MANET Testbeds for Evaluation of Real-Time Controls in Multimedia Transmissions

Panida Veeravuttiphol
Data and Supplementary Network
Total Access Communication, PLC.
Bangkok, 10900, Thailand
panida.v@dtac.co.th

Patrachart Komolkiti
Department of Computer and Network Engineering
Assumption University
Bangkok, 10240, Thailand
patrachart@eng.au.edu

Chaodit Aswakul
Department of Electrical Engineering
Chulalongkorn University
Bangkok, 10330, Thailand
chaodit.a@chula.ac.th

***Abstract*—This paper proposes two real-time control mechanisms, Distributed Real-time Control with Regulating Buffer Threshold (DRC-RBT) and Adaptive Distributed Real-time Control (ADRC), for minimizing the packet jitter. To obtain the reliable results we also construct two platforms of testbeds, wireless and wired, to evaluate the proposed mechanisms. The implemented wireless testbed has been suitably applied in the case of small networks, composed of four mobile nodes, for manageability. In contrast, by using an emulator gateway to model ad-hoc environments, the wired testbed has been applied in case of larger networks possibly with consideration of node mobility. Experimental results reveal that each of the proposed methods performs well in different situations. In static network that packet delay is not significant, DRC-RBT is a preferable method; however, this method can serve only the incoming traffic of constant bit rate nature. On the other hand, by appropriately adjusting the assigned weight value of average delay in ADRC, the obtained numerical tests suggest that ADRC can operate efficiently in both static and dynamic network scenarios.**

Keywords—Jitter control; mobile ad hoc networks; multimedia transmissions; protocol evaluation; testbed

I. INTRODUCTION

In recent times, various technologies of wireless communications have been developed, including MANET (Mobile Ad-Hoc Networks) [1]. MANET is a distributed, mobile, wireless, multi-hop network that can operate without a priori given network infrastructure. Naturally, a node can only communicate directly with its neighbor nodes that are within its transmission range. If a node wants to send a packet to another node outside its transmission range, then that packet must be forwarded by intermediate nodes. As a result, each MANET node must be equipped with a distributed routing capability. Current MANET applications are mainly for emergency and military communications. However, MANET is expected to integrate with wireless sensor networks in a full-scale deployment, as a part of Intelligent Transportation Systems (ITS) in the future [2], [3]. As a result, there have been many research works on developing various MANET protocols in the past few years. Those protocols are often evaluated by computer simulation software [4]–[6]. Computer simulation is an excellent choice for an initial phase of realizing a protocol, thanks to its manageability and ability to achieve any level of emulation complexity as desired. However, despite abundance of wireless channel models, computer programs cannot be trusted to realistically simulate the uncertain dynamics of the physical layer. Neither can simulations catch detailed bugs that may be experienced in the subtle interactions of operating system software, system hardware, and wireless environments. For these reasons, testbed experiments become not only a necessity for a final-stage evaluation platform [7]–[10], but also an incentive of testing real-time control protocols in this paper. Two variations of MANET testbeds have been considered: wired and wireless [7]–[9]. While experiments involving large amount of nodes can be handled with ease in a wired testbed, owing to its controllability, scalability, and repeatability [11], a wireless testbed that uses true air interface is indispensable, since it can capture a real nature of MANETs that a wired testbed cannot. We choose to develop both of them in order to evaluate our proposed mechanisms in a variety of scenarios.

In ITS applications, MANETs are expected to carry real-time multimedia services such as on-demand audio/video retrieval and streaming. To enable effective communications for these services, quality of service (QoS) guarantees are often needed, especially packet delay and delay variance, or jitter. In the past, there were many researchers tackling these QoS problems but most of them focused on wired network environments [12], [13]. QoS controls are much more difficult to design in MANETs because of unreliable characteristics of wireless channels, random node-placements, and dynamic topology changes caused by user movements. These factors can greatly deteriorate the QoS, unless the network has been planned wisely. In this paper, we report our experiences in constructing both wired and wireless ad-hoc network testbeds for the investigation of multimedia transmissions. The focus is also on mean delay and jitter control mechanisms. There were studies of delay and jitter controls performed by the route selection process [7], but we believe that the problem of timing variation should be addressed by the timing regulations of forwarding processes. In this respect, two distributed real-time control mechanisms have been proposed and tested comparatively with a standard benchmark algorithm.

The rest of the paper is organized as follows. In Section II we give an overview of our testbeds. In Section III we introduce our proposed mechanisms. The experimental results of the mechanisms on the testbeds are discussed in Section IV. The paper is concluded in Section V.

This material is based upon research supported by the Cooperation Project between the Department of Electrical Engineering and Private Sector for Research and Development, Chulalongkorn University, Thailand.

II. TESTBED ARCHITECTURE

Two platforms of testbeds, wireless and wired, are considered in this work. A wired testbed that can emulate behaviors of an ad-hoc network is constructed for its controllability, scalability, and repeatability. On the other hand, we realize that transmission algorithms should be verified in the real wireless channel. Thus, a static-topology wireless ad-hoc network testbed is also assembled.

A. Wireless Testbed

The wireless testbed is composed of four laptops acted as mobile nodes. The laptops used in this testbed are IBM ThinkPad R40s, equipped with IntelPro 2100 wireless chipsets. All nodes in this testbed run Linux operating system, communicate to each other via UDP socket, and get connected wirelessly by radio based on the IEEE 802.11b standard in ad-hoc mode. Fig. 1 depicts the architecture of end-to-end process in the testbed. The sender program transmits traffic stream to the receiver program, where delay and jitter are measured. Intermediate programs are employed in intermediate nodes to perform our proposed mechanisms. The tested network consists of four nodes that are placed in four cars. These cars are driven in the form of *Car-Following* [14] along a street at urban speed to perform **dynamic node** experiments, and are parked to perform **static node** experiments, always forming a network topology that each node can only communicate with the spatially adjacent nodes, as depicted in Fig. 2. Node 1 and Node 4 act as the source and destination node, respectively.

B. Wired Testbed

Several characteristics of the wireless LAN channel are emulated in the wired testbed by a script called Emulator Gateway (EmuGw). Fig. 3 illustrates the architecture of the wired testbed. The figure shows that each process is connected via a socket defined by an IP address of that host and a port number. As a result, an ad-hoc network, with a large number of nodes, is effectively emulated in a single computer.

Since the purpose of EmuGw is to emulate the IEEE 802.11b wireless channel characteristics, all packets from all nodes are not directly transmitted to each other, but they must be preprocessed by EmuGw. When EmuGw receives a packet from any node, it will compute a status of the packet, including connectivity, link capacity and packet loss probability. Firstly, connectivity is calculated from a simulated distance between each node pair. Secondly, link capacity is mapped to the amount of transmission delay of each packet, using parameters from the real wireless transmission experiments. Lastly, packet loss probability is set as a constant during each experiment. Fig. 4 depicts functional processes of the EmuGw.

III. DISTRIBUTED REAL-TIME CONTROL MECHANISMS

In this work, two real-time control mechanisms are proposed and compared with the uncontrolled transmissions, as a benchmark. Both mechanisms are applied to each intermediate node for minimizing end-to-end jitter. One of them is only applicable in the case of CBR traffic, while the other one can operate with VBR traffic as well.

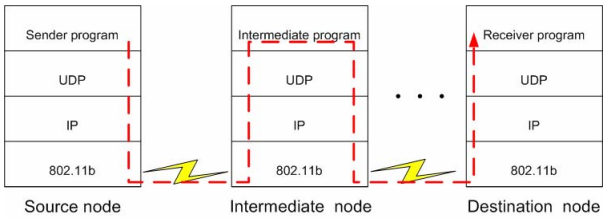

Figure 1. Wireless testbed architecture

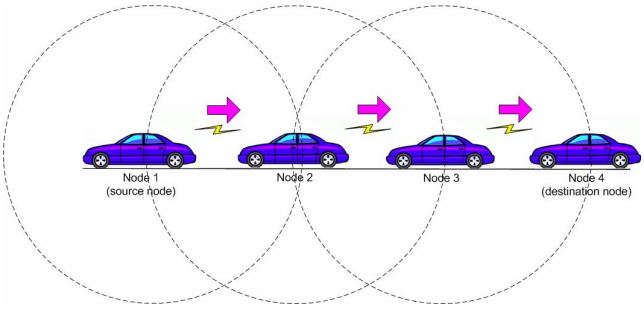

Figure 2. Wireless testbed topology, where arrows indicate direction of traffic flow

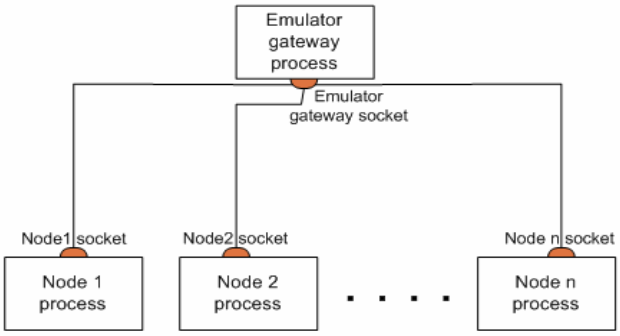

Figure 3. Wired testbed architecture

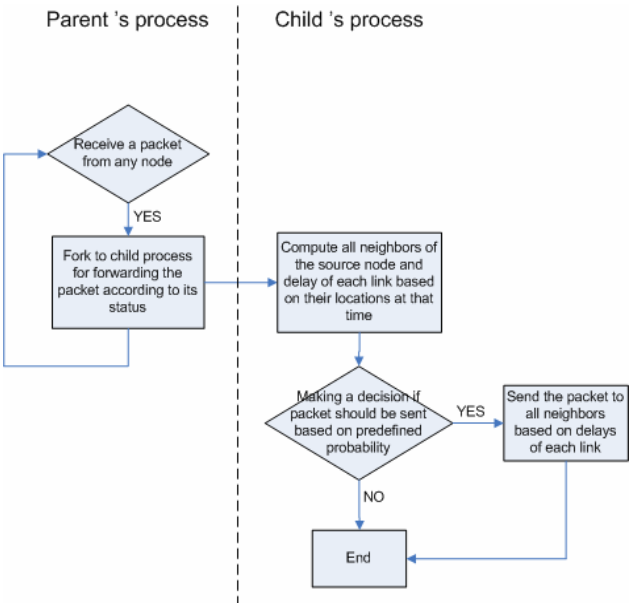

Figure 4. Functional processes of EmuGw

A. *Distributed Real-time Control with Regulating Buffer Threshold (DRC-RBT)*

In this mechanism, each intermediate node accommodates incoming packets in proper buffers, which are assigned on a per-flow basis. Upon receiving the first packet of a flow, the node will calculate the appropriate time duration for holding packets in the buffer before forwarding it to the next node. This duration is computed from a predefined buffer threshold and a transmission interval of each packet. When the first packet of a flow is due, the node starts forwarding packets of that flow according to the sequence of their arrivals and the transmission interval of the flow. To take into account of lost packets, the mechanism can calculate a due time of each of the following packets from their sequence numbers and the transmission interval. Inherently, only CBR traffic is applicable for this mechanism.

Fig. 5 illustrates an example of DRC-RBT process when a buffer threshold is set to 2 frames, i.e., the intermediate node starts forwarding the first packet after it reaches two time slots and transmits the following packets at their due time according to their sequences. Similar to the concept of playout delay, this mechanism is supposed to decrease end-to-end jitter significantly with the price of increased end-to-end delay. However, given the right choice of the regulating buffer threshold, the packet delay should still be made tolerable.

B. *Adaptive Distributed Real-time Control (ADRC)*

As the number of hops increases, DRC-RBT needs a larger buffer threshold to absorb all the increased packet timing uncertainties. This may lead to unacceptable end-to-end delay. To overcome this foreseen problem, ADRC is here proposed. Similar to DRC-RBT, the mechanism of ADRC is distributed and every node is designed to help regulating the packet delays. However, instead of using a buffer threshold as a criterion and always delaying outgoing packets, now each packet may be forwarded immediately, or may be held in a buffer for a moment before being transmitted. ADRC will classify each incoming packet as **early**, **in time**, or **late**, based on each packet's arrival time, and adaptively determine a suitable due time for each packet individually. Whether packet i is early, in time, or late depends on its relative delay, or $relative\_delay_i$, that could be calculated as in (1). When a node receives packet i, ADRC mechanism will update the average delay, or $average\_delay_i$, by using exponentially weighted moving average (EWMA) as shown in (2), where w denotes the weight, and $average\_delay_1 = delay_1$. Delay time is computed according to (3), where R_i and S_i denote the receiving time and sending time of packet i respectively. Here, S_i is assigned as a tag to every incoming packet, based on a timestamp from the preceding node. Without time-synchronization among nodes, the apparent time may differ, but assuming that all nodes have the same clock speed, the offset is a constant.

$$relative\_delay_i = delay_i - ave\_delay_i \quad (1)$$

$$ave\_delay_{i+1} = (1 - w) \times ave\_delay_i + w \times delay_i \quad (2)$$

$$delay_i = R_i - S_i \quad (3)$$

Fig. 6 gives a summary on how ADRC make a decision. In the diagram, c is defined as the lower-bound of relative delay that a node will not add more delay to a packet and d is upper-bound that node will not drop the packet. ADRC does not only greatly reduce delay time compared to DRC-RBT but it can also be applied for CBR and VBR traffic since it does not rely on the assumption of constant transmission interval.

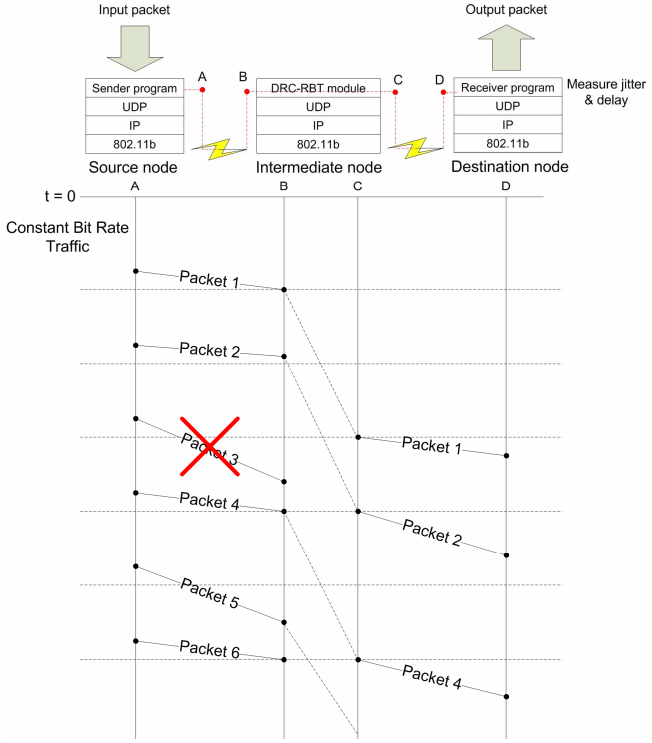

Figure 5. Example of DRC-RBT process

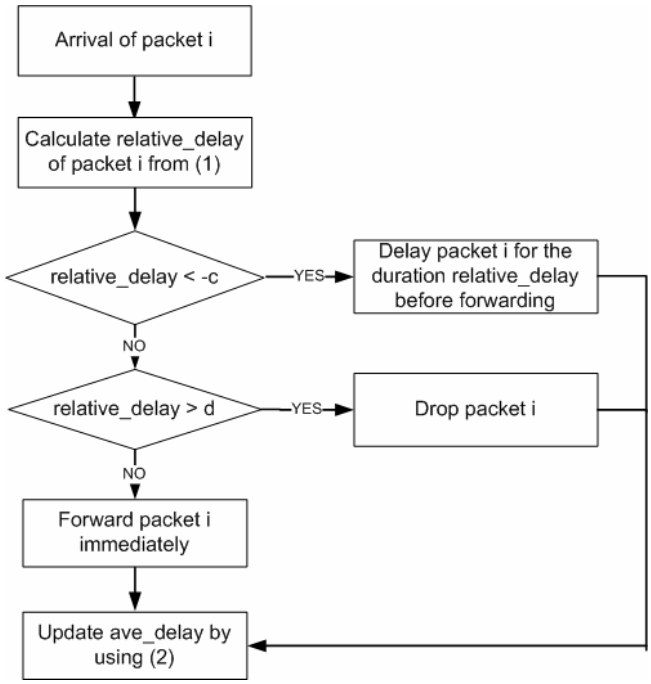

Figure 6. ADRC decision flowchart

IV. Experimental Results

The proposed mechanisms, along with the standard algorithm, are implemented in our testbeds and experiments are carried out in various conditions to observe the effect on end-to-end jitter and delay. Specifically, two conditions are of interest and shown here; node mobility and network size. All tests used source file encoded as MPEG-4 standard. Two types of traffics, CBR and VBR, have been considered in the experiments. For VBR traffic, the bit rate depends on the content of each frame. Frame interval has been set to 66.7 ms (calculated from 15 frames/second frame rate). In CBR case, every packet is transmitted as Constant Bit Rate with 61.9 ms sending interval (average bit rate of VBR traffic). While packet size in both cases is fixed as 1024 bytes. Definitions of the parameters in all experiments are as followed;

th = buffer threshold of DRC-RBT = 10 frames
w = weight of ADRC = 0.02
d = dropping threshold of ADRC = 500 ms
c = delaying threshold of ADRC = 2 ms
l = packet loss probability per hop in the wired testbed

A. Effect of Node Mobility

To observe the effect of node mobility, two testing scenarios are performed. Initially, the wireless testbed is used to observe the effect of environmental changes while maintaining distances among nodes. Alternatively, the wired testbed is used to study the effect of route changes.

1) Environmental Changes: The testing scenario as described in section II–A is performed, where traffic from Node i is imposed to go only to Node $i + 1$, $1 \le i \le 3$, without route alteration. In the test, we compared jitter and delay in both moving and non-moving node scenarios. Figs. 7 and 8 show the jitter and delay, respectively, of the CBR case with uncontrolled transmissions, DRC-RBT, and ADRC. For the VBR case with uncontrolled transmissions and ADRC, as DRC-RBT is not applicable to VBR traffic, the jitter and delay are shown in Figs. 9 and 10, respectively.

Testing in wireless scenario cannot be performed for a long period of time because of several limitations. Therefore, we would not directly compare their values. In this experiment, we focus on the aspect of how well each mechanism can perform in various situations. From the results, it is clear that environmental changes can cause unpredictable delay, which simply causes high jitter in the uncontrolled transmissions. DRC-RBT copes with this problem by increasing the end-to-end delay, while ADRC solves this by its weighing technique. In ADRC, delay gradually decreases upon weighing value after it abruptly increases, which implies that jitter can be reduced.

2) Route Changes: This scenario is tested on the wired testbed to allow us to control node's mobility pattern, using the *Car Following Model*. The theory of this model rests on the assumption that all vehicles travel in one lane with no overtaking. The speed and acceleration of each vehicle may change over time, according to the speed and acceleration of the vehicles in front. However, there is no restriction on the distance between each car. Thus, for $j \ge 1$ and $k \ge 2$, a route change may occur whenever a distance between Node j and Node $j + k$ is small enough that Node j and Node $j + k$ can communicate directly. This experiment uses a 30-node network, with the traffic originated from Node 1 and destined to Node 30, and the packet loss probability per hop $l = 0.005$. The jitter and delay of uncontrolled transmissions, DRC-RBT and ADRC are shown in Figs. 11 and 12, respectively.

From the results, we can see that when route changes occur, DRC-RBT no longer works, because this mechanism relies on each intermediate node to perform distributed timing compensation. Thus, whenever a node is skipped, the proper timing offset is lost. Consequently, DRC-RBT should only be used in static networks with invariable hop counts. On the other hand, ADRC has no such problem, and it always outperforms the uncontrolled transmissions, especially in the dynamic scenario, although it has higher jitter than DRC-RBT in the static scenario.

B. Effect of Network Size

The wired testbed is a more appropriate choice in this scenario for the reason of tractability, since the number of mobile nodes has to be varied. In this experiment, all nodes are static, while the network size is varied from 10 hops to 50 hops, with $l = 0$. Fig. 13 displays the average jitter and delay of uncontrolled transmissions compared with DRC-RBT and ADRC.

Intuitively, end-to-end delay is proportional to the number of hops in either case. Nevertheless, jitter has no such relationship, especially in DRC-RBT and ADRC. For a larger network, there is a higher possibility that delay variation can be compensated. Moreover, the jitter becomes almost independent to the number of hops when all intermediate nodes perform the proposed jitter control mechanisms. This is a desirable property in multimedia transmissions which can help minimize the amount of buffer allocated at receivers.

V. Conclusion

The works presented in this paper are two-fold; proposing distributed real-time control mechanisms for minimizing end-to-end jitter in MANETs, and creating testbeds for evaluating them. The results have shown that not only new techniques are needed for MANETs, but they must also be tested in the real environments. Obtained results from the testbed experiments suggest that these mechanisms, while increasing the packet delay, can decrease the end-to-end jitter significantly. DRC-RBT mechanism is suitable for transmitting CBR traffic in static networks, especially when the end-to-end delay is not crucial. On the contrary, ADRC is appropriate not only for both types of traffic, but also for static and dynamic networks. We also discover that packet delays in MANETs using IEEE 802.11b air interface in the dynamic node scenario are highly unpredictable, as it is not designed to support mobility. This observation truly emphasizes the necessity of testbed in protocol designing. We are looking forward to performing experiments in more scenarios in the future; including when there are competing traffic streams. Ultimately, we firmly believe that the proposed mechanisms are going to greatly improve the QoS in term of jitter for multimedia transmissions in MANETs that employ different MAC protocols as well.

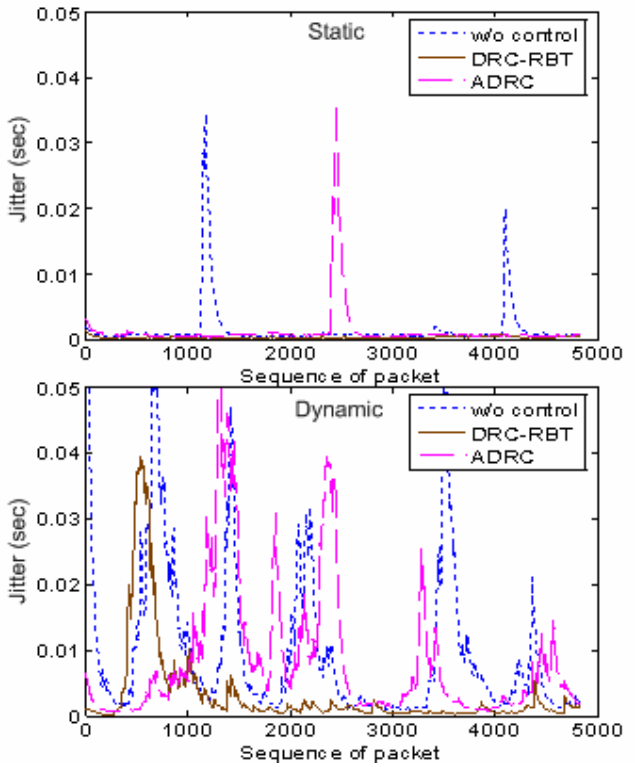

Figure 7. Jitter of CBR traffic in wireless static and dynamic node scenarios

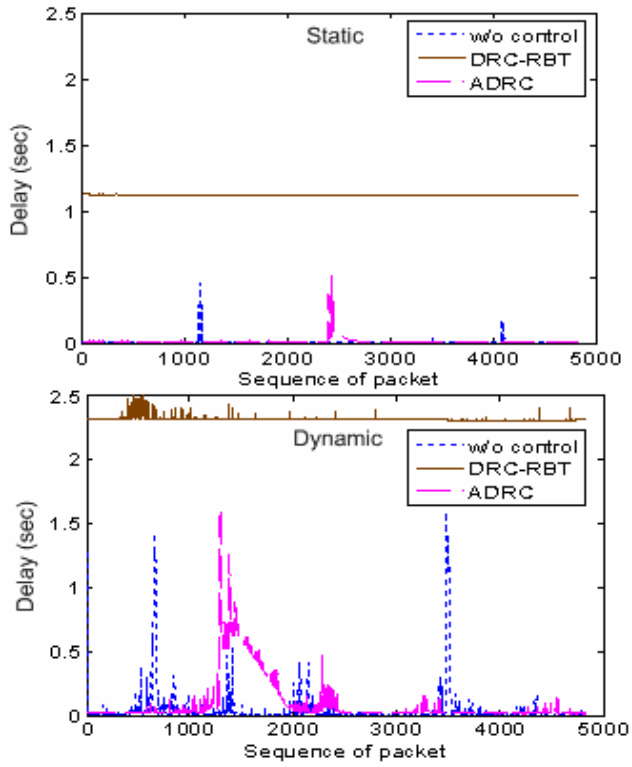

Figure 8. Delay of CBR traffic in wireless static and dynamic node scenarios

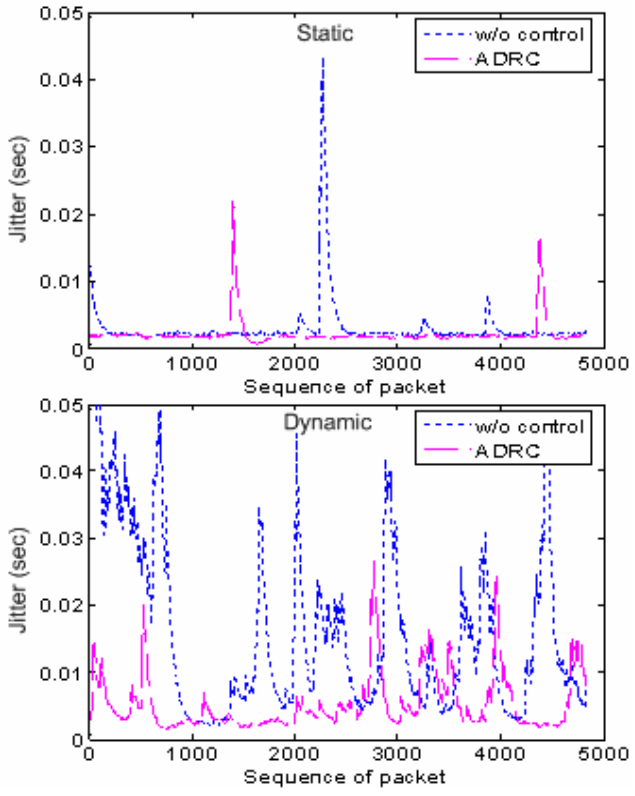

Figure 9. Jitter of VBR traffic in wireless static and dynamic node scenarios

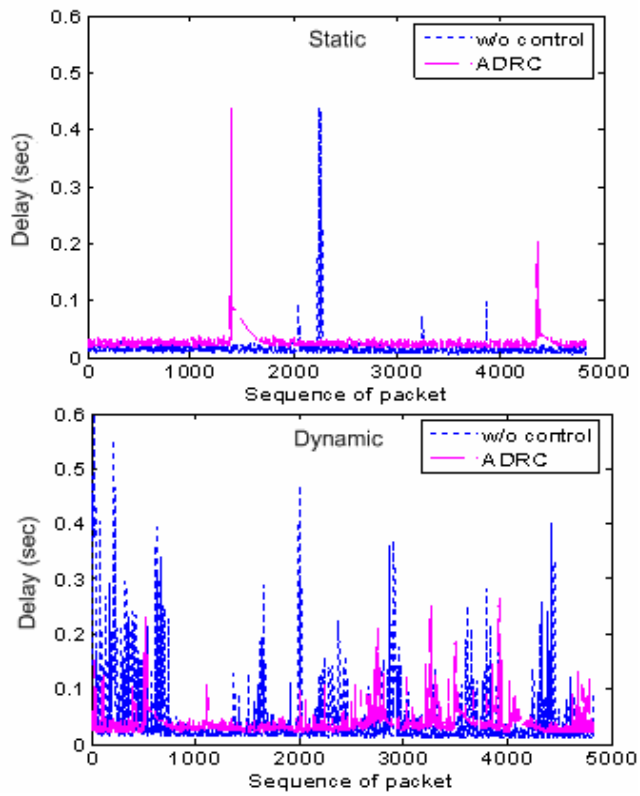

Figure 10. Delay of VBR traffic in wireless static and dynamic node scenarios

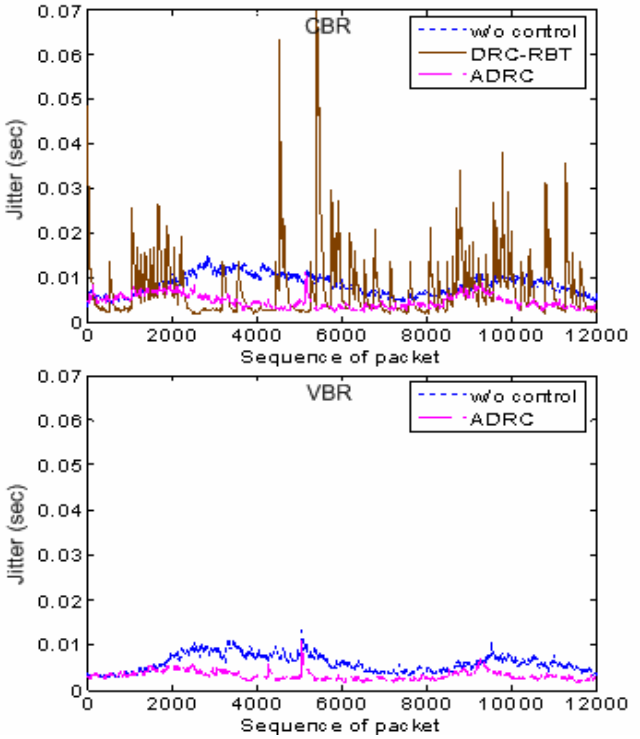

Figure 11. Jitter of CBR and VBR traffic in route change scenarios

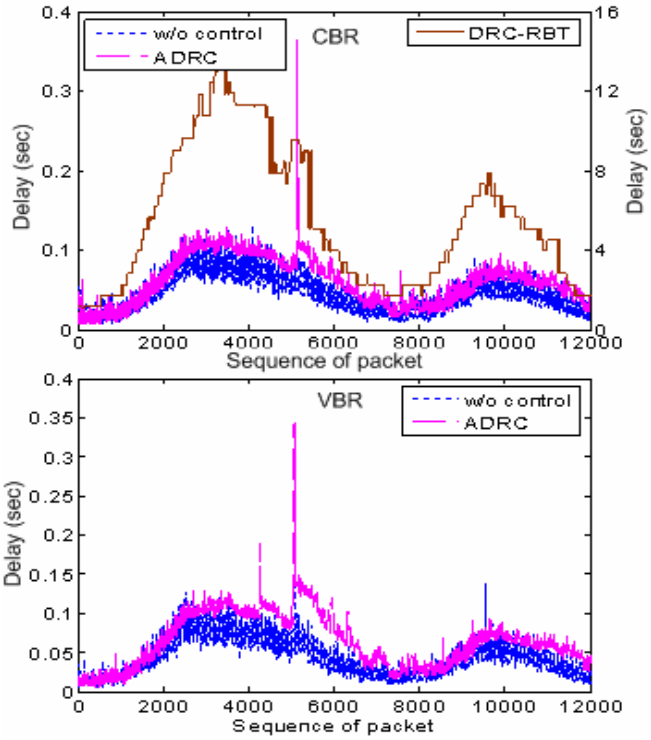

Figure 12. Delay of CBR and VBR traffic in route change scenarios

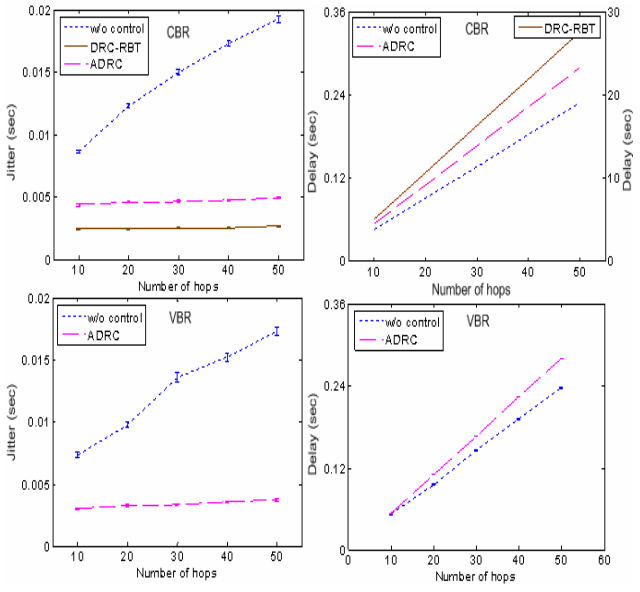

Figure 13. Average jitter and average delay of CBR and VBR traffic vs. number of hops

REFERENCES

[1] The Internet Engineering Task Force (IETF) Mobile Ad-hoc Networks (MANET) Working Group. (April 2006). Available: www.ietf.org/html.charters/manet-charter.html

[2] J P. Horrell. (September 2004). "Intelligence: Behold the all-seeing, self-parking, safety-enforcing, networked automobile", *Popular Science, special section: the future of the car*. Available: www.popsci.com

[3] M. Rabel, A. Schmeiser and H. P. Grobmann, "Ad-hoc in-car networking concept", *IEEE Trans. Intell Transp Syst*, September 2005.

[4] ns-2. Available: www.isi.edu/nsnam/ns

[5] GloMoSim. Available: pcl.cs.ucla.edu/projects/glomosim

[6] OPNET Modeler. (September 2006). Available: www.opnet.com/products/modeler/home-1.html

[7] D. A. Maltz and J. Broch, "Lessons from a full-scale multihop wireless ad hoc network testbed", *IEEE Pers Commun*, February 2001.

[8] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordstrom and C. Tschudin, "A large-scale testbed for reproducible ad hoc protocol evaluations", *IEEE WCNC*, 2002.

[9] Y. Zhang and W. Li, "An integrated environment for testing mobile ad-hoc networks", ACM MOBIHOC, 2002.

[10] R. Ramanathan and R. Hain, "An ad hoc wireless testbed for scalable, adaptive QoS support", *IEEE WCNC*, 2000.

[11] S. Sanghani, T. Brown, S. Bhandare and S. Doshi, "EWANT: The Emulated Wireless Ad hoc Network Testbed", *IEEE WCNC*, 2003.

[12] D. Verma, H. Zhang, and D. Ferrari, "Delay jitter control for real-time communication in a packet switching network", *IEEE TriCom*, 1991.

[13] S. F. Bush, A. Kulkarni, S. Evans and L. Galup, "Active jitter control", *7th International IS&N Conference, Intelligence in Services and Networks* (*ISN*), February 23–25, 2000.

[14] R. W. Rothery, "Car following models", *Traffic flow theory*, Transportation Research Board, Special Report. Chap. 4, 1992.