

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

A Study on Bandwidth Estimation for IEEE 802.11-Based Ad Hoc Networks

K S R Murthy¹

Research scholar

CSSE, Andhra University College of Engineering
AP, India**Dr. G. Samuel Vara Prasad Raju²**

Professor

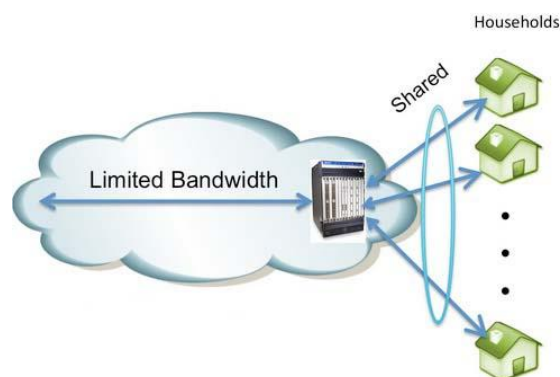
Andhra University
AP, India

Abstract: Since 2005, IEEE 802.11-based networks have been able to provide a certain level of quality of service (QoS) by the means of service differentiation, due to the IEEE 802.11e amendment. However, no mechanism or method has been standardized to accurately evaluate the amount of resources remaining on a given channel. Such an evaluation would, however, be a good asset for bandwidth-constrained applications. In multihop ad hoc networks, such evaluation becomes even more difficult. Consequently, despite the various contributions around this research topic, the estimation of the available bandwidth still represents one of the main issues in this field. In this paper, we propose an improved mechanism to estimate the available bandwidth in IEEE 802.11-based ad hoc networks. Through simulations, we compare the accuracy of the estimation we propose to the estimation performed by other state-of-the-art QoS protocols, BRuIT, AAC, and QoS-AODV.

Keywords: Wireless communication, IEEE 802.11, Quality of Service, available bandwidth estimation.

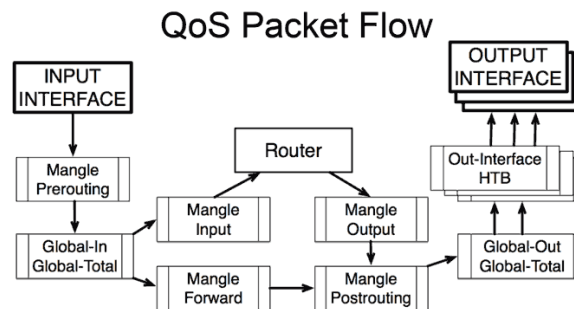
I. INTRODUCTION

Bandwidth is not the only property for a good internet connection. Properties like delay, jitter, and reliability have become the main focus area in the recent years. Together these four properties make up the basis for quality of service (QoS). The Quality of service issues in ad hoc networks are now extensively studied and more and more QoS protocols are proposed. Many protocols concern QoS routing. The goal of such a routing is either to provide the best routes in function of some parameters (like bandwidth, delay, packet loss, etc.) or to find routes that will offer guarantees on some of these parameters.



Many QoS routing protocols have been proposed so far and many of them consider the bandwidth parameter. To design an efficient QoS routing, it is very important to get very accurate information on the used and available bandwidth. The nodes share the medium and their perception of the used bandwidth or the available bandwidth can be very different from one mobile to another, but it also needs to know the available bandwidth which it may share the medium. They can be classified into two main categories:

1. Intrusive techniques that send probe packets for the estimation.



2. Passive techniques that are based on a local computation of the available bandwidth and sometimes on a sparse exchange of this information.



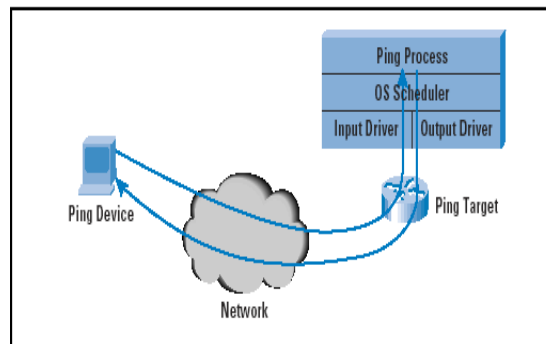
In this article, possible to improve the accuracy in the available bandwidth estimation for ad hoc networks. Indeed, the intrusive solutions consume too much bandwidth while the passive solutions mainly compute the available bandwidth with the formula capacity minus used bandwidth, which is sufficient.

Available bandwidth estimation techniques can be divided in two major approaches:

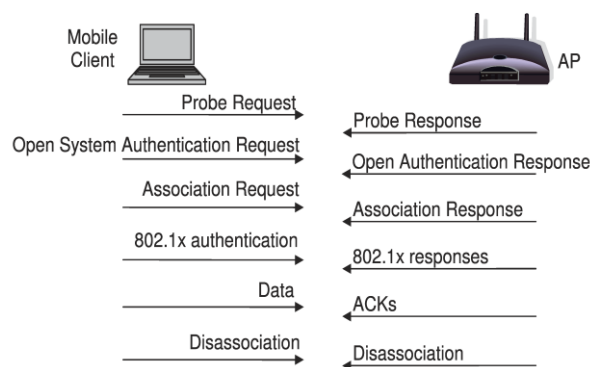
1. Active estimation methods

- (1) Path Construction
- (2) Packet Marking Procedure
- (3) Router maintenance Termination
Packet Number (Tpn) generation.
- (4) Re-Construction Path.

Active bandwidth estimation tools can not only provide network operators with useful information on network characteristics and performance, but also can enable end users (and user applications) to perform independent network auditing, load balancing, and server selection tasks, among many others, without requiring access to network elements or administrative resources. Two main stages in these tools are identified: measurement and estimation. Measurement involves the generation of a probe packet pattern, its transmission through the network, and its reception and measurement. Estimation comprises statistical and heuristic processing of measurements according to some network model. A major component is what we call probe packets generation system; both the measurement and the estimation stages depend on it. Isolation of this function as a component allows for the efficient implementation of tools for a set of measurement techniques, since facilities for common traffic patterns generation can be reused. The probe packet generation component should provide generic services such as generation of packet trains, which currently can be found implemented as functions or methods in most tools based on packet trains.



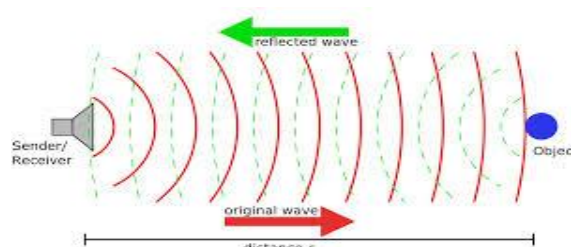
Many active bandwidth estimation techniques have been proposed for wired networks. A detailed survey of the different techniques is proposed. The Self-Loading Periodic Streams (SLoPS) technique measures the end-to-end available bandwidth by sending packets of equal size and by measuring the one-way delays of these probing packets. The source increases the rate of the probing packets; as soon as there is a variation in the delay, one can say that the path is saturated and that the measured point, just before the variation, corresponds to the available bandwidth. The Trains of Packet Pairs (TOPP) technique is based on the same principle. The main difference between these two methods concerns the rate increasing function:



TOPP increases linearly the rate whereas SLoPS uses a binary search. Based on TOPP method, DietTOPP has been developed for a wireless environment. The main idea of is that a probe packets delay higher than the maximum theoretical delay can characterize the channel utilization. The authors propose a method to compute the medium utilization from the delays and then to derive the available bandwidth from the utilization. All these techniques are active as they use end-to-end probe packets to characterize the channel. When every node in an ad hoc network needs to perform such an evaluation for several destinations, the number of probe packets introduced in the network can be important. Therefore, such techniques have mainly two drawbacks: they consume much bandwidth and they have an impact on the on-going traffic they measure.

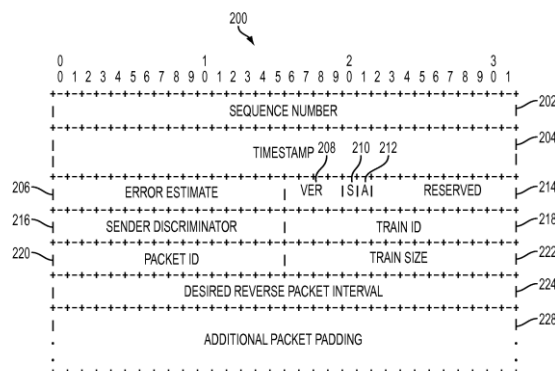
2. Passive estimation methods

Each mobile estimates the available bandwidth by computing the channel utilization ratio and using a smoothing constant. The channel utilization ratio is deduced from a permanent monitoring of the channel status (idle or busy). The QoS routing protocol designed in this article is only based on the available bandwidth at each node and does not consider the possible distant interfering nodes. A bandwidth reservation protocol, called Bandwidth Reservation under InTerferences (BRuIT) that takes into account the notion of carrier sensing area in the available bandwidth estimation. Indeed, with CSMA protocols. Two nodes within carrier sensing range share the medium and thus the bandwidth, even if they cannot directly communicate. Therefore, each node needs not only to know the channel occupancy in its communication range, but also in its carrier sensing range.



BRuIT attempts to compute the channel usage in the carrier sensing area. In BRuIT, the carrier sensing area is approximated by the two-hop neighborhood. Each node provides information about the total bandwidth it uses to route knows and about its neighbors and their usage of the bandwidth, by periodically broadcasting a Hello message containing this information. Then, each node can compute the bandwidth usage in its two hop neighborhood and can then approximate the used bandwidth and derive the available bandwidth in its carrier sensing area. This last computed value is then added to the Hello messages. Thus, each node knows the available bandwidth of its two-hop neighbors. The main drawback of this method is that the two-hop neighborhood may not correspond exactly to the carrier sensing area. the Contention Aware Admission Control. Protocol (CACP). The goal is, like in BRuIT, to determine the available bandwidth of the nodes in the carrier sensing area. First, each node computes the local idle channel time fraction by a permanent monitoring of the radio medium. Then, the authors propose three different techniques: to use, like in BRuIT, Hello messages to broadcast this information over the two-hop neighborhood; to increase the transmission power of nodes such that all the nodes in the carrier sensing area could be reached; or to reduce the sensitivity of the mobiles in order that each node takes into account the bandwidth used in its carrier sensing area. Thus, each node deduces, without an exchange of messages, the available bandwidth of the nodes in its carrier sensing area.

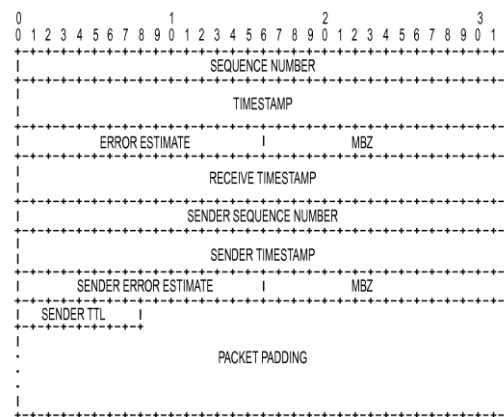
QoS-AODV is a per node available bandwidth estimation. To estimate the available bandwidth, the authors propose a metric called BWER (Bandwidth Efficiency Ratio) that computes the ratio between the number of transmitted and received packets. To collect the neighbors' available bandwidth, Hello messages are periodically broadcasted in the one-hop vicinity. Then, the available bandwidth of a node is considered as being the minimum of the available bandwidth between the one-hop neighbors and current node.



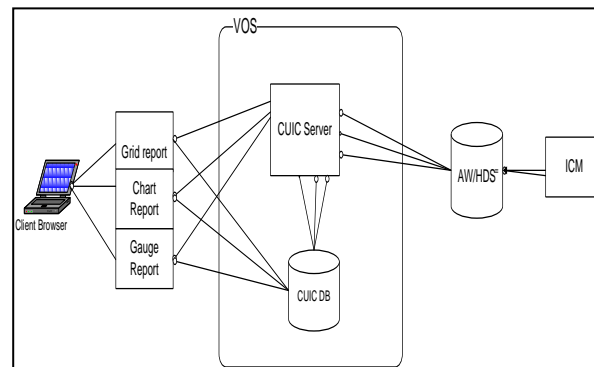
In the protocol AAC, each node estimates its local used bandwidth by simply adding the size of sent and sensed packets over a period of time. The packet size is computed by estimating the medium occupancy time. Therefore this method considers data sent in the carrier sensing area. Finally, the link available bandwidth is the minimum available bandwidth of all nodes belonging to the carrier sensing areas of the sender and the receiver. AAC also estimates the contention count of nodes along a QoS path to solve intra-flow contention problem.

We estimate the available bandwidth of a link based on the previous discussion. In our proposal, we combine three approaches: A time-based listening approach to estimate local and interfering used bandwidth by monitoring the channel utilization. This approach only allows an evaluation of the bandwidth available to a node.

A probabilistic evaluation of the overlap of the silence periods of the two end-points of a link. Measuring this overlap is very difficult in practice as it requires time synchronization between the two peers and additional communication, that's why a probabilistic approach seems more suited. . An estimation of the collision probability on links. These two last estimates require to exchange bandwidth-related information between nodes. This exchange does not rely on dedicated probe packets but can be included in broadcasted packets like Hello messages found in many routing protocols. Therefore, the method we propose can be classified as passive and not intrusive.

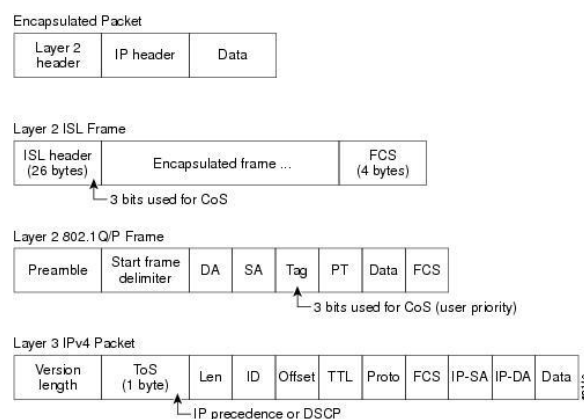


Taking into account collisions



The method presented in the computes an estimation of the overlapping of the silence periods of both emitter and receiver in a probabilistic manner. Therefore, a flow emitting frames at a rate equal to this computed available bandwidth on a link cannot have the guarantee that this throughput will be achieved. It is possible that when a packet is emitted, the receiver available to receive it, leading to a collision. Such collisions lead to retransmissions and to an increase of the contention window, which reduces the real throughput of the flow. For example, the method of suffers from the same limitations as BRuIT, CACP or AAC on configurations like the one depicted. The difference between evaluated and real available bandwidths comes in this case from collisions at node B. Therefore, to provide an accurate estimation, we need to evaluate the collision probability at the receiver side of the link. As we will use Hello messages to exchange information on the available bandwidth per node, we can compute a collision probability based on these Hello messages, denoted by

pHello:



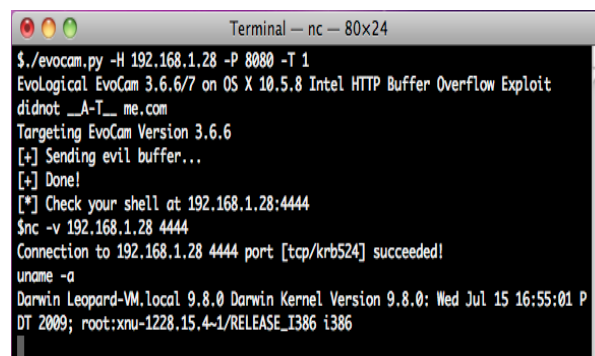
$p_{\text{Hello}} = \frac{\text{number of collided Hello packets}}{\text{number of expected Hello packets}}$

number of expected Hello packets

To compute these values, we consider that each node periodically sends a Hello message and that the period is identical and known. As soon as a node receives a Hello message from one neighbor, it can deduce how many Hello packets it should receive from this neighbor during the next measurement period. This value corresponds to the number of expected Hello packets. The number of collided Hello packets correspond to this expected value minus the number of Hello packets actually received during the considered measurement period. Considering the number of frames that should have been received in a time interval may mix congestion-related effects with collision-related losses. However, when a sender does succeed in sending as many Hello packets as it should due to an overloaded medium, the links associated to this sender will have a low available bandwidth and the inaccuracy in the computation of pHHello will not have a strong impact on the evaluation. Moreover, the introduced error will imply an underestimation on the link, which is preferable to an overestimation. Another strategy could be to rely on sequence numbers of Hello packets to infer the collided packets, nevertheless, it does not give any indication when a hello packet is not received during a consequent time. These two strategies will be compared in future work and could be used side by side.

II. BUFFER OVERFLOW

Buffer overflow is on web pages. The attacker simply feeds a program far more data than it expects to receive. A buffer size is exceeded, and the excess data spill over into adjoining code and data locations. Perhaps the best known web server buffer overflow is the file name problem known as iishack. This attack is so well known that it has been written into a procedure.



```
Terminal -- nc -- 80x24
$ ./evocam.py -H 192.168.1.28 -P 8080 -T 1
EvoLogical EvoCam 3.6.6/7 on OS X 10.5.8 Intel HTTP Buffer Overflow Exploit
didnot _A-T_ me.com
Targeting EvoCam Version 3.6.6
[*] Sending evil buffer...
[*] Done!
[*] Check your shell at 192.168.1.28:4444
$ nc -v 192.168.1.28 4444
Connection to 192.168.1.28 4444 port [tcp/krb524] succeeded!
uname -a
Darwin Leopard-VM.local 9.8.0 Darwin Kernel Version 9.8.0: Wed Jul 15 16:55:01 P
DT 2009; root:xnu-1228.15.4~1/RELEASE_I386 i386
```

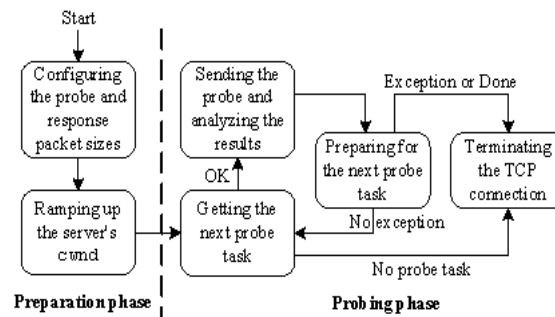
To execute the procedure, an attacker supplies as parameters the site to be attacked and the URL of a program the attacker wants that server to execute. Other web servers are vulnerable to extremely long parameter fields, such as passwords of length 10,000 or a long URL padded with space or null characters.

III. DOT-DOT AND ADDRESS PROBLEM

Web server code should always run in a constrained environment. Ideally, the web server should never have editors, Extern and Telnet programs, or even most system utilities loaded. By constraining the environment in this way, even if an attacker escapes from the web server application, no other executable programs will help the attacker use the web server's computer and operating system to extend the attack. The code and data for web applications can be transferred manually to a web server or pushed as a raw image. But many web applications programmers are naive. They expect to need to edit a web application in place, so they expect to need editors and system utilities to give them a complete environment in which to program. A second, less desirable, condition for preventing an attack is to create a fence confining the web server application. With such a fence, the server application cannot escape from its area and access other potentially dangerous system areas (such as editors and utilities). The server begins in a particular directory subtree, and everything the server needs is in that same subtree.

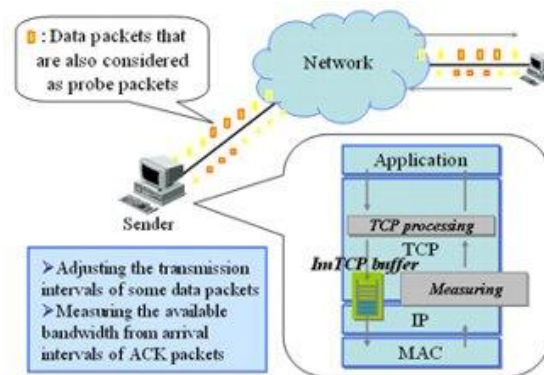
IV. APPLICATION CODE ERRORS

A user's browser carries on an intricate, undocumented protocol interchange with the web server. To make its job easier, the web server passes context strings to the user, making the user's browser reply with full context. A problem arises when the user can modify that context.



V. TRAFFIC REDIRECTION

A router is a device that forwards traffic on its way through intermediate networks between a source host's network and a destination's. So if an attacker can corrupt the routing, traffic can disappear. Routers use complex algorithms to decide how to route traffic. No matter the algorithm, they essentially seek the best path (where "best" is measured in some combination of distance, time, cost, quality, and the like). Routers are aware only of the routers with which they share a direct network connection, and they use gateway protocols to share information about their capabilities. Each router advises its neighbors about how well it can reach other network addresses. This characteristic allows an attacker to disrupt the network. In spite of its sophistication, a router is simply a computer with two or more network interfaces. Suppose a router advertises to its neighbors that it has the best path to every other address in the whole network. Soon all routers will direct all traffic to that one router. The one router may become flooded, or it may simply drop much of its traffic. In either case, a lot of traffic never makes it to the intended destination.

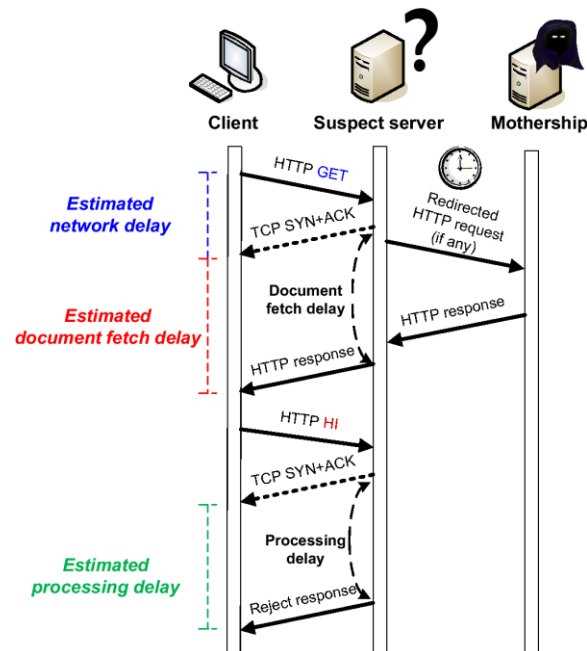


A domain name server (DNS) is a table that converts domain names like ATT.COM into network addresses like 211.217.74.130; this process is called resolving the domain name. A domain name server queries other name servers to resolve domain names it does not know. For efficiency, it caches the answers it receives so it can resolve that name more rapidly in the future. In the most common implementations of Unix, name servers run software called Berkeley Internet Name Domain or BIND or named (a shorthand for "name daemon"). There have been numerous flaws in BIND, including the now familiar buffer overflow. By overtaking a name server or causing it to cache spurious entries, an attacker can redirect the routing of any traffic, with an obvious implication for denial of service.

VI. ESSENTIAL RESOURCES

In the joint allocation of the processor and bandwidth resources, if a certain resource is never the bottleneck, then the fair allocation strategy degenerates to the fair allocation of just the other resource. For example, if the available bandwidth is large enough that no flow experiences congestion due to lack of bandwidth alone, one only needs to worry about the allocation of the processing resources. Fair allocation of a single bottleneck resource has been studied extensively in the literature and has led to a large number of practical algorithms that are in use today in Internet routers, operating systems, and transport-level protocols. This chapter, on the other hand, answers the question of what is a fair allocation when more than one resource is congested and extends the notions of fairness applied to a single resource to systems with multiple heterogeneous resources. We define an

essential resource as one for which a flow's demand does not reduce with an increase in the allocation of other resources to the flow. A number of resources such as the link bandwidth, processor or power, in most contexts, are essential resources. On the other hand, buffer resources in a network are often non-essential resources; for example, in a system with a buffer and a link, a flow uses the buffer only if the link resource is currently unavailable to it, and thus a flow's demand for the buffer resource reduces as more of the link bandwidth is allocated to it. In the system model used in this chapter, we assume that the flows are in competition for resources that are all essential.



We define a pair of resources as related to each other if a flow's demand for one resource uniquely determines its demand for the other resource. Resources in a set are said to be related if each resource is related to every other resource in the set. Resources in real scenarios are almost always related since the demands of a flow for different individual resources are often related to each other. For example, since each packet is associated with certain processing and bandwidth requirements, a specific increase in a flow's demand for link bandwidth is typically associated with a specific increase in its demand for processing resources. A simpler example, involving multiple resources of the same kind, is a tandem network with multiple links where the demand of a flow for bandwidth is the same on all the links. We assume multiple resources that are related, although we make no assumptions on the specific nature of the relationship between a flow's demand for different resources. The existence of a relationship between the demands of a flow for various resources calls for the joint allocation of these resources, as opposed to an independent and separate allocation of the resources.

VII. POCKET-BY-POCKET PROCESSOR AND LINK SHARING

It is apparent that FPLS is an ideally fair but un-implementable policy, in the same sense as GPS. In reality, network traffic is always packetized, and therefore, we next present a practical approximation of FPLS, called *Packet-by-packet Processor and Link Sharing (PPLS)*. The PPLS algorithm extends one of the most practical and simple scheduling strategies, Deficit Round Robin (DRR), used in the allocation of bandwidth on a link. Please refer to a brief description of DRR. The pseudo-code of PPLS. The PPLS algorithm approximates the ideal FPLS in a very similar fashion as DRR achieves an approximation of GPS. The PPLS scheduler maintains a linear list of the backlogged flows, *FlowList*. When the scheduler is initialized, *FlowList* is set to an empty list. For each flow, two variables, instead of one as in DRR, are maintained in the PPLS algorithm: a *processor deficit counter (PDC)* and a *link deficit counter (LDC)*. The link deficit counter is exactly the same as the deficit counter in DRR, which represents the deviation of the bandwidth received by the flow from its ideally fair share.

Packet	Word	Lo Byte	Hi Byte
1 (Frame 1)	1	Data 1 Lo	BOF
	2	Data 2 Lo	Data 2 Hi
	3	Data 3 Lo	Data 3 Hi
	4	Data 4 Lo	Data 4 Hi
2 (Frame 1)	1	Data 5 Lo	Data 5 Hi
	2	Data 6 Lo	Data 6 Hi
	3	Data 7 Lo	Data 7 Hi
	4	Data 8 Lo	Data 8 Hi
3 (Frame 1)	1	Data 9 Lo	Data 9 Hi
	2	Data 10 Lo	Data 10 Hi
	3	Data 1 Lo	BOF
	4	Data 1 Lo	BOF
4 (Frame 2)	1	Data 1 Lo	BOF
	2	Data 3 Lo	Data 2 Hi
	3	Data 3 Lo	Data 3 Hi
	4		

The processor deficit counter, on the other hand, represents the deviation of the processing time allocated to the flow from its ideally fair share.

Thus, each flow in PPLS is assigned two quantum values, a *processor quantum (PQ)* and a *link quantum (LQ)*. When a new packet arrives, the *Enqueue* procedure is invoked. If this packet comes from a new flow, the *Enqueue* procedure appends this flow to the end of the *FlowList* and initializes both of its deficit counters to 0 (lines 8-9). The *Dequeue* procedure (lines 11-38) functions as follows. It serves all flows in the *FlowList* in a round-robin fashion. When the scheduler visits flow *i*, it first increments each of the two deficit counters of this flow by the value of the corresponding quantum (lines 16-17). It then verifies whether or not these two deficit counters exceed their upper bounds respectively, and if they do, it resets them to the maximum possible values (lines 18-23). The rationale behind this bounding process will be discussed later in detail. After the deficit counters of flow *i* are updated, a sequence of packets from flow *i* are scheduled as long as the total length of these packets is smaller than the link deficit counter, and the total processing cost is smaller than the processing deficit counter, as in the **while** loop in lines.

1 *Initialize:*

2 *FlowList* ← NULL;

3 *Enqueue:* /* Invoked whenever a packet arrives */

4 *p* ← *ArrivingPacket*;

5 *i* ← *Flow(p)*; /* Flow of packet *p* */

6 **if** (*ExistsInFlowList* (*i*) = FALSE) **then**

7 Append flow *i* to *FlowList*;

8 *PDCi* ← 0;

9 *LDCi* ← 0;

10 **end if**;

11 *Dequeue:* /* Always running */

12 **while** (TRUE) **do**

13 **if** (*FlowList* ≠ NULL) **then**

```

14  $i \leftarrow \text{HeadOfFlowList}$ ;
15 Remove  $i$  from  $\text{FlowList}$ ;
16  $\text{PDCi} \leftarrow \text{PDCi} + \text{PQi}$ ;
17  $\text{LDCi} \leftarrow \text{LDCi} + \text{LQi}$ ;
18 if ( $\text{PDCi} > \text{maxPDCi}$ ) then
19    $\text{PDCi} \leftarrow \text{maxPDCi}$ ;
20 end if;
21 if ( $\text{LDCi} > \text{maxLDCi}$ ) then
22    $\text{LDCi} \leftarrow \text{maxLDCi}$ ;
23 end if;
24 while ( $\text{QueueIsEmpty}(i) = \text{FALSE}$ ) do
25    $p \leftarrow \text{HeadOfLinePacketInQueue}(i)$ ;
26   if ( $\text{Size}(p) > \text{LDCi}$  OR
27      $\text{Processing Cost}(p) > \text{PDCi}$ ) then
28     break; /* escape from the inner while loop */
29   end if;
30    $\text{PDCi} \leftarrow \text{PDCi} + \text{Processing Cost}(p)$ ;
31    $\text{LDCi} \leftarrow \text{LDCi} + \text{Size}(p)$ ;
32   Schedule  $p$ ;
33 end while;
34 if ( $\text{QueueIsEmpty}(i) = \text{FALSE}$ ) then
35   Append queue  $i$  to  $\text{FlowList}$ ;
36 end if;
37 end if;
38 end while;

```

Pseudo-code of the Packet-by-packet Processor and Link Sharing (PPLS) algorithm. In the meantime, when a packet is scheduled, both deficit counters are decremented by the corresponding cost of this packet. Finally, when the scheduler finishes serving a flow and the flow still remains backlogged, the scheduler places the flow back at the end of the *FlowList*.

Recall that in DRR, for each flow, the quantum is set to be proportional to its weight, therefore, each flow receives in each round, on average, a service with total amount proportional to its weight. In this chapter, the sum of a certain quantity over *all* flows is denoted by dropping the subscript for the flow in the notation. For example, w is the sum of the weights for all flows.

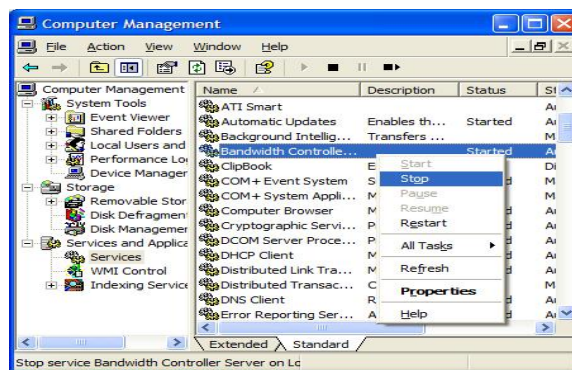
Link Utilization	Mean Data Queuing Delay (ms)	Mean Probe Queuing Delay (ms)	Percentage Error	Computed Data Queuing (ms)	Percentage Error
30%	2.68	1.24	-53.77%	3.19	18.9%
35%	4.30	2.10	-51.12%	4.91	14.08
40%	7.01	3.44	-50.95%	7.34	4.67%
45%	11.33	5.78	-49.01%	11.42	0.79%
50%	18.29	10.65	-41.77%	19.72	7.79%
55%	29.54	19.34	-34.52%	33.40	13.08
60%	47.79	33.74	-29.41%	54.28	13.57
65%	78.90	56.93	-27.84%	85.42	8.27%
70%	153.56	119.24	-22.34%	167.88	9.33%

VIII. CONCLUSION

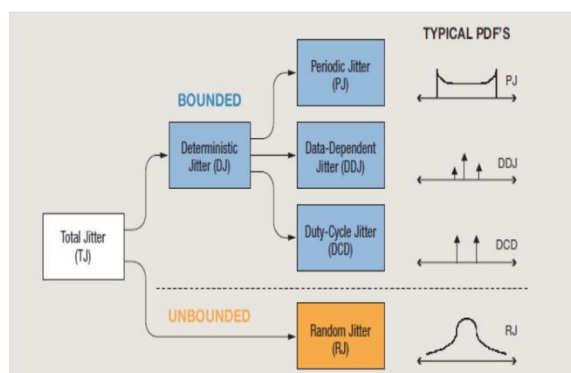
Path selection schemes

We describe the four path selection schemes we evaluate. The schemes are distinguished based on the choice of the key measured network performance metric.

Loss based path selection (LPS): In LPS, we monitor the average loss rate in all candidate paths during 3-second periods. The path with the minimum loss rate is selected. If the currently used path has zero loss rate, then we do not switch to another path even if there are other loss-free paths.



Jitter based path selection (JPS):



The jitter of successive packet pairs is also measured over 3-second periods. The path with the minimum 90th percentile of jitter measurements is selected. If the minimum jitter is practically the same in more than one paths, then JPS selects the path with the lowest loss rate. If the loss rate is also equal, then JPS stays at the current path if that is one of the best paths, or it randomly picks one of the best paths otherwise.

Available-bandwidth based path selection APS): This scheme has two variations. In the first, we use the average available-bandwidth (A-APS). In the second, we use the lower bound of the available-bandwidth variation range (L-APS). A new available-bandwidth estimate results in almost every 3 seconds, similar to LPS and JPS. If the available-bandwidth estimate (average or lower bound) in the currently selected path is greater than twice the video transmission rate, then we stay in that path. Otherwise, we choose the path with the highest available-bandwidth estimate; note that this may still be the currently selected path.

Fairness is an intuitively desirable property in the allocation of resources in a network shared among multiple flows of traffic from different users. During the last decade or two, research on achieving fairness in networks has primarily focused on the allocation of bandwidth. As flows of traffic traverse a network, however, they share various types of network resources such as buffer, processor and power as in mobile systems. A framework based on which one can define fairness in allocation of multiple resources has not yet been established. In this dissertation, we investigate the challenge of achieving fairness in the joint allocation of multiple heterogeneous resources. We generally categorize the systems with multiple resources into two groups: those with *prioritized* resources such as the system with a shared buffer and a shared link, and those with *essential* resources such as the system with a shared processor and a shared link. For both types of systems, we have established fundamental principles to define and measure the fairness in the joint allocation of the shared resources within the system under consideration. These principles, namely the *Principle of Fair Prioritized Resource Allocation* or the FPRA principle and the *Principle of Fair Essential Resource Allocation* or the FERA principle, are simple but powerful generalizations of any given notion of fairness defined on a single shared resource, such as max-min fairness, proportional fairness and utility max-min fairness. We further apply the FPRA principle to the system with a shared buffer and a shared link, and apply the FERA principle to the system with a shared processor and a shared link. Using the notion of max-min fairness as an example, we have developed ideally fair, though un-implementable, allocation strategies in both systems, i.e., the *Fluid-flowFair Buffering (FFB)* and the *Fluid-flow Processor and Link Sharing (FPLS)*, which may be used as benchmarks in the evaluation of the fairness of various practical and implementable allocation schemes in such systems. We anticipate that these algorithms will serve the same purpose as GPS does in research studies on the allocation strategies of a single shared resource. We have presented the *Packet-by-packet Fair Buffering (PFB)* and the *Packet-by-packet Processor and Link Sharing*

(*PPLS*), each of which is an implementable, computationally feasible and *provably fair* approximation of the corresponding ideally fair strategy. We have demonstrated the fairness of PFB and PPLS through extensive simulation experiments using real traffic traces.

Our study in the joint allocation of buffer and bandwidth resources shows that overall fairness is not determined by the exit scheduler alone, but instead by the combination of the entry and the exit policies. This work reveals that use of a fair exit policy such as DRR does not ensure overall fairness when buffer resources are constrained or when packet dropping is used as a congestion control policy. In fact, our study shows that, even though the fairness of exit policies has received far greater attention in the research literature, the entry policy is more critical to overall fairness than the exit policy.

Our study in the joint allocation of processing and bandwidth resources also leads to a similar conclusion. It is illustrated that, in a system with multiple essential resources, achieving fairness with respect to each resource alone does not guarantee the overall fairness in the entire system. In fact, neither the fair allocation of processing resource alone nor the fair allocation of bandwidth resource alone achieves the fair allocation in the overall system with a shared processor and a shared link. Only when these resources are allocated in a coordinated manner, does the allocation strategy such as PPLS provide overall fairness.

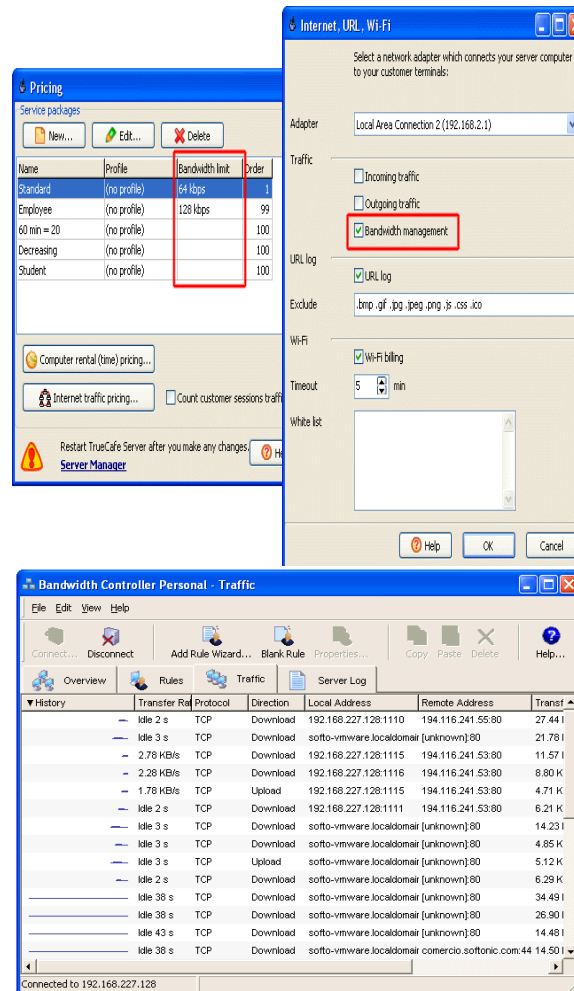
In addition, in order to extend our work to systems with multiple output links, an approach based on system decomposition is also presented in this dissertation. In this method, we decompose a multiple output link system with output links into two types of subsystems:

An *unshared link subsystem* and *shared link subsystems*. Consider a system with a shared buffer and a shared link as an example. The unshared link subsystem consists of *sessions*, each of which contains all flows headed to the same output link, and corresponds to a shared link subsystem associated with the output link. In the unshared link subsystem, the only shared resource is the buffer; in each shared link subsystem, the set of shared resources includes both the buffer and the corresponding link. Therefore, the principles developed in the study of single output link systems can be applied into each of these subsystems, and the fairness in the entire system can be defined based on the fairness in each subsystem.

IX. CONCLUDING REMARKS FUTURE WORK

The Random Early Detection (RED) algorithm has been widely employed in Internet routers to cooperate with TCP end users as a congestion avoidance strategy. The end-to-end congestion control algorithms, implemented in various versions of TCP protocol, depend not only on the bandwidth allocation in the network but also on the packet loss rate as the indication of congestion. Therefore, an unfair management of buffers can cause biased packet loss rates for different flows, and thus lead to a failure in providing end-to-end fairness. Incorporating the principles developed in this dissertation in the design of enhancements to RED-like algorithms for congestion avoidance will likely lead to true fairness in resource allocation as well as an overall improvement in the utilization of the network resources.

Quality-of-service is often an end-to-end issue, of which end-to-end fairness is a critically important piece. Fair bandwidth allocation algorithm such as Weighted Fair Queuing (WFQ) has frequently been used as a component of an overall mechanism that ensures end-to-end delay guarantees. Similarly, mechanisms to achieve end-to-end fairness in the use of all the resources in the network will play a significant role in achieving true end-to-end quality-of-service guarantees. This dissertation focuses on the principles and the design of such mechanisms, serving as the basis for achieving end-to-end fairness. This dissertation is the first attempt to develop theoretical frameworks to define fairness in the allocation of multiple resources. While this work has established a foundation for achieving this goal, it also raises many other possibilities for further investigation. In this dissertation, we have defined the fairness in systems with multiple output links by using Consider buffer allocation in multiple link systems. As one can observe from the fairness definition, the fair allocation policy needs to implement a PFB-like algorithm for each shared link subsystem. In addition, for the unshared link subsystem, the fair allocation policy has to be able to take into consideration the different peak rates of all output links, and achieve a fair distribution of resource dividends among sessions. While we have defined in this dissertation the fairness in the joint allocation of multiple prioritized resources and in the joint allocation of multiple essential resources, a more complicated system may consist of both prioritized and essential prioritized and essential resources exist. Specifically, the shared buffer and the shared link comprise a subsystem with two prioritized resources, while this subsystem and the processor P are both essential to all flows. A future research goal is to develop a definition of fairness for such a system, potentially using the principles proposed in this dissertation. Given that most switches and routers have both prioritized and essential resources, it will be worthwhile to also develop practical strategies for resource allocation in such systems. It is our hope that this dissertation will facilitate future research in the design of *provably* fair strategies for achieving *overall* fairness in *joint* resource allocation.



This phase is based on the SDD. Iteratively, the active user with the highest SDD is the user that degraded in terms of its bandwidth requirements. Calls are degraded according to Table	Degraded Bit-rate
Original Bit-rate	
384Kbps	144Kbps
144Kbps	64Kbps
64Kbps	16Kbps
16Kbps	Not Degraded Further

References

1. R. Prasad, M. Murray, C. Dovrolis, and K. Claffy, "Bandwidth Estimation: Metrics, Measurement Techniques, and Tools," IEEE Network, vol. 17, no. 6, pp. 27-35, Nov. 2003.
2. M. Jain and C. Dovrolis, "End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput," IEEE/ACM Trans. Networking (TON '03), vol. 11, no. 4, pp. 537-549, Aug. 2003.
3. B. Melander, M. Bjorkman, and P. Gunningberg, "A New End-to-End Probing Analysis Method for Estimating Bandwidth Bottlenecks," Proc. Fifth Global Internet Symp. (Global Internet) held in conjunction with Global Comm. Conf. (GLOBECOM '00), Nov. 2000.
4. F.Y. Li, M. Haugea, A. Hafslund, O. Kure, and P. Spilling, "Estimating Residual Bandwidth in 802.11-Based Ad Hoc Networks: An Empirical Approach," Proc. Seventh Int'l Symp. Wireless Personal Multimedia Comm. (WPMC '04), Sept. 2004.
5. A. Johnsson, B. Melander, and M. Björkman, "Bandwidth Measurement in Wireless Network," technical report, Mälardalen Univ., Mar. 2005.
6. S.H. Shah, K. Chen, and K. Nahrstedt, "Dynamic Bandwidth Management for Single-Hop Ad Hoc Wireless Networks," Proc. First IEEE Int'l Conf. Pervasive Computing and Comm. (PerCom '03), Aug. 2003.
7. K. Xu, K. Tang, R. Bagrodia, M. Gerla, and M. Bereschinsky, "Adaptive Bandwidth Management and QoS Provisioning in Large Scale Ad Hoc Networks," Proc. Military Comm. Conf. (MILCOM '03), Oct. 2003.

8. R. de Renesse, M. Ghassemian, V. Friderikos, and A.H. Aghvami, "QoS Enabled Routing in Mobile Ad Hoc Networks," Proc. IEE Fifth Int'l Conf. 3G Mobile Comm. Technologies (IEE 3G), 2004.
9. C. Chaudet and I.G. Lassous, "BRuIT—Bandwidth Reservation under InTerferences Influence," Proc. European Wireless (EW '02), Feb. 2002.
10. C. Chaudet and I.G. Lassous, "Evaluation of the BRuIT Protocol," Proc. IEEE 61st Semiann. Vehicular Technology Conf. (VTC Spring '05), May 2005.
11. Y. Yang and R. Kravets, "Contention Aware Admission Control for Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 4, pp.363-377, 2005.
12. K. Sanzgiri, I.D. Chakeres, and E.M. Belding-Royer, "Determining Intra-Flow Contention along Multihop Paths in Wireless Networks," Proc. First Int'l Conf. Broadband Networks (BROADNETS '04), Oct. 2004.
13. R. de Renesse, M. Ghassemian, V. Friderikos, and A.H. Aghvami, "Adaptive Admission Control for Ad Hoc and Sensor Networks Providing Quality of Service," technical report, King College London, May 2005.
14. V. Bharghavan, A.J. Demers, S. Shenker, and L. Zhang, "MACAW: A Media Access Protocol for Wireless LAN's," Proc. ACM SIGCOMM '94, pp. 212-225, 1994.
15. C. Sarr, C. Chaudet, G. Chelius, and I.G. Lassous, "A Node-Based Available Bandwidth Evaluation in IEEE 802.11 Ad Hoc Networks," Int'l J. Parallel, Emergent and Distributed Systems, vol. 21, no. 6, 2006.
16. G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," J. Selected Areas in Comm., vol. 18, no. 3, pp. 535-547, Mar. 2000.
17. L. Chen and W. Heinzelman, "QoS-Aware Routing Based on Bandwidth Estimation for Mobile Ad Hoc Networks," IEEE J. Selected Areas of Comm., vol. 3, 2005.

AUTHOR(S) PROFILE



Mr K S R Murthy an Assistant professor of the Department of Information Technology Sreenidhi Institute of science and technology, Hyderabad, A..P, India. He is an active Research Scholar at Andhra University in the areas of Ad hock networks Network Security, Cryptography and Mobile Computing, Data Mining.



Dr. G. Samuel Vara Prasada Raju received the M.Tech degree in CSE from Andhra University in 1993. He received PhD degree from Andhra University in 1996. From 1994 to 2003 worked as Asst. Professor in the Dept. of CS&SE in Andhra University College of Engineering. From 2003 onwards worked as Associate Professor and Director of the CS&SE Department for School of Distance Education of Andhra University College of Engineering His research interest includes ecommerce, Network Security, Cryptography and Data Mining. He has more than 20 years experience in Academics and Research