ISSN:2249-5789
Santosh Hosamani et al, International Journal of Computer Science & Communication Networks,Vol 2(2), 218-223

The Dynamic Buffer Sizing Strategies for 802.11 Based Wireless Networks

Santosh Hosamani[1], G.S.Nagaraja[2]
[1]M.Tech Student, Department of Computer Science & Engg, R V College of Engineering, Bangalore (hosamani.santosh@gmail.com)
[2]Associate Professor, Department of Computer Science & Engg, R V College of Engineering, Bangalore (nagarajgs2000@gmail.com)

Abstract

Wireless networks face a number of fundamental issues that do not arise in wired networks. We consider the sizing of network buffers in 802.11 based wireless networks. 802.11 is a set of standards for implementing wireless local area network (WLAN) computer communication using different frequency bands. They are created and maintained by the IEEE LAN/MAN Standards Committee (IEEE 802). All internet routers contain buffers to hold packets during the time of congestion. Buffers are used to reduce the packet loss and to ensure high link efficiency. The widely used general rule-of-thumb is to have buffers size as the bandwidth-delay product (BDP) of the network, In this paper we argue that the use of the fixed size buffers in 802.11 based wireless networks results in either undesirable channel under-utilization or unnecessary high delays and the increased packet loss. Our objective is to maintain high network utilization while providing low queuing delays in 802.11 wireless networks through dynamic buffer sizing algorithms.

Keywords: *Wireless LAN, 802.11, TCP, Buffer Sizing.*

1. Introduction

Buffer size refers to the size allocated for temporary storage of data. Buffers are used at network routers to temporarily store incoming packets when the arrival of packets received exceeds the capacity of the egress link. Buffer sizing is an active research topic in the wireless network community. In communication networks buffers are used for the short time packet bursts. The importance of buffering in communication networks is to reduce the packet drops and to maintain the high link efficiency.

Buffer sizing is an important network configuration parameter: under-buffered networks lead to frequent packet loss and subsequent underutilization of network resources, while over buffered networks lead to increased queueing delays.

Buffer sizing is an active research topic in both wired and the wireless networks [1]. In case of the wired scenario we consider the buffer sizing for the wired routers mainly based on the classical rule of thumb. In the wired case the buffer sizes are set as the product of the bandwidth of the link and the average delay (round trip time or the RTT) of the flows utilizing this link. This rule is also called as the Bandwidth Delay Product (BDP) rule.

Buffer sizing acts as an important metric in 802.11 based wireless networks. The problem is complicated by the time varying capacity of the wireless channel as well as the random access mechanism of 802.11 MAC protocol. The present scenario in which the fixed size buffers are considered for buffering may easily leads to the poor performance of the network. In case of wireless scenario the transmissions are broadcast in nature. Thus the packet service times at different stations in a WLAN are strongly coupled. Hence the mean service rate at a wireless station is strongly dependent on the level of channel contention and thus on the number of active stations and their load. Even when the network load is fixed the packet inter-service times at a station are not fixed but vary stochastically due to the random nature of the CSMA/CA operation. As a result, neither the bandwidth nor the delay in 802.11 WLANs are constant, in contrast to the wired links. We therefore do not have a fixed BDP value available to provide a basis for sizing buffers. These facts affect statistical multiplexing and buffer backlog behaviour, and thus the choice of buffer sizes. Wireless stations dynamically adjust the physical transmission rate/modulation used in order to regulate non-congestive channel losses. This rate adaptation, whereby the transmit rate may change by a factor may induce large and rapid variations in required buffer sizes.

In this paper we study the analysis of the rule of thumb and we find that it shows requirement for adaptation of buffer size in response to changing network conditions. This leads naturally to the consideration of dynamic buffer sizing strategies that adapt to changing conditions. We propose two dynamic buffer sizing algorithms for 802.11 based WLANs.

This paper is organized as follows. Section II describes the literature survey on sizing the router buffers and the rule of thumb. We also analyze the various buffer sizing rules. In section III we describe the methodology used for the TCP fairness and the buffer sizing for the TCP flows in 802.11 WLANs. Section IV gives the strategies proposed for the dynamic buffer sizing. The section V gives the conclusion.

2. Literature Survey

The purpose of this literature survey is to provide the background information on the issues to be considered in this paper and to emphasize the relevance of the present study.

Router buffers are traditionally sized with two primary objectives in mind.

(i) *Accommodating short-term packet bursts.*

The current internet traffic tends to be bursty mainly because of the nature of TCP. A router may not show capacity to process all of the packets immediately when too many packets arrive in a short interval of time. The primary characteristic of the router buffer is to lower down the packet losses which occur because of the bursts. These packets are accommodated in the router buffer until all the packets are serviced.

(ii) *Ensuring AIMD throughput efficiency.*

The TCP makes use of the additive increase multiplicative decrease (AIMD) congestion control algorithm. Whenever the TCP detects the network congestion it decreases the number of packets in flight by half of the total number of packets by using the AIMD congestion control algorithm. A back off action results if router buffers used are too small. This back off action will cause them to empty with a corresponding reduction in link utilisation.

A. *Sizing the Router Buffers and the Rule of Thumb*

The size of the buffers is determined by the TCP's congestion control algorithm. To be more specific the goal is to make sure that when a link is congested, it is busy for all the time. In other words it is equivalent to making sure that its buffer never goes empty. Generally the router buffers are sized based on a rule-of-thumb. It states that each link needs a buffer of size according to a relation B = RTT × C, where RTT is the average round-trip time of a flow passing across the link, and C is the data rate of the link. The rule of thumb does indeed make sense for one long-lived TCP flow. But this rule doesn't holds good in case of the backbone routers. The reason behind this drawback is that large number of flows or the TCP connections are multiplexed together on a single backbone link [2]. Router buffers are sized mainly on a condition that when TCP flows pass through them, they should not underflow and should not lose the throughput, and this is the situation where the rule-of-thumb comes from. The main metric we will use is throughput, and our goal is to determine the size of the buffer so as to maximize throughput of a bottleneck link. The basic idea followed is that when a router has packets buffered then its outgoing link is always busy. In the case if the outgoing link is a bottleneck, then we want to keep it busy as much of the time as possible. In order to achieve it we just need to make sure the buffer never underflows and it never goes empty. Thus we can say that the rule-of-thumb is the amount of buffering needed by a single TCP flow, so that the buffer at the bottleneck link never underflows, and so the router doesn't lose throughput.

The dynamics of TCP's congestion control algorithm yields the rule of thumb. To be specific, a single TCP flow passing through a bottleneck link requires a buffer size equal to the bandwidth-delay product in order to prevent the link from going idle and resulting in losing the throughput.

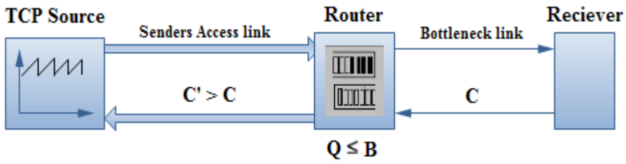

Figure 1: An access link of latency l_{Acc} and link capacity C_{Acc} and a bottleneck link of capacity C and latency l in a single flow topology.

Consider the simple topology in Figure 1 in which a single TCP source sends an infinite amount of data. The data packets sending are of constant size. The flow passes through a single router. The sender's access link is much faster than the bottleneck link of capacity C of the receiver. Due to this difference in link capacity the packets are queued at the router. The propagation time of the packets between sender and receiver or between receiver and the sender is denoted by Tp. Now assume that the TCP flow has settled into the additive-increase and multiplicative-decrease (AIMD) congestion avoidance mode. When the sender transmits a packet an ACK is received each time and gradually increases the number of outstanding packets i.e. the window size. This gradually results in filling up of the buffer. Eventually a packet is dropped, and the sender doesn't receive an ACK. At this time it halves the window size and pauses. The sender now has too many packets outstanding in the network: it sent an amount equal to the old window, but now the window size has halved. So the sender pauses and it waits for ACKs to arrive so that it can resume transmitting the packets.

The key idea to sizing the buffer is to make sure that while the sender pauses, at this time the router buffer should not go empty and it should not force the bottleneck link to go for idle state. By determining the rate, we can determine the size of the reservoir needed to prevent the buffer from going empty by determining the rate at which the buffer drains. It shows that this is equal to the distance in bytes between the peak and trough of the sawtooth representing the TCP window size and this can be easily mapped to the rule-of-thumb.

B. *The Analysis of Buffer Sizing Rules*

In this section we briefly present the different buffer sizing rules and the assumptions on which these rules depend. We also notice that different rules lead to different buffer sizes.

(i) *Rule of thumb:*

The rule of thumb which states that $B = T \times C$ assumes there is a single long lived TCP flow going through the bottleneck link. In this case the bandwidth B is determined by the shape of the TCP window size. Because the window size follows the well known sawtooth, with a distance from peak to trough of $T \times C$, then we need this much amount of buffering to ride out reductions in window size to make sure the bottleneck buffer doesn't go empty and lose throughput. When we consider the validation it is very easy to show from inspection, simulation or in the lab that with a single long lived TCP flow we need $B = T \times C$ to maintain full utilization [3]. Villamizar and Song's first experiments in 1994 consisted of one to eight flows. With such a small number of flows, the sawtooths tend to synchronize because losses hit each flow at roughly the same time.Therefore, the aggregate window size process is also a sawtooth with the same amplitude, and hence the buffer size doesn't change.

(ii) *Small Buffers Rule :*

When there are N long-lived TCP flows sharing the link Appenzeller proposed reducing buffers by a factor of square root of N [4]. The point which should be considered is that if there are sufficiently large numbers of flows, they tend to desynchronize. It seems to start happening with a hundred flows or so. According to central limit theorem as the number of flows increases, the amplitude of the aggregate window size process decreases and hence the traffic smooths. In the absence of another need for buffers, we can easily reduce the buffer size as we increase N.

The small buffers rule makes two main assumptions: The first one is that utilization is the right metric for buffer sizing in a router, and secondly when there are many flows, they aren't synchronized. Utilization is an operator-centric metric. When a congested link operates at 100% throughput then it makes efficient use of the operator's congested resource. It's not necessarily ideal for an individual end-user as the metric doesn't guarantee a short flow-completion time (i.e. quick downloads), or that there won't be too many packet drops. But, there is reason to think that this metric reflects short flow-completion times and appropriate numbers of packet drops. If the buffers are smaller, but not so small as to reduce throughput, then the round-trip time is reduced which for TCP leads to higher throughput for each flow, and they will complete quickly. To understand the relationship between the number of flows and their synchronization, Raina and Wischik modelled a network with various buffer sizes [3].They concluded that with the small buffers rule, the network is not stable, and should have low throughput due to the periodic changes in the aggregate window size, which is a direct consequence of synchronization.

(iii) *Drop based buffers Rule :*

Dhamdhere and Dovrolis [5] studied a particular network example to argue that when packet drop rates are considered, we can conclude that much larger buffers are needed, perhaps larger than the buffers in place today. They studied an example where a large number of flows share a heavily congested low capacity bottleneck link towards the edge of the network, and showed that one might get substantial packet drop rates (up to 17%). In their example, a 50Mb/s link carries 200 long-lived TCP flows, as well as some additional short lived flows. The effective RTT of the flows is 60ms (i.e. the average congestion window size is about two packets). If $B = C \times T$ then the buffer will contain about 1500 packets. The small buffers rule suggests a buffer size of only about 100 packets. Because of the high drop rate they measured, the authors propose increasing buffer sizes.

In this scenario, the problem comes from congestion window dropping to such a low value that TCP starts to drop a lot of packets. Increasing the buffer size doesn't directly reduce the drop-rate in the way we might expect. Increasing the size of the buffer will increase the propagation delay of each flow which in turn, increases the average congestion window size to greater than two packets, and then the drop-rate goes down. It's not clear if we always want to keep the drop-rate low on a heavily congested link. After all, if the link is congested, we'd like to get the notice quickly to the sources so they can reduce their window size. Increasing the buffer size only delays the feedback to the sender. On the other hand, large drop-rates eventually cause TCP performance to fall apart. This suggests a lower-bound on the buffer size that may or may not come into play, depending on the speed of the link.

(iv) *Tiny buffers Rule :*

Tiny buffers are most widely used for all-optical routers. It is interesting to analyze how performance of the network would be affected if we reduced the size of the buffers to just 10-20 packets. For the analysis of the amount of lost capacity and the drop-rates Raina and Wischik [3] suggested that we could build a network with tiny buffers if we are willing to sacrifice a small amount of throughput. When access links are much slower than the core links we have a natural smoothing of packet arrivals into core routers and with only a few dozen packets we can gain small drop rates and a throughput of 85-90% and also when access links have rates comparable to the core links, one can get the same results in the analysis.

3. Methodology for the Buffer Sizing for TCP flows in 802.11 WLANs

In this section we study the TCP fairness and impact of buffer sizing for TCP flows in 802.11e WLANs.

A. *TCP Fairness in 802.11e WLANs*

A cross-layer interactions between the 802.11 MAC and the flow/congestion control mechanisms employed by TCP typically lead to gross unfairness between competing flows, and indeed sustained lockout of flows. It is analyzed how to use the flexibility provided by the new 802.11e MAC to resolve the transport layer unfairness in infrastructure WLANs [6]. TCP uploads and downloads, or the mixtures of both are considered. To analyze the TCP unfairness over 802.11 WLANs we consider unfairness between competing TCP upload flows and between competing upload and download flows.

For the restoring of the fairness the existing approaches to reduce the gross unfairness between TCP flows competing over 802.11 WLANs work within the constraint of the current 802.11 MAC, resulting in complex adaptive schemes requiring online measurements and perhaps, per packet processing. We instead consider how the additional flexibility present in the new 802.11e MAC might be employed to alleviate transport layer unfairness.

To address TCP's performance problems two issues must be addressed. The first one is asymmetry between the TCP data and TCP ACK paths that disrupts the TCP congestion control mechanism, and the second one is network level asymmetry between TCP upload and download flows. A simple solution is developed that uses the 802.11e arbitration inter frame spacing (*AIFS*), CW_{min} parameters to ensure fairness between competing TCP uploads and downloads.

B. *Buffer Sizing for TCP Flows in 802.11e WLANs*

This methodology includes that the mean service rate is dependent on the level of channel contention and the packet inter service times that vary stochastically due to the random nature of CSMA/CA operation. It is mainly focussed on the typical deployment scenario where an infrastructure mode WLAN is configured with the Access Point (AP) acting as a wireless router between the WLAN and the Internet. Considering the performance with fixed buffering, in contrast to wired networks, the mean service rate at a wireless station is not fixed but instead depends upon the level of channel contention and the network load. The throughput and delay of a download flow are plotted as a function of the AP buffer size when the number of competing upload flows (with one upload flow per wireless station) is varied. Similarly for wired networks, the throughput always increases monotonically with the buffer size, reaching a maximum above a threshold buffer size. Also note that download throughput falls as the number of competing uploads increases.

One possible approach is to size buffers based on the conditions requiring the largest buffering to achieve high throughput. But this comes at the cost of high latency while ensuring high throughput. The queueing delay at the AP depends on the service rate, which in turn depends on the number of contending wireless stations and their offered load. Conversely, sizing the buffer to achieve lower latency across all network conditions comes at the cost of reduced throughput. In addition to variations in the mean service rate, it is also important to note that the random nature of 802.11 operations mainly leads to short time-scale stochastic fluctuations in service rate [7]. It directly affects the buffering behaviour. Stochastic fluctuations in service rate results in the early queue overflow and reduced link utilisation. The stochastic fluctuations in service rate lead to a need to increase the buffer size above the BDP value in order to accommodate the effect of these fluctuations. A simple but effective approach is to over-provision by a fixed number of packets above the BDP.

C. *Adaptive Buffer Sizing for TCP Flows in 802.11e WLANs*

In this methodology the provision of Access Point buffers in WLANs is considered. The use of static buffers in WLANs leads to either undesirable channel underutilisation or unnecessary high delays, which motivates the use of dynamic buffer sizing. In this methodology an algorithm to exploit statistical multiplexing gains in WLANs is designed. The objective is to simultaneously achieve both high throughput efficiency and low delay. This strategy involves feedback control of buffer size based on measurements of the buffer idle and busy time. Also in order to ensure efficient link utilisation, the buffer should not lie empty for too long time. Increasing the buffer size results in the reduction of link idle time. Moreover, to ensure low delays, the buffer should be as short as possible and a trade-off therefore exists. This behaviour suggests the following approach. We observe the buffer occupancy over an interval of time. If the buffer rarely empties, we decrease the buffer size to avoid high delay. Conversely, if the buffer is empty for too long period, we increase the buffer size to maintain high throughput.

4. The Dynamic Strategies for Buffer Sizing

The key observations from sections II and III is that there exists no fixed buffer size capable of ensuring both high throughput efficiency and reasonable delay across the range of physical rates and offered loads experienced by modern WLANs. Any fixed choice of buffer size necessarily carries the cost of significantly reduced throughput efficiency and/or excessive queueing delays. Therefore it leads naturally to the consideration of adaptive approaches to buffer sizing, which dynamically adjust the buffer size in response to changing network conditions to ensure both high utilization of the wireless link while avoiding unnecessarily long queuing delays.

A. *Emulating Bandwidth Delay Product (eBDP) Algorithm*

The Emulating BDP is a simple adaptive algorithm based on the classical BDP rule. A wireless station can measure its own packet service times by direct observation. It records the time interval between a packet arriving at the head of the network interface queue t_s and the time at which it is successfully transmitted t_e. It is indicated by receiving accurately the corresponding MAC acknowledgment. Now averaging out these service times per packet yields the mean service time denoted by T_{serv}. The mean service time is not constant. Therefore to facilitate the time-varying nature of the mean service time, this average can be taken over a sliding window. In this strategy we consider the use of exponential smoothing. This exponential smoothing is represented by the expression $T_{serv}(k + 1) = (1 - W)\, T_{serv}(k) + W\, (t_e - t_s)$ to calculate a running average since this has the merit of simplicity and statistical robustness by the central limit arguments. Here the W factor is considered as the smoothing parameter.

When it is provided an online measurement of the mean service time T_{serv}, the classical BDP rule yields the following eBDP buffer sizing strategy. Let T_{max} be represented by the target maximum queuing delay. We calculate the ratio T_{max}/T_{serv}. It is obvious that $1/T_{serv}$ is the mean service rate. According to this mean service rate we select buffer size Q_{eBDP} based on the expression $QeBDP = min(T_{max}/T_{serv},\ Q^{eBDP}_{max})$ where Q^{eBDP}_{max} is the upper limit on the buffer size. This value effectively regulates the buffer size in order to equal the value of current mean BDP. In order to maintain an approximately constant queuing delay of T_{max} seconds the buffer size decreases when the service rate falls and increases when the service rate rises. In a similar way as we measured the mean service time we may measure the flows' round trip times to derive the value for T_{max}. But here we simply use a fixed value of 200ms since this is an approximate upper bound on the round trip time of the majority of the current Internet flows.

We observe that the classical BDP rule is originated from the behaviour of TCP congestion control, in particular, the reduction of cwnd by half on packet loss. The classical BDP rule also assumes a constant service rate and fluid-like packet arrivals. Therefore at the low service rates the BDP rule suggests the use of extremely small buffer sizes. Note that, in addition to accommodating TCP behaviour, buffers have the additional role of absorbing short-term packet bursts and, in the case of wireless links, short-term fluctuations in packet service times. Therefore we modify the eBDP update rule to QeBDP = min $(T_{max}/T_{serv} + c,\ Q^{eBDP}_{max})$. Here the parameter c is an over provisioning amount to accommodate short-term fluctuations in service rate. Obtaining an analytic value for c is intractable, mainly because of the complex nature of the service time at a wireless station which is coupled to the traffic arrivals at other stations in the WLAN and the TCP traffic arrival process where feedback creates the coupling to the service time process. Thus based on measurements it found empirically that a value of c=5 packets works well across a wide range of network conditions.

B. *Adaptive Limit Tuning (ALT) Algorithm*

The above discussed eBDP algorithm is analytically simple and effective, but it fails in a condition when multiple flows share the same link. At this situation it is unable to take the advantage of the statistical multiplexing of TCP cwnd backoffs. Therefore we find a need to design a measurement based algorithm which is capable of taking advantage of the statistical multiplexing opportunities. The algorithm which we develop here we call it as Adaptive Limit Tuning (ALT) algorithm. The objective in this algorithm is to simultaneously achieve both efficient link utilization and low delays in the face of stochastic time variations in the service time. Generally for efficient link utilization we need to ensure that there is a packet available to transmit whenever the station wins a transmission opportunity. Here our main aim is to minimize the time that the station buffer lies empty, which in turn can be achieved by making the buffer size sufficiently large. It is important note that under fairly general traffic conditions, the buffer occupancy is a monotonically increasing function of the buffer size [8]. Moreover, use of large buffers leads to high queuing delays, and to ensure low delays the buffer should be as small as possible. This concept suggests the following approach. First we observe the buffer occupancy over an interval of time. Then if the buffer rarely empties, we decrease the buffer size, and if the buffer is empty for too long time then we decrease the buffer size.

To begin with the Adaptive Limit Tuning (ALT) algorithm we define a queue occupancy threshold q_{thr} and let $t_i(k)$ be the idle time i.e. the duration of time that the queue spends at or below this threshold in a fixed observation interval t, and let $t_b(k)$ be the busy time i.e. the corresponding duration spent above the threshold. Also note that $t = t_i(k) + t_b(k)$ and the aggregate amount of idle/busy time *ti* and *tb* over an interval can be easily observed by a station. Note that the link utilisation is lower bounded by $t_b = (t_b + t_i)$. Let $q(k)$ denote the buffer size during the k-th observation interval. Then the buffer size is then updated according to the rule given by $q(k+1) = q(k) + a_1 t_i(k) – b_1 t_b (k)$ where a1 and b1 are the design parameters.

5. Conclusion

Buffer sizing is an important network configuration parameter. Buffers play a key role in 802.11 wireless networks. Buffers are used to accommodate short term packet bursts so as to mitigate packet drops and to maintain high link efficiency. Packets are queued if too many packets arrive in a sufficiently short interval of time during which a network device lacks the capacity to process all of them immediately. The use of fixed size buffers in 802.11 networks inevitably leads to either undesirable channel under-utilization or unnecessary high delays. In this paper the adaptive buffer sizing strategies are used to maintain high network utilization while providing low queuing delays in 802.11 based wireless networks through the dynamic buffer sizing algorithms.

6. References

[1] G. Vu-Brugier, R. Stanojevic, D. Leith, and R. Shorten, “A Critique of Recently Proposed Buffer-Sizing Strategies,” *ACM Computer Communication Review*, vol. 37. no. 1, Jan. 2007.

[2] G. Appenzeller, I. Keslassy, and N. McKeown, “Sizing router buffers,” in *Proc. of the ACM SIGCOMM ’04*, Sep. 2004, pp. 281–292.

[3] C. Villamizar and C. Song. High performance TCP in ANSNET. ACM Computer Communications Review, 24(5):45–60, 1994.

[4] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing router buffers. In SIGCOMM ’04, pages 281–292, New York, NY, USA, 2004. ACM Press.

[5] A. Dhamdhere and C. Dovrolis. Open issues in router buffer sizing. ACM Sigcomm Computer Communication Review, 36(1):87–92, January 2006.

[6] A. Detti, E. Graziosi, V. Minichiello, S. Salsano, and V. Sangregorio, “TCP fairness issues in IEEE 802.11 based access networks,” submitted paper.

[7] D. J. Leith, P. Clifford, D. Malone, and A. Ng, “TCP fairness in 802.11e WLANs,” *IEEE Commun. Lett.*, vol. 9, no. 11, pp 964–966, 2005.

[8] K. Kumaran, M. Mandjes, A.L. Stolyar, “Convexity Properties of Loss and Overflow Functions,” *Operations Research Letters*, vol. 31, no.2, pp. 95-100, 2003.

Santosh Hosamani, received the Undergaduate degree in Electronics & Communication Engineering from Guru Nanak Dev Engineering College, Bidar, in 2010. Currently he is pursuing post graduation degree in Computer Network Engineering (2010-2012) from RVCE, Bangalore-59. His area of interest includes Networking and Cloud computing.

Dr.G.S.Nagaraja, presently working as Associate Professor in the department of Computer Science & Engineering, RVCE, Bangalore-59. He obtained Ph.D degree in the year February 2011. His research interests include Computer Networks, Network Management and Computer Architecture. He has presented 11 papers in national level conferences, 5 papers in international conferences and 4 papers in referred international journals. Currently he is guiding 4 Ph.D students.