TCP Performance in Mobile Ad-hoc networks: Overview and discussion of an enhancement

Abhuday Aggarwal
Dept of Computer Science
University of Texas, Arlington
Term Paper for CSE 6345
abhuday@hotmail.com

Abstract

Transmission Control Protocol (TCP) performs well in a wired network. However, in the field of mobile ad-hoc networks, significant work has been performed to enhance TCP performance. This is because transport connections set up in a mobile ad-hoc network face many problems. These problems occur due to the nature of a wireless ad-hoc network. Wireless networks are prone to losses due to mobility and transmission errors. These lead to high bit error rates. Mobility of nodes causes frequent re-computation of routes and network partitions. Bandwidth in a wireless channel is limited and varies during transmission due to mobility and obstacles. Packet losses due to transmission errors are very common in wireless networks. Running TCP over such connections leads to low throughput since TCP treats lost or delayed acknowledgements as congestion.

In this paper, I discuss reasons for low TCP performance in ad-hoc networks and describe a technique put forth to provide a better connection protocol in a wireless ad-hoc network and overcome the limitations of TCP. Specifically, I discuss one TCP enhancement technique: ATCP. This technique suggests a solution to improve the performance of the transport connections in a wireless ad-hoc network. ATCP is implemented as a thin layer between the existing TCP and IP layers.

Introduction

Wireless ad-hoc networks are gaining increasing popularity and indications are that these will play an important role in the future. Ad hoc networks are formed by collection of “peer” hosts which may be mobile. These nodes could be laptops mounted on vehicles or carried by people; or simple autonomous sensors. The nodes are capable of communicating with each other without help from a fixed infrastructure. The connections between nodes can change on a continual and arbitrary basis. Nodes within each other’s radio range communicate directly via wireless links, while those that are far apart use other nodes as relays in a multi-hop routing style. These types of networks are also very useful in situations where temporary network connectivity is needed, such as disaster relief areas or in a battle zone.
Ad-hoc networks; therefore, have certain characteristics that are different from traditional computer networks. Firstly, there is no fixed infrastructure. The mobile hosts themselves serve as peer-to-peer relays since there are no dedicated routers. Secondly, hosts in the network are mobile. From the perspective of end-to-end connections, not only are the end hosts mobile, but the “routers” are also mobile. Thirdly, the communication channel is shared and lastly, these networks are characterized by bandwidth that is not only scarce but also varies significantly during transmission due to mobility or some obstacles.

TCP is a connection-oriented transport layer protocol that provides reliable, in-order delivery of data. This protocol is designed to perform well in wired networks where packet losses are mainly due to congestion. Research has shown that the performance of these TCP control mechanisms is insufficient for a wireless network. This is because wireless networks are characterized by losses due to mobility and transmission errors due to high bit error rates. In mobile ad-hoc networks, node connectivity changes over time. There are a couple of effects of this change in node connectivity. Nodes may have to re-compute routes to some of the destinations and it is likely that the ad-hoc

network may be temporarily partitioned due to node mobility. This change in node connectivity can cause throughput to drop to very low levels.

In the next section, we discuss the characteristics of a mobile ad hoc network and what problems it causes with TCP. The section following that discusses an enhancement technique to TCP. Several schemes have been proposed to alleviate the effects of non-congestion-related losses on TCP performance over networks that have wireless links. These schemes choose from a variety of mechanisms, such as local retransmissions, split-TCP connections, and forward error correction, to improve end-to-end throughput. Recent work has also focused on developing MAC layer protocols and routing protocols for these types of networks. In this paper, however, we pay attention to a transport layer protocol – ATCP. We will discuss the design of this protocol and provide pros and cons of this approach. We then succinctly compare a few other enhancement techniques. Finally, I provide the summary and conclusions.

Effects of a mobile ad hoc network on TCP

Let us first consider the negative effects of each of the characteristics of a mobile ad-hoc network on TCP throughput:

High Bit Error Rate

Bit errors lead to packet corruption, which result in TCP data segments or acknowledgements getting lost. The TCP sender is expecting the acknowledgement to arrive within a certain amount of time (retransmit timeout – RTO). If the acknowledgement does not arrive within this time, the sender retransmits the segment, exponentially backs off its retransmit timer for the next transmission, and invokes congestion control. Repeated errors will cause the congestion window to remain small resulting in low throughput.

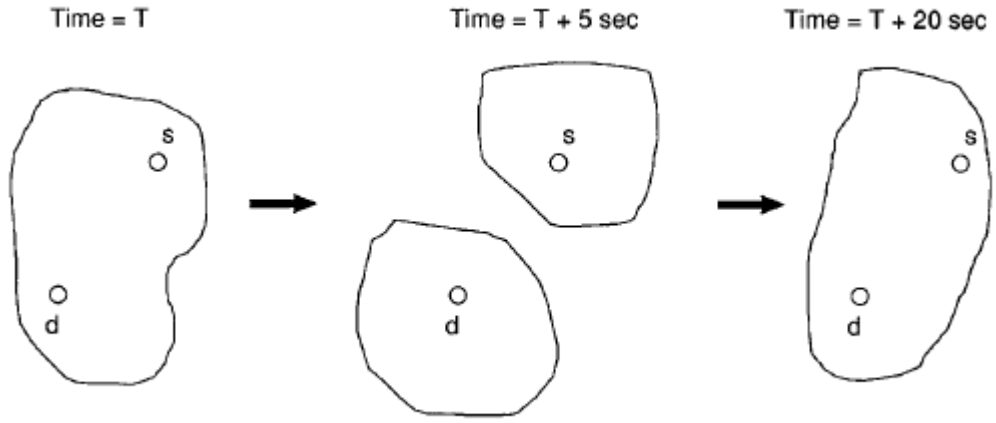
Route Re-computations

The network layer at the sender tries to compute new routes when old routes are no longer available. There are a few different ad hoc routing protocols – DSDV, DSR and AODV. We will not get into details of these routing protocols, since this paper concentrates at the transport layer. However, brief mentions to these protocols will be made as necessary. The route computations are done via route discovery messages in DSR (Dynamic Source Routing) while in DSDV (destination-sequenced distance-vectoring) table exchanges are triggered that eventually result in a new route being found. It is possible that discovering a new route may take much longer than the RTO at the sender. As a result, the TCP sender times out, retransmits the packet, and invokes congestion control. Thus, when a new route is discovered, the throughput will continue to be small for some time because TCP at the sender grows its congestion window using the slow start and congestion avoidance algorithm. This is not the desired behavior because the TCP connection will be very inefficient. If we imagine a network in which route computations are done frequently (due to high node mobility), the TCP connection will never get an opportunity to transmit at the maximum negotiated rate. This means that the congestion window will always be significantly smaller than the advertised window size from the receiver.

Network Partitions

It is possible that the ad hoc network may occasionally get partitioned for several seconds at a time. If the sender and the receiver of a TCP connection are in different partitions, all the sender's packets get dropped by the network resulting in the sender invoking congestion control. If the partition lasts for a significant amount of time (several times longer than the RTO), the situation gets even worse because of a phenomena called serial timeouts. A serial timeout is a condition wherein multiple consecutive retransmissions of the same segment are transmitted to the receiver while it is disconnected from the sender. All these retransmissions are, thus, lost. Since the retransmission timer at the sender is doubled with each unsuccessful retransmission attempt (until it reaches 64 s), several consecutive failures can lead to inactivity lasting one or two minutes even when the sender and receiver get reconnected.

The picture below [1] shows a network getting partitioned at time T+5, causing the source and destination nodes to lie in different partitions.



(b) Partitions are formed and recombined by mobility

Multipath Routing

Some routing protocols, such as temporally-ordered routing algorithm (TORA) maintain multiple routes between source destination pairs, the purpose of which is to minimize the frequency of route re-computation. This sometimes results, undesirably, in a significant number of out-of-sequence packets arriving at the receiver. The effect of this is that the receiver generates duplicate acknowledgments which cause the sender (on receipt of three duplicate acknowledgements) to invoke congestion control.

ATCP (Ad hoc TCP)

ATCP is implemented as a thin layer between IP and TCP. It does not modify TCP itself to maintain compatibility with the standard TCP/IP suite. They carefully consider what congestion window means in ad hoc networks - The congestion window in TCP imposes an acceptable data rate for a particular connection based on congestion information that is derived from timeout events as well as from duplicate acknowledgements.

In an ad-hoc network, since routes change during the lifetime of a connection, the relationship between the CWND size and the tolerable data rate for the route is lost. In other words, the CWND as computed for one route may be too large for a newer route, resulting in network congestion when the sender transmits at the full rate allowed by the old CWND. This approach utilizes network layer feedback (from intermediate hops) to put the TCP sender into either a persist state, congestion control state or retransmit state. Thus, when the network is partitioned, the TCP sender is put into persist mode so that it does not needlessly transmit and retransmit packets. Alternatively, when packets are lost due to error, as opposed to congestion, the TCP sender retransmits packets without invoking congestion control. Finally, when the network is truly congested, the TCP sender invokes congestion control normally. ATCP listens to the network state information provided by ECN (explicit congestion notification) messages and by ICMP "Destination Unreachable" messages and then puts TCP at the sender into the appropriate state. Thus, on receipt of a "Destination Unreachable" message, TCP state at the sender is frozen (the sender enters the persist state) until a new route is found ensuring that the sender does not invoke congestion control.

When the TCP connection is initially established, ATCP at the sender is in the "normal" state and does nothing. If the connection from the sender to the receiver is lossy, segments may be lost or may arrive out-of-order. The receiver generates duplicate ACKs in response. ATCP, in its "normal state", counts the number of duplicate ACKs received for any segment and forwards the first two to TCP. On receiving the third duplicate ACK, it is not forwarded and TCP is put into "persist" mode. Similarly, when RTO expires, ATCP moves TCP to "persist" mode. Hence, TCP sender does not invoke congestion control because both these events correspond to lossy links wherein both data segments and ACKs could be lost. Then ATCP enters "loss" state and retransmits all the unacknowledged segments from the TCP's buffer, reusing the TCP's timers. Finally, when a new ACK arrives ATCP forwards it to TCP, which removes it from "persist" mode and ATCP returns to "normal" state.

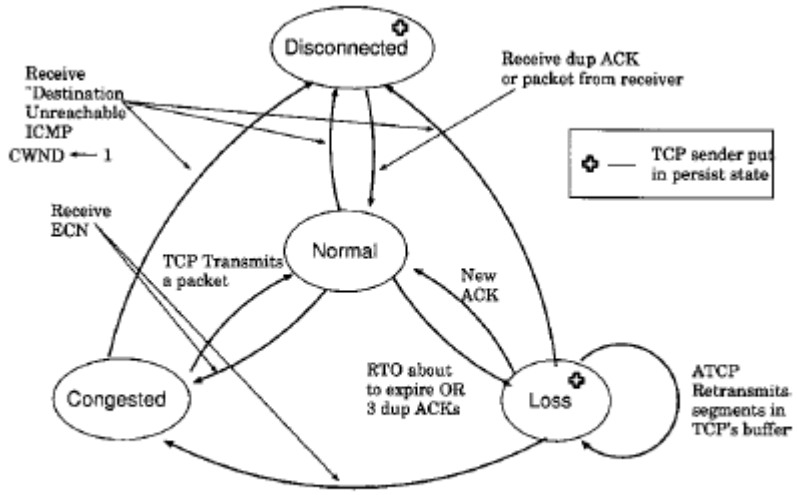


Fig. 2. State transition diagram for ATCP at the sender.

When the network gets congested, the sender receives ECNs. Then ATCP moves to “congested state” and does not interfere with TCP’s congestion control behavior. If ATCP is in “loss” state, ATCP removes TCP from “persist” mode. ATCP ignores all the duplicate ACKs and RTO expirations. Once TCP transmits a new segment, ATCP returns to “normal” state.

Node mobility could result in either route re-computation or temporary network partition that result in the generation of ICMP “destination unreachable” messages. When the sender receives this message, ATCP (in “normal”, “congested” or “loss” state) puts TCP into “persist” mode, sets TCP’s CWND to one segment and itself into “disconnected” state. Setting CWND to one segment ensures that TCP probes the network to determine the correct value of CWND using the new route. In “persist” mode, TCP periodically generates “probe” packets, with the interval between them equal to RTO. When the receiver gets connected to the network, it responds to the “probe” packets with an ACK (or a data packet). This removes TCP from “persist” mode and ATCP goes back to “normal” state.

Benefits

The benefits of the ATCP solution include the fact that the TCP/IP suite is unmodified. ATCP is not visible to TCP; this allows for interoperability between nodes running ATCP and those without.

Drawbacks

ATCP heavily depends on ECN messages to recognize a congested network. There is a possibility that an ECN sent by the receiver will not reach the sender.

Since ATCP allows for interoperability with nodes not running ATCP, nodes without ATCP will see all the usual problems of running standard TCP over a mobile ad hoc network.

Another issue is that, when ATCP is in “loss” state, all unacknowledged packets in TCP’s buffers are retransmitted unnecessarily. Probably the receiver did not receive only one segment and it suffices if only that particular segment is re-transmitted. In an ad-hoc network, each mobile node is a router and the queue size of the mobile nodes is limited. Hence retransmitting all the packets in the TCP’s buffer could also lead to unnecessary congestion in the network.

Another drawback is that it consumes more power of all the intermediate nodes unnecessarily.

Possible Solutions

To handle scenarios when ECNs do not make it to the sender, a possible solution in [3] has been proposed. This is an enhancement that times-out only a certain number of

times and later gives control back to TCP. A threshold is maintained for this purpose. This threshold needs to be fixed through experimentation. Another possible enhancement is that when ATCP is in the “loss” state, only the lost packet be re-transmitted.

Other related TCP enhancements

Earlier research work in this area concentrated mostly on wireless networks with a fixed infrastructure and did not directly apply to ad hoc networks. However, recently, some research on improving TCP performance over mobile ad-hoc networks has also gained some momentum. Most of these works depend on notification or feedback from the network or a lower layer. They differ in how to obtain feedback and how to respond accordingly. Some of the other relevant enhancements include Explicit Link Failure Notification (ELFN), TCP-Feedback (TCP-F) and fixed RTO. We quickly introduce ELFN and TCP-F and provide some general thoughts on these schemes.

ELFN is described in [5]. This technique is also based on feedback. The main objective is to provide the TCP sender with information about link and route failures so it does not treat these failures similar to failures due to congestion. ELFN is based upon the DSR routing protocol. DSR is an on-demand routing protocol where the sender finds a route to the destination by flooding route request packets. To implement this technique the route failure message of DSR was modified to carry a payload similar to the ICMP message - "Host Unreachable". When a TCP sender receives an ELFN, it disables its retransmission timers and enters a stand-by mode. In the stand-by mode, the TCP sender periodically sends a packet to probe the network to see if a route has been established. When a new route is found, TCP leaves the stand-by mode, restores its transmission timers and resumes transmission as normal.

TCP-F is described in [4]. TCP-F relies on the network layer at intermediate hosts to detect route failures due to mobility of downstream neighbors along the route. A sender

can be in an active state or a snooze state. In the active state, transport is controlled by the normal TCP. As soon as an intermediate host detects a link failure, it explicitly sends a route failure notification (RFN) packet to the sender and records this event. After receiving the RFN, the sender goes into the snooze state by stopping sending further packets and freezing the values of state variables such as retransmission timer and congestion window size. The sender remains in the snooze state until it is notified of the restoration of the route through a route reestablishment notification (RRN) packet from an intermediate host. Then it enters the active state again.

Critics of this method mention that in TCP-F the source continues using the old congestion window size for the new route. This is a problem because the congestion window size route specific. It also does not consider the effects of congestion, out-of-order packets and bit error.

Conclusions and Discussions

Ad hoc networks have characteristics like high BER, frequent route re-computations, network partitions and multi-path routing that lead to bad TCP performance over these networks. This is mainly because TCP is a transport layer protocol designed for wire line networks. Older techniques that addressed issues of TCP performance over wireless networks did so by keeping the TCP sender unaware of the loss characteristics of the wireless link, and thus preventing those from affecting the congestion control mechanism of TCP. However, such techniques cannot be deployed over ad hoc networks because they require infrastructure support. They also did not address the issue of losses due to route failures.

Recent work suggests various transport layer mechanisms to solve the problems caused due to mobility. Some of the techniques for improving TCP performance over multi-hop mobile ad-hoc networks include explicit link failure notification technique (ELFN) and

TCP-Feedback. In this paper, I focused on a different technique however, called ATCP, which is implemented as a thin layer between IP and TCP.
It is easy to conclude that many components of TCP are not suitable for the characteristics of ad-hoc networks. Various reasons are discussed.
ATCP addresses many of the issues TCP faces when deployed over ad-hoc networks, and thus shows considerable performance improvement.

REFERENCES

[1] *Jian Liu and Suresh Singh*, “Atcp: Tcp for mobile ad hoc networks,” *IEEE J-SAC*, 2001.

[2] *Feng Wang and Yongguang Zhang*, "Improving TCP performance over mobile ad-hoc networks with out-of-order detection and response" International Conference on Mobile Computing and Networking. Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing

[3] *Vinay Sridhara and Nagendra Subramanya* “Evaluating Different Techniques to Improve TCP Performance over Wireless Ad Hoc Networks” www.eecis.udel.edu/~sridhara/wireless/ee650_paper.pdf

[4] K*. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, “*A feedback-based scheme for improving TCP performance in ad hoc wireless networks” IEEE Personal Communications Magazine, 8(1):34–39, Feb. 2001.

[5] G*. Holland and N. Vaidya, “*Analysis of TCP performance over mobile ad hoc networks” In Proceedings of ACM MobiCom’99, Seattle, Washington, Aug. 1999.

[6] *Karthikeyan Sundaresan , Vaidyanathan Anantharaman , Hung-Yun Hsieh , Raghupathy Sivakumar*, “ATP: a reliable transport protocol for ad-hoc networks” Proceedings of the fourth ACM international symposium on Mobile ad hoc networking & computing, June 01-03, 2003, Annapolis, Maryland, USA