# Generalized Load Sharing for Packet-Switching Networks I: Theory and Packet-Based Algorithm

Ka-Cheong Leung, *Member*, *IEEE*, and Victor O.K. Li, *Fellow*, *IEEE*

**Abstract**—In this paper, we propose a framework to study how to effectively perform load sharing in multipath communication networks. A generalized load sharing (GLS) model has been developed to conceptualize how traffic is split ideally on a set of active paths. A simple traffic splitting algorithm, called packet-by-packet weighted fair routing (PWFR), has been developed to approximate GLS with the given routing weight vector by transmitting each packet as a whole. We have developed some performance bounds for PWFR and found that PWFR is a deterministically fair traffic splitting algorithm. This attractive property is useful in the provision of service with guaranteed performance when multiple paths can be used simultaneously to transmit packets which belong to the same flow. Our simulation studies, based on a collection of Internet backbone traces, reveal that PWFR outperforms two other traffic splitting algorithms, namely, packet-by-packet generalized round robin routing (PGRR), and packet-by-packet probabilistic routing (PPRR).

**Index Terms**—Computer communications, dispersity routing, high speed networks, inverse multiplexing, load sharing, multipath routing, multiprotocol label switching, network striping, performance modeling, traffic dispersion, traffic engineering.

## 1 Introduction

Load sharing is a channel aggregation approach that permits data traffic to be dispersed on multiple channels to reduce network load fluctuations. Given a set of paths connecting a source to a destination, the key to multipath routing is the traffic splitter, a scheduler which distributes traffic to the paths based on the given optimal traffic split.

The most common form of traffic splitting distributes packets to a set of active paths in a round robin fashion [2]. Such an algorithm is quite simple to implement. However, it can only support uniform traffic splitting or cyclic dispersion. With heterogeneous paths, the best way to spread traffic along multiple paths may not be by cyclic dispersion, since it may not achieve the objective, such as minimizing the end-to-end path delay. Such limitation can be alleviated by using a splitter which routes packets in a generalized round robin fashion, according to the given traffic split. Nevertheless, these routing approaches implement the desired traffic split in terms of the number of packets being routed, rather than the actual workload in bytes. Since packets are generally of different sizes in packet-switched networks, except in Asynchronous Transfer Mode (ATM) networks, the actual workloads to paths may deviate unboundedly from the expected workloads. A set of standardized protocols that currently support packet-based load sharing includes inverse multiplexing for ATM (IMA) [2], multilink Point-to-Point Protocol (MP) [11], and link aggregation for carrier sense multiple access with collision detection (CSMA/CD) [6].

### 1.1 Our Contributions

The focus of this work, first described in [8], is to propose a framework to study how to effectively perform load sharing in multipath communication networks. A generalized load sharing (GLS) model has been developed to conceptualize how traffic is split ideally on a set of active paths. A simple traffic splitting algorithm, called packet-by-packet weighted fair routing (PWFR), has been devised to approximate GLS with the given routing weight vector by transmitting each packet as a whole. We have developed some performance bounds for PWFR and found that PWFR is a deterministically fair traffic splitting algorithm. This attractive property is useful in the provision of service with guaranteed performance when multiple paths can be used simultaneously to transmit packets which belong to the same flow.

A technique has been developed in [1] to transform a causal fair queueing algorithm to a load sharing algorithm. A causal fair queueing algorithm, such as round robin, depends only on the previous packets sent to choose the current queue (or flow) to serve. By applying the proposed transformation technique, the function to push packets to paths for any converted load sharing algorithm is completely characterized by the state of the splitter. However, many well-known fair queueing algorithms providing tight delay bounds, such as weighted fair queueing (WFQ) [5], also known as packet-by-packet generalized processor sharing (PGPS) [10], and worst-case fair weighted fair queueing ($WF^2Q$) [3], are noncausal. Moreover, the function to assign packets to paths for PWFR does not merely depend on the state of the splitter, but also on the sizes of the packets to be sent in order to minimize the maximum

• *The authors are with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong, China. E-mail: kcleung@ieee.org, vli@eee.hku.hk.*

*Manuscript received 2 Feb. 2004; revised 17 Feb. 2005; accepted 2 June 2005; published online 25 May 2006.*
*Recommended for acceptance by J. Wu.*
*For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-0038-0204.*

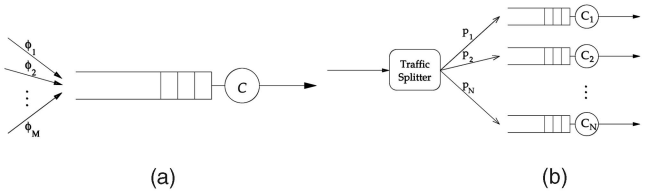



Fig. 1. The roles of processor sharing and load sharing. (a) Processor sharing. (b) Load sharing.

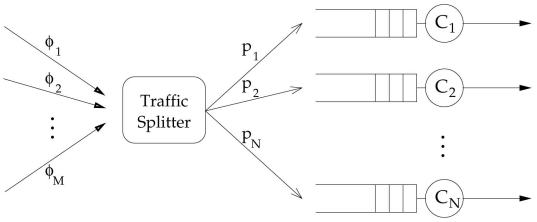



Fig. 2. A generalized sharing model.

residual workload among all paths. Though PWFR is similar to WFQ (or PGPS), it cannot be transformed from any noncausal fair queueing algorithm by employing the transformation technique described in [1].

### 1.2 Organization of the Paper

This paper is organized as follows. Section 2 develops a generalized load sharing model to conceptualize how traffic is split on a set of active paths. Section 3 presents a packetized weighted fair routing algorithm to achieve the best load balancing according to the given traffic split. Section 4 outlines a set of traffic splitting algorithms for performance studies, defines the performance metric that will be used to compare various splitting algorithms, briefly describes the characteristics of the traffic traces used for the simulation, and examines the simulation results of the proposed traffic splitting algorithms. Section 5 concludes and discusses some possible extensions to our work.

## 2 The Generalized Load Sharing Model

In [1], [4], it has been observed that there is a close relationship between link sharing and load sharing. Processor sharing is a channel multiplexing technique that allows different traffic flows to share the same link or path simultaneously to improve link utilization, whereas load sharing is a channel aggregation approach that permits data traffic to be dispersed on multiple channels at the same time to reduce network load fluctuation. The roles of these two techniques are diagrammed in Fig. 1.

Both techniques are aimed at the efficient use of network resources to minimize network congestion and they can be applied in a complementary manner. For example, a set of $M$ traffic flows can be multiplexed to share a set of $N$ outgoing paths, as shown in Fig. 2.

A methodology that is similar to generalized processor sharing (GPS) [10] can be applied for load sharing. Our load sharing model consists of a network node and a set of $N$ (logical) outgoing links or paths, namely, Path 1, Path 2, ..., Path $N$, as shown in Fig. 1b. Without loss of generality, we assume that there is a single class of traffic for traffic splitting. In other words, for ease of presentation, all incoming traffic to the captioned network node will be treated by the same traffic splitter, installed with a routing weight vector. Indeed, it is simple to generalize to cases with multiple classes, each of which can correspond to a destination with a quality of service (QoS) class. Incoming traffic is fed into a traffic splitter for load sharing, according to its traffic class.

When the node receives an incoming packet destined to another node, a traffic splitter installed in the network node is invoked to decide which outgoing path the packet is forwarded to. Once an assignment is made, the packet will be dispatched to an output queue of the corresponding path to wait for transmission. Suppose $p_i$ is the routing weight for Path $i$. It denotes the portion of dispersed traffic to be routed on Path $i$, where $\sum_{i=1}^{N} p_i = 1$. Denote the routing weight vector by $\boldsymbol{p} = (\, p_1 \quad p_2 \quad \ldots \quad p_N \,)$. Given the routing weight vector $\boldsymbol{p}$, the quality of a traffic splitter depends on how close to $\boldsymbol{p}$ the actual load distribution is at all times.

Let $L_i(\tau, t)$ be the amount of traffic (in bytes) prepared to be sent on Path $i$ during the time interval $(\tau, t]$. Since departing traffic will be routed on one of the outgoing paths, the total amount of incoming traffic to be forwarded to the destination during the time interval $(\tau, t]$, $T(\tau, t)$, is equal to the sum of all traffic scheduled to be routed on each outgoing path to the same destination during the same time interval. That is,

$$T(\tau, t) = \sum_{i=1}^{N} L_i(\tau, t) \tag{1}$$

for any period $(\tau, t]$.

A generalized load sharing (GLS) traffic splitter is one that can divide traffic to the set of outgoing paths *exactly* according to the given routing weight vector $\boldsymbol{p}$. Thus, for any period $(\tau, t]$, a GLS traffic splitter[1] is defined [4] as one for which

$$L_i(\tau, t) = p_i \, T(\tau, t) \tag{2}$$

or

$$\frac{L_i(\tau, t)}{L_j(\tau, t)} = \frac{p_i}{p_j} \tag{3}$$

for every $i, j = 1, 2, \ldots, N$.

If a traffic source is constrained by a leaky bucket, the following lemma states that all dispersed flows split by a GLS traffic splitter are also leaky bucket constrained.

**Lemma 1.** *Suppose a traffic source is constrained by a leaky bucket with parameters $(\sigma, \rho)$, where $\sigma$ and $\rho$ represent the maximum size of the bucket token pool and the average token generation rate, respectively. If it is fed into a GLS traffic splitter, a dispersed flow to be routed on Path $i$, where $i = 1, 2, \ldots, N$, is constrained by another leaky bucket with parameters $(p_i \, \sigma, \, p_i \, \rho)$.*

To summarize, there are two major attractive features for GLS. First, it distributes traffic *exactly* according to

1. In [4], this type of ideal load balancing is termed *ideal traffic splitting*.

a set of prescribed routing weights. It is always fair deterministically, where the actual traffic routed to any path is equal to the expected one, for all time intervals. Second, if a GPS server works with a GLS traffic splitter, it is possible to make worst-case network queueing delay guarantees when the flows are constrained by leaky buckets. It follows directly from [10] that the worst-case network queueing delay of a leaky bucket constrained source is bounded under a GPS system.

## 3 Packet-by-Packet Weighted Fair Routing

The generalized load sharing (GLS) approach is an idealized scheme that assumes input traffic to be infinitely divisible for routing, but the smallest possible data unit for routing in the packet-switched networks is a packet. Thus, a load sharing approach, called packet-by-packet weighted fair routing (PWFR), is proposed. The main philosophy of the proposed scheme is to minimize the deviation of the actual load distribution from the given routing weight vector in making each routing decision. In PWFR, a set of packets is split on a set of (logical) outgoing links or paths.

### 3.1 Algorithm

Suppose there is a sequence of packets, namely, Packet 1, Packet 2, ..., to be split on a set of $N$ (logical) paths or channels. Denote the size of Packet $k$ by $S(k)$ bytes. The routing weight for Path $i$ is given as $p_i$, where $\sum_{i=1}^{N} p_i = 1$. Define the routing weight vector as $\boldsymbol{p} = (p_1 \quad p_2 \quad \dots \quad p_N)$. Let $W_i^p(k)$ and $\hat{W}_i^p(k)$, respectively, be the expected workload (in bytes), based on $p_i$, and the actual workload (in bytes) to be sent on Path $i$, just after the routing decision for the $k$th packet has been made. Without loss of generality, we assume $p_i > 0$ and define

$$W_i^p(0) = \hat{W}_i^p(0) = 0 \tag{4}$$

for all $i = 1, 2, \dots, N$.

For an idealized load sharing performed by a GLS traffic splitter,

$$W_i^p(k) = p_i \cdot \sum_{j=1}^{k} S(j), \tag{5}$$

where $i = 1, 2, \dots, N$.

By the conservation of traffic, there is neither creation nor destruction of traffic when a routing decision is made by any traffic splitters. This means that the total traffic to be routed by all paths is independent of what traffic splitter is chosen. Therefore,

$$\sum_{i=1}^{N} W_i^p(k) = \sum_{i=1}^{N} \hat{W}_i^p(k) = \sum_{j=1}^{k} S(j). \tag{6}$$

The idea of PWFR is to simulate GLS as closely as possible, with the constraint that each packet is transmitted on a path as a whole, so as to minimize the maximum residual workload among all paths. The assignment of a complete packet to a path may cause a *transient* load imbalance with respect to the targeted routing weight vector. That is, some paths may be fed more traffic than expected temporarily while other paths may have less, after the routing decision for a certain packet, say, Packet $(k-1)$, is made. Those paths fed with more traffic than expected have the tendency of not having the next packet, i.e., Packet $k$, routed on them. However, if Packet $k$ is large and an overloaded path has a large routing weight, it may still be preferable to send the packet on this overloaded path. Therefore, both the current level of load imbalance as well as the size of the next successive packet are required for the traffic splitter to make the next routing decision.

**procedure PWFR_Packet(***Packet* **)**
**begin**
  $S \leftarrow$ *Packet.size*
  for each path *i* from 1 to N
    $R_i^p \leftarrow R_i^p + p_i\ S$
  choose a path *j* such that $R_j^p$ is maximized [a]
  place *Packet* to the output queue of Path *j*
  $R_j^p \leftarrow R_j^p - S$
**end.**

[a] Ties are broken by a path with the largest routing weight, and, if still unresolved, the smallest path identification number.

Fig. 3. The PWFR algorithm.

To quantify the above selection criterion, a metric is introduced to measure the traffic underload on a path. The residual workload of Path $i$, where $i = 1, 2, \dots, N$, just *before* the routing decision for Packet $k$ is made, $R_i^p(k)$, is defined as the amount of work (in bytes) that should be fed on Path $i$ in order to achieve the expected workload $W_i^p(k)$. In other words,

$$R_i^p(k) = \begin{cases} W_i^p(1) & \text{if } k = 1; \\ W_i^p(k) - \hat{W}_i^p(k-1) & \text{otherwise} \end{cases} \tag{7}$$

and, hence,

$$\sum_{i=1}^{N} R_i^p(k) = S(k). \tag{8}$$

We use $R_i^p(k)$ to measure the traffic underload on Path $i$, just before the routing decision of Packet $k$ is made. If $R_i^p(k) > 0$, Path $i$ has been injected with less traffic than expected and, hence, Packet $k$ can be sent on this path. On the other hand, if $R_i^p(k) < 0$, Path $i$ has too much traffic being routed on it and, hence, Packet $k$ should not be transmitted on this path. In other words, $R_i^p(k)$ provides an indicator to the traffic splitter for deciding which path Packet $k$ should be transmitted on. Packet $k$ is to be sent on Path $j$ when $R_j^p(k)$ is maximized along all participating paths. Ties are broken by a path with the largest routing weight and, if still unresolved, the smallest path identification number. The complete PWFR algorithm is summarized in Fig. 3.

As far as the time and space complexities are concerned, PWFR takes $O(N)$ time for processing each packet as it searches for a path $j$ such that $R_j^p$ is maximized. PWFR needs $O(N)$ counters to store its working variables. As the number of paths $N$ is generally small and fixed, we consider that the computational and storage costs are minimal with respect to the number of traffic flows maintained at each router, nowadays. In the following section, we are going to show that, given that all packets are bounded in sizes and the number of paths is fixed, PWFR is a deterministically fair traffic splitting algorithm.

### 3.2 Performance Bounds

A traffic splitting algorithm is *deterministically fair* if, for any sequences of packets to be dispersed, the absolute difference between the actual workload and the expected workload (in bytes) allocated to each path is always bounded by a finite constant. This means that any deterministically fair traffic splitter attempts to approximate GLS such that the workload deviation on every path is limited by a certain constant. In this section, we are going to show that, if all packets are bounded in sizes and the number of paths is fixed, PWFR is a deterministically fair traffic splitting algorithm.

**Lemma 2.** *For every positive integer $k$, there exists a positive residual workload on some active Path $i$ just before the routing decision for a positive-size Packet $k$ is made. That is,*

$$R_i^p(k) > 0 \tag{9}$$

*for some $i \in \{1, 2, \ldots, N\}$.*

**Proof.** If $k = 1$,

$$R_i^p(1) = W_i^p(1) = p_i\, S(1) > 0 \tag{10}$$

for all $i = 1, 2, \ldots, N$.

Suppose $k > 1$. By the conservation of traffic,

$$\sum_{j=1}^{N} W_j^p(k-1) = \sum_{j=1}^{N} \hat{W}_j^p(k-1). \tag{11}$$

This implies that either

$$W_i^p(k-1) = \hat{W}_i^p(k-1) \tag{12}$$

for all $i = 1, 2, \ldots, N$, or

$$W_i^p(k-1) > \hat{W}_i^p(k-1) \tag{13}$$

for some $i = 1, 2, \ldots, N$.

Since Packet $k$ is positive in size,

$$W_i^p(k) = W_i^p(k-1) + p_i\, S(k) > \hat{W}_i^p(k-1) \tag{14}$$

for some $i = 1, 2, \ldots, N$.

Combining, the lemma is proved. □

**Lemma 3.** *For every positive integer $k$, the actual workload allocated to Path $i$ cannot exceed the expected workload by an amount equal to or more than the maximum packet size $S_{\max}$ bytes. That is,*

$$\hat{W}_i^p(k) - W_i^p(k) < S_{\max}, \tag{15}$$

*where $i = 1, 2, \ldots, N$.*

**Proof.** When $k = 1$,

$$\hat{W}_i^p(1) - W_i^p(1) \le (1 - p_i) \cdot S_1 < S_{\max}. \tag{16}$$

Suppose $\hat{W}_i^p(k) - W_i^p(k) < S_{\max}$ is true for some positive integer $k = \xi$. That is,

$$\hat{W}_i^p(\xi) - W_i^p(\xi) < S_{\max} \tag{17}$$

for all $i = 1, 2, \ldots, N$.

There are two possible cases to consider.

Case 1: Packet $(\xi + 1)$ is not routed on Path $i$.

This implies

$$\hat{W}_i^p(\xi+1) = \hat{W}_i^p(\xi), \tag{18}$$

$$\hat{W}_i^p(\xi+1) - W_i^p(\xi+1) = \hat{W}_i^p(\xi) - [W_i^p(\xi) + p_i\, S_{\xi+1}] < S_{\max}. \tag{19}$$

Case 2: Packet $(\xi + 1)$ is routed on Path $i$.

This implies

$$\hat{W}_i^p(\xi+1) = \hat{W}_i^p(\xi) + S_{\xi+1}. \tag{20}$$

By applying Lemma 2,

$$R_i^p(\xi+1) = W_i^p(\xi+1) - \hat{W}_i^p(\xi) > 0, \tag{21}$$

$$\hat{W}_i^p(\xi+1) - W_i^p(\xi+1) = [\hat{W}_i^p(\xi) + S_{\xi+1}] - W_i^p(\xi+1) < S_{\max}. \tag{22}$$

Combining, the actual workload allocated to Path $i$ is always upper bounded, where $i = 1, 2, \ldots, N$. □

**Lemma 4.** *For every positive integer $k$, the actual workload allocated to Path $i$ cannot lag behind the expected workload by an amount equal to or more than $(N-1) \cdot S_{\max}$ bytes, where $N$ is the number of paths and $S_{\max}$ is the maximum packet size in bytes. That is,*

$$W_i^p(k) - \hat{W}_i^p(k) < (N-1) \cdot S_{\max}, \tag{23}$$

*where $i = 1, 2, \ldots, N$.*

**Proof.** By using Lemma 3,

$$\begin{aligned} W_i^p(k) - \hat{W}_i^p(k) &= \sum_{\substack{j=1 \\ j \neq i}}^{N} [\hat{W}_j^p(k) - W_j^p(k)] \\ &< (N-1) \cdot S_{\max}. \end{aligned} \tag{24}$$

Combining, the actual workload allocated to Path $i$ is always lower bounded, where $i = 1, 2, \ldots, N$. □

**Theorem 1.** *Given that all packets are bounded in sizes and the number of paths is fixed, PWFR is a deterministically fair traffic splitting algorithm.*

**Proof.** The result follows directly from Lemmas 3 and 4. □

In practice, the routing weight vector may change with time. This means that the routing weight vector when Packet $k$ arrives may be different from that when Packet $(k+1)$ arrives, for some positive integer $k$. It is easy to show that the lemmas and theorem in this section still hold when the routing weight vector changes between any two successive packet arrivals, as their proofs do not require any specific relationship among routing weights.

### 3.3 Provision of Guaranteed Performance Service

The following lemma shows that, when a leaky bucket constrained traffic source is divided on a set of paths by a PWFR traffic splitter, the offered workload to each path is also leaky bucket constrained.

**Lemma 5.** *Suppose a traffic source is constrained by a leaky bucket with parameters $(\sigma, \rho)$, where $\sigma$ and $\rho$ represent the maximum size of the bucket token pool and the average token generation rate respectively. Let $S_{\max}$ be the maximum packet size in bytes. If the traffic source is fed into a PWFR traffic splitter, a dispersed subflow to be routed on Path $i$, where $i = 1, 2, \ldots, N$, is constrained by another leaky bucket with parameters $(\sigma_i, \rho_i)$, where*

$$\begin{cases} \sigma_i = p_i\,\sigma + S_{\max} \\ \rho_i = p_i\,\rho. \end{cases} \tag{25}$$

**Proof.** Since a GLS traffic splitter can divide traffic ideally according to the given routing weight vector $\boldsymbol{p}$,

$$W_i^p(\cdot) \le (p_i\,\sigma) + (p_i\,\rho)\cdot(t-\tau) \tag{26}$$

for all $i = 1, 2, \ldots, N$.

By applying Lemma 3, consider any positive integer $k$,

$$\begin{aligned} \hat{W}_i^p(k) &< W_i^p(k) + S_{\max} \\ &\le (p_i\,\sigma + S_{\max}) + (p_i\,\rho)\cdot(t-\tau) \end{aligned} \tag{27}$$

for some $t \ge \tau$.

The lemma is proved. □

Compared with Lemma 1, a PWFR traffic splitter requires an expansion of the bucket token pool by the maximum packet size to constrain a dispersed subflow. This means that a PWFR traffic splitter may give a dispersed subflow with a larger burstiness than the one outputted from the idealized GLS traffic splitter. Yet, the difference can become insignificant if a traffic source is very bursty in nature (with a large $\sigma$). This is generally true for multimedia traffic found in the current Internet.

Based on the result from Lemma 5, we can derive performance bounds for the provision of guaranteed quality of service (QoS) using multiple paths for a single flow, which is leaky bucket constrained by the parameters $(\sigma, \rho)$. The performance bounds are useful as they can be adopted to routing policed traffic with end-to-end QoS guarantees, such as in [9], with the extension of supporting the concurrent use of multiple routes. Further research is needed to devise an efficient and practical algorithm for QoS multipath routing.

Denote by $g_i$ (where $g_i \ge \rho_i$), $h_i$, $C_{ij}$, and $\zeta_{ij}$, respectively, the reserved bandwidth on Path $i$, the number of hops for Path $i$, the link capacity at the $j$th hop for Path $i$, and the propagation delay at the $j$th hop for Path $i$. It can be shown [12] that the delay bound for Path $i$, $D_i(p_i, g_i)$, the delay jitter bound for Path $i$, $J_i(p_i, g_i)$, and the buffer space requirement at the $j$th hop router for Path $i$, $B_{ij}(p_i)$, respectively, can be written as

$$\begin{cases} D_i(p_i, g_i) = \frac{\sigma_i + h_i\,S_{\max}}{g_i} + \sum_{j=1}^{h_i}\left(\frac{S_{\max}}{C_{ij}} + \zeta_{ij}\right) \\ J_i(p_i, g_i) = \frac{\sigma_i + h_i\,S_{\max}}{g_i} \\ B_{ij}(p_i) = \sigma_i + j\,S_{\max}. \end{cases} \tag{28}$$

Denote the reserved bandwidth vector by

$$\boldsymbol{g} = (\,g_1 \quad g_2 \quad \cdots \quad g_N\,). \tag{29}$$

The end-to-end delay bound can be found as

$$D(\boldsymbol{p}, \boldsymbol{g}) = \max_{i=1}^{N}(D_i(p_i, g_i)) \tag{30}$$

and the end-to-end delay jitter bound can be calculated as

$$\begin{aligned} J(\boldsymbol{p}, \boldsymbol{g}) &= D(\boldsymbol{p}, \boldsymbol{g}) - \min_{i=1}^{N}(D_i(p_i, g_i) - J_i(p_i, g_i)) \\ &= D(\boldsymbol{p}, \boldsymbol{g}) - \min_{i=1}^{N}\left(\sum_{j=1}^{h_i}\left(\frac{S_{\max}}{C_{ij}} + \zeta_{ij}\right)\right). \end{aligned} \tag{31}$$

These end-to-end delay and jitter bounds can be easily extended when a PWFR traffic splitter is used concurrently with some packet-based fair queueing algorithms which closely imitate GPS with bounded fairness among different traffic flows, such as WFQ (or PGPS) [5], [10] and worst-case fair weighted fair queueing ($WF^2Q$) [3].

## 4 Performance Evaluation

In this section, we first outline a set of traffic splitting algorithms for performance studies. A performance metric is then defined for comparing various splitting algorithms. Afterward, we briefly describe the characteristics of the traffic traces used for the simulation and, finally, we examine the simulation results of the proposed traffic splitting algorithm.

### 4.1 Traffic Splitting Algorithms

We study three basic traffic splitting algorithms, namely, packet-by-packet weighted fair routing (PWFR), packet-by-packet generalized round robin routing (PGRR), and packet-by-packet probabilistic routing (PPRR). The PWFR approach, which is described in Section 3, simulates generalized load sharing (GLS) as closely as possible, subject to the granularity constraint imposed by transmitting each packet as a whole.

PGRR distributes packets to each path in a cyclical fashion so that the arrival instants of any two packets to each path is as uniformly as possible. Indeed, a way to implement PGRR is to apply PWFR by setting all packet sizes to one unit, or any other constant.

According to the routing weight vector, PPRR divides packets to each route at random. Thus, PPRR is equivalent to a packet-based perfect hashing routing scheme, which provides a performance benchmark to all other hashing-based traffic splitting algorithms.

### 4.2 Performance Metric

Our performance metric for a traffic splitter is the mean squared workload deviation, which measures the variation of the actual workloads allocated by the traffic splitter to a set of $N$ paths from the expected ones distributed under GLS. A good traffic splitting algorithm should divide traffic according to the given routing weight vector and, hence, it should be able to keep the value of the mean squared workload deviation as small as possible.

Suppose a set of $M$ packets, namely, Packet 1, Packet 2, ..., Packet $M$, which belong to connectionless traffic for

| Trace File | Duration in Sec. | # TCP Calls | # TCP Packets | # non-TCP Packets | # TCP Bytes | # non-TCP Bytes | $S_{max}$ |
|---|---|---|---|---|---|---|---|
| ***SDZ-19940714*** | *3 622.845238* | *121 628* | *7 680 715* | *2 245 811* | *1 874 271 141* | *674 803 625* | *1 500* |
| ***FXW-19960918*** | *298.712016* | *217 590* | *3 491 749* | *1 529 446* | *1 431 367 056* | *435 232 031* | *4 348* |
| ***12SDC-19991115a*** | *88.921181* | *5 388* | *435 936* | *1 196 674* | *312 679 995* | *1 573 644 381* | *4 458* |
| ***12SDC-19991115b*** | *88.839599* | *6 806* | *451 451* | *1 250 882* | *281 740 032* | *1 611 799 302* | *4 458* |
| ***12SDC-19991215a*** | *88.879599* | *4 862* | *418 442* | *235 128* | *294 629 822* | *116 560 362* | *9 180* |
| ***12SDC-19991215b*** | *89.287843* | *4 876* | *509 902* | *290 316* | *360 136 635* | *140 277 214* | *9 180* |
| ***IND-20010815*** | *90.094056* | *87 353* | *1 321 518* | *1 525 490* | *729 404 800* | *537 911 885* | *4 426* |

Fig. 4. The characteristics of traffic traces.

non-TCP connections, are split on a set of paths. Let $W_i^p(k)$, and $\hat{W}_i^p(k)$, respectively, be the expected and actual workloads (in bytes) allocated to Path $i$, just after the routing decision for the $k$th packet has been made. The mean squared workload deviation for the set of packets is defined as

$$E[(\hat{W}^p - W^p)^2] = \frac{\sum_{k=1}^{M}\sum_{i=1}^{N}(\hat{W}_i^p(k) - W_i^p(k))^2}{MN}. \quad (32)$$

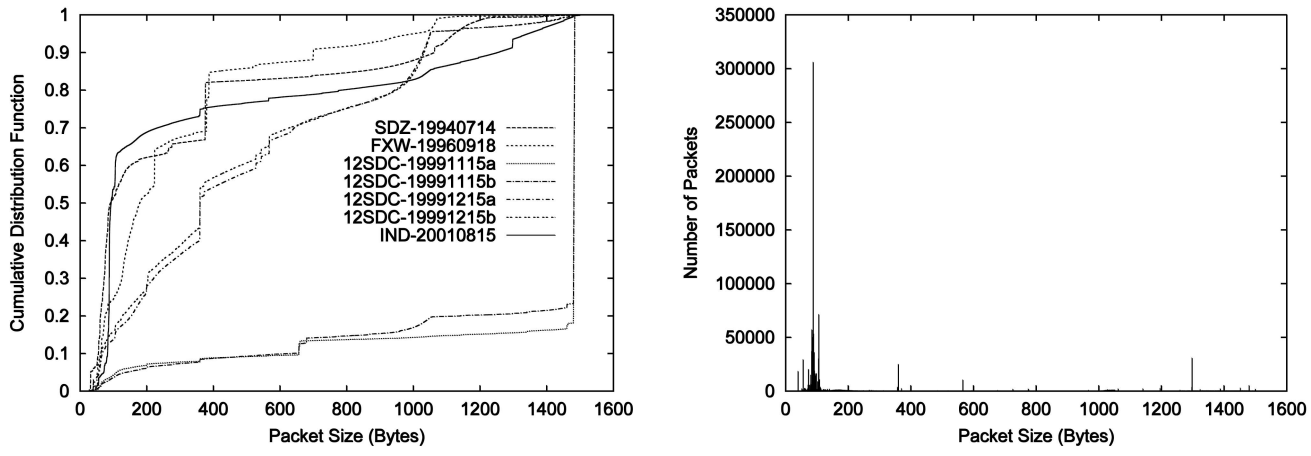
### 4.3 Traffic Traces

The simulation is based on a set of real packet traces collected on the Internet backbones for the Passive Measurement and Analysis (PMA) Project over four trunks, namely, the SDSC FDDI DMZ, the FIX-West facility at NASA Ames (FDDI interface), the SDSC OC12mon vBNS connection, and the Indiana University GigaPOP. A total of seven traces, from July 1994 to August 2001, are employed for our simulation studies. The characteristics and the packet size distributions for non-TCP connections of these traffic traces are summarized in Figs. 4 and 5, respectively.

### 4.4 Simulation Results

Our simulation experiments are based on a network node which routes incoming traffic on a set of outgoing paths. Each simulation run is executed with a complete traffic trace, and the statistics for computing the performance metric are collected only when the routes for the first 10,000 packets have been determined. For PWFR and PGRR, a single run is sufficient to determine the mean squared workload deviation for a certain setting. However, since PPRR involves the use of random numbers, a total of 10 runs have been done to find an average value of the performance metric, and a 95 percent confidence interval for each average value of the metric is also computed. Due to space limitations, plots showing similar results are left out and, hence, only plots corresponding to the trace file "IND-20010815" are presented here.

The results are provided in three sets. The first set examines the performance of different traffic splitters for multipath routing with different number of homogeneous paths used, where the routing weight for each path is the same. Fig. 6 exhibits the mean squared workload deviation for connectionless traffic. The number of paths used varies from one to 10. When all packets travel on a single path, no traffic splitting is needed and, thus, the mean squared workload deviation for all traffic splitters considered is always zero. By using two or more paths, we observe that the mean squared workload deviation when PWFR is employed is substantially smaller than when PGRR or PPRR is used. The performance of PWFR is more or less the same with the number of paths used. Obviously, it is due to




(a) (b)

Fig. 5. Packet size distribution plot for non-TCP traffic. (a) All seven traffic traces. (b) Trace file "IND-20010815."

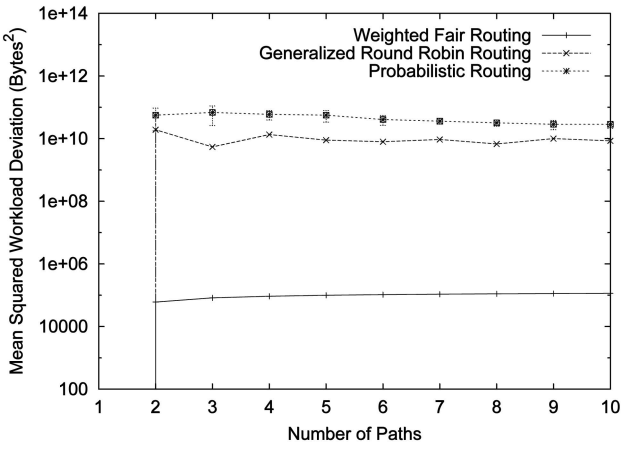



Fig. 6. Workload deviation plot for homogenous multipath routing.

the fact that PWFR aims to minimize workload deviation each time the route of each packet is assigned. Though PGRR tries to distribute packets to paths as closely to the routing weight vector as possible, it does not take the dynamics of packet size into account. PPRR routes packets at random and, hence, it can be expected that it can split the workload approximately equal to the expected load distribution over a very long time period. This means that, over short time periods, the workload may be very different from the expected one. Thus, this approach cannot minimize workload deviation in general.

The second set of results compares the effectiveness of different traffic splitters for heterogeneous multipath routing, where the routing weight for one path can differ from that for another path. It is sufficient to consider only three paths for the extensive study as it has been shown [7], under a wide range of scenarios, that multipath routing is effective in using a small number of paths, say, up to three. The routing weight vector, $\boldsymbol{p} = (\,p_1 \quad p_2 \quad p_3\,)$, has been set such that $p_3 = 0$ or $0.35$. When $p_3 = 0$, $p_1$ varies between 0.001 and 0.5, with a total of 11 data points. Due to symmetry, it is not necessary to perform duplicate experiments when $p_1$ is greater than 0.5. For instance, the result of $p_1 = 0.7$ is the same as that of $p_1 = 0.3$ since, for both cases, the routing weight of one path is 0.3 and that of another is 0.7, and, thus, the results should be the same. Similarly, when $p_3 = 0.35$, $p_1$ varies between 0.001 and 0.3, with a total of seven data points.

Fig. 7a shows the mean squared workload deviation for connectionless traffic when the routing weight for Path 3 is 0, i.e., no traffic will be routed on Path 3. The routing weight for Path 1 varies between 0.001 and 0.5. Similar to our first set of results, we see that the mean squared workload deviation when PWFR is employed is significantly lower than when PGRR or PPRR is used. Fig. 7b exhibits the mean squared workload deviation for connectionless traffic when the routing weight for Path 3 is 0.35, where the routing weight for Path 1 varies between 0.001 and 0.3. The results are similar to cases when the routing weight for Path 3 is 0. Combining, PGRR is very effective in splitting traffic to paths as closely to any routing weight vector as possible.

The third set of results demonstrates the effectiveness of our proposed algorithm on load balancing by means of plotting a set of sample load vectors. Each sample load vector consists of two sample loads, each corresponding to a path. A sample load is taken at intervals of 0.001 seconds. Fig. 8 illustrates the distribution of sample load vectors for connectionless traffic with the routing weight vector $(0.35 \quad 0.65 \quad 0)$. It is clear that, when PWFR is used, the sample load vectors are concentrated on a region of a thin diagonal stripe, where the slope of that stripe is equal to the ratio of the routing weights between Path 1 and Path 2, i.e., 1.857. There is a much thicker stripe for using PGRR, and even a "cloud" of sample load vectors when PPRR is employed. This means that, for connectionless traffic, only PWFR can effectively balance the workloads to paths in proportion to their routing weights.

## 5 Conclusions

In this paper, we have proposed a framework to study how to effectively perform load sharing in multipath communication networks. A generalized load sharing (GLS) model

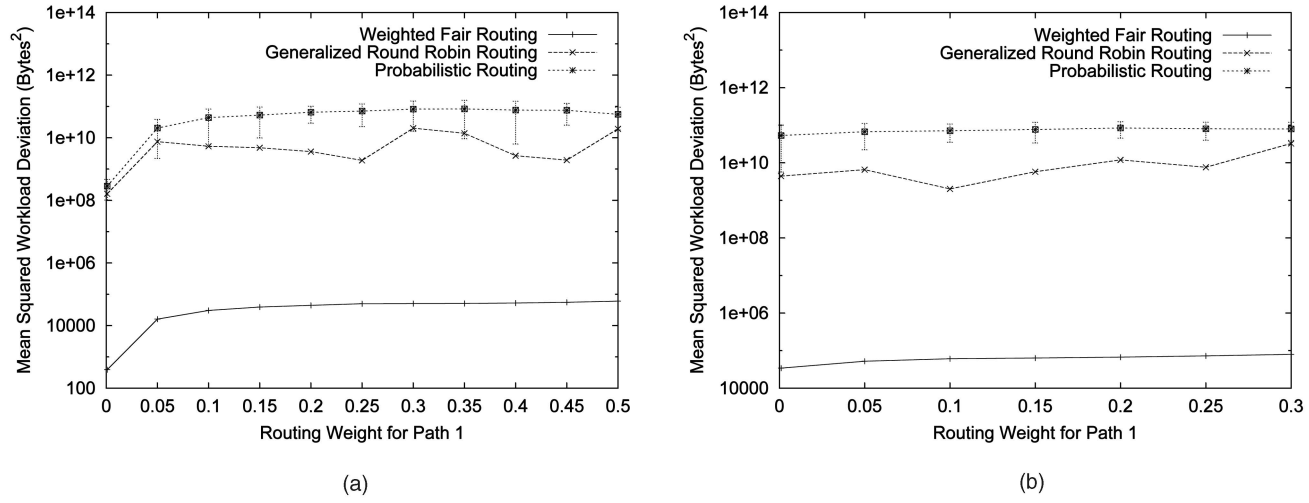



Fig. 7. Workload deviation plot for heterogenous multipath routing. (a) $p_3 = 0$. (b) $p_3 = 0.35$.

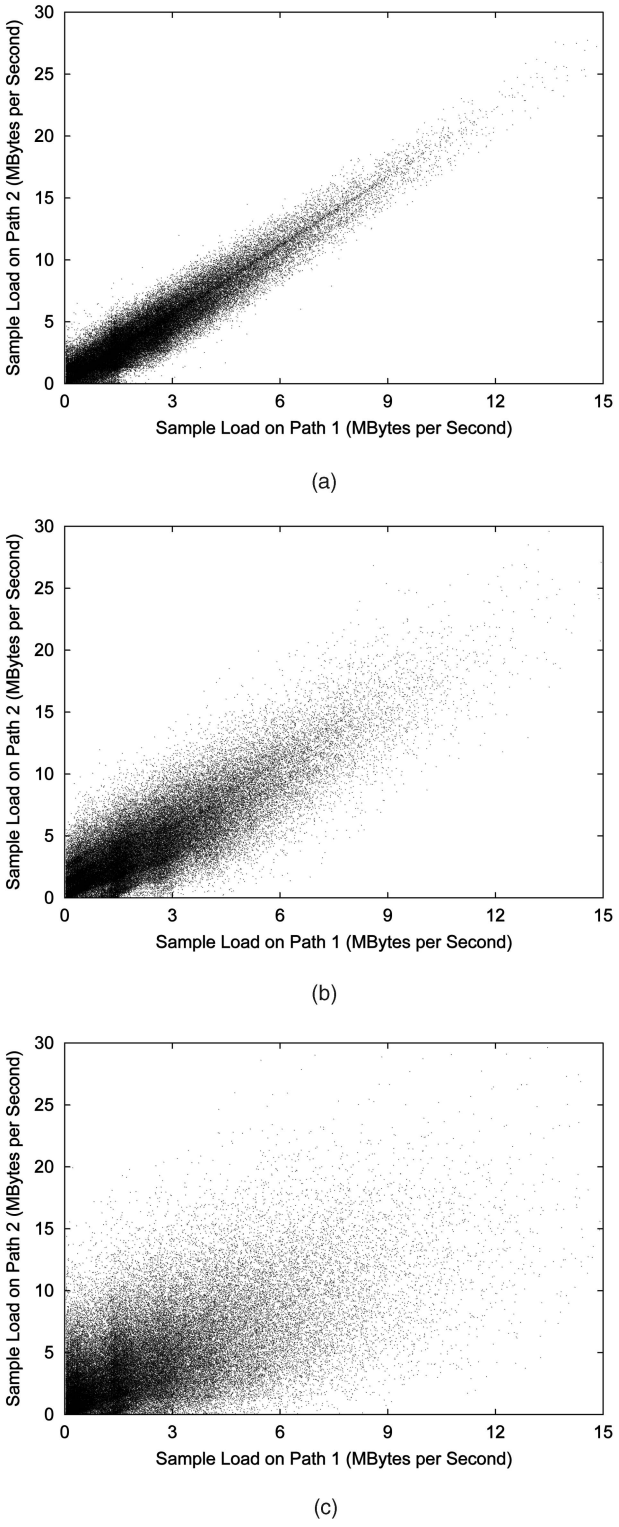



Fig. 8. Sample load plot for connectionless traffic with $p_1 = 0.35$, $p_2 = 0.65$, and $p_3 = 0$. (a) Packet-by-packet weighted fair routing. (b) Packet-by-packet generalized round robin routing. (c) Packet-by-packet probabilistic routing.

has been developed to conceptualize how traffic is split ideally on a set of active paths. A simple traffic splitting algorithm, called packet-by-packet weighted fair routing (PWFR), has been devised to approximate GLS with the given routing weight vector by transmitting each packet as a whole. We have developed some performance bounds for PWFR and found that PWFR is a deterministically fair traffic splitting algorithm. This attractive property is useful in the provision of service with guaranteed performance when multiple paths can be used simultaneously to transmit packets which belong to the same flow.

A total of seven historical Internet backbone traces have been used in our simulation studies. For each of the traffic trace, we investigated connectionless traffic from non-TCP connections. Our simulation studies, based on these traffic traces, reveal that PWFR outperforms two other traffic splitting algorithms, namely, packet-by-packet generalized round robin routing (PGRR), and packet-by-packet probabilistic routing (PPRR). These promising results form a basis for designing future adaptive quality of service (QoS) or constraint-based multipath routing protocols.

## ACKNOWLEDGMENTS

This research is supported in part by the Areas of Excellence Scheme established under the University Grants Committee of the Hong Kong Special Administrative Region, China (Project Number AoE/E-01/99). The authors would like to thank the US National Science Foundation Cooperative Agreement Numbers ANI-0129677 (2002) and ANI-9807479 (1998), and the National Laboratory for Applied Network Research, Measurement and Network Analysis Group for making their real Internet backbone packet traces available for their research, and the anonymous reviewers for their valuable comments and suggestions which assisted them in improving the quality of the paper.

## REFERENCES

[1] H. Adiseshu, G. Varghese, and G. Parulkar, "An Architecture for Packet-Striping Protocols," *ACM Trans. Computer Systems,* vol. 17, no. 4, pp. 249-287, Nov. 1999.
[2] *Inverse Multiplexing for ATM (IMA) Specification Version 1.1,* ATM Forum Technical Committee, Mar. 1999.
[3] J.C.R. Bennett and H. Zhang, "$WF^2Q$: Worst-Case Fair Weighted Fair Queueing," *Proc. IEEE INFOCOM '96,* vol. 1, pp. 120-128, Mar. 1996.
[4] Z. Cao, Z. Wang, and E. Zegura, "Performance of Hashing-Based Schemes for Internet Load Balancing" *Proc. IEEE INFOCOM,* vol. 1, pp. 332-341, Mar. 2000.
[5] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Internetworking: Research and Experience*, vol. 1, no. 1, pp. 3-26, Sept. 1990.
[6] *IEEE Standard 802.3-2002,* IEEE CS, Mar. 2002.
[7] K.-C. Leung and V.O.K. Li., "A Resequencing Model for High Speed Networks," *Proc. IEEE Int'l Conf. Comm.,* vol. 2, pp. 1239-1243, June 1999.
[8] K.-C. Leung and V.O.K. Li, "Generalized Load Sharing for Packet-Switching Networks" *Proc. IEEE Int'l Conf. Network Protocols,* pp. 305-314, Nov. 2000.
[9] A. Orda, "Routing with End-to-End QoS Guarantees in Broadband Networks," *IEEE/ACM Trans. Networking,* vol. 7, no. 3, pp. 365-374, June 1999.
[10] A.K. Parekh and R.G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case," *IEEE/ACM Trans. Networking,* vol. 1, no. 3, pp. 344-357, June 1993.
[11] K. Sklower, B. Lloyd, G. McGregor, D. Carr, and T. Coradetti, *The PPP Multilink Protocol (MP),* Request for Comments, RFC 1990, Network Working Group, Internet Eng. Task Force, Aug. 1996.
[12] H. Zhang, "Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks, *Proc. IEEE,* vol. 83, no. 10, pp. 1374-1396, Oct. 1995.

**Ka-Cheong Leung** received the BEng degree in computer science from the Hong Kong University of Science and Technology in 1994, and the MSc degree in electrical engineering (computer networks), and the PhD degree in computer engineering from the University of Southern California, Los Angeles, in 1997 and 2000, respectively. He worked as a senior research engineer at Nokia Research Center, Nokia Inc., Irving, Texas, from 2001 to 2002. He was an assistant professor in the Department of Computer Science at Texas Tech University, Lubbock, from 2002 to 2005. Since June 2005, he has been with the University of Hong Kong, where he is a visiting assistant professor in the Department of Electrical and Electronic Engineering. His research interests include wireless packet scheduling, routing, congestion control, and quality of service guarantees in high-speed communication networks, content distribution, high-performance computing, and parallel applications. He is listed in the 60th (2006) edition of *Marquis Who's Who in America* and is a member of the IEEE.

**Victor O.K. Li** received the SB, SM, EE, and ScD degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, in 1977, 1979, 1980, and 1981, respectively. He joined the University of Southern California (USC), Los Angeles, in February 1981, and became a professor of electrical engineering and director of the USC Communication Sciences Institute. Since September 1997, he has been with the University of Hong Kong, where he is the chair professor of information engineering. He also served as managing director of Versitech Ltd., the technology transfer and commercial arm of the university, from September 1997 to June 2004. He serves on various corporate boards. His research is in information technology, including all-optical networks, wireless networks, and Internet technologies and applications. Sought by government, industry, and academic organizations, he has lectured and consulted extensively around the world. He is very active in the research community, and has chaired various international conferences and served on the editorial boards of various international journals. He has given distinguished lectures at universities around the world, and keynote speeches at many international conferences. He has received numerous awards, including, most recently, the UK Royal Academy of Engineering Senior Visiting Fellowship in Communications, the KC Wong Education Foundation Lectureship, the Croucher Foundation Senior Research Fellowship, and the Bronze Bauhinia Star, Government of the Hong Kong Special Administrative Region, China. He was elected an IEEE fellow in 1992.