Instantaneous Fairness of TCP in Heterogeneous Traffic Wireless LAN Environments

Young-Jin Jung[1] **and Chang Yun Park**[1*]
[1] School of Computer Science and Engineering, Chung-Ang University, Seoul, Korea
yjjung@cnlab.cse.cau.ac.kr, cypark@cau.ac.kr
*Corresponding author: Chang Yun Park

Received March 3, 2016; revised May 30, 2016; revised July 6, 2016; accepted July 16, 2016; published August 31, 2016

Abstract

Increasingly, numerous and various Internet-capable devices are connected in end user networks, such as a home network. Most devices use the combination of TCP and 802.11 DCF as a system platform, but whereas some devices such as a streaming video persistently generate traffic, others such as a motion sensor do so only intermittently with lots of pauses. This study addresses the issue of performance in this *heterogeneous traffic wireless LAN environment* from the perspective of fairness. First, *instantaneous fairness* is introduced as a notion to indicate how immediately and how closely a user obtains its fair share, and a new time-based metric is defined as an index. Second, extensive simulation experiments have been made with TCP Reno, Vegas, and Westwood to determine how each TCP congestion control corresponds to the instantaneous fairness. Overall, TCP Vegas yields the best instantaneous fairness because it keeps the queue length shorter than the other TCPs. In the simulations, about 60% of a fair share of the effective user bandwidth is immediately usable in any circumstance. Finally, we introduce two simple strategies for adjusting TCP congestion controls to enhance instantaneous fairness and validate them through simulation experiments.

Keywords: TCP Fairness, Instantaneous Fairness, Heterogeneous Traffic, TCP Congestion Control, 802.11 DCF

This research was supported by the Chung-Ang University Graduate Research Scholarship in 2016.

http://dx.doi.org/10.3837/tiis.2016.08.018 **ISSN : 1976-7277**

1. Introduction

Increasingly, numerous and various Internet capable devices are connected in end user networks, such as a home network or an in-car network. Those devices include not only general purpose devices such as tablets and laptop computers but also dedicated purpose devices such as streaming video devices and motion capture sensors. Unless power is restricted, TCP/IP and 802.11 DCF remain a popular system platform for such devices. However, traffic patterns often vary from the conventional assumption that nodes are homogeneous with similar traffic patterns. Some devices persistently generate traffic, and others do so only intermittently with lots of pauses. As the number of the devices grows, contention often arises for wireless link bandwidth and Internet link access. We call this network usage pattern a *heterogeneous traffic wireless LAN environment*.

Fairness is one of the major performance metrics [1] that has been studied extensively in computer networks. However, the results of previous studies do not apply well to a heterogeneous traffic environment because they generally assume homogeneous traffic and focus on how equally each traffic source achieves throughput. For example, for a device that generates traffic only intermittently, measuring fairness based on throughput is meaningless. Moreover, when that device starts to send data after a pause, its traffic could be delayed by the backlogged traffic excessively generated from neighbors during its pause. From the viewpoint of that device, it is acceptable that other devices use its share while it is paused, but it should be able to access its fair share immediately when it starts communicating, regardless of other devices' traffic behavior. *Instantaneous fairness* is the close existing term for that property. However, existing studies, for example [2-4], do not apply because their analyses are based on different assumptions from our heterogeneous traffic wireless LAN environment, where intermittent and persistent traffics are mixed and neither centralized queue management nor traffic reservation are supported.

A fundamental solution for this problem might be to make a wireless LAN QoS support, such as the 802.11e extension. However, it is unlikely that all the various devices in the networks would implement the QoS option, and even if they did, managing network QoS would create another hardship for an ordinary user. A more practical solution might be to use the standard device platform as the default and make performance predictable by simply controlling the number of devices. In an end user network, obtaining fair access to the network in some degree might be sufficient rather than providing a strict performance guarantee. Therefore, we choose fairness as the performance metric rather than a response time guarantee for intermittent traffic.

TCP and 802.11 DCF are major components of the most commonly used platform. The congestion control mechanism of TCP supports long-term fairness among flows sharing the link; thus, the traffic for each source is controlled in some way. On the other hand, 802.11 DCF in principle provides a fair chance to transmit to all contending devices for every frame transmission. Therefore, they provide not only long-term fairness in homogeneous traffic but also the possibility of instantaneous fairness to some degree even in heterogeneous traffic. The objective of this study is to figure out how TCP corresponds to instantaneous fairness in a heterogeneous traffic wireless LAN environment.

This study, first defines the traffic model and the quality of instantaneous fairness in a heterogeneous traffic environment, and introduces a new simple new index for instantaneous fairness. Next, using the index, we have evaluated the instantaneous fairness of three

representative TCPs, Reno, Vegas and Westwood, by simulation. Overall, TCP Vegas performs the best among them; Vegas's collision avoidance keeps queue lengths low, which allows intermittent traffic to recover its share quickly at the end of a pause. Finally, based on that experimental analysis, we have introduced two strategies for adjusting TCPs to achieve better instantaneous fairness and validated them using simulation results.

Most previous studies on quality of services in a user network address reservation-based performance guarantee, which requires new or extended system platforms. We believe that our approach is practical because it can be applied on the platforms currently used, and it is sufficient to estimate service quality because performance at a node in a user network can be approximated based on the degree of fairness provided. A simple and intuitive metric is introduced to measure how closely and how instantly a device can obtain its expected performance on average. We have found that selecting a proper TCP version, a device can achieve about 60% of the fair share of the effective TCP bandwidth immediately, regardless of its neighbors' behavior. We also show that immediacy can be improved even more without sacrificing the aggregate throughput.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 explains the heterogeneous traffic wireless LAN environment in detail and defines instantaneous fairness and its index. Extensive experiments have been made to figure out how existing TCPs perform vis-à-vis instantaneous fairness in section 4. Section 5 introduces two adjustment policies to enhance instantaneous fairness and validates them by simulation. Finally, section 6 concludes the study.

2. Related Work

There have been numerous studies on fairness in computer networks. We summarize those most related to this study in three subareas: TCP fairness, instantaneous fairness, and fairness index. TCP fairness is one of the requirements for TCP congestion control methods; therefore every TCP version satisfies long-term fairness as long as all traffic sources use the same TCP version. For example, [5] proves it in a concise way. Most TCP fairness studies address fairness in a mixed TCP situation in which different TCP versions compete [6-8]. They have found that loss-based TCPs such as Reno are significantly more favored in general than delay-based TCPs such as Vegas. TCP fairness in a wireless LAN has also been covered in many studies, and unfair sharing between upstream and downstream traffic is one of the major issues [9-11]. Interestingly, [12] has addressed the fairness issue between short-lived flows and long-lived flows, which is similar to the intermittent and persistent traffic, respectively, in our heterogeneous traffic environment. That study introduces new queue management schemes that improve unfair treatment against short-lived flows, but it does not consider a fair share. In this study, we measure the degree of fairness as how quickly and closely a device achieves its fair share and suggest TCP adjustments for improving fairness in a distributed way.

Since long-term fairness covers only the average behavior, fairness could be broken in a short interval. For stronger fairness, researchers have also studied instantaneous fairness or short-term fairness [13]. Analyzing fairness at any instance requires the description of each source's throughput as a function of time, which can be very complicated in a general network and traffic environment. Hence most studies on instantaneous fairness assume either homogeneous traffic [3,14] or centralized queue management [2,15]. Our approach in this study is practical in the sense that analysis is done not on general time but per

transmission-task with arbitrary data length. Also, I assume a common real-life environment: heterogeneous traffic across a distributed wireless LAN.

Many definitions of a fairness index exist, and Jain's Index [16] is the representative example in TCP fairness. Depending on how an index value is calculated, they can be broadly grouped into two types. The first is calculated from the performance values that each competitor actually obtains through contention. The other is calculated based on the ideal performance value achievable in a perfectly fair allocation. Jain's Index is an example of the first type, and examples of the second type include [3], which is the ratio to the throughput of GPS (generalized processor sharing) scheduling, and [2], which is the variational distance from the case of uniform distribution. Our instantaneous fairness index is of the second type, the ratio to the case the fair share is perfectly allocated. The distinctive difference is that our index is time-based, which means that the index value is calculated from the finish time of each transmission session, whereas almost all existing indexes are throughput-based. Because the instantaneous fairness index should address immediacy, it is believed that a time-based index is more intuitive than a throughput-based index.

3. Instantaneous Fairness

3.1 Network and Traffic Model

The network environment in this study is an 802.11 wireless LAN, as a user link of Internet such as a home network, where there is a base station, aka AP and several nodes are associated with it. As 802.11 NIC is inexpensively available, it is widely used not only in traditional computing equipment such as tablets but also in a variety of devices such as streaming video devices, CCTVs, and home appliances. Therefore, we assume 802.11 DCF as a link access protocol because it is a default setting that all those devices can support. There is no feature of fixed allocation; fair bandwidth sharing is achieved by on-demand contention. Since our study goal focuses on TCP fairness, we assume for simplicity that the channel conditions of all nodes are the same.

From the point of view of network traffic, some nodes, such as a wireless CCTV camera, are traffic intensive, whereas others, such as a motion sensor, generate traffic sporadically or periodically. Although there are no hard real-time requirements, a fair chance of transmission and quick response are desirable. Most fairness studies have assumed a homogeneous traffic environment in which all competitors generate the same patterns of traffic requests; for example, all the nodes always have backlogged data to transmit. However, that is unrealistic in the environment above. This study concerns fairness in a heterogeneous traffic environment. As in all fairness studies, we assume that enough traffic exists to cause competition among the nodes.

Some nodes, such as sensors, might use not TCP but UDP as a transport protocol, but most of the traffic will still come from TCP. Because UDP does not have a traffic control mechanism, most fairness studies exclude UDP traffic. This study also assumes that UDP traffic does not cause fairness problems; more precisely, the aggregation of all UDP traffic is limited under some portion of the link bandwidth. All TCP users compete for the remaining link bandwidth, and the fairness in that situation is the concern in this study. We also assume that all nodes use the same version of TCP.

This study categorizes user traffic, i.e., application traffic into two types: *persistent* and *intermittent*. Persistent traffic describes data sent without a pause, though the user needs not generate traffic continuously. It is enough that the length of data is unbounded, and hence the

transmission has no end point. Examples include the traffic from a video camera, an Internet radio, or a home cloud server heavily used. Intermittent traffic comes from a user that pauses and sends data in a periodic or sporadic manner. Examples include the traffic from an energy consumption sensor or image capturing sensor. The amount of the intermittent traffic is usually small, but it can sometimes occur in large bursts.

Ideally, a user should be able to use its fair share of the bandwidth whenever it wants, regardless of how other users behave. However, in the real network environment of TCP and 802.11, high utilization is also one of the performance objectives, and bandwidth unused by one user can be used by others. In our traffic classification, the unused fair shares of intermittent user can be used by the persistent users. Because long-term fairness is already provided in the environment, every persistent traffic user can use at least its fair share, and hence, there is no need to consider fairness for them. In fact, the persistent traffic is considered background traffic that is always consuming its share and waiting for the chance to use more. The cost for this opportunistic use is instantaneous fairness for the intermittent user. When an intermittent user starts to transmit after a pause, it should compete for its fair share with the background traffic users already using excessive bandwidth. Therefore, it is worth discussing how quickly and how closely an intermittent traffic user can achieve its fair share.

Our traffic model is formally defined as follows. We call an action in which a user sends data without an intermission a *transmission session*. The *j-th* transmission session of user i is denoted by T_i^j, and each transmission session is specified as (s,l) where *s* is the session start time and *l* is the data length. The traffic of user i is described as a sequence of transmission sessions $U_i = <T_i^0, T_i^1, T_i^2, \cdots T_i^n>$ where *n* is the total number of transmission sessions of user i. Persistent traffic is described as $U_i = <T_i^1 = (0,\infty)>$, which means that a user has one transmission session that starts at time 0 and sends an unbounded length of data. $U_i = <T_i^1 = (0,\infty)>$ corresponds to the infinitely backlogged traffic model frequently used in many fairness studies.

It is worth noting that our traffic model is designed to analyze only fairness: it specifies not the rate of a transmission but only its length. Our reasoning is that the transmission length is determined by the user, but the actual sending rate is determined by the TCP. Also, we assume one TCP user per node, i.e., only one application working at a node, for simplicity in analysis. The case of multiple users at a node is translated in the following simple ways. All the persistent traffic users are merged into one persistent user. For each intermittent traffic user, a new node is added in the network and the user is moved to that node. Although this translation does not keep the exact equivalence in performance, the result would be similar enough to the original to address our objective in this study: the instantaneous fairness of intermittent traffic.

3.2 Definition of Instantaneous Fairness

Instantaneous fairness has been used as a specific terminology to emphasize momentariness. Fairness means the quality of equal resource sharing, and when fairness is used alone, it usually signifies long-term fairness, in which equal sharing is considered for an interval long enough for the system to go into a stable state. Instantaneous fairness, sometimes called short-term fairness, addresses equal sharing for any short interval. From the dictionary definition, another aspect is immediacy. That is, instantaneous fairness could also be used to address how quickly equal sharing is achieved. To the best of our knowledge, no study on TCP

fairness has explicitly addressed this immediacy, which could be understood in the following ways.

First, most fairness studies cover only long-term fairness, and in long-term fairness, momentary unequal sharing does not matter. For example, an earlier arrival of a periodic transmission could cause momentary unequal sharing, but in the fairness evaluation, it is considered to have leveled off over the long interval. The latency for going into a fair state also does not matter. Second, existing fairness studies have assumed a homogeneous traffic environment, often one in which all sources always have backlogged transmissions. Some studies have addressed instantaneous fairness by analyzing the status of sharing for a momentary interval in a microscopic way, but those studies have shown little care for inherently uneven traffic, as in the heterogeneous traffic environment this study assumes.

Traffic requests vary over time in a heterogeneous traffic environment, and hence short-term fairness also varies with time. A few studies have addressed instantaneous fairness in a heterogeneous traffic environment, for example [2], but their time-varying fairness indexes are not intuitive. Moreover, for a node with intermittent transmissions, the fairness during a pause does not matter, and thus it is meaningless to evaluate the fairness for all nodes together. Ideal fairness describes a situation in which the exact fair share is provided immediately whenever a transmission session begins. In summary, fairness in a heterogeneous traffic environment should address 1) how much of its fair share a node can use while it transmits and 2) how immediately a node reaches its fair share after it begins transmitting. Certainly, long-term fairness is inadequate to determine those things. From the dictionary meaning, the term instantaneous fairness seems to be the best-fit, but existing definitions do not include immediacy.

In principle, 802.11 DCF provides instantaneous fairness. Every time the network decides who will access the link, all nodes with data to transmit compete in a memory-less and fair manner: a pause in the past creates neither favor nor disfavor. Ignoring the one time effect of the frame length, each node can access its fair share immediately at any moment. However, TCP is not perfect in our definition of instantaneous fairness, though it provides long-term fairness. With respect to immediacy, different methods for controlling the congestion window have different latencies of effect. Also, because of buffering at intermediate nodes between the TCP ends, such as an AP, the memory-less property is invalid. In short, instantaneous fairness is complicated to analyze and varies widely depending on TCP versions. One of the goals in this study is to figure out which TCP gives better instantaneous fairness and why.

3.3 Metrics: Instantaneous Fairness Index

Many researches have defined a fairness index, and Jain's Index [16] and the Relative Fairness Bound [17] are representative examples. TCP fairness indexes are mostly oriented from Jain's Index. Existing instantaneous fairness indexes are defined similarly to long-term fairness indexes, but they are interval-specific. Almost all existing indexes are throughput-based, which means that the index value is calculated from the throughputs of each competitor. According to our definition of instantaneous, the fairness index should address immediacy. Because the throughput does not directly specify latency, a throughput-based fairness index cannot indicate instantaneous fairness directly. Hence, we introduce a new time-based instantaneous fairness index.

Let B be the bandwidth of a shared link in byte and N be the number of nodes in the link. Then B/N is the ideal link bandwidth share per node without loss from any overhead, such as medium access control overhead. However, due to the inevitable overhead, the effective link bandwidth is less than B. Moreover, the effective user bandwidth at the end, which is the real

throughput applications can get, might be even less than that at the link because of the overhead of the protocol software processing at the node.

Let ρ be the ratio of the effective user bandwidth at the end to *B*. Since our goal is to compare the fairness among different types of TCP congestion control methods, figuring out the exact value of ρ is unnecessary. In other words, the amount of overhead does not need to be analyzed because it is imposed equally, whichever congestion control the TCP applies. From the same reasoning, we assume a single flow per node for simplicity, which means that each node has one transmitting TCP user. Assuming the overhead is evenly imposed across the nodes, the effective bandwidth share at TCP is:

$$S_{TCP}(N) = \rho \frac{B}{N} \qquad where \quad 0 \le \rho \le 1 \tag{1}$$

Supposing TCP is perfectly efficient and fair, this bandwidth share should always be provided to each application; if an application transmits *L* bytes of data, it should finish in $L / S_{TCP}(N)$. However, this is difficult to achieve in practice. In fact, different versions of TCP give different finish times, and hence, those times could be used in evaluating TCP fairness behavior. The main cause of the difference is the congestion control mechanism. As mentioned above, persistent traffic users can transmit more than their fair share in a heterogeneous traffic environment. The amount of data excessively transmitted varies depending not only on the traffic situation but also on different TCP versions' policies for congestion control. In summary, the finish times to transmit *L* bytes of data vary by TCP version, even in the same network and traffic situation.

In this study, the ratio of real finish time to the ideal finish time for each transmission session is used as the basis for the fairness index. The closer it gets to 1, the better instantaneous fairness is achieved. The new instantaneous fairness index is basically the arithmetic average of all the ratio values of transmission sessions that occurred in the experiment.

It is worth noting that the data length of a transmission session, *l*, affects the ratio and fairness index. As it lasts longer, the weight of immediacy becomes smaller on the real finish time. If it is long enough, the ratio is close to one because TCP provides long-term fairness. In the case of infinite length, for example transferring an infinitely long file, which is frequently used in throughput experiments, the finish time cannot be defined. Therefore, transmission sessions with infinite data length are not part of the instantaneous fairness index, even though their traffic indirectly affects the ratio for others.

The formal definition of the new instantaneous fairness index is as follows. Let r_i^j be the ratio of the finish times of transmission session T_i^j. Let the real measured finish time be m_i^j. The ideal finish time is estimated from the fair share determined in Equation (1) and the data length of T_i^j and is denoted by e_i^j. Let l_i^j be the data length of T_i^j (i.e., $T_i^j = (t, l_i^j)$ in the traffic model in Section 3.1). If l_i^j is ∞, m_i^j cannot be measured, and e_i^j is not calculable. Thus, for $l_i^j \neq \infty$, r_i^j is decided as follows.

$$r_i^j = \frac{e_i^j}{m_i^j} = \frac{l_i^j / S_{TCP}(N)}{m_i^j} = \frac{l_i^j}{m_i^j \cdot S_{TCP}(N)} \tag{2}$$

It is worth noting that each ratio is determined not by throughput measurement but by time measurement; that is, it is a time-based metric. In addition, in equation (2), the ratio looks like a function of *N*, but in practice it is almost independent of *N* because the measured finish times, m_i^j, also varies with *N*, the number of users. The value of *N* indirectly affects the amount of contention overhead.

The time ratio in Equation (2) can also be interpreted as the conventional throughput-based fairness definition. **Fig. 1** illustrates the behaviors of transmission rate variation for two cases in an abstract way. The left side is the ideal case with perfect instantaneous fairness and efficient resource sharing, and the right side is a simplified trace of a general TCP transmission rate. Assuming no errors in transmission, the transmission rate is converted to the throughput. There are two traffic users in the network: the upper one is persistent with infinitely backlogged data, and the lower one is intermittent with a transmission session starting at t_{start} with data length L. In the ideal case, the persistent traffic transmits at two times its fair share S until t_s as it takes advantage of the bandwidth left unused by the intermittent traffic. At t_{start}, the intermittent traffic user immediately recovers its fair share and transmits data for the time of L/S. It is the perfect resource sharing case, and hence we use it as the basis for determining the degree of instantaneous fairness.

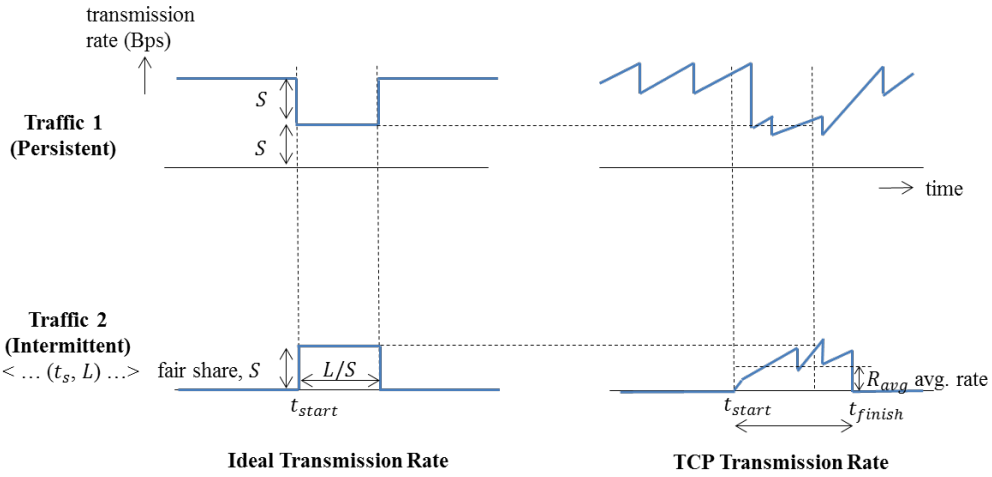


Fig. 1. Abstract Transmission Rate Behavior in Heterogeneous Traffic (Ideal vs. TCP)

The transmission rate of TCP continually varies around the available bandwidth in the shape of a saw blade, which is basically the result of adjusting the congestion control window in an additive increase and multiplicative decrease (AIMD) manner. At the start or after a long timeout the available bandwidth is newly estimated through the slow start process. Hence, in the case of TCP, it takes some time after t_{start} till the intermittent traffic user transmits up to the fair share S. This time lag may be lengthened because the feedback of an intermittent traffic packet suffers delay from competition with packets from existing persistent traffics. Meanwhile a persistent traffic user does not decrease its transmission rate unless a negative feedback is detected. The fair share of the intermittent traffic user will be eventually recovered due to the fair access of 802.11 networks, and the time latency depends on how much and frequently the TCP controls the transmission rate of its traffic. In summary, the time to finish

transmitting L bytes varies depending on which congestion control mechanism the TCP uses.

Suppose the intermittent traffic user finishes its transmission at t_{finish}. Then the average transmission rate R_{avg} in the transmission interval is $L/(t_{finish}-t_{start})$, and this R_{avg} is interpreted as the throughput in the interval. The relative degree of fairness in that interval becomes

$$\text{degree of fairness} = \frac{R_{avg}}{S} = \frac{L/(t_{finish}-t_{start})}{S} = \frac{L}{S\cdot(t_{finish}-t_{start})}$$

L and S are just an instance of l_i^j and $S_{TCP}(N)$ in Equation (2), respectively, and $(t_{finish}-t_{start})$ corresponds to m_i^j, a measured finish time of a transmission session. Therefore, our definition of instantaneous fairness based on time is compatible with the conventional definition based on throughput.

Finally, the instantaneous fairness index *IF* is the arithmetic average of those values of the valid ratio. Let *K* be the total count of the measured finish times (i.e., $K = \left|\left\{m_i^j\right\}\right|$), then *IF* is defined as follows.

$$IF = \frac{\sum r_i^j}{K} \quad \text{where } m_i^j \text{ is available and } l_i^j \neq \infty \tag{3}$$

As usual for fairness indexes, the closer *IF* gets to 1, the better instantaneous fairness is achieved. Moreover, because *IF* is basically the ratio between the ideal and real finish times, the performance of a communication task can be estimated using *IF*. For example, if the *IF* is 0.6 in a wireless LAN with 5 devices, a device can obtain 60% of its fair share, i.e., 60% of one fifth of the maximum user-level bandwidth, at any time. It can also finish its communication task no later than 1.66 (1/0.6) times of the best finish time it can get. Put another way, if the performance estimated using *IF* is unacceptable, the network should be redesigned to either reduce the number of nodes in the network or provide QoS support.

Finally, it is worth mentioning that the index is calculated from not response times but transmission finish times. Given the inevitable errors and delays in transmission, the response time is a more precise performance parameter. This study uses the finish time for the following reasons. First, from the viewpoint of measuring how quickly a user can use its fair share, the finish time at the sender is more intuitive than the response time. Second, purely practically, finish times are easier to acquire, and we found few differences in the results between response times and finish times in our experiments. The only exceptional cases are when the data length of a transmission session is so short that the session is finished without any feedback, i.e., TCP Ack.

4. Experiments: Comparing the Instantaneous Fairness of TCP Versions

4.1 Approximating a Fair Bandwidth Share in 802.11

Computing the instantaneous fairness index requires a knowledge of the fair share of effective bandwidth for each user. Since the objective in the experiments is comparing how well each

TCP version supports instantaneous fairness, the exact value of a fair share is unnecessary. We instead, simply approximate it by a simulation measurement as follows.

First, we estimate the effective bandwidth of a link. For 802.11b, the link bandwidth is 11M bps, but the effective bandwidth is much less than that because of the various sources of overhead. Finding the scenario with the maximum effective bandwidth is not straightforward. For example, as the number of nodes to transmit increases, concurrent transmissions can increase the total throughput by reducing idle times, but the overhead for collision resolution also increases. **Table 1** shows the aggregate throughputs as the sum of the throughputs at all nodes in simulating various network situations. For each of UDP/Download, UDP/Upload, TCP/Download and TCP/Upload, the best scenario to maximize the aggregate throughput is selected (shown in boldface font), and their value will be used as an effective bandwidth for each test case. Instead of the link bandwidth, throughputs at the transport protocol, i.e., end-to-end bandwidths are measured, which is eventually used in computing a fair share.

Table 1. Aggregate Throughputs in Various Situations (Measured in NS2 802.11b)

Test Case		Download	Upload
UDP	1 node	7.83M bps	8.00M bps
	2 nodes	7.83M bps	8.92M bps
	4 nodes	**7.84M bps**	**9.04M bps**
	8 nodes	7.84M bps	8.89M bps
TCP	1 node	4.88M bps	4.97M bps
	2 nodes	4.91M bps	5.02M bps
	4 nodes	**4.96M bps (0.62M Bps)**	5.51M bps
	8 nodes	4.80M bps	**6.88M bps (0.86M Bps)**

Three factors are varied in the test cases: the number of nodes in the wireless LAN, the type of transport protocol (TCP/UDP), and the direction of traffic (upload/download). Because UDP has much less protocol overhead than TCP and no control transmission rate, the UDP cases give higher throughputs than the corresponding TCP cases. It is reasonable that the UDP throughput is the effective link bandwidth and the TCP throughput is the effective end-to-end bandwidth. TCP throughput is shown in the unit of Bps (bytes per second). As discussed in many studies, in 802.11 networks, the throughput for upload traffic is usually higher than that for download traffic. Our NS2 simulation experiments have also shown the same pattern. In most cases, the highest throughput is observed when the number of nodes in a basic service set is 4. The only exception is the TCP upload case, which is highest with 8 nodes transmitting. Overall, the simulation results are consistent with the existing analytic results, for example [11] and the common sense performance measures of 802.11.

As ρB in Equation (1) is determined, $S_{TCP}(N)$ can be simply calculated. For a download traffic user in a network with N nodes, the fair bandwidth share is about $620/N$ KBps. For example, when a network contains 4 nodes, each TCP user should be able to receive 155 Kbytes per second if fairness is provided. In the same network, a fair share for upward traffic users is 215 KBps.

4.2 Instantaneous Fairness of Existing TCP Versions

Because it is practically infeasible to build a controllable network environment, we have evaluated performance by simulation. Since the goal in the experiments is to compare the

performance behaviors of TCP versions, precise and realistic performance measures are unnecessary. The well-known NS2 [18, 19] is used as a simulation tool.

The experimental environment is as follows: A wireless LAN consists of one AP and *n* wireless nodes working in 11 Mbps 802.11 where *n* is an experimental variable that must be an even number. All wireless nodes are placed at the same distance from the AP, about 5 meters, and their channel conditions are considered to be the same. The AP is connected with a server through a wired network whose latency is set to 10 milliseconds. With respect to traffic, one half of the *n* wireless nodes generates persistent traffic, and the other half does intermittent traffic with periodic transmission sessions. The length of a transmission session is also an experimental variable, but for each experiment, all nodes have transmission sessions whose length and period are the same but start times are different. Because a node's fair share in upload is different from that in download, we have made separate experiments in each direction such that all sources generate traffic in a single direction. Bidirectional cases, with half the traffic downloaded and the other half uploaded, have been tested, but those results not shown here because they are basically the average of the results in each direction.

First, we have compared the instantaneous fairness index values of representative TCP versions with download traffic while varying the number of nodes in the network from 2 to 10. Half of the nodes runs the *ftp* application without length, generating as much persistent traffic as possible. Each of the other nodes generates intermittent traffic by running the *ftp* application with 10 Kbytes in the period of 10 seconds. For each of those transmission sessions, the finish time is measured and its ratio to the ideal finish time is calculated using Equation (2) to determine the instantaneous index. The experiments are made separately by changing the TCP version of all nodes. The results of three TCP versions, TCP Reno, TCP Vegas, and TCP Westwood, are shown here because they are well-known representatives with different congestion control policies. We have also tested several other TCP versions, including Linux and Westwood-NR, but they all, even Westwood, give results similar to Reno's in our bit-error-free simulation environment. Therefore, TCP Reno and TCP Vegas are the major comparison targets. The results are shown in **Fig. 2-(a)**. TCP Vegas always gives better instantaneous fairness than TCP Reno and Westwood. With 4 nodes, the index value for TCP Vegas is two times higher, which means that TCP Vegas completes a transmission session two times faster than the other TCPs. As the number of nodes increases, the difference increases; the index values for Reno and Westwood stay mostly the same, but that for Vegas gets higher.

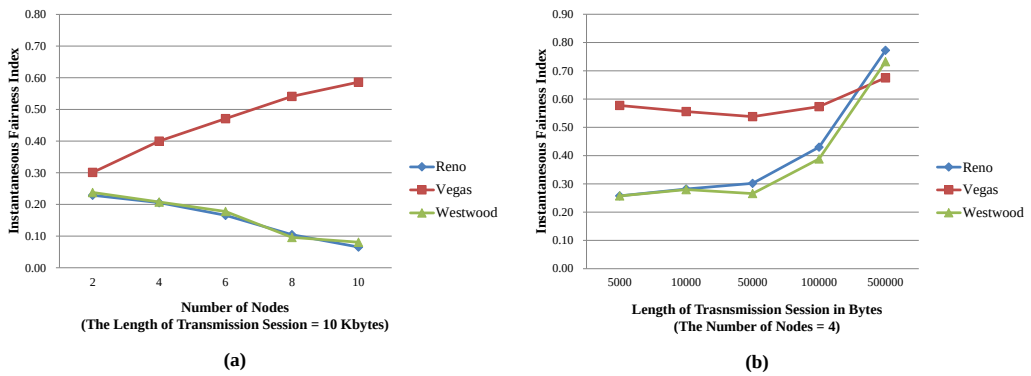


Fig. 2. Instantaneous Fairness Index with Download Traffic

TCP Reno and Westwood keep aggressively sending data until they detect a packet loss, and thus the buffer at the AP is continuously filled with packets from the persistent traffic. It takes a relatively long delay for the initial packets from an intermittent transmission to pass through the buffer. On the other hand, with persistent traffic, TCP Vegas mostly works in the congestion avoidance mode, which manages the congestion window in a precautious manner to avoid an over-accumulation of packets at the buffer [20, 21]. Thus when using TCP Vegas, the initial packets from an intermittent traffic user suffer a shorter queuing delay at the AP buffer, and then the faster TCP feedback speeds up the whole transmission process. That is why TCP Vegas always yields a shorter transmission finish time and a higher instantaneous fairness index value than the other TCPs. As the number of nodes increases, the link and the AP buffer get more congested, but the fair share of a node decreases. The congestion control mechanism in every TCP version maintains a congestion level to some degree, and the finish times increase roughly in proportion to the number of nodes. Because TCP Vegas proactively avoids congestion, the increasing ratio of the finish times could be slower than that of the fair share, and hence its slope is up.

Next, we have repeated the same experiments while varying the data length of the transmission session. Because the performance behavior is basically the same regardless of the number of nodes, the number of nodes is fixed at 4. Transmission length is an important factor in instantaneous fairness because it determines the weight of latency for a TCP to go from the idle state to the fair share. If the data length is sufficiently long, the latency is less important than the throughput achieved in the fair state. The results of these second experiments are shown in **Fig. 2-(b)**. The index values for TCP Vegas vary little, which means that TCP Vegas consistently provides some portion, specifically about 60%, of the fair share from the beginning to the end of every transmission session. With TCP Reno and Westwood, the index values increase as the length of the transmission session grows. When the length is 500 Kbytes, the index values are greater than those of TCP Vegas because the shortcoming in immediacy caused by the aggressiveness of other traffic sources fades as the transmission time increases. In fact, a 500 Kbytes transmission session requires longer than three seconds even in ideal fair sharing, so it looks like a persistent transmission. One thing to note here is that this result does not mean that TCP Vegas is always worse in a long transmission; the length of 500 Kbytes is just one of cases in which TCP Reno and Westwood do better than TCP Vegas. The throughput issue will be explained later.

We have performed the same validation processes for upload traffic. As shown in **Fig. 3**, the overall performance behavior pattern is similar to the download case, but there are differences in detail. When varying the number of nodes, the index values are lower than those for the download traffic overall, and the index values for TCP Reno and Westwood consistently decrease as the number of nodes increases. When varying the length of a transmission session, the reversal point comes earlier than with the download, at 50 Kbytes, and all index values for TCP Vegas are less than 0.5.

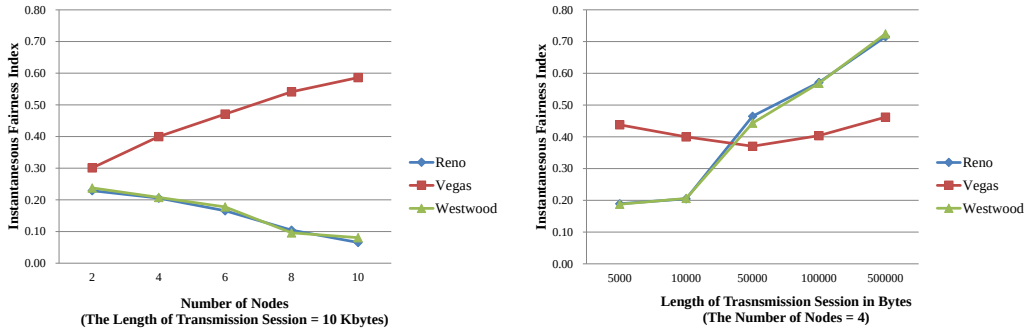


Fig. 3. Instantaneous Fairness Index with Upload Traffic

There are two main reasons for this difference. First, uploading involves multiple data traffic sources, whereas only one source, i.e., AP, is involved in downloading. In principle, because there are more chances to fill the link in uploading than in downloading, the potential end-to-end bandwidth is higher in uploading than in downloading. In effect, as shown in Section 4.1, the effective end-to-end bandwidth and fair share are both higher in uploading. As the fair share increases, the index values decrease in most cases. In TCP Reno and Westwood, more aggressive sources of persistent traffic result in longer latency for a transmission session to go into a fair state, and hence the index values decrease as the number of uploading nodes grows. The second reason is that TCP Vegas does not realize a high actual throughput from the high end-to-end bandwidth during uploading, though TCP Reno and Westwood do well. In competing with multiple major transmitters in 802.11, the reactive, opportunistic, and aggressive approach of TCP Reno and Westwood is a better fit than the proactive, cautious, and altruistic approach of TCP Vegas. As the length of a transmission session grows, the intermittent traffic gets closer to the persistent traffic, and the index value becomes more dependent on the transmission rate than the latency. That explains why Reno and Westwood yield higher index values than Vegas when the length is longer than or equal to 50000, as shown in **Fig. 3-(b)**.

One might doubt that the higher instantaneous fairness index values of TCP Vegas are by-products of its lazier execution in filling the link. To answer this we have also measured and compared throughputs during the fairness experiments described above. **Fig. 4** shows the aggregate throughput as the sum of the goodputs of all traffic sources, including both persistent and intermittent traffic, while varying the number of nodes in download and upload. Half of the nodes are the source of intermittent traffic whose length per transmission session is 10 Kbytes. The other half of the nodes persistently attempt to transmit as much data as possible.

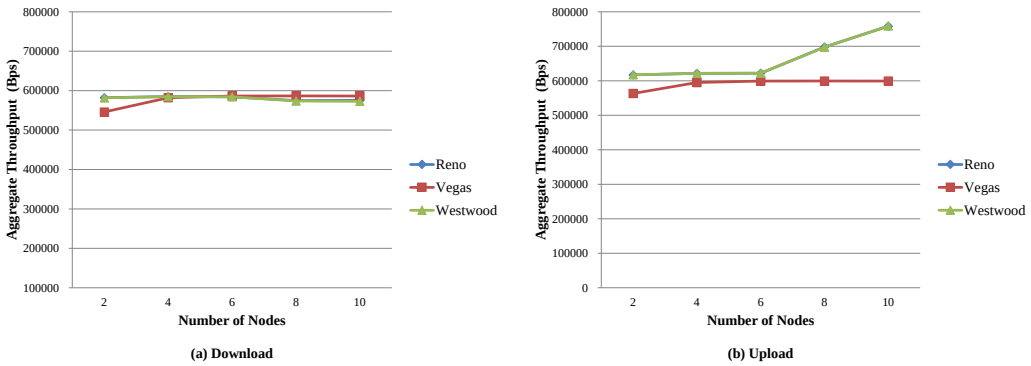


Fig. 4. Aggregate Throughputs When Varying the Number of Nodes

In the download case shown in **Fig. 4-(a)**, TCP Vegas always yields at least a comparable throughput to Reno and Westwood except when the number of nodes is 2. Therefore, the better instantaneous fairness of Vegas does not come from underutilization of the link. It results instead from sharing the link in a manner that maintains instantaneous fairness without sacrificing throughput. In the upload case shown in **Fig. 4-(b)**, Reno and Westwood yield a higher throughput than Vegas when there is no contention or high contention, specifically, when only one or more than 4 nodes are uploading persistently. It is true that Reno and Westwood sacrifice instantaneous fairness in favor of throughput. However, that does not mean that Vegas uses less of the link in favor of fairness. Vegas is just too cautious to be as highly competitive as Reno and Westwood. Considering that downloading accounts for most of the traffic in 802.11, this shortcoming of Vegas does not seem too critical.

5. Simple Adjustments for TCPs To Enhance Instantaneous Fairness

5.1 Two Adjustment Strategies

The experimental results show that two policies of the existing TCPs prevent instantaneous fairness. The one is aggressiveness, which keeps the congestion window higher, especially in TCP Reno. It keeps the buffer at the AP filled with persistent traffic, and therefore when an intermittent transmission begins, a long initial delay occurs. The other is cluelessness about intermittent traffic. TCP is oriented to long-term throughput and does not take much account of intermittent traffic spurts. Not only does idle time in the past not give any favor in the current transmission session, but also the state information from the past session is forgotten during the pause. It takes a long latency for an intermittent transmission to get a fair share, beginning from a slow start in many cases.

To improve instantaneous fairness, these two polices should be adjusted. Each of the following strategies is introduced as the solution for each policy, respectively.

- limit the aggressiveness of persistent traffic,
- encourage intermittent traffic users to be more aggressive.

Before explaining the implementation of those adjustments in detail, it is worth noting that our implementation efforts have been minimal because our goal is to validate the appropriateness and feasibility of the adaptation strategies. Both implementations of the adjustments are far from complete; they are neither optimal nor fine-tuned for better performance. They are only complex enough to show the effects of the adjustments.

The first adjustment is to limit the effective window size. It is applied to TCP Reno because TCP Vegas already keeps the window size lower. In the NS2 simulator it is simply implemented by setting the maxcwin_ parameter of a TCP agent, which means the upper bound on the congestion window for a TCP connection, to some value. The default setting in Reno is 0, which means there is no upper bound. As shown in the previous section, TCP Vegas manages comparable throughputs while keeping the congestion window value around 10. Hence, we simply set the upper bound to 10 and called it "TCP Reno-MaxWin10." TCP Reno-MaxWin10 has the obvious drawback of limiting the maximum throughput a TCP connection can obtain. However, that drawback is not realized as long as multiple sources are competing. For example, in our simulation environment, the drawback isn't a problem as long as more than one node sends persistent traffic.

The second adjustment is to favor an aggressive window increase at the beginning stage of a transmission session after a pause. We have applied it to TCP Vegas to observe the adjustment effect separately and to check whether even TCP Vegas can be improved. Here, we adjusted the Vegas parameters α and β as most Vegas extensions have done at the point of outputting data down to the lower layer protocol [22]. **Fig. 5** shows the code segment added to the TCP Vegas code of NS2 in pseudo code. The modified version is called "TCP Vegas-Fair."

Every time a TCP data segment is sent, Vegas-Fair computes the interval between the current time and the most recent time that it sent a segment. If the interval is greater than a given threshold defining a pause (PAUSE_INTERVAL), a new transmission session is considered detected. The amount of data sent is initialized, and α and β are incremented by one and two, respectively. Because α indicates the stop point for increasing and β does the trigger point for decreasing the congestion window, incrementing both of them gives more chance to increment and less chance to decrement the congestion window. This aggressive management of the congestion window is confined only to the beginning of a transmission session. If the amount of data sent during this transmission session is greater than a given threshold (INTRO_LENGTH), Vegas-Fair decides the introductory phase is over, and the two parameters return to their original values. The values of those thresholds are important because they decide when and for how long a favor is done. In this study, we simply set them to 1 seconds and 5000 bytes to check the effectiveness of the adjustment strategy. Determining optimal values for them would be desirable, but it is outside the scope of this paper.

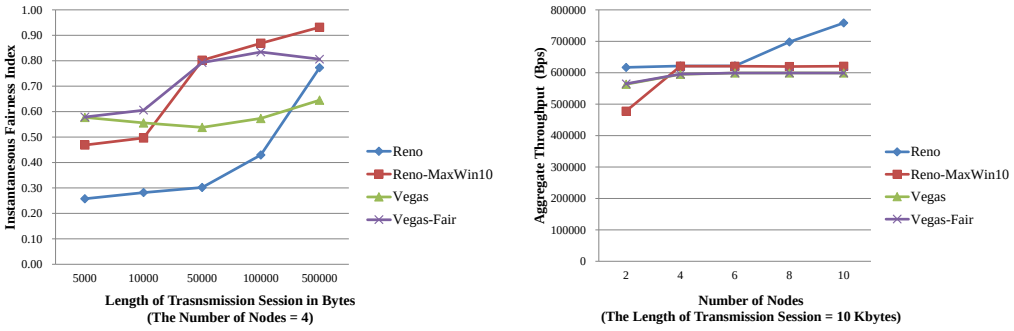
```
...
// inserted at the end of output() of TCP Vegas Code in NS2
currentTime = vegastime();
if (currentTime()-lastSentTime >= PAUSE_INTERVAL) {
    nbyteThisSession = byteSegment;
    if (v_alpha_ == v_alpha_org) {
        v_alpha_ += 1;
        v_beta_ += 2;
    }
} else {
    nbytesThisSession += byteSegment;
    if ( (nbyteThisSession >= INTRO_LENGTH) {
        v_alpha_ = v_alpha_org;
        v_beta_ = v_beta_org;
    }
}
lastSentTime = currentTime();
...
```

Fig. 5. Code Segment of TCP Vegas-Fair

5.2 Validation of the Adjustments

The experimental method for this validation is the same as explained in Section 4.3. We measured the instantaneous fairness indexes and aggregate throughputs of TCP Reno, Reno-MaxWin10, Vegas, and Vegas-Fair. First, as shown in **Fig. 6-(a)**, in download traffic Reno-MaxWin10 always gives higher index values than Reno, and Vegas-Fair gives higher values than Vegas. Thus, each adjustment policy improves instantaneous fairness. It is particularly noteworthy that by simply limiting the maximum of the effective window size, Reno-MaxWin10 behaves similarly to Vegas-Fair.

However, the improvement does not come without cost. **Fig. 6-(b)** shows the aggregate throughputs in upload traffic when varying the number of nodes. Reno-MaxWin10 loses the aggressiveness of Reno and cannot thrive in high competition situations. The more critical problem is that the maximum throughput with little competition is also restricted by the window size limit. It could become worse in a network environment with longer RTTs. An adaptive scheme seems to be required to change the maximum effective window depending on the network and traffic situation. On the other hand, Vegas-Fair shows almost identical throughput behavior with Vegas, indicating the existence of few side-effects. With more tuning of the Vegas-Fair parameters could improve, but more fundamental studies would be required to recapture Reno's throughput in upload traffic. We leave more deliberate adaptation methods for future study.



(a) Download Instantaneous Fairness Index **(b) Upload Aggregate Throughput**

Fig. 6. Aggregate Throughputs When Varying the Number of Nodes

6. Conclusion

In this work, we have addressed performance in a heterogeneous traffic wireless LAN environment from the perspective of fairness. An intermittent traffic user should be able to get fair network access while it intends to communicate. Therefore, we set a fair share of the effective user bandwidth as the ideal goal and call the degree to which a user can immediately achieve it instantaneous fairness. Based on measurements of the finish times of transmission sessions, we have defined a new simple and intuitive index for instantaneous fairness. Then, existing TCPs have been tested because the congestion control methods greatly affect instantaneous fairness. Overall, TCP Vegas provides better instantaneous fairness than Reno and Westwood because its congestion avoidance policy shortens the queue length. It is found that simply selecting a proper TCP version made about 60% of the fair share of the effective TCP bandwidth achievable immediately, regardless of neighbors' traffic behavior. This implies that instantaneous fairness can also be used to estimate network performance. Finally, we have validated that simple TCP adjustments can enhance instantaneous fairness.

More experiments, including measurements in real networks, will give better insight on the usefulness of instantaneous fairness as a performance parameter. Adjusting TCPs to enhance instantaneous fairness has been validated, but more studies are needed to consider and address potential side effects.

References

[1] S. Floyd, "Metrics for the evaluation of congestion control mechanisms," *RFC 5166*, 2008. Article (CrossRef Link)

[2] H. Shi and H. Sethu, "An evaluation of timestamp-based packet schedulers using a novel measure of instantaneous fairness," in *Proc. of the 2003 IEEE International Conference on Performance, Computing, and Communications*, pp. 443-450, 2003. Article (CrossRef Link)

[3] J. Deng, Y. Han, and B. Liang, "Fairness Index Based on Variational Distance," *GLOBECOM 2009*, pp. 1-6, 2009.Article (CrossRef Link)

[4] S. Im, J. Kulkarni, and B. Moseley, "Temporal Fairness of Round Robin: Competitive Analysis for Lk-norms of Flow Time," in *Proc. of the 27th ACM on Symposium on Parallelism in Algorithms and Architectures*, pp. 155-160, 2015. Article (CrossRef Link)

[5] D. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN systems*, vol. 17, no. 1, pp. 1-14, 1989. Article (CrossRef Link)

[6] G. Hasegawa, K. Kurata and M. Murata, "Analysis and improvement of fairness between TCP Reno and Vegas for deployment of TCP Vegas to the Internet," in *Proc. of 2000 International Conference on Network Protocols*, pp. 177-186, 2000. Article (CrossRef Link)
[7] M. Niels and et al., "Inter-protocol fairness between TCP NewReno and TCP Westwood," in *Proc. of 3rd EuroNGI Conference on Next Generation Internet Networks*, vol. 1, pp. 417-439, 2007.
[8] K. Bisoy, Sukant and P. Pattnaik, "Throughput of a Network Shared by TCP Reno and TCP Vegas in Static Multi-hop Wireless Network," *Computational Intelligence in Data Mining—Volume 1*, pp. 471-481, 2016. Article (CrossRef Link)
[9] S. Pokhrel and et al., "TCP Performance over Wi-Fi: Joint Impact of Buffer and Channel Losses," *IEEE Transactions on Mobile Computing*, 2016. Article (CrossRef Link)
[10] S. Priya and K. Murugan, "Enhancing TCP Fairness in Wireless Networks using Dual Queue Approach with Optimal Queue Selection," *Wireless Personal Communications* vol. 83, no. 2, pp. 1359-1372, 2015. Article (CrossRef Link)
[11] S. Pilosof and et al., "Understanding TCP fairness over wireless LAN," *INFOCOM 2003*, vol. 2, pp. 863-872, 2003. Article (CrossRef Link)
[12] Q. Wu, M. Gong, and C. Williamson, "TCP fairness issues in IEEE 802.11 wireless LANs," *Computer Communications*, vol. 31, no. 10, pp. 2150-2161, 2008. Article (CrossRef Link)
[13] N. Al-Ratta, M. Al-Rodhaan, and A. Al-Dhelaan, "FMAC: Fair Mac Protocol for Achieving Proportional Fairness in Multi-Rate WSNs," *Communications and Network*, vol. 7, no. 2, pp.89-105, 2015.Article (CrossRef Link)
[14] H. Jamjoom and K. Shin, "On the role and controllability of persistent clients in traffic aggregates," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, no. 2, pp. 410-423, 2006. Article (CrossRef Link)
[15] F. Baccelli and D. Hong, "The AIMD model for TCP sessions sharing a common router," in *Proc. of the Annual Allerton Conference on Communication Control and Computing*, vol. 39, no. 2, pp. 900-911, 1998.
[16] R. Jain, D. Chiu and W. Hawe, *A quantitative measure of fairness and discrimination for resource allocation in shared computer system*, vol. 38. Eastern Research Laboratory, Digital Equipment Corporation, 1984.
[17] S. Golestani, "A self-clocked fair queueing scheme for broadband applications," *INFOCOM '94.* pp. 636-646, 1994. Article (CrossRef Link)
[18] "Network Simulator ns-2," http://www.isi.edu/nsnam/ns/, 2007. Article (CrossRef Link)
[19] C. Na, J. Chen and T. Rappaport, "Measured traffic statistics and throughput of IEEE 802.11b public WLAN hotspots with three different applications," *IEEE Transactions on Wireless Communications*, vol. 5, no. 11, pp. 3296-3305, 2006. Article (CrossRef Link)
[20] L. Brakmo and L. Peterson, "TCP Vegas: End to end congestion avoidance on a global internet," *IEEE J. of Selected Areas in Communication*, vol. 13, pp. 1465–1480, 1995. Article (CrossRef Link)
[21] M. Massaro, C. Palazzi and A. Bujari. "Exploiting TCP Vegas' algorithm to improve real-time multimedia applications," in *Proc. of Consumer Communications and Networking Conference (CCNC) 12th Annual IEEE*, pp. 316-321, 2015. Article (CrossRef Link)
[22] E. Abolfazli and V. Shah-Mansouri, "Dynamic adjustment of queue levels in TCP Vegas-based networks," *Electronics Letters*, vol.52, no. 5, pp. 361-363, 2016. Article (CrossRef Link)

Young-Jin Jung received the B.S. degrees in computer engineering from Chung-Ang University, Seoul, Korea in 2016. He is currently a M.S. Student in the School of Computer Science and Engineering, Chung-Ang University, Seoul, Korea. His research interests include wireless sensor network, 5G Network, and energy-efficient communication.

Chang Y. Park received the B.S. and M.S. degrees in computer engineering from Seoul National University, Seoul, Korea in 1984 and 1986, respectively, and received the Ph.D. in computer science from the University of Washington, Seattle in 1992. He is currently a Professor in the School of Computer Science and Engineering, Chung-Ang University, Seoul, Korea. His research interests include computer networks, and real-time systems.