Utilizing the Diversities and Invariants of the Internet to Support Multi-Services *

Chuanxiong Guo†, Wenwu Zhu⋆, Zhensheng Zhang⋆, and Zhi-Li Zhang‡
† xguo@ieee.org, the Institute of Communications Engineering, Nanjing, China
⋆ wenwu.zhu@intel.com, Intel Corporation
⋆ zzhang@ieee.org, San Diego Research Center
‡ zhzhang@cs.umn.edu, Minnesota University

ABSTRACT

In this paper, we demonstrate that, by utilizing diverse application requirements and the Internet traffic characteristics, a simple and scalable multi-service support scheme for the Internet can be designed by proposing a harmonious priority-based multi-services support scheme (HARP). In HARP, we use a Discrete Smallest Transmitted Flow first (DSTF) to assign dynamic priorities for Web and background flows based on the number of bytes transmitted, and a fixed priority assignment for real-time and streaming flows based on their delay requirements. HARP harmonizes the co-existence of different flows by providing small queueing delays for real-time and streaming flows, guaranteeing short response time for interactive Web flows, and offering comparable throughput for background flows as in the traditional best effort network. Furthermore, HARP increases the good-put of the network by significantly reducing the packet drop rate. To demonstrate the effectiveness of HARP, simulations are carried out using realistic Internet traffic sources such as Web traffic, real-time and streaming traffic, and traditional best effort traffic. Simulation results indicate that a simple, scalable, and easy-to-deploy scheme can be designed to support multi-services by utilizing the diversities and invariants of the Internet through HARP.

1. INTRODUCTION

To provide multi-services in a single converged network infrastructure is an important and attractive goal, as it provides not only convenience to end users, but also cost saving for service providers in deployment of new services. As the Internet evolves into a universal global information infrastructure, diverse applications such as Web browsing, audio/video streaming, collaborative audio/video conferencing, voice over IP (VoIP) and distant learning can co-exist on the Internet. These diverse applications impose different quality-of-service (QoS) requirements. It is thus natural to expect that the traditional best effort Internet to be enhanced to support the various existing and emerging applications. Research on how to support QoS in the Internet has been carried out for more than a decade, and many architectures (e.g., IntServ [39], DiffServ [4]), frameworks (e.g., H-WF^{2}Q$^+$ [2], CBQ [13], ABE [19], SCORE [32], PDD [11]), and mechanisms (e.g., various fair queueing [10, 15, 29], Priority Queueing) have been proposed and developed.

However, most of the previous works focus on per-packet level QoS parameters such as packet delay, delay variation, bandwidth guarantee, packet loss rate. The per-packet level parameters are most meaningful to real-time and interactive applications that require delay bounds and bandwidth guarantees, while other criteria may be more appropriate for non-real-time, non-interactive applications. For example, per-session level Web response time and average throughput are better criteria for Web browsing and the traditional bulk data transfer applications such as ftp, respectively.

In this paper, based on the Internet measurement studies [5, 6, 7, 14, 21, 25, 33], we observe the following two characteristics in the Internet:

- **Internet applications diversity.** There are various network applications (such as Web browsing, real-time applications, video and audio streaming, and e-mail, ftp, etc.) in the Internet. This is not surprising since the Internet is a platform for all types of applications (known or unknown). These different applications have different requirements (per-packet or per-session) to the network. As we will see in the following example, these requirements may not conflict with each other.

- **Internet traffic invariants.** Measurements show that the Internet traffic has the following invariant characteristics: 1) A few long flows (we call them the elephants thereafter) contribute to most of the traffic volume while the vast short flows (the mice) contribute very few traffic [14, 36]. 2) TCP is the dominant protocol, and Web is the killer application [5, 7, 14, 21]. 3) The traffic volume of the real-time and streaming applications is only a small portion of the total traffic [5, 6, 21, 22]. Though the traffic pattern of the Inter-

*This work was performed while Chuanxiong Guo and Wenwu Zhu were associated with Microsoft Research Asia (MSRA), Zhensheng Zhang and Zhi-Li Zhang were visiting professors at MSRA. Zhi-Li Zhang was supported in part by the National Science Foundation under the grants ITR-0085824 and CNS-0435444

SIGCOMM Asia Workshop'05, Apr. 12–14, 2005, Beijing, China.

net may change as new applications emerge, it is very unlikely that the diverse application requirements and the current Internet traffic characteristics will disappear in the future.

We further observe that, by utilizing the diversity of the application requirements and the invariants of the traffic characteristics of the Internet, the requirements of different applications may be fulfilled by the network harmoniously. Our observation can be demonstrated by the following simple example. Suppose there are 3 flows, f_r, f_w, f_b with queue length 1, 2, 4 packets in the system at time $t = 0$. f_r is a real-time flow, f_w is a Web flow, and f_b is a background flow. The packets are of the same size, and the system transmits 1 packet per unit time. If the packets are transmitted at sequence $S_1 = P_b^0 P_w^0 P_b^1 P_r^0 P_w^1 P_b^2 P_b^3$ (e.g., use fair queueing), the queueing delay for P_r^0 is 3, and the transmit times for f_w and f_b are 5 and 7, respectively. If the packets are transmitted at sequence $S_2 = P_r^0 P_b^0 P_w^0 P_b^1 P_w^1 P_b^2 P_b^3$ (e.g., give high priority to f_r), the queueing delay for p_r^0 is 0, and the transmit times for f_w and f_b are still 5 and 7, respectively. The per-packet queueing time for P_r^0 of S_2 is better than that of S_1 whereas the response times for f_w and f_b are the same. Sequences S_1 and S_2 show that the improvement of packet queueing delay of one flow does not necessarily mean the increase of the transmit time (response time) of other flows. If the packets are transmitted at sequence $S_3 = P_r^0 P_w^0 P_w^1 P_b^0 P_b^1 P_b^2 P_b^3$ (e.g., differentiate f_w and f_b based on flow size), the queueing delay for P_r^0 is 0, and the transmit times for f_w and f_b are 3 and 7, respectively. The response time for f_w of S_3 is better than that of S_2, while the queueing delay for P_r^0 remains the same. In all the sequences, f_b gets the same number of packets transmitted. Sequences S_2 and S_3 show that the improvement of the response time of one flow does not necessarily mean the increase of the per-packet queueing delay of other flows, and different requirements of different applications can be fulfilled by the network harmoniously. This example, though simple, demonstrates that giving higher priority to real-time flows may not increase the response time of Web and background flows (if the traffic volume of the real-time flows is small), and differentiating flows based on their sizes can reduce the mean response time.

Motivated by the above example and observations, in this paper, we propose a novel harmonious priority-based multi-service support mechanism (HARP in short) that utilizes the current Internet traffic characteristics and the complementary requirements of different kinds of applications to provide multi-service support in the Internet. In HARP, Web and background flows are assigned dynamic priority using a DSTF (Discrete Smallest Transmitted Flow first) scheme according to the number of their transmitted bytes, and real-time and streaming flows are assigned fixed priority according to their requirements. Packets then are scheduled via priority queueing. By using HARP, real-time and interactive applications receive small queueing delay, Web based applications have minimized mean response time, and background applications still enjoy the same aggregate throughput as in the traditional best effort network. In other words, the requirements of different applications are fulfilled by HARP harmoniously. It is the very mixture of the Internet traffic that HARP can take advantage of.

The rest of the paper is organized as follows. In Section 2 HARP scheme is presented. DSTF is introduced to assign dynamic priority to Web and background flows and a fixed priority strategy is used to assign priority to real-time and streaming applications. Section 3 analyzes the properties of HARP. Section 4 uses simulations to demonstrate the effectiveness of HARP, i.e., the diverse requirements of different applications can be harmonized by HARP. We discuss related work in Section 5 and conclude the paper in Section 6.

Priority	1	2	3	4	5	6	7	8
W_q	1.11	1.39	1.79	2.38	3.33	5	8.33	66.7

Table 1: Queueing delays of different priority levels.

2. THE HARP SCHEME

HARP is a mixed priority scheduling scheme which combines dynamic and fixed priority queueing. It differentiates Web and best effort flows based on their sizes, and assigns fixed priority to real-time and streaming flows based on their requirements. In the rest of this section, we first give a brief introduction to priority queueing, and then present methods to assign priority to different flows.

2.1 Background: Priority queueing

For a non-preempt priority M/G/1 queueing system with N priorities, the average queueing delays for different priority levels are given by the following equation [3], assume that priority 1 and N are the highest and lowest priority, respectively,

$$W_{q_n}^{prio} = \begin{cases} \frac{1}{1-\rho_1}\frac{\lambda E[X^2]}{2}, & for\ n = 1 \\ \frac{1}{(1-\sum_{i=1}^{n}\rho_i)(1-\sum_{j=1}^{n-1}\rho_j)}\frac{\lambda E[X^2]}{2}, & for\ 1 < n \leq N \end{cases} \quad (1)$$

where λ and X are the arriving rate and the service time of all the packets, respectively, and $\rho_i = \lambda_i E[X_i], 1 < i \leq N$.

For an M/G/1 FCFS system, the mean queueing delay is,

$$W_q^{FCFS} = \frac{1}{1-\rho}\frac{\lambda E[X^2]}{2}. \quad (2)$$

Even though the real Internet traffic can hardly fit the M/G/1 model, the delay trend reflected by (1) and (2) is correct [28]. Suppose that the system has 8 priority levels and $\rho_1 = \rho_2 = \cdots = \rho_7 = 0.1$, $\rho_8 = 0.25$, and $\frac{\lambda E[X^2]}{2} = 1$. Then, $W_q^{FCFS} = 20$ for all the packets under FCFS, whereas the queueing delays for different priority levels under priority queueing are given in Table I.

The comparison shows that most of the packets in priority queueing will experience much smaller queueing delay and only a small portion of packets will experience larger queueing delay under priority queueing than that of FCFS. If the portion experiencing larger queueing delay is from a few very long sized background flows (which is exactly the case since the Internet flow size is heavy-tailed) and does not care much about queuing delay, then the lower queueing delays of the high priority packets are achieved with almost no additional expense. It is this very simple and well-known fact that HARP utilizes to support different traffic.

2.2 Assign priority to Web and best effort flows

For Web flows, session level response time [38] is a more relevant criterion than per-packet level parameters. It is well known from queueing theory that the Shortest Remaining Processing Time first (SRPT) scheduling minimizes the mean response time under the single node case. However, SRPT needs to know the size of the flow at the beginning. This requirement may not be able to be satisfied due to: 1) there are many dynamic generated contents whose size cannot be determined a prior; 2) one TCP connection may be used to transfer multiple files (HTTP 1.1 persistent or pipeline mode), which cannot be determined at the beginning of the request. Further, the flow size cannot be known a prior to routers without a signaling protocol. For these reasons, we introduce a Smallest Transmitted Flow first (STF) scheme to assign dynamic priority for Web and background flows. In STF, a *trans_count* is maintained for every flow to record the number of bytes that have been served. The flow with the smallest *trans_count* has the highest priority. STF thus prefers short flows among competing flows, and it treats flows with approximately equal *trans_count* fairly. When the distribution of flow size is heavy-tailed, only very few flows with large size will experience large queueing delay [see (1)], the difference between STF and SRPT is thus very small. We observe that the performances of STF and SRPT are almost the same in our simulations (see Section 4).

STF has another advantage in that it assigns high priority to packets at the beginning of the flow. This is a very useful property for TCP, since TCP is fragile to packet loss at the beginning due to its slow start behavior. Since STF assigns priority from high to low to a flow other than from low to high as that of SRPT, it will not cause out-of-order packets. From this point of view, STF is more suitable for TCP than SRPT.

We use a discrete version of STF (thus the name DSTF) in practice for the following reasons:

1. Priority can be carried in the packet header (e.g., the TOS byte of IPv4, and the Traffic Class byte of IPv6), thus, packet by packet counting is not needed in the core network.

2. In practice, priority can be assigned without comparing with other flows, thus the time complexity is low and can be achieved in $O(1)$ time.

There are N priority levels, with 1 being the highest priority and N being the lowest priority, and a set of threshold values $\{thresh_1, thresh_2, \cdots, thresh_N\}$ in DSTF, where $thresh_1 = 0$, and $thresh_{i-1} < thresh_i$, $1 < i \leq N$. The priority is set by using the procedure described in Figure 1.

The priority of a packet need to be assigned only once at the host or edge router along its path, then the priority is carried in the packet header. There is a trade-off on simplicity and exactly mimic STF in choosing the threshold values. Certainly the more values we use, the better DSTF can emulate STF, but the harder for the users to choose priority for the real-time and streaming applications. We have studied on how to determine the threshold values of DSTF in [16] by considering the heavy-tail property of Web and background flows. The guideline we use to choose threshold values for DSTF is that the threshold values should make most of the packets experience small queueing delay, whereas only a small portion may experience large queueing delay. In the rest of the paper, the following values are used: $N = 8$ and $thresh_1, thresh_2, \cdots, thresh_8 = 0, 5KB, 10KB, 20KB, 40KB, 100KB, 300KB, 1.2MB$. Since the differences between any pair of adjacent thresholds are larger than the MTU (typically 1500 bytes), the procedure needs to be executed at most twice to assign priority for one packet. Hence, the time complexity of DSTF is $O(1)$.

There are two choices to assign priority for traditional best effort flows. The first is to assign the lowest priority to them. The second is to use DSTF. Since the priority of a flow drops quickly as the number of the transmitted bytes increase, the priority of the long best effort flows will soon drop to the lowest priority if DSTF is used. Therefore, in HARP, we use DSTF for both Web and traditional best effort flows.

---

variables:
  $trans\_count_f = 0$; /* count the already transmitted bytes of flow f */
  $i = 1$; /* memorize the priority assigned to the previous packet of flow f */
when a packet p_f of flow f comes:
loop:
  if ($trans\_count_f \leq thresh_{i+1}$)
    assign priority i to p_f;
  else if ($trans\_count_f > thresh_N$)
    assign priority N to p_f;
  else
    $i++$; goto loop;
  $trans\_count_f += length(p_f)$;

Figure 1: The DSTF dynamic priority assignment procedure. The priority of a flow decreases as the transmitted bytes of the flow increases.

2.3 Assign priority to real-time and streaming flows

In HARP, a fixed priority scheme is used to assign priority to real-time and streaming flows. These flows can choose their priority levels according to their requirements. It can be deduced from (1) that the higher the priority level, the lower the queueing delay. Thus, the stricter the requirement, the higher the priority level should be chosen. We use priorities 1 and 2 for real-time flows, and priorities 3, 4, and 5 for streaming flows in this paper.

Since the real-time and streaming flows generally have higher priorities in the system. Their rate should be monitored and limited to protect other flows. In HARP, the real-time and streaming flows are constrained by a leaky bucket (σ_f, r_f) at the ingress edge of the network, where σ_f and r_f are the depth of the bucket and the average rate of the flow respectively. Note that the leaky bucket filter is only needed to run once at the edge if the edge is trusted by the whole network.

Since real-time and streaming applications occupy only a small portion of the total Internet traffic, we believe HARP priority queueing alone without call admission control (CAC) can deliver good performance to real-time/streaming flows. However, CAC may still be needed to limit the fraction of the real-time/streaming traffic in case this traffic contributes a large portion of the link capacity (e.g., at the edge of the

network where the multiplex gain is not enough; or, as an effect of the deployment of the HARP priority queueing, more real-time/streaming flows are introduced into the Internet). We have designed a simple measurement based CAC scheme for HARP by, not surprisingly, utilizing the property of priority queueing and the Internet traffic characteristics. See [16] for more details.

HARP is not a 'fair' scheme in that packets are treated differently based on their priority. However, due to the Internet traffic characteristics and CAC control, high priority packets will be only a small fraction of the whole traffic, long best effort flows therefore will not be starved. HARP thus harmonizes the co-existence of different flows.

2.4 Discussion

Since HARP needs to assign priority to packets, it is therefore natural to ask the question: where should the priority be tagged? One place is to assign priority at end-hosts. But since not all end-hosts can be fully trusted (end-hosts may be malicious or have been compromised), edge routers therefore have to validate or re-tag the packets anyway. We recommend that edge routers and trusted end-hosts should be the place to assign priority. This means that edge routers must maintain states (bytes transmitted, priority level of a real-time flow, etc.) and perform packet classification. We believe that this is feasible since the number of flows and aggregated traffic at edge routers are not that large.

Another problem faced by HARP is that HARP gives incentive for end-users to split a long flow into several small flows to get better treatment. For Web browsing, it is easy to detect the splitting since the end-users need to use the *Byte Ranges* header field [12] to mark the beginning and end of the data segments. Once the edge router detects that a flow is broken into several small ones, it can punish the flow by lowering its priority level or even terminating the session.

Since HARP gives highest priority to SYN packet of TCP, it may accelerate the propagation of malicious worms since probing is now faster. We believe that multi-service support and Internet worm defense are two research areas that are not tightly coupled, and therefore need to be studied separately. Progresses in Internet worm containment, detection, and prevention can be applied to HARP directly.

3. PROPERTIES OF HARP

Though the design choices of HARP are simple, HARP possesses very good properties: HARP is a scalable scheme and easy-to-deploy; it can provide small queueing delay for real-time flows and the delay distribution decays exponentially fast; it provides small response time for Web flows and the same throughput to background flows; and HARP also achieves better good-put than that of traditional best effort Internet. In what follows, we present and analyze these properties in detail.

Property I: HARP is simple and scalable and can be deployed incrementally.

The two methods to assign priority to packets in HARP are very simple and both can be achieved in $O(1)$ time complexity. Packets are marked at the hosts or the edge routers. After entering the network, they are processed based on priority tags carried in packet header. That is to say, per-flow treatment in HARP is only needed at network edges or end hosts. The priority-based packet scheduling and leaky bucket rate control are also simple schemes and are widely implemented in current routers and hosts. HARP therefore scales well and can be deployed without changing the infrastructure of the current Internet. HARP is useful even if it is partially deployed, e.g., HARP can provide good multi-service support even if it is only deployed at the edges of the network, and the conventional core FCFS network is over-provisioned.

Property II: HARP can provide small mean queueing delay for real-time packets whose priority is high. And the queueing delay distribution of real-time packets decays exponentially.

The small mean queueing delay can be easily derived from (1). Suppose there are N_1 real-time flows with rate X_1, X_2, $\cdots$, X_{N_1}, and N_2 mice flows with rate Y_1, Y_2, $\cdots$, Y_{N_2}, denote $R_X = \sum_{i=1}^{N_1} X_i$, and $R_Y = \sum_{j=1}^{N_2} Y_j$. The rates of the real-time flows are assumed constant. Since the mice flows are short and of high priority, the distribution of the rate of the mice flows can be considered bounded *i.i.d* (identical independent distribution), hence the following result follows via the central limit theorem,

$$\frac{R_X + R_Y - E(R_X + R_Y)}{\sigma(R_X + R_Y)} = \frac{R_Y - E(R_Y)}{\sigma(R_Y)} \to N(0,1). \quad (3)$$

Equation (3) demonstrates that, when the mice flows are of large number (which is exactly the case for the Internet), the aggregate rate of the real-time and the mice flows will fluctuate around the mean value with high probability.

When applying the result to queueing theory, one finds that for a single-server queue, the queue length distribution has asymptotic form

$$P(q > x) \propto e^{-\delta x} \quad (4)$$

where δ is determined by the rate function of the real-time and mice flows arrival process and the service rate of the server (see [23] on how to determine δ). Since queueing delay distribution for real-time flows decays exponentially, large queueing delay is therefore a rare event.

We would like to note that (4) holds in HARP because short TCP flows can be considered i.i.d due to their high priority and short queueing delay. However this i.i.d property may not necessarily hold for the traditional Internet due to the interaction between FCFS/DropTail [1] and TCP congestion control.

Property III: HARP can significantly reduce the mean response time for Web flows when the traffic load is high.

Due to the specifically designed threshold values of HARP, most of the Web packets (except the packets from the very long best effort flows) will experience smaller queueing delay. The shorter the flow size, the smaller the queuing time, hence the smaller the response time. We have the following equation to calculate the response time for a short Web flow.

$$W_{q_n}^{prio} = \begin{cases} RTT\lceil log_2^{FS/MSS+2} \rceil, \\ \quad if FS \le (2^{R_e+1-2})MSS \\ RTT(\lceil (FS/MSS - 2^{R_e+1} + 2)/RWIN \rceil + \\ R_e + 1), if FS > (2^{R_e+1} - 2)MSS \end{cases} \quad (5)$$

where RTT is the round trip time, $RWIN$ is the advertised window of the receiver, $R_e = \lfloor log_2^{RWIN} \rfloor$, FS is the flow size, MSS is the TCP segment size. See [16] for derivation

[1] FCFS for scheduling and DropTail for buffer management. We use FCFS and DropTail interchangeable in this paper.

of (5). From (5), we can see that RT first increases logarithmically when FS is small and then it increases linearly as FS increases.

Only flows with very long size may experience larger response time under HARP. However due to the heavy-tailed property of flow size, the number of flows with very long size is only a very small fraction of the total number of flows. Note that the improvement of the response time when the network is light-loaded may be small, since the reduction of the queueing delay is negligible as compared with the propagation and transmission delays in this case.

Property IV: The aggregate throughputs of the long best effort flows under HARP and the traditional best effort FCFS scheme are the same.

This property can be explained as follows: note that the aggregate throughputs of the Web and background traffic [= (total throughput) - (throughput of the real-time/streaming flows)] are the same under HARP and FCFS. Due to the heavy-tailed property of the Web and background flows, which is determined by the user behavior instead of the scheduling schemes of the network, the traffic volume (hence the throughput) contributed by the long best effort flows will be approximately the same under HARP and FCFS. However, the very long sized flows may experience lower throughput as compared with FCFS due to the larger queueing time (hence larger response time, see Table I and simulation results of Section 4.2.3). As a consequence, there will be more very long sized flows existed simultaneously under HARP to make the aggregate throughput of these flows invariant.

Property V: HARP can decrease the packet drop rate of the network significantly as compared with DropTail under the assumption that the flow size is heavy-tailed.

This counter intuition property of HARP can be explained as follows. From [27], the sending rate of a TCP connection can be expressed by the packet drop rate p as,

$$B(p) = \frac{MSS}{RTT\sqrt{\frac{2bp}{3}} + T_0 min(1, 3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \quad (6)$$

where MSS is the segment size, RTT is the TCP round trip time, T_0 is the initial timeout interval, and b is the number of packets that are acknowledged by a received ACK. Suppose there are N flows competing for the bandwidth whose amount equals C, we have

$$B(p) = \frac{C}{N}. \quad (7)$$

Thus,

$$N = \frac{C(RTT\sqrt{\frac{2bp}{3}} + T_0 min(1, 3\sqrt{\frac{3bp}{8}})p(1+32p^2))}{MSS}. \quad (8)$$

Therefore, once the number of competing TCP flows is determined, the packet drop rate can be calculated from (8). Since packets are prioritized by its flow size under HARP, the drop probability of high priority packets is almost zero. Since the flow size is heavy-tailed, most of the traffic is thus contributed by a few long flows. Therefore, the whole traffic volume C (suppose the link is fully utilized) is at most several times of the traffic volume from long flows C', whereas the total flow number N is of several magnitude of the flow number of long flows N'. Hence based on (8), the packet drop rate of HARP p' is significantly smaller than the packet drop rate p of DropTail. HARP therefore improves the goodput of the network by reducing packet drop rate.

The difference between p' and p can be illustrated by the following example: suppose $C = 10Mb/s$, $RTT = 100ms$, $b = 1$, $T_0 = 1s$, The CDF of the flow size is described by the Pareto distribution $F(x) = 1 - (\frac{k}{x})^{\alpha}$, with $k = 3KB$ and $\alpha = 1.2$. p is 6.5% when $N = 70$ under DropTail by using (8). Based on the property of Pareto distribution, the traffic volume C' and the number of flows N' for flows with sizes larger than 100KB are $C' = 5Mb/s$ and $N' = N(1-F(100KB)) = 1.05$, respectively. Therefore, from (8), we get the packet drop rate under HARP $p' < 0.1\%$, which is much smaller than that of DropTail (6.5% in this case). As we can see in simulation, p' in practice is larger than the value derived from (8) due to the burstiness of higher priority packets. However, it is still much smaller than that of DropTail.

It is worthy to note that the effectiveness and correctness of HARP depend on the traffic distribution and the diverse application requirements. For example, HARP cannot provide good support for media streaming servers where almost all the packets are real-time or streaming packets. In this case HARP performs almost the same as FCFS. Nonetheless, HARP can improve the performance of Web Servers since it treats long and short Web flows differently.

In the next section, we use simulation to verify the properties of HARP and to demonstrate HARP's effectiveness to support multi-services. We would like to point out that the purpose of these simulations is not to show how real the network topology and traffic models are, but to demonstrate the effectiveness of HARP under the diverse application requirements and the Internet traffic characteristics.

4. SIMULATION

In this section, we carry out simulation to study the performance of HARP. The simulation software we use is NS [26] with a NSWEB package [35], in which we added SRPT and DSTF and enhanced the Web client so that it can request pages at the pre-defined time with pre-defined page id, and fixed some bugs of NSWEB. In the following simulations, we compare the performance of HARP with other three schemes. In the first of the three schemes, DropTail is used for Web and background flows and WFQ is used to isolate the real-time and streaming flows from each other and the best effort traffic, and name it DropTail-WFQ (DT-WFQ for abbreviation). DT-WFQ is chosen because WFQ is a well studied scheme that can provide guaranteed bandwidth and delay for real-time flows, and DropTail is the widely used scheme for best effort traffic. In the second of the three schemes, we use SRPT to assign priority for Web and background flows, and fixed priority scheme to assign priority for real-time and streaming flows, and name it SRPT-FP for description convenience. SRPT-FP is chosen to compare the performances of SRPT and STF. In the third of the schemes, we use DropTail. DropTail is chosen since it is the de facto scheme for the traditional best effort Internet.

The topology of the network is shown in Figure 2 and the parameters of the links are shown in Table II. R_0 and R_3 are edge routers, R_1 and R_2 are core routers. In HARP and SRPT-FP, the following threshold values are used: $N = 8$ and $thresh_1$, $thresh_2$, $\cdots$, $thresh_8 = 0$, 5KB, 10KB, 20KB, 40KB, 100KB, 300KB, 1.2MB. The buffer limits of

Link	Bandwidth (Mb/s)	Propagation delay (ms)
$R_0 - R_1$	10	10
$R_1 - R_2$	20	20
$R_2 - R_3$	10	10
$ws_0 - R_0, wb_i - R_3$ $(0 \leq i < 300)$	100	1
$sf_i - R_0, sfsink_i - R_3$ $(1 \leq i \leq 20)$	100	1
$h_i - R_1 (i = 0, 1)$	100	11
$h_i - R_2 (i = 2, 3)$	100	11

Table 2: The bandwidth and propagation delay of each link.

R_0R_1, R_1R_2, and R_2R_3 are set to 125, 250, and 125 packets under DropTail, HARP and SRPT-FP, respectively. The buffer size is set based on the bandwidth delay product [34]. The packets with the lowest priority will be dropped if buffer overflows under HARP and SRPT-FP. In DT-WFQ, each real-time flow is assigned with a weight proportional to its rate, the packets from R_0R_1 are guaranteed 10Mb/s bandwidth at R_1R_2, packets from real-time flows are not dropped, the total buffer assigned to the best effort flows are 100, 200, and 100 packets for R_0R_1, R_1R_2, and R_2R_3 respectively (proportional to the bandwidth assigned to the best effort traffic), and packets from best effort traffic will be dropped from the tail if buffer overflows.

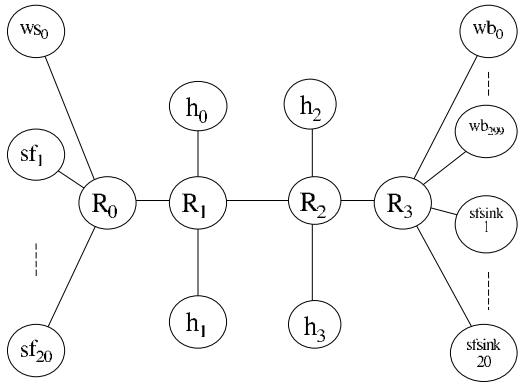


Figure 2: Network topology of the simulation experiments.

The traffic of the simulation is designed as follows. There are 20 CBR flows (representing real-time and streaming flows) between $sf_i - sfsink_i (1 \leq i \leq 20)$, each with rate $10 \times i$ kb/s and packet size 250 bytes. There are 300 clients located in $wb_i (0 \leq i \leq 300)$ using pre-generated access sequences to download contents from the server located in ws_0. ws_0 stores 30,000 pages, a page is composed of an initial file and several embedded files. The size of the initial files on ws_0 is described by the Pareto distribution [i.e., the CDF is described by $F(x) = 1 - (\frac{k}{x})^{\alpha}$] with ($k$=3KB, α=1.2), the distribution of the number of the embedded files in each page and the size of embedded files are described by Pareto distributions with (k=2,α=1.5) and (k=3KB, α=1.2), respectively.

There are also 2 ftp sessions between $h_0 - h_3$, 2 ftp sessions between $h_1 - h_2$. These ftp sessions are always on and are used to consume the surplus capacities of R_1R_2.

The transport protocol used by Web clients is TCP, and the application protocol is HTTP 1.0. We use HTTP 1.0 instead of HTTP 1.1 in this simulation because the flow size cannot be pre-determined for SRPT-FP since one TCP connection may be used to transfer multiple files under HTTP 1.1. We did compare HARP, DropTail, and DT-WFQ using HTTP 1.1, and got similar results. One client can establish at most 4 simultaneous TCP connections. The traffic from the server to the clients represents interactive Web browsing traffic (short Web flows) and the background traffic (long flows).

The access sequences used by wb_i are pre-generated by the following procedure. In this procedure, the simulation environment is the same as described above and HARP is used. Each client randomly selects a page from the page pool of the server, requests the page, and then thinks/pauses for some time after the page (with all its embedded files) is downloaded. The think/pause time of the clients is described by a Pareto distribution with ($k = t$ second, α=1.5). By adjusting t, different aggregate traffic volume can be generated. The time when the new request is issued, the page id, and the client id are logged to be used by the following experiments.

The traffic models we use in this simulation are based on recent progresses on Internet traffic measurement [5, 14] and Web modeling [8, 20].

In the following simulations, two access sequences are used, one sequence is generated with think time (k= 24 second, α=1.5) and the other with (k= 12 second, α=1.5). In the former case, the network is not congested, whereas the network is congested in the later case. We then show the performances of the schemes (the end-to-end delays of the real-time flows, the Web response time, and the throughput of the best effort traffic) under these 2 access sequences. All the simulations are run for one hour and the data collected during the first 100s is ignored for steady statistics purpose.

4.1 When R_0R_1 is not congested

In this simulation, the priority of the CBR/real-time flows is set to 1 under HARP and SRPT-FP, the rate reserved for the CBR flows is equal to their source rate under DT-WFQ, and the pre-defined access sequence of wb_i is generated with think/pause time (k=24 seconds, α=1.5). Figure 3 illustrates the output rate at link R_0R_1 averaged in 5 seconds under HARP. It can be observed that the traffic is very burst, which reflects the traffic characteristics of the Internet. The average link utilizations of R_0R_1 under all the 4 schemes are about 0.65 in this simulation. The total packet loss rates under all the schemes are very low (<0.03%).

4.1.1 End-to-end delays of the real-time flows

The end-to-end delays of the CBR flows are shown in Figure 4(a) and 4(b). The delays of the CBR flows under DT-WFQ are in inverse proportion to the reserved bandwidth which reflects the property of WFQ [29, 30], and the delays for the low bit rate flows are rather large; for example, the average and maximum delays for a 40kb/s flow are 79ms and 172ms respectively. This experiment demonstrates that fair queueing is not suitable for low bit rate flows, especially considering that the network is not congested in this simulation.

Even though the network is not congested, the maximum

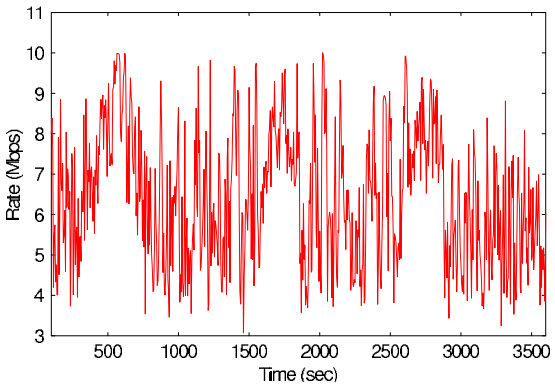
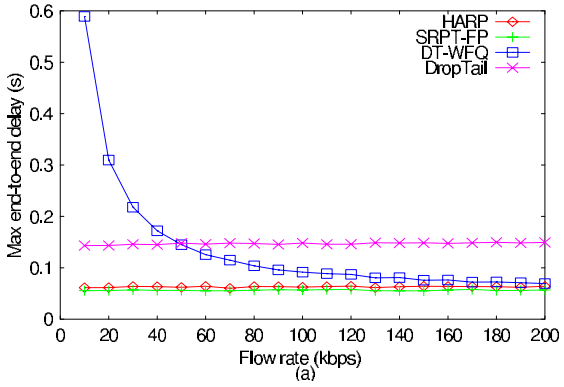


Figure 3: The output rate at R_0R_1 generated by the simulation, which demonstrates the burstiness of the generated traffic.

end-to-end delays of the CBR flows are large (150ms) under DropTail. The result also shows that the end-to-end delays of the CBR flows under HARP and SRPT-FP are good and are not coupled with the rates of the flows. The result is interesting in that even though WFQ can provide guaranteed service which cannot be provided by HARP theoretically, the maximum and mean delays of the real-time flows provided by HARP are much better than that of WFQ at least in this simulation.



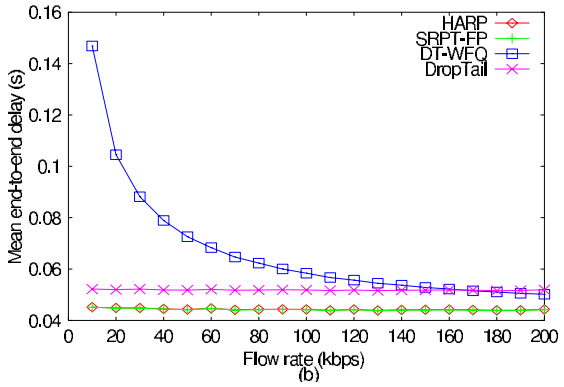


Figure 4: (a) Maximum and (b) mean end-to-end delays of the CBR flows.

4.1.2 *Response times of the Web and background flows*

Figure 5 illustrates the mean response times of the Web and background flows versus flow sizes. Note that the scale of the x-y axis is lognormal. The response time is defined as the time interval from the time the client begins to request a page to the time the client receives the last byte of the page (that is, all the embedded files are transferred). Since the network is light-loaded in this simulation, the response times do not have much difference under all the 4 schemes.

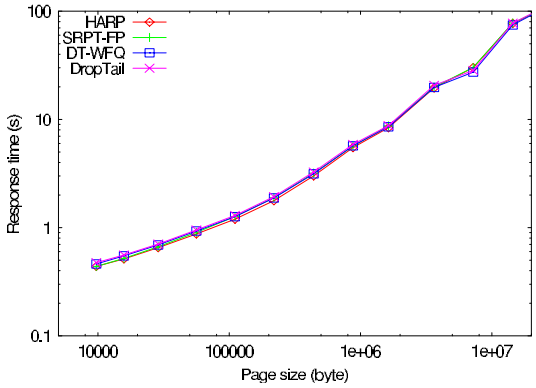


Figure 5: Response time versus flow size (log-log scale). The response times under the four schemes are almost the same when the network is light-loaded.

4.1.3 *Aggregate throughput of long flow*

We also study the throughput of the long flows of the Web and best effort flows (flow is defined by the 5-tuple <protocol, src, src_port, dst, dst_port>), and the result shows that about 44% (830M bytes) of the Web and background traffic is from the long flows (>100KB) under all the 4 schemes. The reason why the long flows get the same kind of traffic volume in all the 4 schemes has been addressed in Property IV of Section 3.

This experiment demonstrates that HARP improves the delays of the real-time flows without increasing the response time of the Web and background flows as observed in the example of Section 1.

4.2 When R_0R_1 is congested

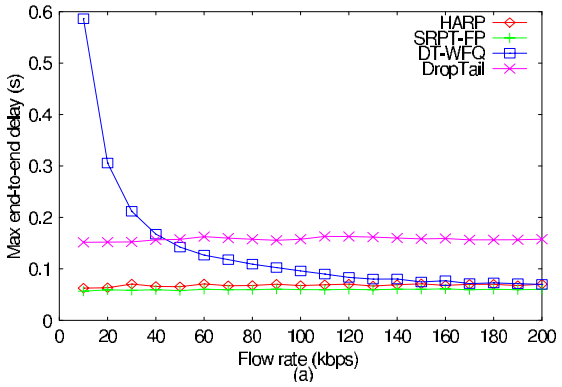
In this simulation, the parameters of the think time of the pre-generated access sequence are (k=12 seconds, α=1.5). The other conditions are the same as in the previous simulation. Thus, the requests from the clients will be doubled if the bandwidth is large enough. In this experiment, the link utilizations at R_0R_1 of the 4 schemes are perfect (approaching 100%). The packet drop rates are 0.4%, 0.2%, 3.3%, 6.5% under HARP, SRPT-FP, DT-WFQ, and DropTail, respectively. The packet drop rate under HARP is significantly smaller than that of DropTail, which confirms Property V.

4.2.1 *End-to-end delays of the real-time flows*

As depicted in Figure 6, under DT-WFQ, the worst-case delays of the CBR flows are almost identical to the previous simulation and the mean delays of the CBR flows are much worse than that of Figure 4(b) due to network congestion. The result also indicates that the performance of DropTail is rather bad in terms of delay (max: 160ms, mean: 120ms) and packet drop rate (6.5%). However, the performances of HARP and SRPT-FP (no one CBR packet is dropped and

the delays are small and not affected by the network congestion) are much better than that of DT-WFQ and DropTail.

The maximum delays of the CBR flows are very close to the mean delays under HARP in both simulations. The results indicate that large queueing delay under HARP is actually a very small probability event, which conform to property II.



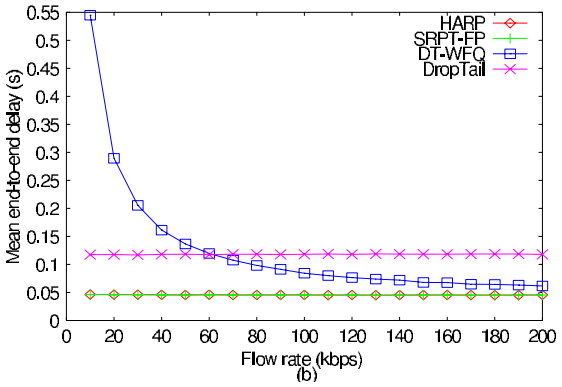


Figure 6: (a) Maximum and (b) mean end-to-end delays of the CBR flows.

4.2.2 *Response times of the Web and background flows*

The response times versus the page sizes of the 4 schemes in this simulation are illustrated in Figure 7. The result indicates that both HARP and SRPT-FP provide much better response time than the other two schemes when the network is heavily loaded; for example, the response times are 0.44s, 0.44s, 2.3s, 2.1s for a 10KB page, 1.4s, 1.4s, 7.5s, and 5.5s for a 100KB page under HARP, SRPT-FP, DT-WFQ, and DropTail, respectively. Therefore, both HARP and SRPT-FP can improve the response time for short flows dramatically. The result also indicates that the response times for very large-size pages (>4MB) under HARP and SRPT-FP are larger than that under DT-WFQ and DropTail. However, this kind of discrimination may be tolerable due to the fact that,

1. Among the 200,000 flows transferred in the simulation, only 42 (0.02%) flows are larger than 4M bytes, and the response time is less critical for large file downloading;

2. The discrimination in response time does not mean that the long flows get less traffic volume. As we will see in Section 4.2.3, the aggregate throughputs of even the very long flows are approximately the same in all the 4 schemes.

The result also shows that the response times of HARP and SRPT-FP are almost identical, which conforms that DSTF and SRPT have almost identical performance.

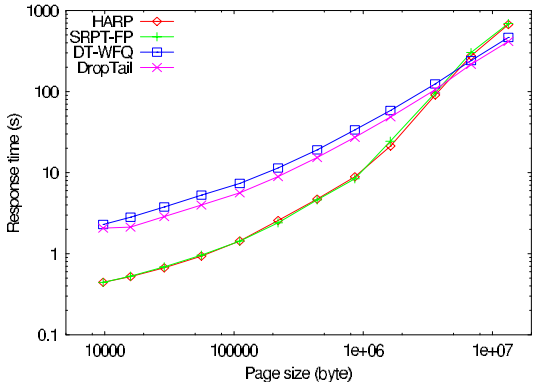


Figure 7: Response time versus flow size (log-log scale). The result demonstrates that the response times of HARP and SRPT-FP are almost identical and are much better than that of the other two schemes when the network is congested.

4.2.3 *Aggregate throughput of long flows*

This simulation demonstrates again that about 44% (1.5 GB) of the Web and background traffic are from flows with sizes > 100KB; about 17% (600MB) of the Web and background traffic are from 42 very long flows with sizes > 4MB, under all the 4 schemes. This result suggests that, though the response time of a single very long best effort flow may be large under HARP (hence the throughput may be low), the aggregate throughputs of the long best effort flows are approximately the same under HARP and the traditional DropTail scheme even under congested condition.

Both experiments under congestion and light-loaded conditions demonstrate that the requirements of the real-time flows, the interactive Web flows, and the background flows can be harmoniously fulfilled by HARP without degrading the performance of each other, which also conforms the observation of the example of Section 1.

We also varied the priority assigned to the CBR flows, and measured the end-to-end delays of the CBR flows and the response times of the Web and best effort flows. Due to space limitation, we only give the result here. The result demonstrates that priority 1 and 2 can provide good worst-case and mean delays for real-time traffic and priority 3, 4, and 5 can be used for streaming flows, and assigning different priority to the real-time and streaming flows does not affect the performance of the Web and background flows under HARP. Therefore, HARP is a robust scheme that different applications do not interfere with each other.

5. RELATED WORK

Supporting multi-services in a converged Internet is an important and attractive research goal. Many architectures (e.g., IntServ [39], DiffServ [4]), frameworks (e.g., H-WF^{2}Q$^+$ [2], CBQ [13], SCORE [32], PDD [11]), and mechanisms (e.g., various FQ, priority queueing) have been introduced

along this line of research. However, in most of the schemes, more attention is given to real-time applications.

IntServ uses per-flow resource reservation to guarantee QoS. Parekh and Gallager's work [29, 30] is the theoretical foundation for this approach. It states that if a flow f is constrained by a leaky bucket with parameters (σ_f, r_f), and Weighted Fair Queueing (WFQ) [10] is used as the packet scheduler in nodes along the path the flow traverses, the end-to-end delay D_f of the flow is bounded and is in inverse proportion to r_f. This result is attractive in that the delay bound is only related to the properties of the flow itself and the number of hops it traverses. Besides the scalability and deployment issues faced by IntServ, we further mention two not-so-obvious shortcomings of FQ.

- D_f may be very large for a low bit rate real-time flow,thus making the end-to-end delay meaningless in practice. For example, the maximum and the average end-to-end delays of a 10kb/s flow in Section 4.1.1 are 0.59s and 0.15s even if the network is not congested. The delays are much larger than the 0.1s upper bound that people can bear for real-time applications like VoIP. We further argue that giving different delay bounds to different flows based on their reserved rates may not be a feasible choice, since the delay requirement of a real-time flow may have nothing to do with its rate. We note that this is also the problem faced by SCORE [32] since it has the same worst-case delay property as WFQ.

- For the dominant Web applications where per-session response time is a much better criterion than the per-packet level QoS metrics [38], FQ may not be a good choice (perhaps the worst choice). This is because FQ schedules flows by emulating GPS (Generalized Processor Sharing), thus the response times for the Web flows will be equally bad.

In DiffServ, the network is separated into the core and the edge, and packets are aggregated into classes described by different PHBs (Per-Hop Behavior). DiffServ thus scales well. HARP is very similar in spirit to DiffServ. It can be considered as a new mechanism that fits well into the DiffServ architecture. HARP can be used to implement the Class Selector behaviors PHBs [24]. The priorities of HARP can be mapped into codepoints 000000b to 111000b naturally.

In ABE [19], packets belong to one single best effort class, and are marked either green or blue. Green packets are expected to receive low bounded delay, and blue packets are expected to receive same or better services as compared with the traditional best effort network. Green packets may experience high packet loss rate and low throughput in ABE. ABE is an alternative best effort approach, thus may not suitable to provide multi-service support in the Internet.

Since Web is the most important application in the Internet nowadays, many studies have been performed on how to improve the performance of Web [1, 9, 17, 18, 31, 37]. SRPT scheduling is used to improve the response time of the short Web flows in [9] and [18]; LAS scheduling [31], which is similar to our STF, is shown to perform very close to SRPT to minimize mean response time; a new TCP version SAReno is proposed to differentiate flows based on their residual sizes in [37]; and RIO-like schemes are designed to differentiate flows into two classes in [17]. All the schemes utilize the Web length distribution to provide better support for short web flows, thereby improving the mean response time, but ignore the requirement of other types of flows

Though HARP benefits a lot from the previous lessons and techniques, we consider HARP is unique in that it is a multi-service support scheme designed from 'bottom up', which means that HARP tries to understand the diverse application requirements and the Internet traffic characteristics first, it then utilizes them to achieve its goal (i.e., multi-service support) by designing simple schemes. HARP is also unique in that it uses different evaluation criterion for different applications (i.e., max and mean delays for real-time and streaming flow, response time for Web flow, and throughput for background flow), and makes all the flows co-exist harmoniously in the network with their requirements fulfilled simultaneously.

6. CONCLUSION

The contributions of this paper can be summarized as follows.

- This paper makes the observation that the network can support applications with different requirements harmoniously by utilizing the diverse application requirements and the Internet traffic characteristics.

- HARP is proposed to realize the above observation. HARP is a priority-based multi-service support mechanism which differentiates flows by using different priority assignment strategies for different applications based on their requirements. A Discrete Smallest Transmitted Flows first (DSTF) scheme is proposed for Web and background applications based on the number of their transmitted bytes and a fixed priority assignment is used for real-time and streaming applications, by utilizing the property of priority queueing and the Internet traffic characteristics. DSTF is a simple and practical scheme which does not need to know the flow size a prior and can work well with the TCP protocol.

- The properties of HARP are analyzed and simulations were carried out under realistic traffic models: real-time and streaming flows, Web flows, and best effort data transfer. Both the analysis and the simulation results indicate that under HARP, real-time and streaming flows experience very low end-to-end delay and enjoy bandwidth guarantee, Web flows have short response time, while background traffic still enjoy the same aggregate throughput, and furthermore, the packet drop rate can be significantly reduced.

The diverse application requirements and the unique characteristics of the Internet traffic are considered as two invariants of this dynamic network at application and network level respectively. HARP thus brings a new and promising idea for providing multi-services in a converged Internet by taking advantage of these two invariants due to its simplicity, scalability, and easy-to-deploy.

7. ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their valuable comments.

8. REFERENCES

[1] Nikhil Bansal and Mor Harchol-Balter. Analysis of SRPT scheduling: Investigating unfairness. In *Proc. Sigmetrics*, 2001.

[2] J. Bennett and H. Zhang. Hierarchical packet fair queueing algorithms. In *Proc. SIGCOMM*, 1996.

[3] Dimitri Bertsekas and Robert Gallager. *Data Networks*. Prentice Hall, 2 edition, 1992.

[4] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services, December 1998. RFC2475.

[5] Nevil Brownlee and Kc Claffy. Understanding Internet traffic streams: Dragonflies and tortoises. *IEEE Communications Magazine*, 40(10), Oct. 2002.

[6] Maureen Chesire, Alec Wolman, Geoffrey M. Voelker, and Henry M. Levy. Measurement and analysis of a streaming-media workload. In *Proc. USITS*, 2001.

[7] K. Claffy. The nature of the beast: Recent traffic measurements from an Internet backbone, 1998. http://www.caida.org/outreach/papers/1998/Inet98/index.xml.

[8] M. Crovella and A. Bestavros. Self-similarity in World Wide Web: Evidence and possible causes . *IEEE/ACM trans. Networking*, 5(6), Dec. 1997.

[9] M. Crovella, R. Frangioso, and M. Harchol-Balter. Connection scheduling in Web servers. In *Proc. USITS*, 1999.

[10] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *Proc. SIGCOMM*, 1989.

[11] C. Dovrolis, D. Stiliadis, and P. Ramanathan. Proportional differentiated services: Delay differentiation and packet scheduling. *IEEE/ACM trans. Networking*, 10(1), Feb. 2002.

[12] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1, June 1999. RFC2616.

[13] Sally Floyd and Van Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM trans. Networking*, 3(4), Aug. 1995.

[14] C. Fraleigh, S. Moon, C. Diot, B. Lyles, and F. Tobagi. Packet-level traffic measurements from a tier-1 IP backbone. Technical Report TR01-ATL-1101011, Sprint ATL, November 2001.

[15] Chuanxiong Guo. SRR: An O(1) time complexity packet scheduler for flows in multi-service packet networks. In *Proc. SIGCOMM*, 2001.

[16] Chuanxiong Guo, Wenwu Zhu, Zhensheng Zhang, and Zhi-Li Zhang. Utilizing the Diversities and Invariants of the Internet to Support Multi-Services. Technical Report MSR-TR-2004-46, Microsoft Research, 2004.

[17] L. Guo and I. Matta. The war between mice and elephants. In *Proc. ICNP*, 2001.

[18] Mor Harchol-Balter, Bianca Schroeder, Mukesh Agrawal, and Nikhil Bansal. Size-based scheduling to improve Web performance, 1998. http://www-2.cs.cmu.edu/~harchol/Papers/insubmission.ps.

[19] P. Hurley, Mourad Kara, J. Y. Le Boudec, and P. Thiran. ABE: Providing a low-delay service within best effort. *IEEE Network*, 15(3), May/June 2001.

[20] Bruce A. Mah. An empirical model of HTTP network traffic. In *Proc. infocom*, 1997.

[21] Sean McCreary and K. Claffy. Trends in wide area IP traffic patterns. http://www.caida.org/outreach/papers/2000/AIX0005/AIX0005.htm.

[22] Art Mena and John Heidemann. An empirical study of real audio traffic. In *Proc. infocom*, 2000.

[23] R. Mortier, I. Pratt, C. Clark, and S. Crosby. Implicit admission control. *IEEE JSAC*, 18(12), Dec. 2000.

[24] K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers, December 1998. RFC2474.

[25] NLANR. The NLANR project. http://www.nlanr.net.

[26] NS. The NS network simulator. http://www.isi.edu/nsnam/ns.

[27] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Reno performance: A simple model and its empirical validation. *IEEE/ACM trans. Networking*, 8(2), Apr. 2000.

[28] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, F. Tobagi, and C. Diot. Analysis of measured single-hop delay from an operational backbone network. In *Proc. infocom*, 2002.

[29] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services network: The single node case. *IEEE/ACM trans. Networking*, 1(3), June 1993.

[30] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services network: The multiple node case. *IEEE/ACM trans. Networking*, 2(2), Apr. 1994.

[31] Idris A. Rai, Guillaume Urvoy-Keller, and Ernst W. Biersack. Analysis of scheduling for job size distributions with high variance. In *Proc. Sigmetrics*, 2003.

[32] I. Stoica. *Stateless core: a scalable approach for quality of service in the Internet*. PhD thesis, Carnegie Mellon University, Department of Electrical and Computer Engineering, December 2000.

[33] K. Thompson, G.J. Miller, and R. Wilder. Wide area internet traffic patterns and characteristics. *IEEE Network*, 11(6), Nov. 1997.

[34] C. Villamizar and C. Song. High performance TCP in ANSNET. *Computer Communication Review*, 24(5), October 1994.

[35] Jorg Wallerich. NSWEB. http://www.net.uni-sb.de/~jw/nsweb/.

[36] WorldCup98. World Cup 98's WWW access traces. http://ita.ee.lbl.gov/html/contrib/WorldCup.html.

[37] Shanchieh Yang and Gustavo de Veciana. Size-based adaptive bandwidth allocation: Optimizing the average QoS for elastic flows. In *Proc. infocom*, 2002.

[38] Mazen Zari, Hossein Saiedian, and Muhammad Naeem. Understanding and reducing Web delays. *IEEE Computer*, 34(12):30–37, Dec. 2001.

[39] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new Resource ReServation Protocol. *IEEE Network*, 7(5), September 1993.