PAPER

Service-Sensitive Routing in DiffServ/MPLS Networks

Nai-Bin HSU[†], ***Student Member***, **Ying-Dar LIN**[†], **Mao-Huang LI**[††], ***and*** **Tsern-Huei LEE**[††], ***Nonmembers***

SUMMARY This study investigates the problem of unfairness when QoS routing does not consider the mix of traffic classes. Unfairness is mainly caused by routing different traffic flows of the same class through paths with extremely different traffic mixtures, involving various service classes. Next, a new routing scheme—*Service-sensitive Routing* (SSR), which takes the state of traffic mixture of the various service classes into account, is proposed. To determine the QoS route for a flow request, SSR not only considers the available bandwidth and delay of the candidate paths, but also considers the mix of traffic classes on the paths. Additionally, the hybrid granularity routing decision in SSR scheme is scalable and suitable for the *Differentiated Services* and MPLS networks. Extensive simulations show that SSR can effectively reduce the variance of the average of queuing delays, for example by approximately 20% to 35% for a moderate offered load, compared to the shortest path routing. Furthermore, this routing scheme reduces the fractional reward loss and bandwidth blocking probability.

key words: *service-sensitive, service balance index, MPLS, DiffServ*

1. Introduction

The Internet is providing users diverse and essential Quality of Services (QoS), particularly given the increasing demand for a wide spectrum of network services. Many services, previously only provided by traditional circuit-switched networks, can now be provided on the Internet. These services, depending on their inherent characteristics, require certain degrees of QoS guarantees. Many technologies are therefore being developed to enhance the QoS capability of IP networks. Among these technologies, the *Differentiated Services* (DiffServ) [3], [4], [11], [13] and *Multi-Protocol Label Switching* (MPLS) [5], [7], [14], [15] are the enabling technologies that are paving the way for tomorrow's QoS services portfolio.

The DiffServ is based on a simple model where traffic entering a network is classified, policed and possibly conditioned at the edges of the network, and assigned to different behavior aggregates. Each behavior aggregate is identified by a single *DS codepoint* (DSCP). At the core of the network, packets are fast-forwarded according to the *per-hop behavior* (PHB) associated with the DSCP. By assigning traffic of different classes to different DSCPs, the DiffServ network provides different forwarding treatments and thus different levels of QoS.

MPLS integrates the label swapping forwarding paradigm with network layer routing. First, an explicit path, called the *label switched path* (LSP), is determined, and established using a signaling protocol. A label in the packet header, rather than the IP destination address, is then used for making forwarding decisions in the network. Routers that support MPLS are called *label switched routers* (LSRs). The labels can be assigned to represent routes of various *granularities*, ranging from as coarse as the destination network down to the level of each single flow. Moreover, numerous traffic engineering functions have been effectively achieved by MPLS. When MPLS is combined with DiffServ and constraint-based routing, they become powerful and complementary abstractions for QoS provisioning in IP backbone networks.

To provide various guarantees to traffic running over the network, time-sensitive scheduling and QoS routing mechanisms must also be implemented in the network elements. Many QoS routing mechanisms have been proposed [1], [2], [10], [19], and could be incorporated with DiffServ/MPLS according to and depending on the QoS objectives and policies of service providers, such as load balancing. However, a routable QoS path may not be schedulable. Considering the following observations: (1) the traffic running over the network links is continuously fluctuating, (2) the scheduling disciplines, packet dropping disciplines, and so on configured in each of the network elements are unlikely to change dynamically with respect to the instantaneous states of the traffic running over the links in the network.

The outcome if two separate flows of the same service class traverse two different routes with significantly different states of traffic mixture is interesting. Intuitively, since the mix of traffic classes is different, the queue lengths of the same class on the two routes will also differ. Therefore, the queuing delays and packet dropping probabilities suffered by these two flows of the same service class could differ significantly. More-

Manuscript received December 18, 2000.
Manuscript revised March 19, 2001.

[†]The authors are with the Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan.
[††]The authors are with the Department of Communication Engineering, National Chiao Tung University, Hsinchu, Taiwan.

over, one of the two flow requests may be impossible to schedule under the service constraints, while the other flow may be set up successfully. More details are presented in the next section. Obviously, this situation is unfair to the customer who gets worse treatment, since he is paying the same price for the same class of service as the other customer.

This study investigates the problem of unfairness when QoS routing does not consider the mix of traffic classes. Next, a new routing scheme, *Service-sensitive Routing* (SSR), is proposed, which takes the state of traffic mixture of various service classes into account when determining the QoS route for a flow request. Simulation results show that SSR can effectively reduce the variance of the traffic mixture among all network links, and thus reduce the unfairness problem in terms of average queuing delay.

The rest of this paper is organized as follows. Section 2 describes the problem of unfairness and the motivation behind this study by providing an empirical example. Meanwhile, Sect. 3 presents definitions and the link cost function. Section 4 then describes the novel routing scheme, and Sect. 5 presents simulation results and a performance evaluation. Finally, Sect. 6 concludes this work.

2. Problem and Motivation

To deliver predictable guarantees in a packet network requires the use of a traffic shaper and a scheduling algorithm in the routers. A shaper smoothes the burstness of the input traffic. The scheduling algorithm selects the packet to be transmitted in the next cycle from the available packets of the flows sharing the output link. Numerous packet scheduling algorithms, such as *First In First Out* (FIFO), *Virtual Clock* (VC) [18], *Weighted Fair Queuing* (WFQ) [6], *Deficit Round Robin* (DRR) [16], and *Self-Clocked Fair Queuing* (SCFQ) [8] etc. have been developed, each with distinctive characteristics and performance.

To demonstrate the unfairness problem caused by traversing different routes with different mixtures of traffic of various service classes, a simple experiment was executed. Assume that the traffic shaper and scheduler are employed in the edge router and scheduler in the core router. The following definition applies:

Definition 2.1: Given an input flow F whose amount of traffic $A(t) \leq b+rt$ is satisfied over the t period, then the flow F is said to conform to a (b, r)-*leaky bucket*, where b is the bucket depth and r is the mean rate.

As shown in [9], we obtain the following theorem:

Theorem 1: Given a (b, r)-*leaky bucket* flow F, then the maximum end-to-end delay D of the packets of the flow F satisfies

$$D = D_p + D_s + \frac{b}{r} + \frac{Mh}{r}, \tag{1}$$

where D_p is the propagation delay of the routing path, D_s is the sum of the scheduling delay in each node on the path, M is the maximum packet size of the flow, and h is the number of hops of the path.

For an input flow F, the distribution of traffic classes of the nodes on the routing path, D_s, suffered by the flow F differ, and consequently, the end-to-end delays of the flow, D, also differ.

Assume two classes of services are offered in the network, namely $C1$ and $C2$. Further assume that traffic streams of service $C1$ and $C2$ consist of packets with fixed packet size and exponential inter-arrival times. Since in the current experiment the problem is independent of the service discipline, we simply assume that the SCFQ scheduling is used. The SCFQ scheme uses a virtual time function which depends on the progress of work in the actual queuing system. Details can be found in [8]. Figure 1 shows the simulation model in this experiment. According to the DSCP classification, incoming packets are separated into priority buffer queues. We assume that queue-1 has a higher priority than queue-2, and that the capacity of the outlet server is 100 Mbps.

Table 1 lists the simulation cases. Assume that the volume of traffic is 66 Mbps in case A and 90 Mbps in case B. Meanwhile, both cases consist of $C1$ and $C2$ traffic. Each case involves three scenarios with different ratios of $C1$ and $C2$, as shown in the parentheses. These scenarios can be viewed as snapshots at different time points.

Table 1 also lists the simulation results, revealing different queuing delays for packets of $C1$ and $C2$. The phenomenon indicates that under different traffic mix conditions, packets of the same class suffer different queuing delays, creating the problem of unfairness.

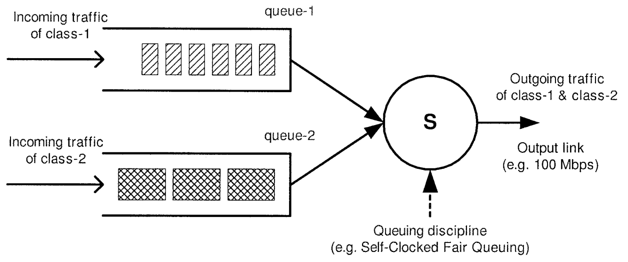

Incoming traffic of class-1
queue-1
Outgoing traffic of class-1 & class-2
S
Incoming traffic of class-2
queue-2
Output link (e.g. 100 Mbps)
Queuing discipline (e.g. Self-Clocked Fair Queuing)

Fig. 1 Model of scheduling with the mix of traffic classes.

Table 1 Queuing delay under various ratios of traffic, link=100 Mbps.

case	scen.	$C1$ traffic Mbps	$C2$ traffic Mbps	$C1$ queuing delay (μs)	$C2$ queuing delay (μs)
A	A1	22.0 (1/3)	44.0 (2/3)	82	353
	A2	16.5 (1/4)	49.5 (3/4)	55	367
	A3	26.4 (2/5)	39.6 (3/5)	172	273
B	B1	30.0 (1/3)	60.0 (2/3)	154	723
	B2	22.5 (1/4)	67.5 (3/4)	66	726
	B3	36.0 (2/5)	54.0 (3/5)	583	374

Furthermore, comparing the simulation results of cases A and B reveals that the unfairness problem is worse as the link utilization is heavy. Other queuing systems, such as *Deficit Round-Robin* (DRR) [16], may result in a more serious unfairness problem.

Existing QoS routing mechanisms do not consider the state of the traffic mixture when determining a feasible flow path. Thus, flows of the same class could suffer from being treated very differently in current routing schemes. This work proposes a new QoS routing scheme called *Service-Sensitive Routing* (SSR) to resolve the unfairness problem presented in the above example. This routing scheme considers the traffic mixture of links to be part of the cost of determining a feasible route. Therefore, the SSR not only finds a route with abundant residual capacity for a flow but also balances the traffic mix and maintains a high degree of fairness among flows of the same service class.

3. Definitions and Link Cost Function

Consider a DiffServ network consisting of a set of routers, V, connected by a set of unidirectional links, E. Such a network is modeled herein as $G(V,E)$ with a function $Cap : E \mapsto \mathbf{Z}_0^+$, where $Cap(e)$ denotes the capacity of link e. The DiffServ network handles K traffic classes, labeled $k = 1,...,K$. Traffic flows of the same class are assumed to share the same traffic characteristics and QoS requirements, and the effective bandwidth required for setting up a class k flow on a link is b_k.

Definition 3.1: Given the number of service classes K offered in a network $G(V,E)$, the *Service Balance Index* (SBI) Ψ_e of the link e is defined as

$$\Psi_e = \sum_{k=1}^{K} \left| \frac{r_k^e}{R^e} - \frac{s_k^e}{S^e} \right|, \forall e \in E, \tag{2}$$

where r_k^e denotes the amount of traffic of class k requested on the link e, s_k^e denotes the configured capacity for class k on the link e. The total request bandwidth of link e is denoted as $R^e = \sum_{k=1}^{K} r_k^e$, and the total service bandwidth of link e is denoted as $S^e = \sum_{k=1}^{K} s_k^e$.

Notably, $Cap(e)$ is partitioned into K classes, and s_k^e can be obtained from the *Service Level Agreement* (SLA) contracts signed with customers or from historical traffic statistics. Additionally, the scheduling discipline, the buffer management, and the queuing policy of this node are also considered. Index Ψ is used to estimate the degree of balance of different traffic classes. Obviously, as the request ratio $(\frac{r_k^e}{R^e})$ approaches to the service ratio $(\frac{s_k^e}{S^e})$, Ψ_e shrinks. That is, if distribution of traffic load of different classes is perfect, Ψ_e will be zero. From the definition of the SBI, we define the link cost function $cost(e)$ as follows.

Definition 3.2: Given the number of service classes K offered in a network $G(V,E)$, the residual bandwidth is B^e, and the SBI is Ψ_e of the link e. When a flow request F with b_k requirement arrives at a edge router, we define the cost of carrying a class k flow on link e as follows:

$$cost(e) = \begin{cases} \frac{1}{B^e(1+\gamma)}, & \text{if } b_k \le B^e, \\ \infty, & \text{otherwise.} \end{cases} \tag{3}$$

where $\gamma = \eta \cdot \Delta\Psi_e$ is the *inflation ratio* and η is the *inflation factor.* $\Delta\Psi_e = \Psi_e(current) - \Psi_e(admit\ F)$, is the improvement of SBI if we admit the incoming request.

The residual bandwidth B^e in this definition is inflated by the γ. As the approach in [12], this cost function considers not only the residual bandwidth, but also the number of hops on the routing path. Additionally, $cost(e)$ is sensitive to the mix of service classes of the links.

The inflation ratio assigned to a link depends on its mix of traffic classes. When the admission of the flow and the SBI is reduced, a positive inflation is added to the residual bandwidth, making the link more attractive to the SSR algorithm. On the other hand, if the admission of the flow and the SBI are increased, a negative inflation is added to the residual bandwidth, making the link less attractive to the SSR algorithm. Therefore, an important property of this cost function is that it tracks the mix of traffic conditions and automatically adapts to find a feasible route which will balance the traffic in the network.

The sensitivity of the cost function is adjusted by the value of η. To be effective, the cost function must be sufficiently large that the skew (heavy imbalanced) link becomes unattractive as the path-finding function is invoked. Furthermore, the change in the value of the cost function should be smooth. This cost function is used to compute the route in the following section and to provide a route that allows the flow to achieve a greater balance of service classes. Consequently, the offered traffic will be more schedulable and interoperate with other technologies, thus delivering a more robust end-to-end QoS solution.

4. Routing Algorithm and Labeling

This section presents a hybrid granularity scheme using a routing *mark* as part of the label in MPLS. Sets of labels distinguish destination address, service class, forwarding path, and probably also privacy. In MPLS, edge devices perform most of the processor-intensive work, performing application recognition to identify flows and classify packets according to the network policies. Herein, when a flow request arrives at an edge router, it is routed to the nearly best path given the current network state, where the "best" path is defined

src, dst	LSP$_1$	.....	LSP$_m$
s, d_1	π_{11}, $width_{11}$, $delay_{11}$, ρ_{11}	.....	π_{1m}, ...
s, d_i	π_{i1}, $width_{i1}$, $delay_{i1}$, ρ_{i1}	.....	.....

Fig. 2 Routing cache in the *Service_Sensitive_Routing.*

as the least costly feasible path. Flows between an S-D pair are routed on several different paths and marked accordingly at the source and edge routers. Notably, flows of the same routing path may require different qualities of service. The core router in a DiffServ domain uses the label to determine which output port (interface) a packet should be forwarded to, and to determine service class. Core devices expedite forwarding while enforcing QoS levels assigned at the edge.

By limiting the number of routing marks, say to m, the routing algorithm can route flows between each S-D pair along a limited number of paths. The route *pinning* is enforced by stamping packets of the same flow with the same *mark*. Rather than identifying every single flow, the forwarding process at intermediate or core routers is simplified by merely checking the label. The routing cache size of an edge router is bounded to $O(nm)$, where n is the number of network nodes.

Figure 2 illustrates the format of the routing cache, which provides a maximum of m feasible routes (LSPs) between an edge node (s) and each other destination ($d_i, i = 1 \cdots n, d_i \neq s$). The first path entry π_{i1}, can be pre-computed, or the path information can be flushed and computed on-demand under the network administration policy. Besides the path list, each path entry includes the residual bandwidth ($width$), maximum delay ($delay$), and utilization (ρ). Information on the entry can be flushed by the management policy, for instance, refresh timeout or reference counts. Regarding labeling and forwarding, the approach is scalable and suitable for the *Differentiated Services* and MPLS networks.

The algorithm in Fig. 3 describes that upon a flow request F, the *Service_Sensitive_Routing* algorithm first attempts to extract the least costly feasible path π from the routing cache. If the extraction is negative, the scheme attempts to compute the least costly feasible path, termed σ. If σ is found, *Service_Sensitive_Routing* assigns a new mark to σ, inserts this new mark into the route cache, and then labels/routes the flow request F explicitly through σ. Meanwhile, if π is found and the path is only lightly utilized, the *Service_Sensitive_Routing* marks the flow F and routes it to path π. Otherwise the flow is blocked. If the utilization of path $\rho(\pi)$ exceeds a pre-defined threshold, say θ, the *Service_Sensitive_Routing* can either route F to a π held in the cache, or route F through a newly computed path σ, whichever is least costly. Consequently, flows between an S-D pair may be routed on a maximum of m different paths, where m is the maximum number of routing classes. In the SSR algorithm, function

```
Service_Sensitive_Routing($F, s, d, b_r, D_r$)
flow $F$;                /* from $s$ to $d$ with req. $b_r$ and $D_r$ */
cache entry $\Pi(s,d)$;/* set of routing paths from $s$ to $d$ */
extracted path $\pi$;
computed path $\sigma$;
Begin
    initiate $cost(NULL) \leftarrow \infty$
    extract $\pi \in \Pi(s,d)$
        that $cost(\pi)$ is the least & satisfy constraint
            { $width(\pi) \geq b_r$, $delay(\pi) \leq D_r$, ... }
    case ($\pi$ not found):
        $\sigma \leftarrow FindRouteLeastCost(s, d, b_r, D_r)$
        if ($\sigma$ not found) then "path not found"
        insert/replace($\sigma$, $\Pi(s,d)$),
        label($F$) & route($F$) through $\sigma$
    case ($\pi$ is found):
        if ($\rho(\pi)$ lightly utilized) then
            label($F$) & route($F$) to $\pi$
        endif
        $\sigma \leftarrow FindRouteLeastCost(s, d, b_r, D_r)$
        if ($\sigma$ not found) then "path not found"
        if ($cost(\sigma) < cost(\pi)$) then /* $\sigma$ better */
            insert/replace($\sigma$, $\Pi(s,d)$),
            label($F$) & route($F$) through $\sigma$
        else /* $\pi$ better */
            label($F$) & route($F$) to $\pi$
        endif
End
```

Fig. 3 Algorithm of the *Service_Sensitive_Routing.*

FindRouteLeastCost computes the least cost path using the inflated link state database based on the service-sensitive function in Sect. 3. Additionally, *constrained shortest path* or *widest-shortest path* heuristics can be used in the path-computing method.

5. Performance Evaluation

This section evaluates various aspects of the performance of SSR, including the Service Balance Index (SBI), with various path utilization thresholds (θ), queuing delay, fractional reward loss, and bandwidth blocking probability.

Three routing schemes are used in the simulation, namely Shortest Path Routing (SPR), variant of Service-Sensitive Routing (SSR*), and Service-Sensitive Routing (SSR) as proposed in the previous section. SSR* is a variant of SSR and uses the value of service balance index (SBI) Ψ rather than the $\Delta\Psi$ in the *inflation ratio* γ of the link cost function.

Although enhancing the traffic balancing, SSR behaves differently from SSR* in different loading situations. The function *FindRouteLeastCost* of SSR attempts to find a path which results in more balanced traffic, and the *FindRouteLeastCost* of SSR* is encouraged to find a path with lightest loading. For example, assume there are two paths from s to d, say σ_a and σ_b. There is one flow of class $C1$ and one flow of class $C2$ on the path σ_a while nothing is on the path σ_b. If the incoming flow from s to d is class $C3$, function *FindRouteLeastCost* of SSR* will find the path σ_b to route the $C3$ flow. However, *FindRouteLeastCost* of SSR will obtain σ_a as the least cost path to balance the traffic classes.

5.1 Network Model

Random graphs [17] are used to simulate network models and ensure that the effects of the different routing algorithms are independent of any specific network. For all possible pairs (u, v) of nodes, the edges can be added to the graph by using the probability function:

$$P_e(u,v) = \beta \ \exp\left(\frac{-d(u,v)}{\alpha L}\right), \tag{4}$$

where $d(u,v)$ denotes the *Euclidean* distance between nodes u and v, L is the maximum possible distance between any two nodes in the graph, and α and β represent parameters of real numbers in the range (0, 1). Setting the values of parameters α and β to 0.25 and 0.2, respectively, can generate graphs which appear to resemble geographical maps of major nodes in the Internet.

In this model, n nodes are randomly distributed over a rectangular coordinate grid, and the distance between each pair of nodes is calculated. Edges are then introduced by the probability P_e. The average degree of the nodes in these graphs is in the range [3.5, 5]. Each link is assumed to be STM-1 with 155 Mbps.

5.2 Traffic Model

As listed in Table 2, three classes of QoS traffic exist in the experiments herein, namely $C1$, $C2$, and $C3$, to carry the compressed real-time voice, video connections, and high precedence data or pictures, for instance. Service classes $C1$, $C2$, and $C3$ are given a service weight of 3, 2, and 1, respectively. Meanwhile, service class $C1$ has higher priority than $C2$, and $C2$ has higher priority than $C3$.

Flow requests of class k are assumed to arrive at an S-D pair according to a *Poisson* process with a rate of λ_k. The mean holding time for a flow of class k is assumed to be exponentially distributed with mean $1/\mu_k$, as shown in Table 2. Flow requests are evenly destined among other nodes in the network. A request whose QoS requirement cannot be satisfied is dropped without further affecting the system.

A *target utilization* of 80% is set for QoS flows to protect the best-effort traffic from starvation, and thus the best-effort traffic does not influence the simulation results.

Table 2 Traffic model of the simulation.

service class	traffic type	weight	arrival ratio	BW req. (Kbps)	mean holding time (sec)
C1	VBR	3	45%	32	180
C2	VBR	2	5%	256	600
C3	VBR	1	50%	384	72

5.3 Simulation Results

The simulation results mainly examine the behavior of the SSR routing scheme under various loadings. SSR intends to balance the traffic classes while keeping the fractional reward loss and the bandwidth blocking probability lower than the other routing schemes. The 95% *confidence interval* lies within 5% of the simulation average for all the statistics reported herein. Unless otherwise specified, the results presented herein are based on the 40-node and 170-link random topologies as described in Sect. 5.1, and the *widest-shortest path* computation is used.

5.3.1 Service Balance Index

Figure 4 shows the average link SBI statistics under various load and *path utilization threshold*, θ, for the SSR routing scheme, while Fig. 5 presents the improvement of SSR in terms of the percentage reduction in averaged link SBI compared with *SPR*. According to Figs. 4 and 5, the lower the threshold implies a better

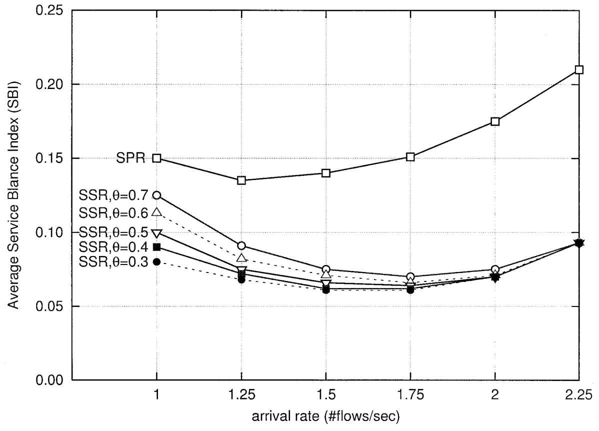


Fig. 4 Performance of the SSR (SBI).

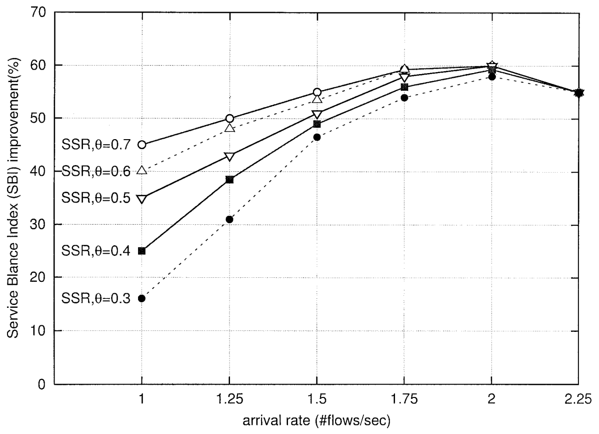


Fig. 5 SBI improvement of the SSR (Compared to SPR).

SSR performance. This effect is especially conspicuous as the traffic loading of the network reduces. Notably, although performance can be enhanced by using a lower threshold, a threshold value that approximately equals the average utilization of each path between the S-D pair is recommended.

In these experiments, SSR reduces the averaged link SBI by 40% to 60% more than SPR. Given a path utilization threshold θ, the extent of the improvement increases with the load. This phenomenon occurs because using alternative routes is set by the threshold θ, and thus the lower the traffic load, the more imbalanced it becomes. On the other hand, as the traffic load becomes extremely high, the improvement is reduced since some of the links saturated and become the bottleneck of service balancing. Note that the unfairness is less significance under the light traffic load.

According to Fig. 6, the links are categorized into several utilization levels. Both SSR and SSR* have effects on the traffic balancing. SSR* performs better (lower link SBI) than SSR as the offered load is light. On the other hand, SSR scheme is better for links with higher utilization levels. Furthermore, SSR scheme is less affected by the increasing of loading than SSR* scheme. Notably, service balancing is generally a greater concern over highly utilized links than with little utilized links.

Figure 7 shows the average SBI of individual links in the network while Fig. 8 presents the variance of the average SBI of the links. From the figures, with SSR and SSR*, the network links not only have much lower average SBI, but also much lower variance of SBI compared with SPR.

Obviously, the average value of SBI is a concave function of the arrival rate or the offered load; this effect becomes more apparent as threshold θ becomes larger. When the offered load is light, most of the paths held in cache are lightly utilized, flow request simply follows the cached route to carry the data traffic. As the number of flow requests is increased, SSR has more opportunities to balance the traffic. Therefore, the values of SBI of the links are reduced. However, as the loading of some links becomes heavy or even bottlenecks occur, less choices of the route from source to the destination are available. Consequently, more links are imbalanced and the values of SBI increase with the load.

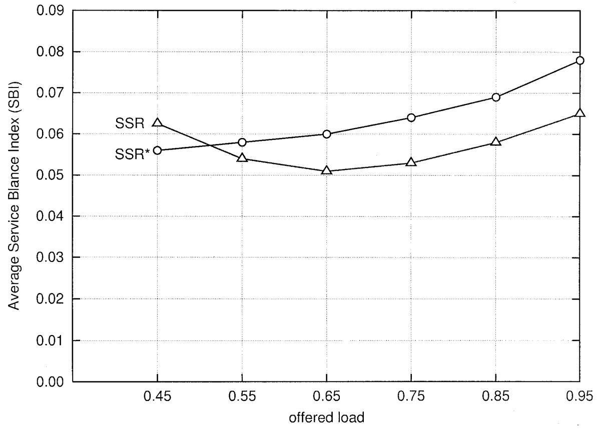


Fig. 6 Effects of the SSR* vs. SSR, λ = 1.75, θ=0.3.

5.3.2 Queuing Delay

A *Self-Clocked Fair Queuing* (SCFQ) system measures queuing delay suffered by packets of flows in different service classes. Table 3 lists the parameters of the simulation. The size of the packets is generated randomly according to the mean size and the spread of the distribution.

Table 4 shows the queuing delay statistics for packets of service classes 1, 2 and 3. Flows of a specific class

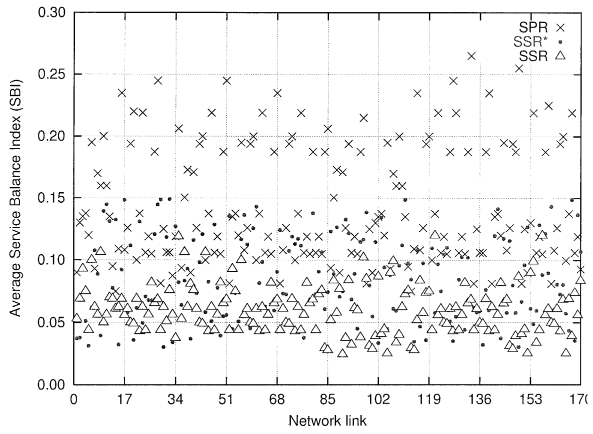


Fig. 7 SBI of each link, λ = 1.75, θ=0.3.

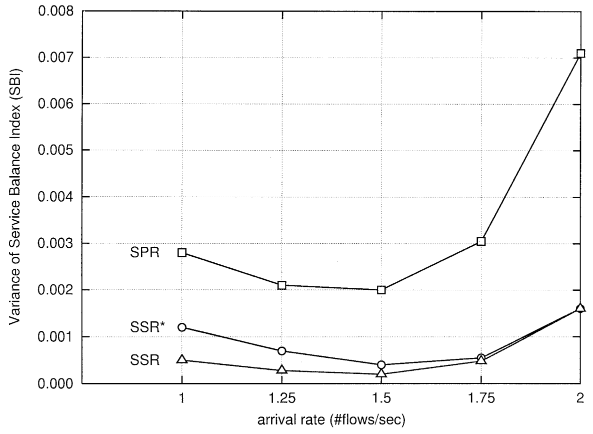


Fig. 8 Variance of the average SBI, θ=0.3.

Table 3 Parameters of the SCFQ system.

service class	weight	mean packet size (byte)	bandwidth allocation (%)
C1	3	100	20
C2	2	1000	40
C3	1	1500	40

Table 4 Average queuing delay, threshold θ=0.3, λ=arrival rate(#flows/sec), $\bar{x}$=mean(μs), $\hat{s}$=std. deviation, δ=reduction (%).

λ	scheme	C1 $\bar{x},(\delta)$	C1 $\hat{s},(\delta)$	C2 $\bar{x},(\delta)$	C2 $\hat{s},(\delta)$	C3 $\bar{x},(\delta)$	C3 $\hat{s},(\delta)$
1.00	SPR	28(-)	12(-)	110(-)	27(-)	177(-)	69(-)
	SSR*	27(4)	7(36)	105(5)	15(47)	160(10)	29(59)
	SSR	27(4)	7(36)	105(5)	14(47)	160(10)	28(59)
1.25	SPR	34(-)	16(-)	130(-)	53(-)	293(-)	346(-)
	SSR*	32(6)	11(34)	119(9)	33(37)	212(28)	201(42)
	SSR	31(7)	10(39)	118(9)	30(43)	202(31)	167(52)
1.50	SPR	42(-)	19(-)	157(-)	74(-)	552(-)	806(-)
	SSR*	39(8)	14(27)	139(12)	52(30)	347(37)	511(37)
	SSR	37(10)	13(32)	133(15)	49(35)	320(42)	443(45)
1.75	SPR	49(-)	21(-)	189(-)	91(-)	916(-)	1178(-)
	SSR*	45(10)	16(24)	163(14)	69(24)	604(34)	925(22)
	SSR	43(13)	15(28)	155(18)	65(28)	550(40)	850(28)
2.00	SPR	59(-)	22(-)	232(-)	103(-)	1539(-)	1537(-)
	SSR*	52(12)	17(22)	196(15)	82(20)	1031(33)	1308(15)
	SSR	50(15)	16(24)	186(20)	80(22)	923(40)	1248(19)
2.25	SPR	73(-)	23(-)	298(-)	109(-)	2562(-)	1780(-)
	SSR*	63(14)	19(19)	252(16)	96(12)	1802(30)	1785(0)
	SSR	60(18)	19(19)	226(24)	96(12)	1537(40)	1778(0)

with a particular arrival rate exhibit a lower mean and standard deviation (variance) of queuing delay, and either a SSR* or SSR scheme is used. Also, as the traffic load increases, the mean of the queuing delay is also reduced. Conversely, the reduction of queuing delay variance decreases with increasing offered load. This phenomenon occurs because in a high offered load situation, most of the network links become saturated. Therefore, no big difference occurs when traversing different routes.

This experiment confirms that a routing scheme that considers the mix of traffic classes could significantly reduce unfairness in suffering queuing delays.

5.3.3 Fractional Reward Loss

Let W denote the set of all possible (s,d) pairs, $W = \{(s,d)|s,d \in V, s \neq d\}$. Flow requests of class k are assumed to arrive at S-D pair w according to a Poisson process with rate λ_k^w. Each class k flow of S-D pair w carried on the network produces $(b_k d_k^w)$ units of revenue, where b_k denotes the bandwidth requirement of flows of class k. Meanwhile, d_k^w is the distance of the shortest path between S-D pair w. From another point of view, the network losses $(b_k d_k^w)$ units of revenue for each class k connection of S-D pair w that is rejected. Formally, the *fractional reward loss* ($\mathcal{L}$) of a network is defined as

$$\mathcal{L} = \frac{reward\ loss}{total\ reward} = \frac{\sum_{k=1}^{K}\sum_{w\in W} b_k d_k^w \lambda_k^w P_k^w}{\sum_{k=1}^{K}\sum_{w\in W} b_k d_k^w \lambda_k^w}, \quad (5)$$

where P_k^w represents the blocking probability of class k traffic of S-D pair w. Minimizing the fractional reward loss is equivalent to maximizing the expected revenues produced by the network. According to Fig. 9, the network with the SSR scheme suffers less fractional reward loss than the others.

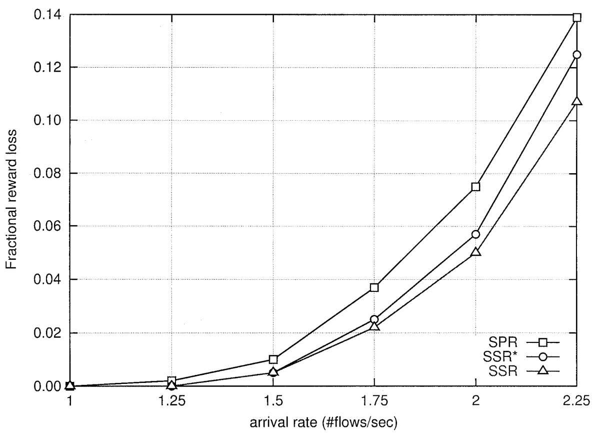


Fig. 9 Fractional reward loss ($\mathcal{L}$), θ=0.3.

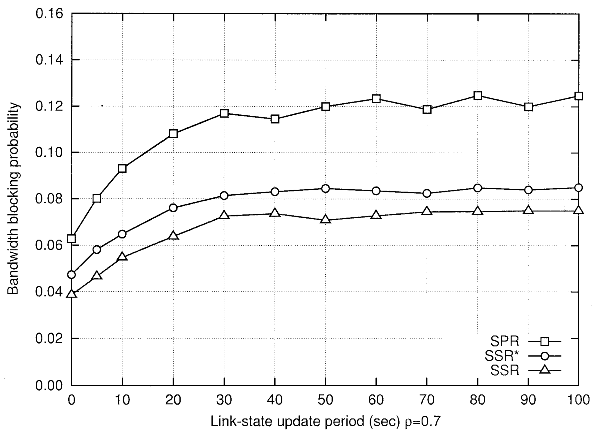


Fig. 10 Bandwidth blocking probability (P_b), θ=0.3.

5.3.4 Bandwidth Blocking Probability

Bandwidth blocking probability, P_b, is defined as

$$P_b = \frac{\sum rejected\ bandwidth}{\sum request\ bandwidth}. \quad (6)$$

This experiment deals with the effects of inaccurate link-state information, due to the link state update periods. Figure 10 shows the bandwidth blocking probabilities, P_b, with an offered load ρ=0.7, and where the mean holding time is adjusted to fix the offered load, and the refresh timeout of the cache entry is 100 units. As expected, longer update periods basically increase blocking. A longer update period causes greater inaccuracy in the link-state, and more changes in network could fail to be noticed. As links approach saturation under the inaccurate state, the link state and residual bandwidth databases viewed from the source router are likely to be unchanged, meaning that an infeasible path may be mistaken for a feasible one. In that case, flows are blocked in the signaling phase and only admitted if other flows leave.

SPR scheme achieves a higher blocking probability than other schemes. Traffic tends to form bottleneck links in the SPR scheme, and is more imbalanced than with other schemes. Conversely, the traffic obtains a feasible path more flexibly in the SSR routing networks, and has more chances to find alternative paths.

Traffic engineering in MPLS networks is the process of arranging how traffic flows through the network to avoid network congestion caused by uneven network utilization. QoS routing is an important part of traffic engineering which may find a longer but lightly or suitably loaded path better than a heavily or unevenly loaded shortest path. Hence QoS routing can help make the traffic engineering process automatic. SSR scheme couples QoS routing and provisioning can lead to efficient selection of routes, increasing the likelihood of meeting the applications' end-to-end QoS requirements.

6. Conclusion

QoS routing is an important component of traffic engineering in MPLS networks. This study investigated the problem of unfairness when QoS routing does not consider the mix of traffic classes. To resolve the unfairness of queuing delay, this work presents a new routing scheme—*Service-sensitive Routing* (SSR), which takes the state of traffic mixture of various service classes into account. To determine the QoS route for a flow request, SSR not only considers the available bandwidth and delay of the candidate paths, but also considers the mix of traffic classes on the paths. Additionally, the hybrid granularity routing decision in the SSR scheme is scalable and suitable for labeling and forwarding in the *Differentiated Services* and MPLS networks.

Extensive simulations show that the SSR scheme does significantly decrease the average *Service Balance Index* (SBI) of the network and the variance of average queuing delays compared to the existing *Shortest Path Routing* (SPR), for example by approximately 20% to 35% with a moderate offered load. Furthermore, this routing scheme maintains a fractional reward loss and bandwidth blocking probability significantly lower than the SPR scheme. In summary, the proposed *Service-sensitive Routing* leads to efficient selection of routes, increasing the likelihood of meeting the services' end-to-end QoS requirements.

Acknowledgement

The authors wish to acknowledge the constructive comments from the anonymous reviewers.

References

[1] G. Apostolopoulos, S. Kamat, and R. Guerin, "Implementation and performance measurements of QoS routing extensions to OSPF," Proc. Infocom'99, IEEE, July 1999.

[2] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda, and T. Przygienda, "QoS routing mechanisms and OSPF extensions," RFC 2676, Aug. 1999.

[3] Y. Bernet, J. Binder, S. Blake, M. Carlson, S. Keshav, E. Davies, B. Ohlman, D. Verma, Z. Wang, and W. Weiss, "A framework for differentiated services," draft-ietf-diffserv-framework-02.txt, Feb. 1999.

[4] D. Black, S. Blake, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," RFC 2475, Dec. 1998.

[5] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, and A. Viswanathan, "A framework for multiprotocol label switching," draft-ietf-mpls-framework-05.txt, Sept. 1999.

[6] A. Demers, S. Keshav, and S. Shenkar, "Analysis and simulation of a fair queueing algorithm," Internet. Res. and Exper., vol.1, 1990.

[7] F. Le Faucheur, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, and P. Cheval, "MPLS support of differentiated services," draft-ietf-mpls-diff-ext-02.txt, Oct. 1999.

[8] S.J. Golestani, "A self-clocked fair queuing scheme for broadband applications," Proc. Infocom'94, pp.636–646, IEEE, April 1994.

[9] P. Goyal, S.S. Lam, and H.M. Vin, "Determining end-to-end bounds in heterogeneous networks," ACM/Springer-Verlag Multimedia System J., vol.5, no.3, pp.157–163, May 1997.

[10] R. Guerin, A. Orda, and D. Williams, "QoS routing mechanisms and OSPF extensions," Internet Draft, draft-guerin-qos-routing-ospf-03.txt, Jan. 1998.

[11] T. Li and Y. Rekhter, "A provider architecture for differentiated services and traffic engineering(PASTE)," RFC 2430, Oct. 1998.

[12] Q. Ma and P. Steenkiste, "Supporting dynamic inter-class resource sharing: A multi-class QoS routing algorithm," Proc. Infocom'99, IEEE, July 1999.

[13] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the differentiated services field (DS Field) in the ipv4 and ipv6 headers," RFC 2474, Dec. 1998.

[14] E.C. Rosen, Y. Rekhter, D. Tappan, D. Farinacci, G. Fedorkow, T. Li, and A. Conta, "MPLS label stack encoding," draft-ietf-mpls-label-encaps-07.txt, Sept. 1999.

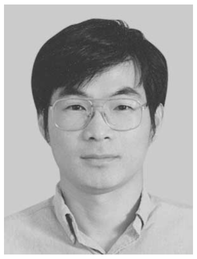
[15] E.C. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," draft-ietf-mpls-arch-06.txt, Aug. 1999.

[16] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round-robin," IEEE/ACM Trans. Networking, vol.4, no.3, June 1996.

[17] B.M. Waxman, "Routing of multipoint connections," IEEE J. Sel. Areas Commun., vol.6, no.9, pp.1617–1622, Dec. 1988.

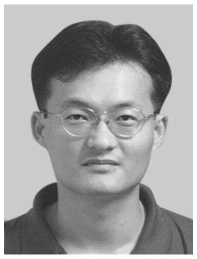
[18] L. Zhang, "Virtual clock: A new traffic control algorithm for packet switching networks," ACM Trans. Comput. Syst., vol.9, no.9, pp.101–124, May 1991.

[19] Z. Zhang, C. Sanchez, B. Salkewicz, and E. Crawley, "QoS extensions to OSPF," Internet Draft, draft-zhang-qos-ospf-01.txt, Sept. 1997.



Nai-Bin Hsu received the B.S. degree from National Taiwan Institute of Technology and the M.S. degree in Computer Science and Information Engineering from National Chiao Tung University (NCTU), Taiwan, in 1984 and 1990, respectively. He joined Telecom. Lab., Chungli, Taiwan in 1984, where he worked on the project of National Information Infrastructure. He is currently working toward the Ph.D. degree in Computer Information Science from NCTU. His research interests include of QoS routing, and performance evaluation of data comm. networks.

Ying-Dar Lin received the B.S. degree in Computer Science and Information Engineering from National Taiwan University in 1988, and the M.S. and Ph.D. degrees in Computer Science from the University of California, Los Angeles in 1990 and 1993, respectively. He joined the faculty of the Department of Computer and Information Science at National Chiao Tung University in August 1993 and is Professor since 1999. His research interests include design, analysis, and implementation of network protocols and algorithms, wire-speed switching and routing, quality of services, and intranet servers.



Mao-huang Li received the B.S. degree in Electrical Engineering from National Sun Yat-Sen University and M.S. degree in Communication Engineering from National Chiao Tung University, Taiwan, in 1991 and 2000, respectively. He joined Lucent Technologies—Taiwan Telecommunications in 1993. He is currently a senior manager of Network Planning and Solutions department.

Tsern-Huei Lee received the B.S. degree from National Taiwan University, Taipei, Taiwan, in 1981, the M.S. degree from University of California, Santa Barbara, in 1984, and the Ph.D. degree from University of Southern California, Los Angeles, in 1987, all in electrical engineering. Since 1987 he has been a member of the Faculty of National Chiao Tung University, Hsinchu, Taiwan, R.O.C., where he is a Professor with the Dept of Communication Engineering and a member of the Center for Telecom Research. His current research interests are in communication protocols, broad-band swich architectures, and traffic management. Dr. Lee received an Outstanding Paper Award from the Institute of Chinese Engineers in 1991.