An Adaptive Selection of Congestion Window for TCP over Ad-Hoc Networks

Jung-Hoon Song°, Sang-Don Lee, and Ki-Jun Han
Department of Computer Engineering, Kyungpook National University
{pimpo°, netopia7}@netopia.knu.ac.kr, kjhan@bh.knu.ac.kr

Abstract

TCP does not distinguish between congestion and packet losses due to route change and link failures, which are prevalent in mobile ad hoc networks. So, TCP does not show satisfactory performance in ad hoc networks since it assumes that all packet losses are due to network congestions. In particular, when a route is reestablished it needs to be adaptively determined CWND according to the new route features. In this paper, we proposed CWND adjustment scheme to improve the TCP performance overad hoc networks. TCP sender effectively adjusts CWND by monitoring the network situation using control packets. Simulation results using NS-2 show that the proposed scheme increases TCP throughput compared with those of general TCP.

1. Introduction

In a mobile ad hoc network, a group of mobile computing devices communicate among themselves using wireless radios, without the aid of a fixed networking infrastructure. Due to their dynamic properties, mobile ad hoc networks have gained significant attention lately as a way of providing continuous network connectivity to mobile computing devices in various areas such as disaster and military applications. In such applications, reliable packet exchange is an indispensable requirement [8]. TCP, however, does not show a satisfactory performance in mobile ad hoc networks due to a high BER (Bit Error Rate) and node mobility. Base TCP does not distinguish between congestion and packet losses due to transmission errors, which are prevalent in mobile ad hoc networks. The node mobility results in frequent route re-computations Aand occasional network partitions. Discovering new routes takes a significantly longer time than the RTO (Retransmission Time Out) interval at the sender. As a result, the TCP sender times out, retransmits the packet and invokes congestion control [7].

In mobile ad hoc networks, routes are frequently re-computed due to high mobility. At this time, the TCP sender will never get an opportunity to transmit at the maximum negotiated rate because its CWND (Congestion Window) will be much smaller than the receiver's advertised window size. It is also likely that an ad hoc network gets periodically partitioned and the sender and receiver of a connection lie in different partitions. Then, all the sender's packets get dropped which results in the sender invoking congestion. If the situation persists for a few seconds, there could be multiple RTO expirations and the RTO may reach its upper bound. Finally, when the receiver and sender get connected, it could take several seconds before there is some kind of transmission [7].

There have been a lot of research works to address the above problems. TCP-Feedback [1], ELFN(Explicit Link Failure Notification) based approach [2] and ATCP (Ad hoc TCP) [3] deal with a route failure due to node mobility. Fixed-RTO [4] and TCP-DOOR (Detection of Out-of-Order and Response) [5] are modified TCP versions which adapt themselves to dynamic ad hoc environments. The main concern is how to avoid invoking unnecessary congestion control in case of packet losses due to node mobility. They, however, did not reflect the dynamic network situation when determining the optimal value of a congestion window, so they could not offer a satisfactory end-to-end throughput. In this paper, we propose a new congestion window scheme which effectively adjusts the TCP's CWND based on network conditions in order to enhance the end-to-end throughout.

The rest of this paper is organized as follows. Section 2 describes our CWND adjustment mechanism. Section 3 shows that our scheme enhances TCP throughput via NS-2 simulation. Finally, the conclusion is presented in Section 4.

2. CWND Adjustment Mechanisms

The TCP assumes that all packet losses are due to network congestion. Previous research works based on feedback messages might distinguish between congestion and packet losses due to route change and link failures, but they could not be applicable to the ad hoc networks since they could not efficiently adjust the congestion window size for dynamic topology change. A new mechanism must be addressed toward an adaptive approach. In this paper, we propose the CWND adjustment scheme to improve the TCP performance over mobile ad hoc networks.

Consider the partial change of an intermediate node due to local repair as shown in Fig. 1. In this case, the sending TCP

stops transmission and waits until the route is reconnected through a detour. After this, it re-computes a new CWND based on the information about the buffer status of nodes along the route. The buffer status means how much buffer is being unoccupied at each node, and is measured as the ratio of the amount of the available buffer space to the buffer capacity at each node. If the new route has a higher available bandwidth than the old one, the sending TCP maintains the old CWND value. Otherwise, the sending TCP sets CWND to 1. In particular, if the buffer availability for nodes along the new route, denoted by Q_{buf}, is below some threshold value (Q_{thr}), we consider that the new route has a less available bandwidth than the old one. Here, the buffer availability (Q_{buf}) is defined as the minimum buffer availability at each node along the route.

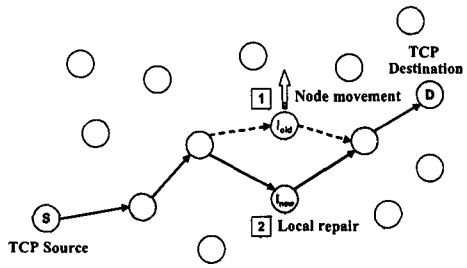

Fig. 1 Local repair at the intermediate node

In AODV routing protocol [6], when there is a link break on an active route, the node upstream of that break chooses to repair the link locally if the destination is not farther than local repair threshold (MAX_REPAIR_TTL) hops away. To repair the link break, the node broadcasts an RREQ message for that destination. The node initiating the repair then waits the discovery period to receive RREPs in response to the RREQ. At this time, in our scheme, information on buffer availabilityis exchanged among nodes over the route via the RREP. During this process, the minimum buffer availability (Q_{buf}) can be found by comparing the value of Q_{buf} field in the RREP with its own buffer utilization. When the repairing node receives an RREP it issues an RERR message for the destination, with the value of Q_{buf} and 'N' bit set. At this time, the node does not compare the hop count of the new route with the hop count field of the invalid route table entry for the destination, which is not case in the original AODV. Following this, the repairing node sends an RERR to the sending TCP with Q_{buf} and 'N' flag set as shown in Fig. 2(a). When the TCP sender receives an RERR with 'N' flag set indicating that there has been a local repair, it determines the value of CWND using the information on the buffer availability.

Now consider the case where a temporary movement of some intermediate node along the route causes link failure or a partition of the network, but the existing route can be reconnected after some time.

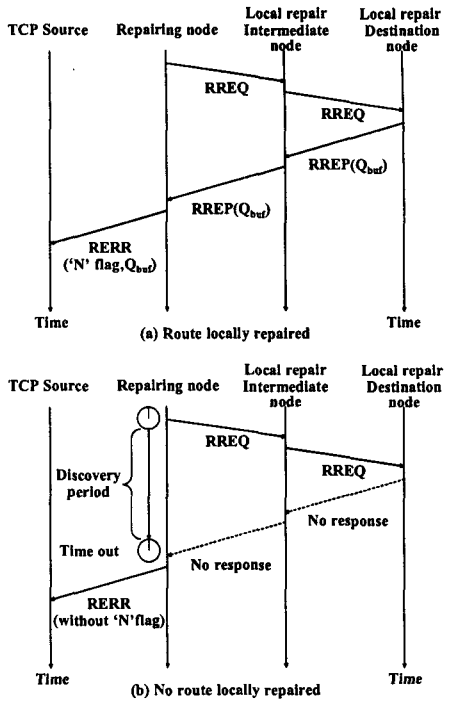

Fig. 2 Signaling for Local repair

At this time, if the sender and receiver of a TCP connection temporarily lie in different partition for a short time as shown in Fig. 3, the sender does not have to invoke congestion control. In our scheme, the TCP sender waits until the existing route is reconnected and then resumes transmission with maintaining the old CWND value. If the partition of the network persists for a few seconds, and as a result, the source node receives an RERR message without 'N' flag as shown in Fig. 2(b), a fresh route is established. At this time, the TCP initializes CWND to 1 and resumes transmission.

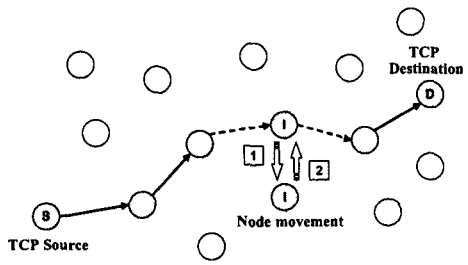

Fig. 3 Route reconnection after movement

As explained so far, in our scheme, the TCP sender monitors the network situation using control packets such as RREP,

RERR or ICMP messages, and efficiently adjusts the CWND value based on network conditions.

3. Simulation

We evaluated he performance enhancements made by our CWND adjustment scheme. We used the NewReno TCP protocol, with and without applying our scheme. To compare our scheme with ATCP, we modified the NewReno TCP. It is similar to ATCP, which stops transmission until the route is established and sets CWND to 1 after re-computing new route. The simulation study is done by the NS-2 network simulator which is a discrete event simulator developed as part of the VINT project at the Lawrence Berkeley National University [11]. The extensions implemented by the CMU Monarch project enable it to simulate mobile nodes that are connected by wireless network interfaces [12]. We modified the CMU extension of the AODV and TCP code to implement CWND adjustment mechanisms.

The main objective of our simulation study is to prove that CWND adjustment scheme shows a noticeable improvement in the throughput. In the simulation, the buffer availability threshold (Q_{thr}) is given by three quarters of the buffer capacity. Local repair procedure is always executed due to movement of the intermediate node.

Fig. 4 shows the throughput of the proposed. Note that our scheme offers a higher throughput in the CBR traffic rate than the conventional ones. Overall, however, the throughput decreases as the offered load is increased at the intermediate node. In case of low offered load, the sending TCP maintains the old CWND value and results in the higher performance in our scheme. Because the minimum buffer availability of the intermediate node is higher than the threshold value. If the intermediate node is offered high rate of traffic, throughput of proposed scheme is similar to conventional ones. This is shown that the minimum buffer availability of the intermediate nodes is lower than the threshold value and CWND is set to one segment.

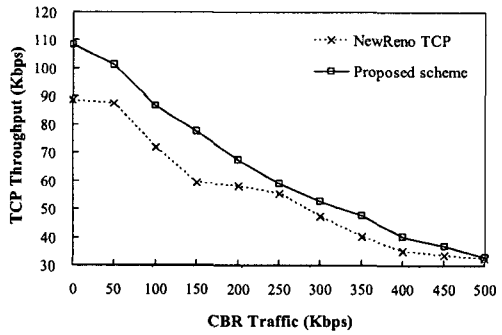

Fig. 4 Simulation result

4. Conclusion

In the existing feedback based schemes, the source could be informed of a route failure so that it could not misinterpret it as congestion and get into the congestion control phase. In case of network topology change, however, they maintained the old value of CWND or just set it to one. This was somewhat extreme approach, and thus degraded the end-to-end throughput. In this paper, we proposed an adaptive CWND adjustment scheme to improve the TCP performance by efficiently coping with dynamic topology changes over mobile ad hoc networks. Simulation results showed that our scheme offered a better performance than the existing schemes.

References

[1] K. Chadran, S. Raghunathan, S. Venkatesan,and R. Prakash, "A feedback based scheme for improving TCP performance in ad-hoc wireless networks," in Proc. 18th International Conference on Distributed Computing Systems (ICDCS'98), Amsterdam, The Netherlands, May 26-29, 1998, pp. 474-479.

[2] G. Holland,and N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in Proc. ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99), Seattle, Washington, USA, Aug. 1999.

[3] J. Liu, and S. Dingh, "ATCP:TCP for Mobile Ad Hoc Networks," IEEE Journal on selected areas in communications, vol. 19, No. 7, July 2001.

[4] T. Dyer, and R. Boppana, "A comparison of TCP performance over three routing protocols in mobile ad hoc networks," in Proc. ACM International Symposium onMobile Ad Hoc Networking and Computing (MobiHoc 2001), Long Beach, California, USA, Oct. 2001.

[5] Y. Zhang, and F. Wang, "Improving TCP performance over mobile ad-hoc networks with out-of-order detection and response," in Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2002), Lausanne, Switzerland, June 2002.

[6] C. Perkins, E. Belding-Royer, and S. Das, "Ad Hoc On-demand Distance Vector (AODV) Routing," IETF RFC 3561,July 2003.

[7] V. Sridhara, "Evaluating Different Techniques to Improve TCP Performance over Wireless Ad Hoc Networks," Research Paper, available from http://www.eecis.udel.edu/~sridhara/wireless/wireless.html.

[8] J. H. Choi, and C. Yoo, "TCP-aware Source Routing in Mobile Ad Hoc Networks," Eighth IEEE International Symposium on Computers and Communications (ISCC'03), Kemer, Antalya, Turkey, July 2003.

[9] R. De Oliveira, and T. Braun, "TCP in Wireless Mobile Ad Hoc Networks," Technical Report, IAM-02-003, University of Bern, Switzerland, March 2003.

[10] W. Stevens, TCP/IP Illustrated (Vol. 1, The Protocols), Addison-Wesley, 1994.

[11] K. Fall,and K. Varadhan, "NS notes and documentation," the VINT Project, UC Berkeley, LBL USC/ISI, and Xerox PARC, available from http://www.isi.edu/nsnam/ns/, Dec. 2003.

[12] CMU Monarch Group, "CMU Monarch extensions to the NS-2 simulator,"