ Message Caches of nodes N_a and N_b respectively. Let nodes N_c and N_d denote upstream nodes obtained by the Parallel MNTMR of nodes N_a and N_b respectively. We have $N_c \neq N_d$, if $|JQ_a \cap JQ_b| \geq 2$ or $|JQ_a \cap JQ_b| = 0$.

The proof is included in [9]. Intuitively, *claim 1* shows that as long as two nodes in different trees do not share exactly one JQ message in their JQ message caches, they will not select the same node as their upstream nodes. Thus in most scenarios, the Parallel MNTMR maintains disjointness between two trees.

4. SIMULATION RESULTS

We use a simulation model based on NS-2[10]. We only consider the continuous mobility case with zero pause time, and vary the maximum speed from 2.5 m/s to 15 m/s. In each run, we simulate a 50 node wireless ad hoc network within a 1500 $\times$ 300 square meters area. Each simulation is 900 seconds long, and results are averaged over 30 runs.

We randomly choose one sender and eight receivers. For MDC we encode one frame into two packets, while for Single Description Coding (SDC) we encode one frame into one packet. We set the frame rate as 8 fps, and GOP size as 15. For fairness, we set the Peak Signal to Noise Ratio (PSNR) of MDC and SDC to be approximately the same, i.e. 33 dB. To achieve similar quality, standard MPEG QCIF sequence Foreman is coded with a Matching Prusuit Multiple Description Video Codec called MP-MDVC [11] at 64.9 kbps for MDC, and with Matching Pursuit Codec [12] at 41.2 kbps for SDC sequence. The playback deadline of each packet is set to 150 milliseconds (ms) after it is generated.

We evaluate the performance using the following metrics:

a. **The ratio of bad frames:** In multicast scenario, the ratio of bad frames is the ratio of the total number of non-decodable frames to the total number of frames that should have been decoded in all the receivers. A description of an I-frame is non-decodable, if the packet corresponding to the description is not received on time. A description of a P-frame is non-decodable, if at the playback deadline, either the packet corresponding to the description is not received or the same description of the previous frame is non-decodable. A frame of a MDC stream is non-decodable, if both of its two descriptions are non-decodable. This metric takes into account the dependency between consecutive frames in a predictive coding scheme, and also reflects the fact that MDC can, to some extent, conceal the undesirable effects caused by missing packets.

b. **The number of bad periods:** A bad period consists of contiguous bad frames. This metric reflects the number of times that received video is interrupted by the bad frames.

c. **Normalized packet overhead:** The total number of control packets transmitted by any node in the network, divided by the total number of video frames received by all the receivers.

d. **Forwarding efficiency:** The total number of data packets transmitted by any node in the network, divided by the total number of packets received by all the receivers.

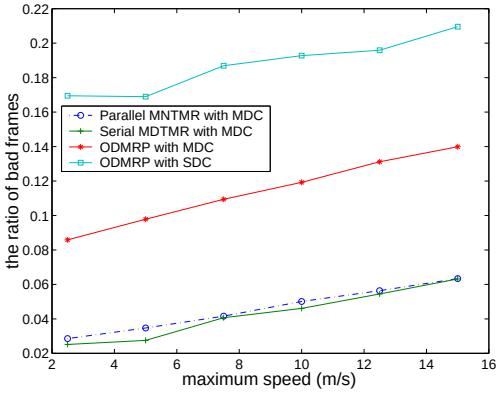
We compare the following four schemes:

- Multiple tree multicast with Parallel MNTMR and MDC;
- Multiple tree multicast with Serial MDTMR [3] and MDC;
- Single tree multicast with ODMRP [6] and MDC;
- Single tree multicast with ODMRP [6] and SDC.

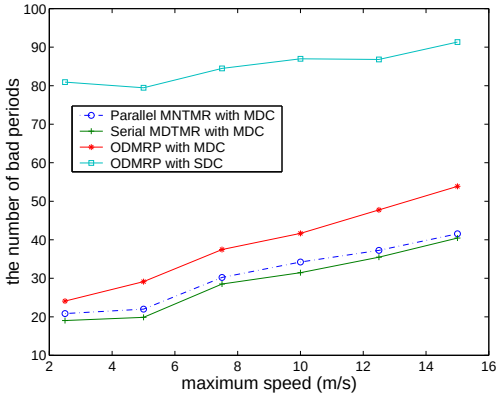
ODMRP [6] builds a single tree by by periodically flooding the network with control packets to create and maintain the forwarding state of each node. Our previously proposed Serial MDTMR [3] builds two disjoint multicast trees in two steps. First it constructs a shortest path multicast tree. Then it constructs the second shortest path tree, by requiring all the middle nodes in the first tree not to be middle nodes of the second tree.

Figures 1(a) and 1(b) show the ratio of bad frames and the number of bad periods of the four schemes respectively. As seen, performances of multiple tree multicast with Parallel MNTMR and Serial MDTMR are very close to each other, and are superior to that of the other two schemes with ODMRP. Over the 30 runs, on average 8% of the nodes are shared between the two trees constructed by Parallel MNTMR. This explains the reason that two multiple tree protocols perform similarly. The combination of our proposed multiple tree multicast protocols and MDC reduces contiguous packet loss caused by broken links of a multicast tree. This is because links of two nearly-disjoint trees fail nearly independent. Note that MDC by itself could also reduce scattered packet loss caused by wireless channel error, or packet collision, thus reducing both the ratio of bad frames and the number of bad periods. Since a packet representing an I-frame is much larger than a packet representing a P-frame, an I-frame is more likely to be dropped compared to a P-frame. Transmitting I-frames twice also benefits MDC, as compared to SDC.

Figure 2(a) shows the normalized control packets for the four schemes. Simulation results show that the number of normalized control packets of Parallel MNTMR is very similar to that of ODMRP, and is around 50 percent lower than that of Serial MDTMR. In order to construct double disjoint trees, Serial MDTMR has to broadcast Route Request mes-

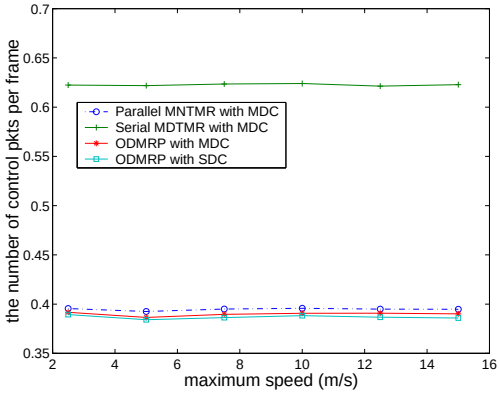


(a)

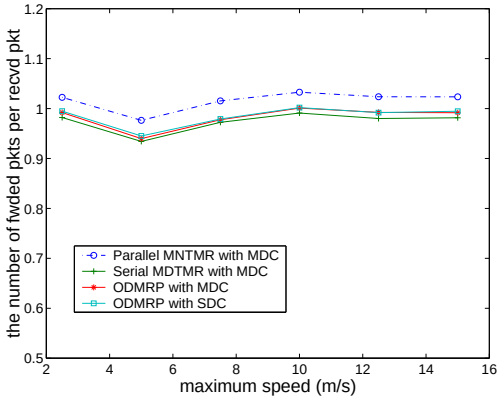


(b)

Fig. 1. Performance evaluation for multiple tree video multicast: (a) The ratio of bad frames; (b)The number of bad periods.



(a)



(b)

Fig. 2. Performance evaluation for multiple tree protocols: (a) The normalized control packets; (b) The normalized forwarded data packets.

sage twice in each routing cycle, while both Parallel MNTMR and ODMRP only broadcast once. Thus we see that Parallel MNTMR has approximately the same control overhead as a single tree multicast protocol.

Figure 2(b) shows that the number of the normalized forwarded data packets is almost the same for all four schemes with Parallel MNTMR being slightly worse. This indicates that the performance gains of Parallel MNTMR and Serial MDTMR are not achieved by forwarding a packet more times than ODMRP, rather by the combined effect of independent trees and MDC. We have also tested the above four schemes in other scenarios, and arrived at similar conclusions.

5. CONCLUSIONS

In this paper, we propose a multiple tree protocol, Parallel MNTMR, which builds two nearly disjoint trees simultaneously in a distributed way. Simulations show the effectiveness of our proposed multiple tree protocol.

6. REFERENCES

[1] S. Mao, S. Lin, S. Panwar, Y. Wang, and E. Celebi, "video transport over ad hoc networks: multistream coding with multipath transport", *IEEE Journal on Sel. Areas in Comm.*, Dec. 2003, pp. 1721 - 1737.

[2] V. Padmanabhan, H. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," *ACM NOSSDAV*, May 2002.

[3] W. Wei and A. Zakhor, "Multipath Unicast and Multicast Video Communication over Wireless Ad Hoc Networks", *Broadnets 2004*.

[4] S. Mao, X. Cheng, Y. Hou, and H. Sherali. "Multiple description video multicast in wireless ad hoc networks," *BroadNets 2004*.

[5] J. Jetcheva and D. Johnson. "Adaptive Demand-Driven Multicast Routing in Multi-hop Wireless Ad Hoc Networks", ACM Mobihoc 2001.

[6] S. Lee, M. Gerla, and C. Chiang. "On-Demand Multicast Routing Protocol", *IEEE WCNC'99*, Sep. 1999, pp. 1298-1302.

[7] S. Sajama, and Z. Haas. "Independent-tree ad hoc multicast routing (ITAMAR)", *ACM Mobile Networks and Applications*, Oct. 2003.

[8] C. Cordeiro, H. Gossain, and D. Agrawal. "Multicast over wireless mobile ad hoc networks: present and future directions", *IEEE Network*, Jan./Feb. 2003.

[9] W.Wei, and A. Zakhor. "Multiple Tree Video Multicast over Wireless Ad Hoc Networks". Technique Report. Available: www-video.eecs.berkeley.edu/∼weiwei/pub/Wei-TR-2005-1.pdf

[10] ns-2: network simulator. http://www.isi.edu/nsnam/ns/

[11] X. Tang, and A. Zakhor. "Matching pursuits multiple description coding for wireless video," *IEEE Trans. on CSVT*, June 2002, pp. 566 -575.

[12] R. Neff and A. Zakhor. "Very Low Bit-Rate Video Coding based on Matching Pursuits", *IEEE Trans. on CSVT*, Feb. 1997, pp. 158-171.