Designing Reliable Delivery for Mobile Ad-hoc Networks in Robots

BJ Tiemessen
Department of Computer Science
211 University Services Center
601 South Howes Street
Colorado State University, Fort Collins, CO 80523
Tiemesse@cs.colostate.edu

Abstract

This study looks at implementing an ad-hoc network in mobile robots performing search and rescue operations. In a mobile ad-hoc network the topology may be changing frequently and each node can act as an end node but may also act as a router to forward packets. Nodes in a mobile ad-hoc network may experience periods of intermittent connectivity which presents new challenges in reliable delivery using Transmission Control Protocol (TCP). To improve reliable delivery and throughput using TCP I propose changing the window size and the packet size based on the distance between the sending and receiving nodes. My results show that changing the window size will increase throughput and decrease the number of lost packets.

1 Introduction

Performing search and rescue operations using autonomous robots presents many problems for researchers such as computer vision and object recognition, collision avoidance, and data communications. In this paper I will look at the data communications and more specifically issues with providing reliable message delivery for Mobile Ad-hoc Networks (MANET) in autonomous robots. A MANET is defined by its lack of infrastructure, mobility, limited bandwidth, and unreliable medium. The absence of a fixed infrastructure in a MANET means that each node performs peer-to-peer communication so nodes need to act as routers to pass messages between peers that are not directly connected. In a MANET the sending and receiving nodes are mobile as well as the intermediate routing nodes resulting in a frequently changing topology. Mobile nodes generally have limited resources, such as power and size, compared to nodes in a wired network. Due to the wireless nature of a MANET all communications in a given area contend for use of the shared channel which has a direct impact on bandwidth. Link reliability in a wireless environment is uncertain due to the mobility of nodes and signal strength that can be influenced by outside factors. These unique characteristics present challenges to consider when implementing reliable delivery in

a MANET.

To get a better understanding of the type of traffic and usage of the network in the robots I started by looking at what communications will be taking place. The robots will be performing search and rescue operations in a disaster area or an area which may compromise the safety of human rescue workers. The robots used for this study are designed to be autonomous and send reports to one or more laptop command center(s) which may take control of any of the robots. The command center may control any of the robots in a first person style by using a video feed and sending movement commands. Taking control of a robot using a first person fashion will require the robot to send streaming video to the command center which can use an unreliable transport such as User Datagram Protocol (UDP). The command center will need to send movement commands to the robot; this communication may use an unreliable transport as the command center will be receiving feedback of the robot's movement via video. The robots are ultimately performing search and rescue so when it believes it has achieved the goal state it will notify the command center and send a still image of its goal for further verification. The image of the suspected goal may need to be further analyzed by image analysis software so the image needs to be high resolution and complete. To ensure that the image will be received at the command center intact, it must use reliable transport. The robots may be fitted with temperature, radioactivity, or other such sensors to help assess the environment. Sensor readings will be logged on the individual robots but may also periodically be sent to the command center; this traffic can use an unreliable delivery unless the reading reaches a critical level. When a robot receives a sensor reading that is outside the acceptable range, its battery is low, is stuck, or has some other mission threatening issue it will send a critical message to the command center. The critical message must use reliable transport to make sure the command center is aware of the issue. The network implemented in the robots will need to provide a means of reliable delivery and be capable of transporting large amounts of data.

TCP (Transmission Control Protocol) is the most widely used transport protocol in the

Figure 1: Evolution Robotics ER1 Robot

Internet but was designed to be used on a wired network so it does not address the unique problems presented in a wireless environment. Previous research shows that TCP does not perform well in a MANET[5][4] [7] [12] [3] [8] [9]. In a pure Ad-hoc network, such as the one presented in this paper, no traffic will ever reach a wired network, so the best solution for reliable delivery may be to implement a new protocol to replace TCP. I chose to implement TCP as the reliable delivery mechanism in this study to better understand why TCP does not work well in a wireless environment and to look at what factors may improve its performance.

The robots in this study are laptop robots, meaning that the robot is nothing more than a frame with wheels, drive motors, power supply, motor controller and a laptop as shown in Figure 1. The laptop uses USB interfaces to communicate with the motor controller to provide movement for the robot. Sight for the robots is accomplished by a USB camera connected to the laptop. Communications for the robots takes place by a standard 802.11 based wireless networking card attached to the laptop running in ad-hoc mode.

The rest of this paper is organized as follows. My definition of states is explained in Section 2. The simulation[1] and methodology is provided in Section 3. I discuss factors of TCP performance in Section 4. My results are presented in Section 5 and Section 6 concludes the paper.

2 Definition of States

In order to better evaluate reliable delivery in a MANET I define three states of transmission in a mobile node. I define the states disconnected, steady, and unsteady as follows:

- Disconnected - The disconnected state is where a sending or receiving node has no other node within radio range or there exists no physical route between sender and receiver.
- Steady - The steady state is defined by the ability to transmit data between a sending a receiving node. During the steady state there exists a valid, but not necessarily optimal, route between a pair of nodes wishing to communicate.
- Unsteady - The unsteady state is defined by the existence of a physical link between communicating nodes but the absence of a valid route.

It is important to define these states to get an accurate evaluation of simulation results. My research focuses on performance loss in TCP and the causes of resulting loss. There are many factors which may cause delay and jitter of TCP communications. I expect to have no communication in the disconnected state and a decrease or possibly no communication during the unsteady state. With a clear definition of these states I can better evaluate the effects on TCP in state transitions and within the states. Figure 2 shows a plot of TCP throughput with vertical lines drawn at the time in which an unsteady state occurs.

3 Methodology

I use the network simulator ns2 for all simulations. The topology used for the simulations is a $3200m \times 300m$ grid with 15 stationary nodes $200m$ apart. A single mobile node starts $195m$ north of the first node in the chain and travels east at a rate of $0.5m/s$. The speed chosen for the mobile node comes from the maximum speed of the robots used. This topology

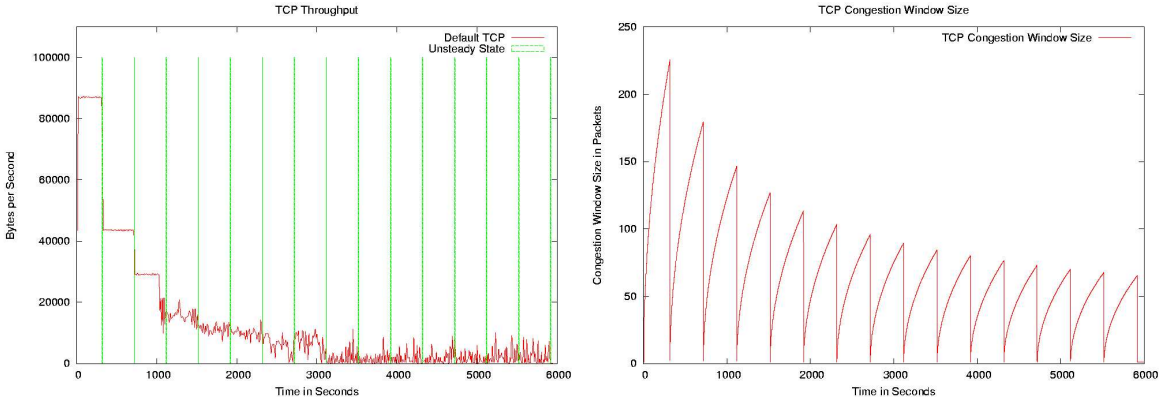

Figure 2: Left: Default TCP new reno throughput. The vertical lines indicate unsteady state. Right: TCP Congestion Window Size

allows us to look at a 1-15 hop chain as well as the effects of route changes in a single experiment. As the mobile node starts to move, it starts a TCP session with the first node in the chain. The traffic is generated by the FTP traffic agent in ns2. Unless otherwise noted I use AODV[2] routing protocol and TCP new reno for the transport protocol. The measurements I use to assess TCP performance are throughput and data received. Throughput is calculated by the bytes of data successfully received divided by the time slice, unless otherwise noted, the default time slice used is $20s$. The data received graphs show the cumulative sum of all data successfully received.

4 TCP Performance

Factors such as mobility, the 802.11 MAC[3] layer, hop length, packet size and window size may cause TCP to perform differently from a wired environment. This section will briefly touch on these factors.

4.1 Mobility

Mobility of a node can cause link disconnection and route changes which affect the flow of data. Link disconnection is an austere concept so I will focus on route changes

caused by mobility. For my simulations I used the AODV routing protocol and no nodes are disconnected until the last few seconds of the simulation when the sender is out of range. As the mobile node reaches the edge of its transmission range to its shortest hop closest neighbor, AODV must update its routes. During the AODV routing update, there is a short period of time that there is no route between the sender and receiver. At this time I notice that the congestion window drops to the initial window value of one. Upon further investigation of the trace files I observe that an ACK[4] is lost so the sending node times out and resends the packet. This retransmitted packet also gets dropped while AODV is updating the route between sender and receiver. The third time the packet is transmitted, AODV finally has a valid route. TCP reset the congestion window to the initial value of 1 during the AODV routing update.

4.2 MAC Layer

In wireless communication, all nodes use the same channel so if more than one node transmits at the same time the transmissions will collide and the receiving node will hear a garbled message. To combat channel contention each packet transmission in IEEE 802.11 communications is proceeded by an RTS/CTS (Request to Send/Clear to Send) handshake. The node wanting to transmit a packet will listen on the channel to make sure there is no other traffic and if it is clear, will send an RTS message. When the receiving node hears the RTS message it will make sure the channel is clear and send a CTS message. This handshake notifies all other nodes in the carrier sense range that it is not clear to transmit data. Current hardware specifies that a node has a transmission range of about 250m and has a carrier sense range of about 550m. The channel contention at the MAC layer essentially makes TCP perform similarly to a stop-and-wait protocol. TCP may send more than one packet but the MAC layer will wait until the channel is clear after sending a packet to send successive packets.

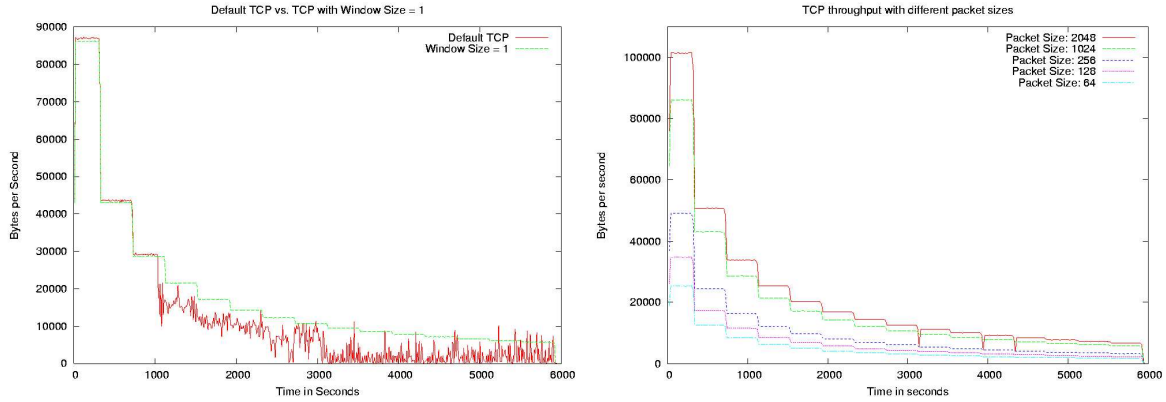

Figure 3: Left: A comparison of default TCP throughput and throughput of TCP with a window size of 1. Right: The effect on TCP throughput by changing the packet size

4.3 Hop Length

In a chain topology the packets travel down the chain hop by hop toward the receiver. As the length of the chain grows or the number of hops grow, each node must contend with other nodes for the channel. My topology is similar to that in [6] with nodes spaced $200m$ apart although I use a mobile node that changes my results slightly. In [6] they claim that the maximum concurrent senders in an h-hop chain can be $h/4$ but my results indicate that the window needs to be decreased as the hop count increases. My findings on window size will be discussed further in Section 4.4 and Section 5.

4.4 Window Size

Channel contention, MAC traffic, routing convergence delay, and congestion in a wireless ad-hoc network can cause packets to be lost or time out which will cause TCP to adapt its window size. Using standard TCP with an unbounded window I observe that the congestion window grows very large for the first 3 hops but time outs and duplicate ACKs keep the window size small for all successive hops. During the steady state of the first 3 hops the window continues to grow until the unsteady state where the window gets reset to 1.

TCP Settings	Data Received (Bytes)
Packet Size 1024 Max Window 1	106497640
Packet Size 1024 Max Window 2	103156752
Packet Size 1024 Max Window 3	95065776
Packet Size 1024 Default Window	80542344
Packet Size 512 Default Window	68925488

Figure 4: Table of Total Data Received at Node 0 in bytes

4.5 Packet Size

Packet size turns out to be a big factor in throughput of TCP traffic as shown in Figure 3. Channel contention and MAC layer RTS/CTS essentially makes TCP perform as a stop-and-wait protocol. The stop-and-wait behavior limits the number of packets that can be sent over the channel, so increasing the packet size allows more data per packet to be sent. While increasing the packet size increases throughput it can also have negative effects. A large packet size means that the transport layer is keeping the channel busy for a longer period of time and further increases channel contention for the MAC layer and routing layer messages. A large packet size also means that if a packet is lost the retransmitted packet is large, which will hinder further data communication for a longer period of time as opposed to a smaller packet size.

5 Results

I began by looking at the behavior of default TCP new reno. I noticed that there was a nice steady throughput to TCP for the first 3 hops, followed by many large spikes and valleys in the throughput of the successive nodes shown in Figure 2. Looking at the congestion window behavior of TCP with default settings I noticed that during the steady state the window would drop to 1 indicating a time out. To study this time out more I looked at the unsteady state between the 1 hop and 2 hop steady states. The TCP round trip time almost doubles from the 1 hop steady state compared to the 2 hop steady state. I

thought that this sudden change in round trip time might be the cause of the time out but further investigation shows that for this unsteady state it takes AODV close to 2 seconds to converge on a new route.

The initial simulation of TCP using default settings indicated that there was a serious problem after about 8 hops as throughput was suffering. I concluded that channel contention was hindering throughput and that the TCP source was overloading the network with traffic. I decided that the sender needed to restrict the amount of data being propagated into the network so I clamped the window size. After using many different window sizes and even window sizes from previous work such as in [10] I noticed that as the hop count grows, the window size needs to decrease. My simulations show that clamping the window size to 1 produces the best overall throughput. A very interesting observation that I made is that for a single hop TCP with an unbounded window and TCP with a window size of 1 have almost identical throughput. After investigating this behavior further I determined that the MAC layer causes TCP to perform similar to a stop-and-wait protocol so the increased window size does not change the throughput. Besides increasing total throughput of TCP by clamping the window size to 1 I also reduced delay and jitter. The graphs of TCP using a window size of 1 produce a nice clean stair step pattern that can be easily predicted. I observe that the throughput for n hops equals

$$T_{h1} \times \frac{1}{n} \tag{1}$$

where T_{h1} is the measured throughput of the first single hop from sender to receiver, the first steady state.

My results show that decreasing the window of TCP for increasing hop count improves throughput so I want to try to decrease the window further. In a wired network decreasing the packet size should effectively decrease the window size of TCP. My theory was that if I could effectively reduce the TCP window using a smaller packet size I could then increase the senders window size in an attempt to further increase the throughput. I ran many simulations with varying packet sizes shown in Figure 3. My results show that for a packet

size s with a window size of 2 I could increase the throughput by increasing s and lowering the window size to 1. Packet size does not effectively lower the window size of TCP in a wireless environment because of channel contention and the MAC layer. Increasing the packet size in a wireless network allows the sending node to keep control of the channel for a longer amount of time resulting in increased data transmission. Figure 4 shows that for a given packet size a window size of 1 produces the largest data throughput.

6 Conclusions and Further Work

Packet size and TCP window size have a large impact on throughput in a wireless environment. My results show that increasing the packet size will increase throughput by allowing a node to control the channel for a longer period of time. Clamping the TCP window to 1 reduces the number of dropped packets and reduces channel contention that occurs when the network is overloaded. For networks with a small hop count there was nearly no change in throughput between an unbounded TCP window and a window size of 1. For networks with a large hop count I increased the throughput by clamping the window size to 1. By changing parameters of TCP I am able to achieve adequate reliable delivery for my robot network. Although I am able to improve TCP performance this study did not address issues such as the effects of TCP in the disconnected state for long periods of time. In wired networks TCP performs well but the best solution for reliable delivery in a MANET may be to develop a new protocol that will address the issues which are unique to a MANET.

7 Acknowledgments

I would like to thank Nischal Piratla for his contribution of ideas and graph analysis. I would also like thank the McNair Scholars program and everyone involved with this program for making this research possible.

List of Figures

1 Evolution Robotics ER1 Robot 4

2 Left: Default TCP new reno throughput. The vertical lines indicate unsteady state. Right: TCP Congestion Window Size 6

3 Left: A comparison of default TCP throughput and throughput of TCP with a window size of 1. Right: The effect on TCP throughput by changing the packet size 8

4 Table of Total Data Received at Node 0 in bytes 9

Notes

[1] All simulations are performed in ns2

[2] AODV - Ad-hoc On-Demand Distance Vector routing

[3] MAC - Medium Access Control, for this paper we only consider the 802.11 MAC layer

[4] ACK - A TCP acknowlegement

References

[1] Biswas, S., and Morris, R. Opportunistic routing in multi-hop wireless networks. In *ACM Workshop on Hot Topics in Networks* (Boston, MA, USA, November 2003).

[2] Broch, J., Maltz, D., Johnson, D., Hu, Y., and Jetcheva, J. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings MobiCom '98* (Dallas, TX, October 1998).

[3] Das, C. P. E. R. S., and Marina, M. Performance comparison of two on-demand routing protocols for ad hoc networks. *IEEE Personal Communications* (February 2001), 16–28.

[4] Fu, Z., Greenstein, G., Meng, X., and Lu, S. Design and implementation of tcp-friendly transport protocol for ad hoc wireless networks. In *Proceedings IEEE ICNP'02* (Paris, France, 2002).

[5] Fu, Z., Meng, X., and Lu, S. How bad tcp can perform in mobile ad hoc networks. In *Proceedings of 7th IEEE ISCC* (2002).

[6] Fu, Z., Zerfos, P., Luo, H., Lu, S., Zhang, L., and Gerla, M. The impact of multihop wireless channel on tcp throughput and loss. In *Proceedings of IEEE InfoCom'03* (San Francisco, CA, April 2003).

[7] Gerla, M., Tang, K., and Bagrodia, R. Tcp performance in wireless multi-hop networks. In *Proceedings of IEEE WMCSA'99* (1999).

[8] Holland, G., and Vaidya, N. Analysis of tcp performace over mobile ad hoc networks. In *Proceedings of ACM MobiCom'99* (Seattle, WA, August 1999).

[9] Hsieh, K. S. V. A. H., and Sivakumar, R. Atp: A reliable transport protocol for ad-hoc networks. In *MobiHoc '03* (Annapolis, Maryland, USA, June 1-3 2003).

[10] Johansson, P., Larsson, T., Hedman, N., Mielczarek, B., and Degermark, M. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Mobicom '99* (1999), pp. 195–206.

[11] Johnson, D. Routing in ad hoc networks of mobile hosts. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications* (December 1994).

[12] Kawadia, V., and Kumar, P. Experimental investigations into tcp performance over wireless multihop networks. In *SigComm'05 Workshops* (Philadelphia, PA, August 2005).

[13] Ke, Q., Maltz, D., and Johnson, D. Emulation of multi-hop wireless ad hoc networks. In *Seventh International Workshop on Mobile Multimedia Communications* (October 2000).

[14] Khurana, S., Kahol, A., and Jayasumana, A. Effect of hidden terminals on the performance of ieee 802.11 mac protocol. In *Proceedings of the 23rd Annual Conference on Local Computer Networks* (1998).

[15] Ye, W., Vaughan, R., Sukhatme, G., Heidermann, J., Estrin, D., and Mataric, M. Evaluating control strategies for wireless-networked robots using an integrated robot and network simulation. In *Proceedings of the 2001 IEEE International Conference on Robotics and Authomation* (May 21-26 2001), vol. vol. 3, pp. pp. 2941–2947.

[16] Zhu, C., and Carson, M. Qos routing for mobile ad hoc networks. In *Proceedings INFOCOM 2002* (New York, 2002).