Venkatesh.Donepudi, vahiduddinshariff / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 3, Issue 2, March -April 2013, pp.242-252

Measurement and Modeling of the congestion-controlled traffic over CSMA based multihop Wireless Mesh Networks

Venkatesh.Donepudi, vahiduddinshariff

Abstract

wireless mesh network is a well suited future technology. Its properties like low cost, high bandwidth and *significant* progress has been made in understandingthe behavior of TCP and congestion-controlled traffic over CSMA based multihop wireless networks. Despite these advances, however, no prior work identified severe throughput imbalances in the basic scenario of mesh networks, in which a one-hop flow contends with a two-hop flow for gateway access. In this paper, we demonstrate via real network measurements, testbed experiments, and an analytical model that starvation exists in such a scenario; i.e., the one-hop flow receives most of the bandwidth, while the two-hop flow starves. Our analytical model yields a solution consisting of a simple contention window policy that can be implemented via standard mechanisms defined in IEEE 802.11e. Despite its simplicity, we demonstrate through analysis, experiments, and simulations that the policy has a powerful effect on network-wide behavior, shifting the network's queuing points, mitigating problematic MAC and transport behavior, and ensuring that TCP flows obtain a fair share of the gateway bandwidth, irrespective of their spatial location.

Introduction

Wireless Mesh Networks(WMN) are next generation wireless networks with theirpromising and low-cost technology. It provides high-speed Internet access forfuture broadband applications. WMNs are flexible, mobile, reliable and scalablewireless networks. WMNs reduce the initial investment and deployment time compared to traditional broadband Internet access technologies [1]. In WMNs802.11 based multi-hop communication is used for delivering fast services to end-users,WMNs are not used for any fixed Access point networks likeWireless LocalArea Networks(WLAN).WMNs are extremely reliable with its mesh connection.If any node fails or drop of packet in the network occurs, it uses another routeas activce route.Wireless mesh networks (WMN) is an emerging technology. Existing definitions are simply based on Mobile Ad hoc Networks (MANETs), which arevariants of WLAN. WMNs are considered as special MANETS [3,5]. Wireless Mesh Networks(WMNs) are self-organized, self-configured, ease and can rapidity change with the network deployment [2, 4, 5]. Wireless Mesh Network consists of three types of nodes. Mesh Points(MP), Mesh Routers(MR), andMesh Clients(MC). In WMN, some Access Points(APs) have wired connections known as mesh points(MPs). APs that don't have wired connections arecalled Mesh Routers(MRs), they can form multi-hop communication with theMPs. Both MPs and MR form the backhaul network, which is used to forwardtraffic between the nodes. MP and MR are usually equipped with multiple wireless interfaces built on either same or different wireless technologies and theyestablish a permanent infrastructure. New MP and MR are easily added, sincethe connection is wireless connection [5]. Compared to conventional wireless routers, a mesh router can achieve the same radio range with much lower transmission power through multi-hop communications.Mesh router and conventionalrouters are usually built based on a similar hardware platform [5]. MPs and MRshave the similar design but MPs are connected to wired networks. Mesh Clients(MC) usually have only one wireless interface and hardware as well as softwareare much simpler than MPs or MRs. MCs do not have routing capabilities andworks as Mesh Router(MR) and as a router [4, 5]. Due to high mobility of meshclients they can leave and join the Mesh Router(MR) at any time [5].There is an increasing need towards portable and mobile computers or workstations with the development of wireless technology and Internet. Wireless networks need to provide communications between mobile terminals, in addition, access to high speed wired networks needs to be provided too. Wireless

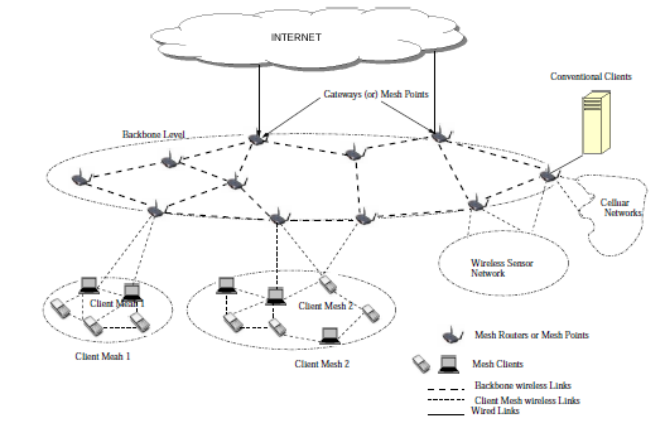

Figure 1: Wireless Mesh Networks

Local Area Networks(WLANs), which provides better flexibility and convenience than their wired counterpart, are being developed to provide high bandwidth access for users in a limited geographical area. IEEE Project 802 recommends an international standard 802.11 for WLANs. The standards include detailed specifications both for MediumAccess Control(MAC) Layer and Physical(PHY) Layer. In WLANs, the physical media, which is shared by allstations and has limited connection range, has significant differences when compared to wired media. The design ofWLAN MAC protocol is further complicated by the presence of hidden terminal and capture effects. Currently, the IEEE 802.11 WLAN standards include a basic medium access protocol Distributed Coordination Function(DCF) and an optional Point Coordination Function(PCF). In 802.11, the DCF is the fundamental access method used to support asynchronous data transfer on a best effort basis. As specified in the standards, the DCF must be supported by all the stations in a basic service set(BSS). The DCF protocol is based on Carrier Sense Multiple Access with Collision Avoidance(CSMA/CA). CSMA/CD is not used because a station is unable to listen to the channel for collision while transmitting. In 802.11 CS is performed both at physical layer, which is also referred to as physical carrier sensing, and at the MAC layer, which is known as virtual carrier sensing. The PCF in the 802.11 is a polling-based protocol, which is designed to support collision free and real time services. This paper focuses on the performance analysis and modeling of DCF in 802.11 WLAN. There are two techniques used for packet transmitting in DCF. The default one is a two-way handshaking mechanism, also known as basic access method. A positive MAC acknowledgement(ACK) is transmitted by the destination station to confirm the successful packettransmission. The other optional one is a four-way handshaking mechanism, which uses request-to-send/clear-to-send(RTS/CTS) technique to reserve the channel before data transmission. This technique has been introduced to reduce the performance degradation due to hidden terminals. However, the drawback of using the RTS/CTS mechanism is increased overhead for short data frames. The modeling of 802.11 has been a research focus since thestandards has been proposed. In this,the effect of capture and hidden terminal and paper gives the theoretical throughput limit of 802.11 based on a *p*-persistent variant. However, none of these captures the effect of the Contention Window(CW) and binary slotted exponential back-off procedure used by DCF in 802.11. Unlike those ones, we use Markov process to analyze the saturated throughput of 802.11 and show that the Markov analysis works well. We believe that the Markov chain analysis method is fit for examining the performance of IEEE 802.11, which is based on binary slotted exponential backoff. In this also uses Markov chain and considers the frame retry limits to analyze the saturated throughput; therefore, a more exact model isproposed in this thesis. On the other hand, with the prosperity of Internet, Transport Control Protocol(TCP), which is the widely used reliable protocol in the Internet, is supposed to work well in heterogeneous environment. Since the WLAN MAC has its own characteristics, such as MAC Layer ACK frame, MAC retransmissions, which are different from traditional wireless medium, the performance evaluation and enhancement will be somewhat different from the research before. The performance of TCP over WLAN is being studied recently; however, none of these give a TCP performance enhancement based on the WLAN MAC layer solutions. In fact, when TCP runs over WLAN, where a shared channel is used for multiple access, the forward TCP data and the backward TCP ACK will compete the channel, which may cause collisions and degrade the overall performance. Meanwhile, 802.11 has been standardized and any proposed enhancement scheme must keep backward compatibility with 802.11, i.e., it can work with 802.11 without introducing performance degradation.

Large-scale mesh network deployments are planned and underway in cities across the world. According to In-Stat, the market will grow from 248 cities in 2005 to 1,500 cities in 2010, becoming a $1B industry in mesh access point sales alone. The prevailing architecture for large-scale deployments is a two-tier architecture in which an access tier connects end users' PCs and mobile devices to mesh nodes and a backhaul tier forwards traffic to and from high-speed gateway nodes. Directional antennas terminating at the gateway are also used to expand coverage and capacity. We have deployed and are operating UrbanMesh,1 a two-tier mesh network serving a user population of nearly 2,000 users in a 3 *km*2 urban community. As most deployments are in the planning or early deployment stages, to the best of our knowledge, this is the highest density urban mesh network operating to date. Unfortunately, we have observed that under heavy load, flow *starvation* will occur in which one-hop flows obtain high throughput whereas competing multi-hop flows obtain near zero throughput. The phenomena occurs under two hardware/software platforms as well as in simulation. Clearly, for mesh networks to be successful, it is critical that network resources are distributed fairly among users, irrespective of their spatial location. In this paper, we (i) perform extensive measurements in UrbanMesh to characterize starvation, (ii) studystarvation's protocol origins, (iii) develop an analytical model to capture the interaction of medium access and

congestion control that leads to starvation and (iv) design, justify, and evaluate a counter-starvation

Policy in which nodes one-hop away from the gateway increase their minimum contention window. In particular, our contributions are as follows. First, we experimentally demonstrate the existence of starvation in TCP wireless mesh networks. Moreover, we design a set of experiments to isolate the factors that cause starvation. We identify that only a one-hop TCP flow coupled with a two-hop TCP flow is sufficient to induce starvation. Moreover, we demonstrate that starvation is not merely a hidden-terminal effect by showing that starvation does *not* occur if the TCP flows are replaced by UDP flows and IEEE 802.11's RTS/CTS handshake is used to counter hidden terminals; yet, the use of RTS/CTS is irrelevant for TCP flow starvation. Second, we describe the protocol origins of starvation as a compounding effect of three factors. First,the medium access protocol induces bi-stability in which pairs of nodes alternate in capturing systemresources. Second, the system's nested congestion control loops utilizing the wireless medium make it more likely that outer loops (multi-hop flows) are disrupted rather than inner loops (single-hop flows). Third, and most critically, the system incurs a high penalty when switching between the two states of the bi-stable system. In particular, a state is normally exited when either a flow "runs out of ACKs" and thus cannot transmit data because it cannot receive feedback, or when a DATA packet is dropped at themedium access layer. Third, we develop an analytical model to both study starvation and to drive the solution to counter starvation. The model omits many intricacies of the system (TCP slow start, fading channels, channel coherence time, etc.) and instead focuses on the minimal elements needed such that starvation manifests. Namely, the model uses a discrete-time Markov chain embedded over continuous time to capture a *fixed* end-to-end congestion window, a carrier sense protocol with or without RTS/CTS, and all end-point and intermediate queues. The model yields a *Counter-Starvation Policy* in which all of the gateway's neighbors should increase their minimum contention window to a value significantly greater than that of other nodes.2 The model also characterizes *why* the policy is effective in that it forces all queueing to occur at the gateway's one-hop neighbors rather than elsewhere. Because these nodes have a perfect channel view of both the gateway and their neighbors that are two hops away from the gateway, bi-stability is eliminated such that the subsequent penalties are not incurred. Finally, we validate the Counter-Starvation Policy by re-deploying aanageable set of MirrorMesh nodes on-site (mirroring a subset of the wireless mesh nodes). The results demonstrate that our solution completely solves the starvation problem for TCP upstream and downstream traffic. We extend our investigation to a broader set of scenarios using simulations and show that our solution enables TCP flows to fairly share the gateway bandwidth in more general scenarios.Wireless mesh backhaul nodes are predominantly deployed on single-story residences with the exception of two schools, two businesses, and a public library within the neighborhood. The current mesh infrastructure is composed of 18 backhaul nodes which coordinate to share the Internet bandwidth from a single fiber that is burstable to 100 Mbps. Fig. 1 depicts the spatial distribution of the wireless mesh backhaul tier and the connectivity map. In the figure, nodes are depicted as connected if a direct transmission can occur between the two backhaul nodes. All links are omni-directional with the exception of a directional link(shown in black) which serves as an additional point of capacity for the network. Each mesh node serves access nodes or clients wirelessly within its immediate proximity of approximately 200-300 m in radius. The radius is limited primarily by heavy tree foliage and densely packed homes with lot sizes averaging only 510 m^2 . At the time of our experiments, there are nearly 2,000 users in the network in an area of nearly 3 Km^2 .,i.e., approximately 600 users and 6 backhaul nodes per Km^2., which makes wireless mesh one of the most dense mesh networks operating to date; for comparison, the St. Cloud mesh network currently serves 120 users per Km^2.. There are 4,760 residents per Km^2. in the served neighborhood; as a point of reference, the average population density of the 20 largest U.S. cities is 2,800 residents per Km^2 .. Planned and deployed commercial mesh networks also employ this two-tier architecture in which Internet gateways feed multiple multi-hop wireless paths and directional links are used to provide high-speed "short-cuts" to the gateway for capacity improvement. The process is then duplicated every 3-10 Km^2. for larger coverage areas.

Wireless mesh networks (WMNs) are being deployed to provide low-cost high-bandwidth Internet access on a large scale. In a WMN, stationary wireless mesh nodes (or routers) provide network connectivity for mobile wireless mesh clients. Multiple wireless mesh nodes are interconnected to form a wireless backbone. Several mesh nodes also serve as the gateways (GW), which connect to the Internet. The mesh architecture, in which all mesh nodes participate in creating an infrastructure for decentralized data transmission, makes a WMN a stable and economically viable solution for providing universal Internet access. In most WMN usage scenarios, the available bandwidth must be utilized in a fair manner by all mesh nodes. Unfortunately, the carrier sense multiple access with collision avoidance

(CSMA/CA) medium access control (MAC) protocol in IEEE 802.11 can lead to unfairness in multi-hop data transmission in WMNs. In addition, in the transport layer, the transmission control protocol (TCP) does not account for fairness among users and sometimes may lead to starvation of some nodes depending upon the location of nodes and their distance from the GW. An undesirable condition is the flow starvation which can cause some flows to attain extremely low data transfer rates compared to other flows. Starvation in wireless networks has been observed in some earlier studies. The poor performance of CSMA/CA-based MAC protocols has been considered through simulations and analytical modeling. It has been shown that when senders observe different channel conditions compared to theirneighbors, some of them can encounter flow starvation. The existence of unfairness in single hop networks, where any two communicating nodes are within transmission range of each other, was reported in [4]. The work in [5] classified various two-flow scenarios into twelve classes and investigated theshort-term and long-term performance in each class. Similar studies for multi-hop networks can be found in [6]–[8]. The work in [9] identified bi-stability as a cause for starvation in WMNs. In a linear topology, bi-stability occurs when nodes closer to the GW enter a phase of successful packet transmissions that forces other nodes to wait for an extended period of time. The work in [9] is closely related to ourwork in this paper. They analytically determined the existence of starvation in a multi-hop network with linear topology, which they called the basic topology. Analytical modeling was carried out for the two-node scenario but not for the three-node scenario. No consideration was given to possible solutions to the exposed terminal problem. Furthermore, they considered the interference model where nodes that are two hops away are outside each other's transmission range and carrier sensing range. Also, another work which is closely related to our work is Idle Sense [10], which adjusts the contention window of the IEEE 802.11 MAC protocol to achieve fairness in wireless local area networks (WLANs). Idle Sense, however, is suitable for single hop WLANs only and does not consider the effect of data flows with multiple hops.In [11] and [12], it has been argued that modifications to the transport layer can provide fairness despite unfair MAC protocols. Multi-channel protocols have also been considered to mitigate starvation [13], as well as rate control techniques [14], multi-path solutions [15], and optimization-based approaches [16]. All these methods are relevant but we believe that solutions at the MAC layer are central to performanceenhancement and we suggest improvements at the MAC layer to limit starvation. In this paper, we establish the existence of, and analyze the extent of, starvation in two-hop and three-hop scenarios where a node is at most two or three hops from the GW. Specifically, we consider the interference model where some adjacent nodes are within carrier sensing range but are not within transmission range of each other. This is a common situation when a WMN is employed to provide network connectivity in dense environments such as a university campus. The contributions of this paper are as follows: 1) We propose a simplified Markov chain model to analyze WMNs with linear topology and numerically compute the degree of unfairness between nodes. 2) We propose a fair binary exponential backoff (FBEB) algorithm to reduce the extent of starvation. Our proposed algorithm uses the intuition that, to improve fairness, mesh nodes that have successfully transmitted a data packet should not be permitted to eagerly transmit more data packets. By delaying the transmission of successive packets we are able to reduce the degree of starvation. This effect is achieved by adapting the contention window for transmissions in the IEEE 802.11 protocol. 3) Our analytical results show that solving the exposed terminal problem alone is insufficient for combating starvation in WMNs. 4) Simulation results also show that our proposed FBEBalgorithm can improve the data rate of starving nodes by a factor of 7 in some cases. Fairness is achieved by limiting the aggressive use of the channel; the overall loss in throughput across all nodes was observed to be about 20%. The loss in overall throughput is, however, acceptable if fairness is a key parameter in WMNs. We also show through our simulations that our proposed FBEB algorithm outperforms other schemes [9], [10] under the 2-hop carrier sense assumption. Our work presumes a linear topology (or a chain) for mesh nodes. This is an elementary structure and understanding thebehavior of nodes arranged in such a topology provides insight into the behavior of larger networks, which could be studied as a collection of chains.

2. Related Work

2.1 Wireless Networks

Multi-hop wireless ad hoc networks offer communications capability to mobile hosts without requiring a fixed infrastructure. In such a network, packets can traverse multiple intermediate nodes en route from the source to the destination. An example of a multi-hop wireless network is shown in Figure 2. This example shows four nodes (labeled A, B, C, D) in a simple \chain" network topology. Many other topologies are possible, though we consider only the simple chain network in this paper. The topologyis called multi-hop because (for example) a packet destined from A to D must use B and C as intermediate routers. Each forwarding step is called a hop. The overall performance achieved within a multi-hop wireless network depends on the Medium Access Control (MAC) protocol and transport protocol used. For mobile nodes, performance also

depends on mobility patterns and the ad hoc routing protocol used. A popular MAC protocol for wireless networks is the IEEE 802.11 MAC [2]. This protocol requires each node to sense the channel before sending a frame. The protocol is called CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance). To address the hidden node problem, the 802.11 MAC uses a Request-To-Send (RTS) and Clear-To-Send (CTS) handshake. A node sends an RTS control frame to indicate that it has a frame to send. Upon receiving the RTS, the intended receiver returns a CTS control frames if it is okay to receive the data frame. In this fashion, a packet traverses one hop at a time toward the destination.

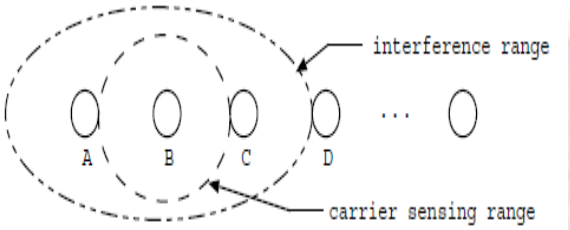

Figure 2 –Multi-hop wireless Network
TCP (Transport Control Protocol)

TCP performance in multi-hop wireless networks is poor [5, 12, 16]. TCP throughput often decreases dramatically with the number of hops traversed by a flow, regardless of the MAC protocol used [5, 12]. The primary reason is link-layer packet losses caused by contention between data packets traveling in the same direction, and collisions between data packets and TCP ACK packets traveling in opposite directions. Several methods have been proposed to remedy these problems. Fu et al. [5] propose a link-layer version of RED [4] to signal the TCP sender about impending congestion, and an adaptive pacing algorithm to distribute TCP data packets evenly across a multi-hop chain. Combined, these algorithms improve throughput by 5%-30%. As another example, Cordeiro et al. [3] propose disjoint routes for forward TCP packets and backward TCP ACKs so that contention is reduced. They report throughput improvements of 90%. Several multi-channel MAC protocols have been proposed to improve the overall ad hoc network capacity [6, 7, 11, 13, 15]. We do not consider multi-channel
approaches in this paper, but we have studied them in prior work [9]

2.2 TCP Protocols

There have been two main approaches to detecting congestion before router buffers overflow: end-to-end methods and router-based mechanisms. End-to-end methods are focused on making changes to TCP Reno. Most of these approaches try to detect and react to congestion earlier than TCP Reno (*i.e.*, before packet loss occurs) by monitoring the network using end-to-end measurements. Router-based mechanisms, such as AQM, make changes to the routers so that they notify senders when congestion is occurring but before packets are dropped.

2.2.1 End-to-End Approaches

TCP/Vegas:

Vegas have mainly showed that it can achieve 40%-70% better throughput than Reno. However, this throughput is achieved not by aggressive retransmissions, but of efficient use of the available bandwidth. Vegas proposed three important techniques to improves throughput and reduce losses.

(1) New Retransmission Mechanism: Vegas estimate the RTT and variance using a fine-grained timer. Vegas read and record the system clock each time a segment is sent. This helps to estimate when to retransmit a lost packet. Also Vegas only decreases the congestion window if the retransmitted segment was previously sent after the last decrease. Whereas, in Reno it is possible to decrease the congestion window more than once for losses that occurred during one RTT interval.

(2) Congestion avoidance mechanism:Thesimple idea that Vegas exploits is thatnumber ofbytes in transit is directlyproportional to the expected throughput, and therefore, as the window size increases the bytes in transit increases resulting in increased throughput of the connection. The goal of Vegas is to maintain the "right" amount of extra data in the network. Vegas congestion avoidance actions are based on changes in the estimated amount of extra data in the network, and not only on the dropped packets.

Expectedthroughput = windowSize/BaseRTTBaseRTT is set to the minimum of all measured round trip times. WindowSize is the size of the current congestion windowModified Slow-Start Mechanism: Vegas uses rate control during slow-start, using a rate defined by thecurrent window size and the BaseRTT.

2.2.2 Router-based approaches

Random Early Detection: Random Early Detection (RED) is an AQM mechanism that seeks to reduce the long-term average queue length in routers. In RED, as each packet arrives, routers compute a weighted average queue length that is used to determine when to notify end systems of incipient congestion. "Marking" a packet performs congestion notification in RED.

While TCP/Vegas also use delay-based congestion control in an effort to increase TCP throughput due to reduced number of packet losses and timeouts, and a reduced level of congestion over

the path. In contrast, TCP-LP uses one-way delay measurements vs. round-trip delays. Moreover, the key difference between TCP-LP and RTT-based 5 congestion control protocols is in their primary objective. While the former aim to achieve fair-share rate allocations, TCP-LP aims to utilize only excess bandwidth.

• **Link-layer protocols**: There have been several proposals for reliable link-layer protocols. The two main classes of techniques employed by these protocols are: error correction (using techniques such as forward error correction (FEC)), and retransmission of lost packets in response to automatic repeat request (ARQ) messages. The link-layer protocols for the digital cellular systems in the U.S. — both CDMA [10] and TDMA [17] — primarily use ARQ techniques. While the TDMA protocol guarantees reliable, in-order delivery of link-layer frames, the CDMA protocol only makes a limited attempt and leaves it to the (reliable) transport layer to recover from errors in the worst case. The AIRMAIL protocol [1] employs a combination of FEC and ARQ techniques for loss recovery. The main advantage of employing a link-layer protocol for loss recovery is that it fits naturally into the layered structure of network protocols. The link-layer protocol operates independently of higher-layer protocols (which makes it applicable to a wide range of scenarios), and consequently, does not maintain any per-connection state. The main concern about link-layer protocols is the possibility of adverse effect on certain transport-layer protocols such as TCP. We investigate this in detail in our experiments.

• **Indirect-TCP (I-TCP) protocol:** This was one of the early protocols to use the split-connection approach.It involves splitting each TCP connection between a sender and receiver into two separate connections at thebase station — one TCP connection between the sender and the base station, and the other between the base stationand the receiver. In our classification of protocols, ITCP is a split-connection solution that uses regular TCPfor its connection over wireless link. I-TCP, like other split-connection proposals, attempts to separate loss recovery over the wireless link from that across the wireline network, thereby shielding the original TCP sender from the wireless link. However, as our experiments indicate the choice of TCP over the wireless link results in several performance problems. Since TCP is not well-tuned for the lossy link, the TCP sender of the wireless connection often times out, causing the original sender to stall. In addition, every packet incurs the overhead of going through TCP protocol processing twice at the base station (as compared to zero times for a non-split-connection approach), although extra copies are avoided by an efficient kernel implementation. Another disadvantage of this approach is that the end-to-end semantics of TCP acknowledgments is violated, since acknowledgments to packets can now reach the source even before the packets actually reach the mobile host. Also, since this protocol maintains a significant amount of state at the base station per TCP connection, handoff procedures tend to be complicated and slow.

• **The Snoop Protocol:** The snoop protocol introduces a module, called the *snoop agent*, at the base station. The agent monitors every packet that passes through the TCP connection in both directions and maintains a cache ofTCP segments sent across the link that have not yet been acknowledged by the receiver. A packet loss is detectedby the arrival of a small number of duplicate acknowledgments from the receiver or by a local timeout. Thesnoop agent retransmits the lost packet if it has it cached and suppresses the duplicate acknowledgments. In ourclassification of the protocols, the snoop protocol is a link-layer protocol that takes advantage of the knowledgeof the higher-layer transport protocol (TCP). The main advantage of this approach is that it suppresses duplicate acknowledgments for TCP segments lost and retransmitted locally, thereby avoiding unnecessary fast retransmissions and congestion control invocations by the sender. The per-connection state maintained by thesnoop agent at the base station is *soft*, and is not essential for correctness. Like other link-layer solutions, thesnoop approach could also suffer from not being able to completely shield the sender from wireless losses.

• **Selective Acknowledgments**: Since standard TCP uses a cumulative acknowledgment scheme, it often does notprovide the sender with sufficient information to recover quickly from multiple packet losses within a single z transmission window. Several studies [6] have shown that TCP enhanced with selective acknowledgmentsperforms better than standard TCP in such situations. SACKs were added as an option to TCP by RFC 1072 [9]. However, disagreements over the use of SACKs prevented the specification from being adopted, and the SACK option was removed from later TCP RFCs. Recently, there has been renewed interest in adding SACKs to TCP. Two of the more interesting proposals are the TCP SACKs Internet Draft [14] and the SMART scheme [12].The Internet Draft proposes that each acknowledgment contain information about up to three non-contiguous blocks of data that have been received successfully. Each block of data is described by its starting and ending sequence number. Due to the limited number of blocks.

3. Analytical Modeling of Flow Starvation

In this, we present an analytical model to study the effect of flow starvation among nodes in

WMNs with a linear topology.We first define the state space and then describe how to determine the state transition probabilities.

3.1 System Model

We consider a linear topology such that each node is within the transmission range of its neighboring nodes (see Fig. 3). For nodes (or wireless mesh routers) that are two hops away from each other, they are within the carrier sensing range but not within each other's transmission range. We consider the Transmission Control Protocol at the transport layer. Nodes that send data packets receive TCPacknowledgments (ACKs). The number of data packets sent by a node before receiving an ACK is less than or equal to the TCP congestion window parameter, cgw, which we assume to be fixed for ease of analysis. We assume that nodes are always backlogged. A source node can attempt to send data packets as long as the number of unacknowledged packets is less than cgw.

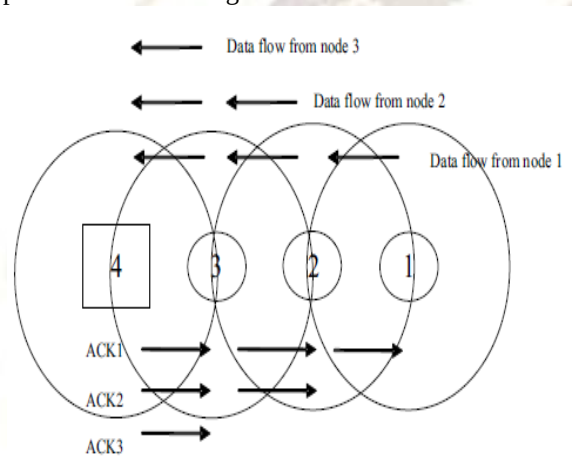

Figure 3. Upstream data transmission in a linear topology with three wireless mesh routers {1, 2,3} and a gateway (GW) {4}.

Any intermediate node may either have data packets to send in the upstream direction (from node to GW) or ACKs to send in the downstream direction (from GW to the node). In the data link layer, each node listens to the channel before attempting a packet transmission. If the channel is busy, the node continuously listens to the channel until it senses that the channel is idle. The node then randomly selects a slot among the next CW slots (where CW is the contention window size) and transmits a packet in that slot if the channel is idle. Packet collision can happen when two neighboring nodes transmit a packet in the same slot. When packet collision occurs, the corresponding nodes double their contention window and start the sending process again. If the contention window reaches its maximum value, it is reset and the packet to be sent is being dropped.

3.2 States and Transition Probabilities

We model the state of each node using a tuple that consists of the number of packets in each queue: a separate queue is maintained for data packets from a particular flow and a separate queue is utilized for ACKs corresponding to a particular flow. Furthermore, the state of each node also includes the size of its contention window. Let N = {1, 2, . . . ,|N|} denote the set of nodes (i.e., wireless mesh routers and GW). In the linear topology, node |N| is GW, node n ∈ {1, 2, . . . ,|N| − 1} is the $(|N| - n)^{th}$ hop downstream neighbor of GW. Thus, node 1 is the farthestnode from GW in terms of the number of hops. Let vectorc = (CW1,CW2, . . . ,CW|N|), where CWn is the contentionwindow size of node n ∈ N. Let F denote the set of flows.A flow f ∈ F corresponds to a source and destination pair.For n ∈ N, we let vector $q_n^d = (Q_{n,1}^d, Q_{n,2}^d, \ldots, Q_{n,|f|}^d)$, where $Q_{n,f}^d$ denotes the number of data packets stored innode n for flow f ∈ F. Similarly, for n ∈ N, we letvector $q_n^a = (Q_{n,1}^a, Q_{n,2}^a, \ldots, Q_{n,|f|}^a)$, where $Q_{n,f}^a$ denotesthe number of ACK packets stored in node n for flow f ∈ F.The system state is an aggregate state that is composed ofthe states of individual nodes. The system state is defined as

$s = (c, q_n^d, q_n^a, \forall\ n \in N)$.(1)

The term "TCP starvation" describes the phenomena of the output queue being filled with larger volumes of UDP, causing TCP connections to have packets tail dropped. Tail drop does not distinguish between packets in any way, including whether they are TCP or UDP, or whether the flow uses a lot of bandwidth or just a little bandwidth. TCP connections can be starved for bandwidth because the UDP flows behave poorly in terms of congestion control. Flow-Based WRED (FRED), which is also based on RED, specifically addresses the issues related to TCP starvation.

4. Implementation and simulation

4.1 Implementation

Implementing NS2 within the Adaptive Transport Protocol requires some effort, even with the aid of a reliable protocol such as TCP. Bandwidth estimation, managing timers, and providing message-oriented, rather than stream-oriented semantics, must be provided on top of the underlying kernel protocol. In addition, TCP's congestion-control scheme represents a potential obstacle to our mechanisms for flow control and assigning priorities to messages. ATP, the complete implementation of NS2 which we describe here, runs at user-level over TCP or UDP, and consists of C++ code.

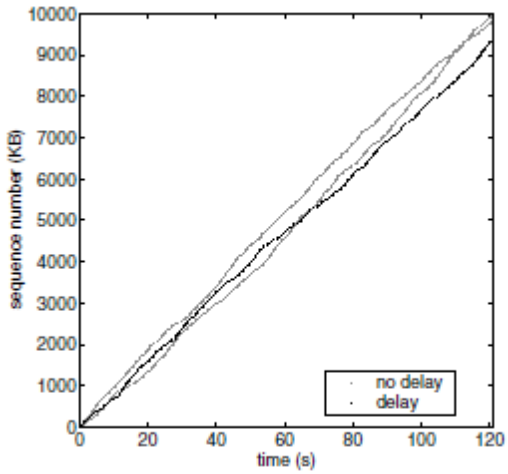

figure 2(a)

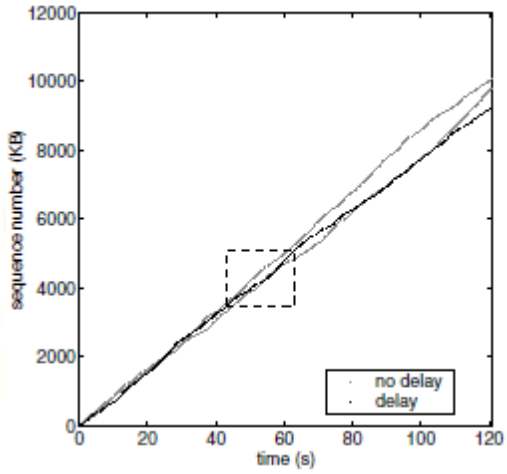

figure 2(b)

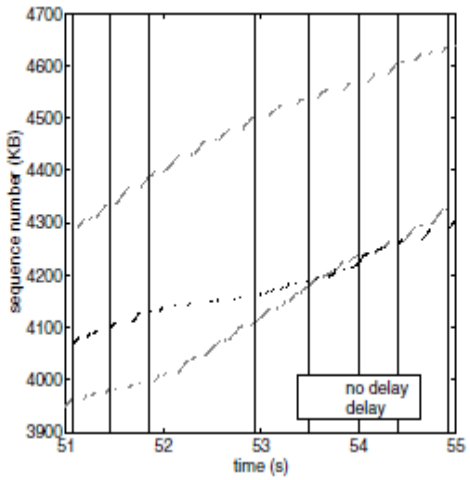

Figure 2(c)

A. Reliable transport subsystem

ATP is a reliable protocol, so it must run on top of a reliabledatagram protocol or a reliable stream protocol. At first glance, TCP would seem to be perfectly adequate, since it hasbeen highly optimised to perform well both in local-area networks and wide-area networks. TCP has also been adapted to cope with the peculiarities of wireless networks, such as high error rates and packet losses [8], [9]. However, implementing ATP on TCP requires considering anumber of alternatives. Transmitting an entire message at once using TCP may result in the message being buffered in the kernel (if there is sufficient buffer space), preventing an application from deferring the send operation or aborting it. Once TCP has copied data into the kernel, it is not easy to determine how much of it has been sent. A final difficulty lies in deciding whether to use multiple streams to connect the sender to the receive r, and, if so, how to allocate messages to streams. Using a single TCP stream will result in all message transmissions being serialised, so that a high-priority message may have to wait for a low-priority message. Using multiple TCP streams may result in unpredictable competition for bandwidth, since TCP is a greedy protocol and most common implementations of TCP do not coordinate congestion control between streams [10].

In our initial version of ATP, we chose to allow message transmission over either TCP, or a reliable datagram protocol on top of UDP, which we will refer to as SPP (Sequenced Packet Protocol). The TCP implementation is the simpler of the two, and runs over a single TCP connection, but sends messages in fixed size segments (1 MTU), rather than sending an entire message at a time. Padding is required for small messages to enable the receiver to detect segment boundaries. It has the natural advantage of being TCP-friendly. The SPP implementation performs its own buffering, retransmissions and duplicates suppression at user level. Since it uses UDP datagrams, it requires no padding to distinguish message boundaries, and is therefore more efficient for transmitting small messages. Unlike TCP, SPP is not optimised for WAN use, and has not been thoroughly tested to determine its fairness or behavior under high error rates. It is robust to the errors we have seen in our 802.11b network, and to packet drops caused by queue overflows in the sender's networkstack. The ATP experiments in this paper use the versionrunning over SPP. For the purposes of comparison, we have also implemented a version of NAI using multiple TCP connections, one per priority level, which we refer to as MTCP (multi-stream TCP, described in more detail in Section V). We conducted some experiments to assess the behavior of concurrent TCP streams and determine if this design was suitable for an NAI

implementation. The graphs in Figure 2 illustrate some cases of competition between concurrent TCP streams which result in an unequal division of bandwidth. We conducted experiments to measure how three concurrent TCP flows compete for bandwidth over a link with a capacity of 256 KB/s. Each flow sends 32 KB messages in a loop for 120 seconds. One of the three senders was delayed for a constant amount of time between the completion of one send call and the start of the next, to investigate the effect of TCP's slow-start restart [11] for idle connections. The expected outcome of these tests would be for the two undelayed senders to receive a roughly equal share of bandwidth,and the delayed sender to receive a smaller share, decreased inproportion to the size of the delay. However, as Figures 2(a)and 2(b) show, the shares of bandwidth are not constant, andat some points the delayed sender outperforms the undelayedsenders! In addition to looking at sequence number plots, wealsocalculated the times taken for individual send operations bythe delayed sender. Figure 2(b) shows a sequence number plot

where the delayed sender waits for 150 ms between send operations;measuring the time between completion of successivesends gives an average of 0.42 seconds, but a range from 0.22to 1.09, almost a five-fold variation (a minimum of 0.22 is possible because the send can buffer data in the kernel and return fast, overlapping with the delay). This compares to an expected mean of 0.375 seconds if all the streams received equal bandwidth. Figure 2(c) shows the region of this test that the send operation taking 1.09 seconds lies in, with vertical bars indicating the completion times of sends. This type of unpredictable and uncontrollable contention effect argues that NAI over multiple TCP streams may ultimately be inferior to other options, particularly in settings where small and infrequent high-priority messages are intermixed with lower-priority bulk data transfers.

B. Bandwidth estimator

Estimating bandwidth is a necessary component of an adaptive transport protocol, since both the application and the protocol itself rely on this value in order to adapt appropriately to network changes. ATP requires a bandwidth estimate to fully implement callback timers, or else a message can never be reported as undeliverable before its callback timer expires. Much work has been devoted to the general problem of estimating bandwidth for flows in a wide-area network. Wide-area bandwidth estimation schemes must arrive at an estimate of limiting bandwidth, which lies at some link along the path between a sender and receiver. In contrast, we assume that the rate of communication is principally limited by the bandwidth on the wireless link. Since all communication between the mobile host and remote hosts must be over this link, we can estimate the total bandwidth available on it, rather than deriving separate estimatesfor each connection or destination host. Our current estimatorassumes that traffic from protocols other than ATP constitutes a negligible fraction of the total traffic, but this restriction would be removed in a kernel version of ATP. A further important difference from wide-area bandwidth estimation is a side-effect of ATP semantics. Since ATP incorporatespriorities for messages, an inaccurate estimate can cause apriority inversion. The difficulty is that the device driver may buffer datagrams when it is unable to transmit as fast as the incoming rate. If an over-optimistic bandwidth estimate causes the kernel to buffer datagrams from low-priority messages, these will hold up the transmission of datagrams from high-priority messages until the send queue is free of low-priority datagrams. It is impractical to remove datagrams from the send queue, but some operating systems allow the length of the queue to be read by applications (for instance, by FreeBSD's ifmib feature). The ATP bandwidth estimator incorporates a heuristic to "back off" and reduce its estimate when it detects a backlog in the send queue.

The bandwidth estimation algorithm: A straightforwardscheme for bandwidth estimation is to count how many send operations, and of what sizes, complete over an interval, and divide to find the estimate. This can be inaccurate because the kernel buffers data, both at the protocol level (in the case of TCP – UDP does no buffering), and at the network device driver. Additionally, the time required to derive an estimate depends on the amount of data being sent. Blocking on a send operation for a large message will delay the estimate until the send completes

```
intcurbw; int staleness = 0;
intpolldelay; // configurable, >= 0
bandwidth_estimate(used, backlog) {
used = maximum(used, filter(used));
if (backlog > MAXIMUM_BACKLOG) {
curbw = used; staleness = 0;
return (used, used - backlog*MTU);
}
else if (used <curbw) {
int probe = PROBE_SIZE;
staleness++;
probe *= staleness / polldelay;
curbw = used;
return (curbw, curbw + probe);
}
else {
staleness = 0;
return (curbw, curbw + PROBE_SIZE);
}
}
```

Figure 3: *The bandwidth estimation algorithm. Every second, ATP invokes the bandwidth estimator to determine a new estimate (this is a tuple of the*

internal estimate and the estimate advertised to the application) The bandwidth used over the preceding second is averaged with the values for the four preceding seconds, then the new estimate is generated, depending on the backlog and staleness of the current estimate .MAXIMUM BACKLOG is set high enough to avoid transient backlogs (10 in our implementation), and PROBE SIZE is half the minimum of the send queue capacity and the current estimate.

Alternatively, the bandwidth used by a TCP connection can be derived from TCP's round-trip time estimate and the send window size. However, this value reflects how much data has been sent on the connection, and not the potential capacity of the connection. ATP derives its estimate by polling the network card for the amount of data sent and received over the course of each second. This quantity is then used as a predictor of the bandwidth over the next second. Since the bandwidth reported by the network card depends on the amount of data which ATP tries to send, this simple estimate is inaccurate if the true bandwidth is higher than the send rate. Accordingly, the bandwidth estimator uses a probing scheme to speculatively increase the estimate. Estimation relies on three statistics: the observed bandwidth, the length of the network interface send queue (backlog), and the staleness of its current estimate. Staleness measures the number of seconds since the last point at which the estimate changed, or since there was a "genuine decrease" in available bandwidth. A genuine decrease can be distinguished from a decrease in the count of bytes transmitted by detecting that the length of the send queue has increased (in fact, we check that it exceeds a threshold of 10 packets; the maximum length allowed is 50 in FreeBSD). Figure 3 shows pseudocode for the bandwidth estimation algorithm. It runs once every second, and computes two values based on the statistics obtained over the previous second curbw and available. The available value is the estimate of available bandwidth on the wireless link, which is suppliedto the application. The curbw value is the amount of bandwidth which ATP should restrict itself to using over the next second.This is an internal ATP statistic, which may be lower thanavailable if the network interface send queue is nonemptyand there is a consequent risk of priority inversion. The available value may also be higher if the estimator is probing thebandwidth. An averaging filter with a window size of 5 is usedto smooth the estimates in order to make them less sensitive totransient spikes.If the application has queued more messages than can be sentin a single second, then the estimator will automatically detectincreases in available bandwidth, since the per-second usage willincrease as the bandwidth increases. However, if the network isunderutilised, then an increase may not be detected without anadditional mechanism. For instance, an application using ATPmay use the bandwidth estimate to determine its mode of operation,and not change into a mode using more bandwidth unlessthe estimate goes up. In this case the surplus bandwidthwould never be exploited. ATP breaks this deadlock in twoways. Firstly, in the trivial case where the application is unwilling
to send any data because it believes the bandwidth iszero, ATP polls the remote host to determine when bandwidthrises above zero, making a simple packet-pair estimate [12].Secondly, ATP probes the bandwidth by speculatively increasingthe estimate it gives the application. The intent behind thismechanism is that the application will eventually decide the estimateis high enough and attempt to exploit it by sending moredata. Figure 3 shows the probe mechanism as part of the bandwidthestimate routine.

References:

[1] A. Aggarwal, S. Savage, and T. Anderson, " Understanding the Performance of TCP Pacing ", Proceedings of IEEE INFOCOM, Tel Aviv, Israel,March 2000.

[2] ANSI/IEEE Standard 802.11b, "Part 11: Wireless LAN Medium Access Control (MAC) and PhysicalLayer (PHY) Speci_cations: Higher-Speed Physical Layer Extension in the 2.4 GHz band", 1999.

[3] C. Cordeiro, S. Das, and D. Agrawal, "COPAS:Dynamic Contention-Balancing to Enhance the Performance of TCP over Multi-hop Wireless Networks", Proceedings of IC3N'02, Miami, FL, October 2002.

[4] S. Floyd and V. Jacobson, \Random Early Detection Gateways for Congestion Avoidance", IEEE Transactions on Networking, Vol. 1, No. 4,pp. 397-413, August 1993.

[5] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The Impact of Multi-hop WirelessChannel on TCP Throughput and Loss", Proceedings of IEEE INFOCOM'03, San Francisco, CA,April 2003.

[6] W. Hung, K. Law, and A. Leon-Garcia, "A Dynamic Multi-Channel MAC for Ad Hoc LAN",Proceedings of 21st Biennial Symposium on Communications, Kingston, ON, Canada, June 2002.

[7] N. Jain, S. Das, and A. Nasipuri, "A MultichannelCSMA MAC Protocol with Receiver-Based Chan- nel Selection for Multi-hop Wireless Networks",Proceedings of the IEEE ICCCN'01, Phoenix, AZ, October 2001.

[8] J. Ke, "Towards a Rate-Based TCP Protocol forthe Web", Proceedings of

MASCOTS'2000, San Francisco, CA, pp. 36-45, October 2000.

[9] T. Kuang and C. Williamson, "A Bidirectional Multi-Channel MAC Protocol for Improving TCP Performance on Multi-Hop Wireless Ad Hoc Networks", submitted for publication, 2004.

[10] T. Kuang, F. Xiao, and C. Williamson, "Diagnosing Wireless TCP Performance Problems: A Case Study", Proceedings of SCS SPECTS Conference, Montreal, PQ, pp. 176-185, July 2003.

[11] J. So and N. Vaidya, "A Multi-channel MAC Protocol for Ad Hoc Wireless Networks", TechnicalReport, Dept. of Computer Science, University of Illinois at Urbana-Champaign, January 2001.

[12] K. Tan and M. Gerla, "Fair Sharing of MAC under TCP in Wireless Ad Hoc Networks", Proceedingsof IEEE MMT'99, Venice, Italy, October 1999.

[13] A. Tzamaloukas and J. Garcia-Luna-Aceves, "A Receiver-Initiated Collision-Avoidance Protocol for Multi-Channel Networks", Proceedings of IEEE INFOCOM'01, Anchorage, USA, April 2001.

[14] VINT Group, "Network Simulator ns-2", available at http://www.isi.edu/nsnam/ns.

[15] S. Wu, Y. Tseng, C. Liu, and J. Sheu, "A Multi Channel MAC Protocol with Power Control for Multi-Hop Mobile Ad Hoc Networks", The Computer Journal, Vol. 45, No. 1, pp. 101-110, 2002.

[16] S. Xu and T. Saddawi, "Does the IEEE 802.11 MAC Protocol Work Well in Multi-hop Wireless Ad Hoc Networks?", IEEE Communications Magazine, June 2001.

Authors:

Venkatesh.Donepudicompleted his M.Tech in IT in VignanUniversity.He is working as Asst.Professor in CSE Department,Jagans College of Engineering and Technology, choutapalem, Nellore. Having 1.5 Years of Teaching Experience.

vahiduddinsheriff Completed his M.Tech in CSE in VignanUnivrsity. Working as Asst.Professor in CSE Department ,CR Reddy college of Engineering and Technology,Eluru. Having 2 years of Teaching Experience.