Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 50 (2006) 3300–3314
Computer Networks
www.elsevier.com/locate/comnet

Rate-distortion based link state update [☆]

Gang Cheng [a], Nirwan Ansari [b,*]

[a] VPISystems Inc., 943 Holmdel Road, Holmdel, NJ 07733, United States
[b] Advanced Networking Laboratory, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07012, United States

Received 18 April 2005; received in revised form 3 August 2005; accepted 5 December 2005
Available online 23 January 2006

Responsible Editor: A. Orda

Abstract

Finding paths that satisfy the performance requirements of applications according to the link state information in a network is known as the quality-of-service (QoS) routing problem and has been extensively studied. However, distributing the link state information may introduce a significant protocol overhead on network resources. In this paper, based on rate-distortion analysis, we investigate the issue on how to update the link state information efficiently and effectively. A theoretical framework is presented, and a high performance link state policy that is capable of minimizing the false blocking probability of connections under a given update rate constraint is proposed. Through theoretical analysis and extensive simulations, we show that the proposed policy outperforms the current state of the art in terms of the update rate and false blocking probability of connections.

Keywords: Link state update; QoS routing; Rate-distortion function; Information theory

1. Introduction

The tremendous growth of the global Internet has given rise to a variety of applications that require quality-of-service (QoS) beyond what is provided by the current best-effort IP packet delivery service. One of the challenging issues is to select feasible paths that satisfy different quality-of-service (QoS) requirements. This problem is known as QoS routing. In general, two issues are critical to QoS routing: the state distribution and the routing strategy [1]. The routing strategy is used to find a feasible path which meets the QoS requirements; this has been extensively studied in the literature [2–7]. The state distribution addresses the issue of exchanging the state information throughout the network and can be further decomposed into two sub-problems: when to update and how to disseminate the link state information. This paper focuses on the state distribution, especially on the former sub-problem. A number of research works have also been reported on how to disseminate the link state

☆ This work has been supported in part by the National Science Foundation under Grant 0435250, and the New Jersey Commission on Science and Technology via NJWINS.
* Corresponding author. Tel./fax: +1 973 596 3670.
E-mail address: ansari@njit.edu (N. Ansari).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2005.12.008

information [8–11], which is, however, beyond the scope of this paper.

Many proposed QoS routing solutions assume that accurate link state information is available to each node. However, this is impossible in real networks. Moreover, to facilitate accurate enough link state information would impose a significant bandwidth and processing overhead on the network resource, i.e., the network resource will be greatly consumed if an update of the link state information is triggered whenever a minor change to the QoS parameters occurs. Current link-state routing protocols such as OSPF [12] recommend that the link state is updated periodically with large intervals. For instance, a link disseminates its state information every 30 min in OSPF. Consequently, because of the highly dynamic nature of link state parameters, the link state information known to a node is often outdated. As a result, the effectiveness of the QoS routing algorithms may be degraded significantly. To overcome this problem, several link state update policies (threshold, equal class, and exponential class based update policies) have been proposed in [13]. Given a predefined threshold value (τ), an update is triggered in the threshold based update policy if $|b_c - b_o|/b_o > \tau$, where b_o is the last advertised value of the available bandwidth, and b_c is the current available bandwidth. However, in equal class and exponential class based update policies, the bandwidth is partitioned into classes and an update is triggered whenever the available bandwidth crosses a class boundary. The only difference between them is that the bandwidth is partitioned into classes of equal size $((0,B),(B,2B),\ldots)$ in equal class based update policy, while it is partitioned into unequal classes, whose sizes $((0,B),(B,(f+1)B),((f+1)B,(f^2+f+1)B),\ldots)$ grow geometrically by a factor f, in the exponential class based update policy, where B is a predefined constant. Recently, many works considering the effects of the stale or coarse-grained information on the performance of QoS routing algorithms were reported in the literature. In [14], extensive simulations were made to uncover the effects of the stale link state information and random fluctuations in the traffic load on the routing and setup overheads. In [15,16], the effects of the stale link state information on QoS routing algorithms were demonstrated through simulations by varying the link state update interval. A combination of the periodic and triggered link state update is considered in [17]. Instead of using the link capacities or instantaneous available bandwidth values, Li et al. [18] used a stochastic metric, Available Bandwidth Index (ABI), and extended BGP to perform the bandwidth advertising.

As reviewed above, although many link state update policies have been proposed, there is still a lack of a rigorous theoretical foundation. As a result, they may not be efficient enough and may waste the network resource. In this paper, we will provide a theoretical framework for link state update, from which we will propose a high performance link state update mechanism. We theoretically demonstrate that from the perspective of QoS routing, our proposed link state policy outperforms its contenders in terms of the update rate and false blocking probability of incoming connections. The rest of the paper is organized as follows. The problem is formulated in Section 2. In Section 3, we provide an insight on link state update based on Information Theory. We propose an efficient link state update policy in Section 4, and present the simulation results in Section 5. Finally, concluding remarks are given in Section 6.

2. Problem formulation

A key challenge for a network simulation is the selection of the network topology and the traffic patterns. Owing to the constantly changing and decentralized nature of current networks, it is rather difficult to define a typical network topology applicable for exploring any protocols [19]. Moreover, the results of a simulation over different network topologies with different traffic patterns may vary dramatically. In this paper, the problem of designing an efficient link state update policy in a network is translated into that of finding an update policy for a single link. We further discover that the performance of a link state update policy in a network can be evaluated over a single link. As a result, we avoid the above problems. Moreover, our proposed link state policy is designed without any assumption on the traffic pattern of the network.

Without loss of generality, we only focus on one link state metric (here, we adopt bandwidth). If a route is successfully found for a connection, the corresponding portion of the bandwidth (the requested bandwidth of the connection) on each link it traverses is reserved. Practically, the protocol overhead will be intolerably high if link state is updated whenever a minor change occurs. Hence, for the sake of reducing the protocol overhead, the staleness of the link state information is inevitably introduced.

In this paper, we focus on minimizing the staleness of the link state information introduced by an update scheme.

Many works have been reported in literature to evaluate the impact of the stale link state information on the performance of QoS routing algorithms [14–16]. However, how to numerically measure the staleness of the link state information has never been discussed. In this paper, in order to propose an efficient update scheme, we introduce a novel method to address this issue.

At any moment, when a connection request arrives at a node, we assume that the node tries to compute a path which meets the QoS requirements of the connection according to its available link state information (source routing). If, from the perspective of the node, there are enough network resources (bandwidth) to accommodate this connection, it starts a setup process for the connection. Otherwise, it rejects the connection request immediately. Ideally, the connection is accepted if there are actually enough network resources, and rejected otherwise. However, due to inaccurate link state information and the adopted routing algorithm, there are two possible undesirable cases due to the staleness of link state information:

- *False positive*: There are actually not enough network resources to accommodate a connection, but is indicated otherwise by its link state information. A setup process will be initialized by the node, thus wasting network resources.
- *False negative*: A connection can actually be accepted by the network, but is rejected by the node because of inaccurate link state information or failure of the adopted routing algorithm in finding a feasible path.

For example, as shown in Fig. 1, assume a connection going to node 3 with a requested bandwidth of 8 Mbps arrives at node 1; a false positive occurs if, from the perspective of node 1, the available bandwidth of link e_2 is 9 Mbps while in reality it is only 6 Mbps (because node 1 thinks that there are enough network resources available to accommodate the connection, and accepts the connection); on the other hand, if the actual available bandwidth of link e_2 is 9 Mbps and the one known to node 1 is 6 Mbps, a false negative occurs. Collectively, both cases are referred to as the false routing in this paper. Hence, *the staleness of link state information can be evaluated in terms of the probability of false routing*. On the other hand, defining c_1 and c_2 as the costs of the case of a false positive and negative, respectively, we can also evaluate the staleness of link state information in terms of the average cost resulted from false routing. Assume that on the arrival of a connection request, the probabilities of a false positive and false negative are p_{fp} and p_{fn}, respectively. Therefore, the average cost due to false routing upon the arrival of a connection request is

$$\bar{c} = p_{\mathrm{fp}}c_1 + p_{\mathrm{fn}}c_2. \tag{1}$$

Intuitively, there is no false routing ($p_{\mathrm{fp}} = 0$ and $p_{\mathrm{fn}} = 0$) and the cost is zero if the link state information is exactly accurate. The price for this optimality is a very large protocol overhead due to the dynamic nature of link state in current networks: huge amount of bandwidth is consumed for disseminating the link state information. Hence, it is practical to accept a limited probability of false routing in order to keep the protocol overhead under a reasonable level. Our objective is to minimize $\bar{c}$ under a given upper bound (network) on the average bandwidth used for the link state dissemination, which in turn yields an upper bound on the average bandwidth (r) consumed by every link for link state update. Assume the link state metric of a link is a time independent random process (with memory), each link can be viewed as a signal source and all

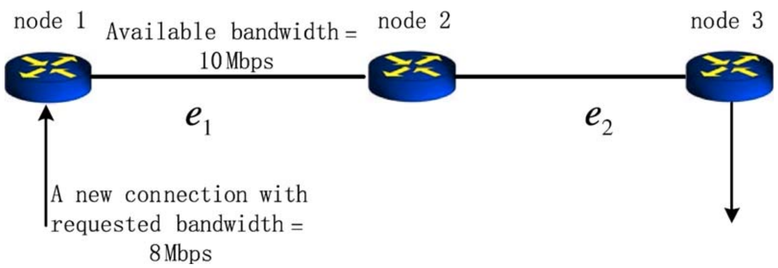

Fig. 1. A 3-node network consists of nodes 1, 2, and 3. The available bandwidth of link e_1 is 10 Mbps. A new connection arrives at node 1, whose requested bandwidth and destination are 8 Mbps and node 3, respectively.

nodes that do not directly connect to this link are receivers. From the perspective of Information Theory, our task become minimizing the source distortion (D) (the cost due to false routings) for a given transmission rate r (or to minimize the transmission rate r for a given source distortion). The rest of the theoretical analysis carried out in the paper is based on this concept.

Given a link, assume its available bandwidth known to each node is b_u, and the actual one is b_a. By definitions, a false positive is possible only if

$$b_u > b_a \tag{2}$$

and a false negative may occur only if

$$b_u < b_a. \tag{3}$$

Assume that the amount of the requested bandwidth of a connection is a random variable x with probability density function $p(x)$. Therefore, denoting $p_{fp}(b_u, b_a)$ and $p_{fn}(b_u, b_a)$ as the probabilities of false positives and negatives under the condition of b_u and b_a, we have

$$p_{fp}(b_u, b_a) = \frac{1}{2}\int_{\min\{b_u,b_a\}}^{\max\{b_u,b_a\}} p(x)\left[1 - \frac{(b_a - b_u)}{|b_a - b_u|}\right]dx \tag{4}$$

and

$$p_{fn}(b_u, b_a) = \frac{1}{2}\int_{\min\{b_u,b_a\}}^{\max\{b_u,b_a\}} p(x)\left[1 - \frac{(b_u - b_a)}{|b_a - b_u|}\right]dx. \tag{5}$$

Collectively, the average cost (source distortion) upon the arrival of a connection request is

$$\begin{aligned} e(b_u, b_a) &= p_{fp}(b_u, b_a)c_1 + p_{fn}(b_u, b_a)c_2 \\ &= \frac{c_1}{2}\int_{\min\{b_u,b_a\}}^{\max\{b_u,b_a\}} p(x)\left[1 - \frac{(b_a - b_u)}{|b_a - b_u|}\right]dx \\ &\quad + \frac{c_2}{2}\int_{\min\{b_u,b_a\}}^{\max\{b_u,b_a\}} p(x)\left[1 - \frac{(b_u - b_a)}{|b_a - b_u|}\right]dx \\ &= \int_{\min\{b_u,b_a\}}^{\max\{b_u,b_a\}} p(x)\left\{\frac{c_1}{2}\left[1 - \frac{(b_a - b_u)}{|b_a - b_u|}\right]\right. \\ &\quad \left. + \frac{c_2}{2}\left[1 - \frac{(b_u - b_a)}{|b_a - b_u|}\right]\right\}dx. \end{aligned} \tag{6}$$

Based on this definition (Eq. (6)), we provide two *interchangeable* definitions for the optimal link state update policy.

Definition 1. For a given source distortion to be less than or equal to D, the optimal link state update policy is the one that minimizes the transmission rate r of each link.

Definition 2. For a given transmission rate to be less than or equal to r, the optimal link state update policy is the one that minimizes the source distortion.

3. An insight from information theory

Many studies have been done to characterize the Internet traffic, revealing interesting facts such as Long-Range Dependence or multi-fractal behaviors [20–22]. Therefore, we cannot assume that the source signal (available bandwidth) is memoryless. Each QoS parameter of a link is viewed as a random process with memory that is independent of time in this paper. By applying the sampling theorem, each continuous random process can be converted into an equivalent discrete-time sequence of samples. Therefore, each QoS parameter on a link can be treated as a time independent continuous random sequence with memory. As shown in Fig. 2, denote $z_1, z_2, \ldots,$ and $y_1, y_2, \ldots$ as the exact sequence of the available bandwidth and the sequence of the available bandwidth known to a node that is not directly connected to the link, respectively, where C, b_{max}, and b_{min} ($0 \leqslant b_{min} \leqslant b_{max} \leqslant C$) are the link capacity, maximum requested bandwidth, and minimum requested bandwidth, respectively.

Note that the definition of an optimal link state update policy is the one that minimizes the transmission rate r of each link while limiting the source distortion to be less than or equal to a given constant D. Hence, we can apply the rate-distortion function in Information Theory to the problem of link state update. Denote $\mathbf{I}(\mathbf{A}; \mathbf{B})$ as the mutual information between $\mathbf{A}$ and $\mathbf{B}$, $\mathbf{H}(\mathbf{A})$ as self-entropy of $\mathbf{A}$, and $\mathbf{H}(\mathbf{A}|\mathbf{B})$ as the conditional entropy between $\mathbf{A}$ and $\mathbf{B}$. By Information Theory, the mutual information between $\mathbf{Y^n}$ and $\mathbf{Z^n}$ is

$$\begin{aligned} \mathbf{I}(\mathbf{Y^n}; \mathbf{Z^n}) &= \mathbf{H}(\mathbf{Y^n}) - \mathbf{H}(\mathbf{Y^n}|\mathbf{Z^n}) \\ &= \mathbf{H}(\mathbf{Y^n}) + \mathbf{H}(\mathbf{Z^n}) - \mathbf{H}(\mathbf{Y^n}\mathbf{Z^n}), \end{aligned} \tag{7}$$

where $\mathbf{Y^n}$ and $\mathbf{Z^n}$ represent the sequences $z_1, z_2, \ldots, z_n$ and $y_1, y_2, \ldots, y_n$, respectively. Note that

$$\mathbf{H}(\mathbf{Y^n}) = -\int \cdots \int_{Y^n} p(\mathbf{Y^n}) \log p(\mathbf{Y^n}) d\mathbf{Y^n} \tag{8}$$

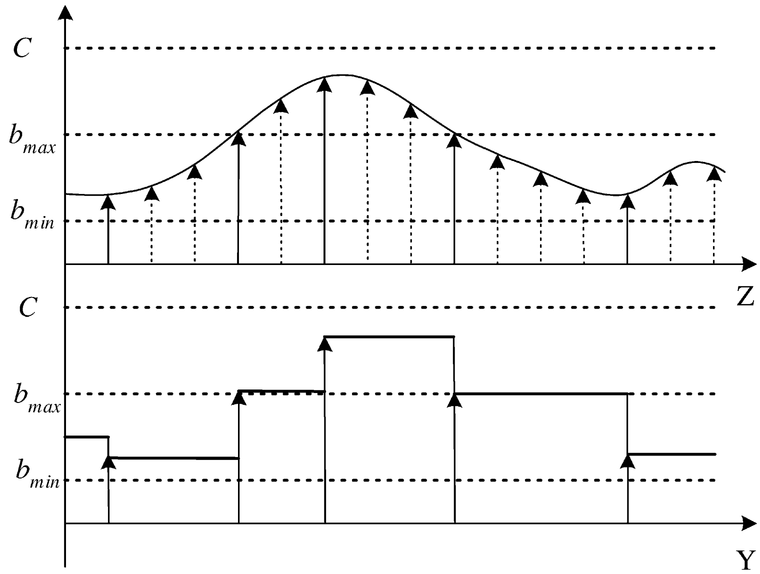

Fig. 2. A demonstration of $z_1, z_2, \ldots$ and $y_1, y_2, \ldots$ The solid arrows represent the corresponding samples that are used for updates while the dash arrows represent the samples that are not used.

and

$$\mathbf{H}(\mathbf{Y^n}|\mathbf{Z^n}) = -\underbrace{\int \cdots \int}_{\mathbf{Z^n}} \underbrace{\int \cdots \int}_{\mathbf{Y^n}} p(\mathbf{Z^n}, \mathbf{Y^n}) \times \log p(\mathbf{Y^N}|\mathbf{Z^n}) \mathrm{d}\mathbf{Y^n} \, \mathrm{d}\mathbf{Z^n}. \tag{9}$$

The optimal link state update policy satisfies

$$\begin{aligned} R(D) &= \min_{\xi_D} \left\{ \lim_{n\to\infty} \left\{ \frac{1}{n} \mathbf{I}(\mathbf{Z^n}, \mathbf{Y^n}) \right\} \right\} \\ &= \min_{\xi_D} \left\{ \lim_{n\to\infty} \left\{ \frac{1}{n} [\mathbf{H}(\mathbf{Y^n}) - \mathbf{H}(\mathbf{Y^n}|\mathbf{Z^n})] \right\} \right\}, \end{aligned} \tag{10}$$

where $R(D)$ is the bandwidth (a single link) used by the optimal link state update policy and ξ_D is the set of transition probabilities from $\mathbf{Z^n}$ to $\mathbf{Y^n}$ subject to the distortion D, i.e., ξ_D satisfies the following condition:

$$\lim_{n\to\infty} \left\{ \frac{1}{n} \sum_{i=1}^{n} \int_{\min\{y_i, z_i\}}^{\max\{y_i, z_i\}} p(x) \varphi(y_i, z_i) \mathrm{d}x \right\} \leqslant D, \tag{11}$$

where

$$\varphi(y_i, z_i) = \frac{c_1}{2} \left[1 - \frac{(z_i - y_i)}{|z_i - y_i|} \right] + \frac{c_2}{2} \left[1 - \frac{(y_i - z_i)}{|z_i - y_i|} \right].$$

Note that

$$\lim_{n\to\infty} \frac{1}{n} \mathbf{H}(\mathbf{Y^n}) = \lim_{n\to\infty} \mathbf{H}(y_n|\mathbf{Y^{n-1}}). \tag{12}$$

Similarly,

$$\lim_{n\to\infty} \frac{1}{n} \mathbf{H}(\mathbf{Z^n}) = \lim_{n\to\infty} \mathbf{H}(z_n|\mathbf{Z}^{\mathbf{n}-1}) \tag{13}$$

and

$$\lim_{n\to\infty} \frac{1}{n} \mathbf{H}(\mathbf{Y^n Z^n}) = \lim_{n\to\infty} \mathbf{H}(y_n z_n|\mathbf{Y^{n-1} Z^{n-1}}). \tag{14}$$

Hence,

$$\begin{aligned} R(D) = \min_{\xi_D} \{ \lim_{n->\infty} [&\mathbf{H}(y_n|\mathbf{Y^{n-1}}) + \mathbf{H}(z_n|\mathbf{Z^{n-1}}) \\ &- \mathbf{H}(y_n z_n|\mathbf{Y^{n-1} Z^{n-1}})] \}. \end{aligned} \tag{15}$$

If the memory lengths of all the sequences $\mathbf{Y^n}$ and $\mathbf{Z^n}$ are $\alpha + 1$, i.e., $\forall k > \alpha$,

$$p(y_k|y_1 y_2 \ldots y_{k-1}) = p(y_k|y_{k-1} y_{k-2} \ldots y_{k-\alpha}) \tag{16}$$

and

$$p(z_k|z_1 z_2 \ldots z_{k-1}) = p(z_k|z_{k-1} z_{k-2} \ldots z_{k-\alpha}), \tag{17}$$

we have

$$\begin{aligned} R(D) = \min_{\xi_D} \{ &\mathbf{H}(y_{\alpha+1}|\mathbf{Y^\alpha}) + \mathbf{H}(z_{\alpha+1}|\mathbf{Z^\alpha}) \\ &- \mathbf{H}(y_{\alpha+1} z_{\alpha+1}|\mathbf{Y^\alpha Z^\alpha}) \} \end{aligned} \tag{18}$$

and ξ_D satisfies that

$$\frac{1}{\alpha} \sum_{i=1}^{\alpha} \int_{\min\{y_i, z_i\}}^{\max\{y_i, z_i\}} p(x) \varphi(y_i, z_i) \, \mathrm{d}x \leqslant D. \tag{19}$$

By Eqs. (10) and (11), a lower bound on the transmission rate of the link state update under the constraint of a given source distortion may be computed. For the cases that sample sequences have memory length of $\alpha+1$, the computation of the lower bound can be simplified by solving Eqs. (18) and (19). In this paper, since no assumption on the distributions of $\mathbf{Y^n}$ and requested bandwidth of connections ($p(x)$) is made, we cannot compute $R(D)$. Practically, we may adopt the common approaches assuming that $\mathbf{Z^n}$ is self-similar distributed such as Pareto distribution in order to derive an explicit expression for $R(D)$, which we will consider in our future works. In this paper, we focus on presenting a link state update policy. Note that $R(D)$ is the lower bound on the transmission rate used for updating link state information under the constraint that the cost due to false routing is no larger than D. In turn, given an upper bound B on the transmission rate used for updating link state information, we can find out the corresponding lower bound on the cost by solving the equation

$$R(D) = B. \tag{20}$$

Observe that if $\varphi(y_i, z_i) = 1$ ($c_1 = c_2 = 1$), $R(D)$ is the minimized transmission rate under the constraint that the probability of false routing is no larger than D. Given an upper bound on the transmission rate, we can also compute the corresponding minimum probability of false routing for the link state update policies. Next, we approximately estimate the probability of false routing when we attempt to set up a connection between two nodes in a network. Denote $e_1, e_2, \ldots, e_m$ as the links in the network. Assume there are totally u possible routes, $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_\sigma$, and route r_k ($1 \leqslant k \leqslant \sigma$) consists of links $e_{k(1)}, e_{k(2)}, \ldots, e_{k(h_k)}$, where h_k is its hop count. If we simply view the link state on different links as independent, we can compute the probability of false routing of route r_k as

$$p_k = 1 - \prod_{i=1}^{h_k}(1 - p(e_{k(i)})), \tag{21}$$

where $p(e_{k(i)})$ is the probability of false routing on link $e_{k(i)}$. Further denote π_k, $1 \leqslant k \leqslant u$, as the normalized number of connection requests that tries to use route r_k. For example, denote the average number of connection requests in a unit of time on using route r_k as α_k, $\pi_k = \frac{\alpha_k}{\sum_{i=1}^{u}\alpha_i}$. Hence, $\sum_{k=1}^{u}\pi_k = 1$. In particular, if we envision a uniform distribution of connection requests among the routes, then $\pi_k = \frac{1}{u}$. Denote $\theta(s,d) = \{r_{\kappa(s,d,1)}, r_{\kappa(s,d,2)}, \ldots, r_{\kappa(s,d,\epsilon(s,d))}\}$ as the set of routes that starts at edge node s and ends at d, where $\epsilon(s,d)$ represents the number of the routes. Hence, we can compute the probability of false routing between nodes s and d as

$$p_{s,d} = \frac{1}{\sum_{i=1}^{\epsilon(s,d)}\pi_{\kappa(s,d,i)}} \sum_{i=1}^{\epsilon(s,d)} \pi_{\kappa(s,d,i)} p_{\kappa(s,d,i)}. \tag{22}$$

4. The proposed link state update policy

For simplicity, we only consider the case that $c_1 = c_2 = c$ in this paper, and thus

$$e(b_\mathrm{u}, b_\mathrm{a}) = c\int_{\min\{b_\mathrm{u},b_\mathrm{a}\}}^{\max\{b_\mathrm{u},b_\mathrm{a}\}} p(x)\mathrm{d}x. \tag{23}$$

The key difference between our proposed link state policy and those in [13] is that instead of partitioning the bandwidth into classes of equal sizes or exponentially growing sizes, we divide the bandwidth into classes by taking into account of the requested bandwidth of the connections and the available bandwidth, for the purpose of minimizing the source distortion under a bandwidth constraint for disseminating the link state information. As a result, our proposed link state update policy can avoid the unnecessary updates from the QoS routing perspective, and therefore, performs better than those in [13] in terms of the average bandwidth for disseminating the link state information and the average costs due to routing and setup failures. We theoretically prove that our proposed link state update policy is an optimal class based update policy in the sense that given the number of classes, it can minimize the average cost. Our proposed link state update policy consists of the following three steps:

1. Sample the available bandwidth. Different sampling strategies can be adopted. For instance, we can sample the available bandwidth every T_s, where T_s is the sampling interval. We can also compute the available bandwidth upon the acceptance and the end of a connection (only after a route with enough bandwidth is found for a connection, it is accepted).
2. Quantize the samples.
3. If the current quantized value is different from the previous updated one, use the current one for update.

Since data traffic in the current Internet is inherently bursty, a QoS parameter may fluctuate

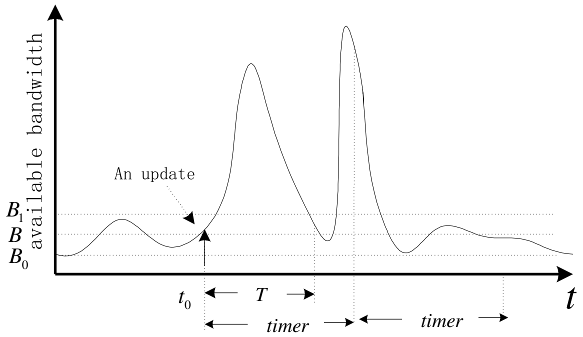

Fig. 3. The bursty traffic of the Internet.

dramatically in a very short time. As a result, some unnecessary updates may occur. For example, as shown in Fig. 3, the available bandwidth has been changed substantially in a very short time T, and many updates are triggered. Therefore, for the purpose of saving the bandwidth used for link state update, similar to [13], we can set a clamp down timer (the minimum interval between two consecutive updates) to make a link state update policy less sensitive to the fluctuation of the available bandwidth. However, it should be noted that setting a clamp down timer also has negative impacts on the accuracy of the link state information. As shown in Fig. 3, assume the average available bandwidth is B, and in most of the time, the available bandwidth is less than B_1 and larger than B_0. At t_0, an update of the link state information is triggered, and at the end of the corresponding timer, another link state information update is triggered because the available bandwidth is very large at that time. Observe that in most of the time of the second timer duration, the available bandwidth is between B_1 and B_0. However, because of the clamp down timer, no link state information update is allowed. Hence, the link state information during this timer period is rather inaccurate. Therefore, the clamp down timer is not recommended here.

Based on Eq. (23), we shall propose our class based update policy. As mentioned before, since all connection requests can be accepted by the link if its available bandwidth is larger than $b_{\max}$, updating link state information is not necessary if the available bandwidth remains larger than $b_{\max}$. Hence, the last class in our proposed class based update scheme is $(b_{\max}, C)$. Similarly, the first class of our proposed update policy is $(0, b_{\min})$. Therefore, we assume the bandwidth is partitioned into $n+2$ classes: $((0,b_{\min}),(b_{\min},b_1),(b_1,b_2),\ldots,(b_{n-1},b_{\max}),(b_{\max},C))$, and denote B_i as the value used for updating the link state information when the current available bandwidth is in class (b_{i-1},b_i) ($b_0=b_{\min}$ and $b_n=b_{\max}$). When the available bandwidth moves into $(0,b_{\min})$ or $(b_{\max}, C)$, we simply update link state information once with the update value of 0 or C. Note that, according to the definition of the class based update policy, the link available bandwidth from the perspective of the nodes is B_i if the actual one is in (b_{i-1},b_i). Hence, we define the average cost due to false positives and negatives (the distortion) as

$$d=\int_0^C f(x)e(\tilde{x},x)\mathrm{d}x, \tag{24}$$

where $e(\tilde{x},x)=\int_{\min\{\tilde{x},x\}}^{\max\{\tilde{x},x\}} p(x)\mathrm{d}x$, $\tilde{x}$ is the corresponding quantized value of x, and $f(x)$ is the probability density function of the available bandwidth. Hence,

$$\begin{aligned} d&=\int_0^C f(x)e(\tilde{x},x)\mathrm{d}x\\ &=\sum_{i=0}^{n-1}\int_{b_i}^{b_{i+1}}\left(\int_{\min\{B_i,x\}}^{\max\{B_i,x\}}p(y)\mathrm{d}y\right)f(x)\mathrm{d}x\\ &=\sum_{i=0}^{n-1}\int_{b_i}^{B_i}\left(\int_x^{B_i}p(y)\mathrm{d}y\right)f(x)\mathrm{d}x\\ &\quad+\sum_{i=0}^{n-1}\int_{B_i}^{b_{i+1}}\left(\int_{B_i}^{x}p(y)\mathrm{d}y\right)f(x)\mathrm{d}x. \end{aligned}\tag{25}$$

Note that

$$\int_0^C f(x)\mathrm{d}x=1 \tag{26}$$

and

$$\int_0^C p(x)\mathrm{d}x=1. \tag{27}$$

Therefore, we can form a Lagrangian Relaxation by incorporating the constraints, Eqs. (26) and (27), into Eq. (25)

$$\begin{aligned} &L(B_0,B_1,\ldots,B_{n-1},b_1,b_2,\ldots,b_{n-1})\\ &\quad=d-\lambda_1\left(\int_0^C f(x)\mathrm{d}x-1\right)\\ &\qquad-\lambda_2\left(\int_0^C p(x)\mathrm{d}x-1\right). \end{aligned}\tag{28}$$

So,

$$\frac{\partial L}{\partial b_i}=f(b_i)\left[\int_{B_{i-1}}^{b_i}p(x)\mathrm{d}x-\int_{b_i}^{B_i}p(x)\mathrm{d}x\right] \tag{29}$$

and

$$\frac{\partial^2 L}{\partial b_i^2} = f'(b_i)\left[\int_{B_{i-1}}^{b_i} p(x)\mathrm{d}x - \int_{b_i}^{B_i} p(x)\mathrm{d}x\right] + 2f(b_i)p(b_i).$$

Note that

$$\int_{B_{i-1}}^{b_i} p(x)\mathrm{d}x - \int_{b_i}^{B_i} p(x)\mathrm{d}x = 0 \Rightarrow \frac{\partial L}{\partial b_i} = 0 \tag{30}$$

and

$$\int_{B_{i-1}}^{b_i} p(x)\mathrm{d}x - \int_{b_i}^{B_i} p(x)\mathrm{d}x = 0 \ \Rightarrow \ \frac{\partial^2 L}{\partial b_i^2} = 2f(b_i)p(b_i) \geqslant 0. \tag{31}$$

Hence, $\forall i$, by letting

$$\int_{B_{i-1}}^{b_i} p(x)\mathrm{d}x - \int_{b_i}^{B_i} p(x)\mathrm{d}x = 0, \tag{32}$$

$L(B_0, B_1, \ldots, B_{n-1}, b_1, b_2, \ldots, b_{n-1})$ is minimized, i.e., the source distortion is minimized. Moreover,

$$\frac{\partial L}{\partial B_i} = p(B_i)\left(\int_{b_i}^{B_i} f(x)\mathrm{d}x - \int_{B_i}^{b_{i+1}} f(x)\mathrm{d}x\right) \tag{33}$$

and

$$\frac{\partial^2 L}{\partial B_i^2} = \acute{p}(B_i)\left(\int_{b_i}^{B_i} f(x)\mathrm{d}x - \int_{B_i}^{b_{i+1}} f(x)\mathrm{d}x\right) + 2p(B_i)f(B_i). \tag{34}$$

By letting

$$\int_{b_i}^{B_i} f(x)\mathrm{d}x - \int_{B_i}^{b_{i+1}} f(x)\mathrm{d}x = 0, \tag{35}$$

$$\frac{\partial^2 L}{\partial B_i^2} = \acute{p}(B_i)\left(\int_{b_i}^{B_i} f(x)\mathrm{d}x - \int_{B_i}^{b_{i+1}} f(x)\mathrm{d}x\right) + 2p(B_i)f(B_i) = 2p(B_i)f(B_i) \geqslant 0 \tag{36}$$

and

$$\frac{\partial L}{\partial B_i} = p(B_i)\left(\int_{b_i}^{B_i} f(x)\mathrm{d}x - \int_{B_i}^{b_{i+1}} f(x)\mathrm{d}x\right) = 0, \tag{37}$$

$L(B_0, B_1, \ldots, B_{n-1}, b_1, b_2, \ldots, b_{n-1})$ is minimized. So, the optimal partition of the bandwidth can be achieved by solving Eqs. (32) and (35). Observe that, if $f(x) = p(x)$, from Eqs. (32) and (35), we have

$$\int_{B_{i-1}}^{b_i} f(x)\mathrm{d}x - \int_{b_i}^{B_i} f(x)\mathrm{d}x = 0 \tag{38}$$

and

$$\int_{b_i}^{B_i} f(x)\mathrm{d}x - \int_{B_i}^{b_{i+1}} f(x)\mathrm{d}x = 0. \tag{39}$$

It follows that $\forall i, j < n \ (i,j > 1)$

$$\int_{b_{i-1}}^{b_i} f(x)\mathrm{d}x = \int_{b_{j-1}}^{b_j} f(x)\mathrm{d}x. \tag{40}$$

Then,

$$\int_0^C f(x)\mathrm{d}x = 1 \ \Rightarrow \ \forall i, \quad \int_{b_{i-1}}^{b_i} f(x)\mathrm{d}x = \frac{1}{n}. \tag{41}$$

Therefore, when both the requested bandwidth and the available bandwidth of connections are uniformly distributed from 0 to C, the partition of the bandwidth of the equal class based update policy proposed in [13] is optimal. With Eq. (25), we can further derive the minimized source distortion when the number of classes of bandwidth is given. Assume the available bandwidth is uniformly distributed, i.e., $f(x) = \frac{1}{C}$. Therefore, by Eq. (35), $B_i - b_i = b_{i+1} - B_i$, for all $0 \leqslant i < n$. Hence, by Cauchy mean-value theorem and Eq. (32),

$$\begin{aligned} \int_{B_{i-1}}^{b_i} p(x)\mathrm{d}x &= \int_{b_i}^{B_i} p(x)\mathrm{d}x \\ &\Rightarrow \ p(\xi_{i-1})(b_i - B_{i-1}) = p(\delta_i)(B_i - b_i) \\ &\Rightarrow \ p(\xi_{i-1})(b_i - b_{i-1}) = p(\delta_i)(b_{i+1} - b_i), \end{aligned} \tag{42}$$

where $\forall i$, $B_{i-1} \leqslant \xi_{i-1} \leqslant b_i$ and $b_i \leqslant \delta_i \leqslant B_i$. Hence, if $p(\delta_i) > p(\xi_{i-1})$, then $b_{i+1} - b_i < b_i - b_{i-1}$, i.e., the size of the ith class is less than that of the $(i-1)$th class, which conforms to our claim that relatively accurate link state information is preferred at the point (available bandwidth) where the value of the probability density function of the requested bandwidth of connections is large. Next, we provide an example of computing $b_1, b_2, \ldots, b_{n-1}$ ($b_0 = 0$ and $b_n = C$) and $B_0, B_1, \ldots, B_{n-1}$. Assume the available bandwidth is uniformly distributed from 0 to C ($f(x) = \frac{1}{C}$). Therefore,

$$\begin{aligned} \frac{\partial f}{\partial b_i} = 0 \ &\Rightarrow \ \frac{\partial}{\partial b_i}\left(\int_{B_{i-1}}^{B_i} \int_{\min\{b_i,y\}}^{\max\{b_i,y\}} p(y)\mathrm{d}y\mathrm{d}x\right) = 0 \\ &\Rightarrow \ \frac{\partial}{\partial b_i}\left(\int_{B_{i-1}}^{b_i} \int_x^{b_i} p(y)\mathrm{d}y\mathrm{d}x + \int_{b_i}^{B_i} \int_{b_i}^{x} p(y)\mathrm{d}y\mathrm{d}x\right) = 0 \\ &\Rightarrow \ b_i = \frac{B_{i-1} + B_i}{2}. \end{aligned} \tag{43}$$

Further assume that the requested bandwidth is also uniformly distributed but from σ to ς ($0 \leqslant b_{\min} = \sigma < b_{\max} = \varsigma \leqslant C$). Hence, by Eq. (32), either

$$p(B_i) = 0 \tag{44}$$

or

$$\int_{b_i}^{B_i} f(x)\mathrm{d}x - \int_{B_i}^{b_{i+1}} f(x)\mathrm{d}x = 0. \tag{45}$$

Since

$$\sigma \leqslant b_{i-1} \leqslant B_i \leqslant b_i \leqslant \varsigma \tag{46}$$

and the requested bandwidth is uniformly distributed from σ to ς

$$p(B_i) = \frac{1}{\varsigma - \sigma}. \tag{47}$$

Therefore,

$$B_i - b_i = b_{i+1} - B_i. \tag{48}$$

On the other hand, since $b_i = \frac{B_{i-1}+B_i}{2}$ (by Eq. (43)), we have

$$B_i = b_{\min} + i\left(\frac{b_{\max} - b_{\min}}{n}\right) = \sigma + \frac{\varsigma - \sigma}{n} \tag{49}$$

and

$$b_i = \frac{B_{i-1} + B_i}{2} = b_{\min} + \left(\frac{b_{\max} - b_{\min}}{n}\right)\left(\frac{2i-1}{2}\right) \tag{50}$$

$$= \sigma + \left(\frac{\varsigma - \sigma}{n}\right)\left(\frac{2i-1}{2}\right). \tag{51}$$

Intuitively, the larger the number of classes, the more sensitive our link state update policy is to the fluctuation of the available bandwidth is. As a result, the bandwidth required for updating the link state information is larger. Therefore, given an upper bound on the bandwidth for link state update, there accordingly exists an upper bound on the number of classes. In this paper, we approximately compute the bandwidth used for link state update under the condition that the number of classes is n. Denote $p(B_i)$ as the probability that the quantized value of a sample is B_i; $q(B_i)$ as the probability that the link state information is updated with B_i, $i = 1,2,\ldots,n-1$; $p(B_j|B_i)$ as the transition probability of two consecutive (quantized value of) samples from B_i to B_j, and T_s as the average interval between two consecutive samples. Note that $q(B_i)$ is different from $p(B_i)$ because B_i is used for an update only when the previous update (not the previous sample) B_j is different from B_i. For example, as shown in Fig. 4, assume there are only two classes

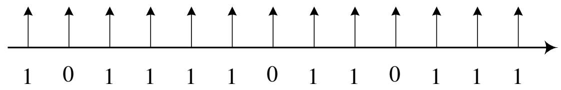

Fig. 4. An example of the link state update with only two classes.

and two quantized values B_0 and B_1, $p(B_0) = 0.2$, $p(B_1) = 0.8$, and $p(B_0|B_0) = 0$ (the transition probability that the current quantized value is B_0 under the condition that the previous one is also B_0). Observe that before and after each B_0, there must be a sample with value B_1 (because $p(B_0|B_0) = 0$). Since an update can occur only when two consecutive samples are different from each other, $q(B_0) = 0.5$, and $q(B_1) = 0.5$. As mentioned before, the sample sequence is a random sequence with memory. Note that a sample is used for update only when it differs from the latest update, i.e., whether a sample is used for update only depends on the previous update. Therefore, for simplicity, we approximately assume that the sample sequence has the Markov memory structure, i.e., $p(B_i|B_{i-1}, B_{i-2}, \ldots, B_0) = p(B_i|B_{i-1})$, where $p(B_i|B_{i-1}, B_{i-2}, \ldots, B_0)$ is the conditional probability that, under the condition that the previous samples are $B_{i-1}, B_{i-2}, \ldots, B_0$, the ith sample is B_i. Therefore, we have

$$\mathbf{p}\begin{pmatrix} p(B_0|B_0) & p(B_1|B_0) & \cdots & p(B_{n-1}|B_0) \\ p(B_0|B_1) & p(B_1|B_1) & \cdots & p(B_{n-1}|B_1) \\ \vdots & & \ddots & \\ p(B_0|B_{n-1}) & p(B_1|B_{n-1}) & \cdots & p(B_{n-1}|B_{n-1}) \end{pmatrix} = \mathbf{p}, \tag{52}$$

where $\mathbf{p} = (p(B_0), p(B_1), \ldots, p(B_{n-1}))$. Accordingly, the sequence of updates exhibits the Markov memory structure. Hence, assuming the transition probability of updates from B_i to B_j is $q(B_j|B_i)$,

$$\mathbf{q}\begin{pmatrix} q(B_0|B_0) & q(B_1|B_0) & \cdots & q(B_{n-1}|B_0) \\ q(B_0|B_1) & q(B_1|B_1) & \cdots & q(B_{n-1}|B_1) \\ \vdots & & \ddots & \\ q(B_0|B_{n-1}) & q(B_1|B_{n-1}) & \cdots & q(B_{n-1}|B_{n-1}) \end{pmatrix} = \mathbf{q}, \tag{53}$$

where $\mathbf{q} = (q(B_0), q(B_1), \ldots, q(B_{n-1}))$. Since a sample is used for update only when it is different from the previous update, $q(B_i|B_i) = 0$ for all $0 \leqslant i < n-1$. Therefore,

$$q(B_j|B_i) = \frac{p(B_j|B_i)}{1 - p(B_i|B_i)}, \quad i \neq j. \tag{54}$$

Hence, we can compute $\mathbf{q}$ by solving Eq. (53).

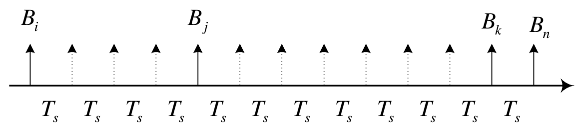

Fig. 5. Solid arrows represent the samples that are used for link state update and dash arrows are not.

As shown in Fig. 5, assume that the average number of samples between any two consecutive updates is m. Accordingly, the average interval between any two consecutive updates is $(m+1)T_s$, and the average number of updates in a unit time, or the update rate, is $\frac{1}{(m+1)T_s}$. Assume that the size of the packets used for link state update is L_u, then the bandwidth used for link state update (a single link) is $\frac{L_u}{(m+1)T_s}$, and the number of link state information packets received by each node of a network in a unit time is $\frac{E}{(m+1)T_s}$, where E is the number of links in the network. Next, we compute m under the assumption that $p(B_j|B_i)$ and $q(B_i)$ $(0 \leqslant i, j < n)$ are known. Note that an update occurs if the current sample is in a different class from the last update. Therefore, if the current update value is B_i, the average number of samples till the next update is

$$\sum_{k=0}^{\infty} kp^k(B_i|B_i)(1-p(B_i|B_i)). \tag{55}$$

Hence, the average number of samples between any two consecutive updates is

$$m = \sum_{i=0}^{n-1} q(B_i)\left(\sum_{k=0}^{\infty} kp^k(B_i|B_i)(1-p(B_i|B_i))\right) \tag{56}$$

and thus the average bandwidth used for link state update can be computed.

5. Simulations

Note that given a blocking probability p of a connection on every single link and without considering the relation among links, its overall blocking probability to traverse through an h-hop path is: $1-(1-p)^h$. Moreover, the number of link state updates of a network is simply the sum of the updates of all the links in the network because each link updates its own link state information independently. Therefore, the performance of a link state update policy in a network can be reflected by its performance on a single link. Hence, instead of conducting simulations in a network, we choose an easier alternative: evaluating the performance of our proposed link state update by comparing it with those in [13] (threshold based, equal class based, and exponential class based update policies) on a single link. For completeness, we briefly review the equal class based and exponential class based update policies.

Definition 3. Equal class based update policy [13] is characterized by a constant B which is used to partition the available bandwidth operating region of a link into multiple equal size classes: $(0,B)$, $(B,2B),\ldots,$ etc. An update is triggered when the available bandwidth on an interface changes to a class that is different from the one at the time of the previous update.

Definition 4. Exponential class based update policy [13] is characterized by two constants B and f $(f > 1)$ which are used to define unequal size classes: $(0,B),(B,(f+1)B),((f+1)B,(f^2+f+1)B),\ldots,$ etc. An update is triggered when a class boundary is crossed.

Note that when the available bandwidth fluctuates around a class boundary, many meaningless updates may be triggered. In order to dampen such oscillatory behavior, the two class based policies are augmented by a hysteresis mechanism in [13]: the generation of an update is suppressed until the available bandwidth reduces sufficiently to cross beyond the middle value of the new classes. Such rule is not applied when the available bandwidth increases and crosses a class boundary.

We adopt two performance indices for the purpose of comparison: the update rate (average number of updates in a unit time) and the false blocking probability of connections, which are respectively defined below

$$\frac{\text{The total number of updates}}{\text{Total simulation time}}, \tag{57}$$

$$\frac{\text{Total number of false blockings of connections}}{\text{Total number of connection requests}}, \tag{58}$$

where the total simulation time is the total number of unit times simulated. Similar to [14], the arrivals of connection requests are generated with a Poisson process with rate $\lambda = 1$ and the duration of each connection is of a standard Pareto distribution with $\alpha = 2.5$ (cumulative distribution of the standard Pareto distribution is $F_x(x) = 1-\left(\frac{\beta}{x}\right)^{\alpha}$, where α is the shape parameter and β is the scale parameter).

Hence, the average duration of a connection is $l = \frac{\alpha\beta}{\alpha-1}$ (the mean of the standard Pareto distribution). By [14], the average traffic load of the link is $\rho = \lambda lb/R$, where b is the average requested bandwidth of a connection. For the purpose of comparison, we do not set the clamp down timer in our simulations. Upon the acceptance and the end of a connection, the available bandwidth is re-computed. The bandwidth requested by each connection is uniformly distributed in $[0, B_{\max}]$. Note that for a single class based link state update policy, the larger number of the classes the bandwidth is partitioned into, the more accurate the link state information is, implying the lower false blocking probability of connections, while the more sensitive it is to the fluctuation of the available bandwidth, thus resulting in a larger update rate. Hence, we can claim that policy 1 outperforms policy 2 if and only if, for any given number of classes to policy 2, an appropriate number of classes can always be found for policy 1 such that it achieves better performance on both the update rate and false blocking probability of connections. By extensive simulations, we found that our proposed link state update policy outperforms the equal and exponential class based link state update policies for any given number of classes. In this paper, due to the page limit, we only selectively present the simulation results of the cases that the numbers of classes of the equal class based update policy are 5 and 10 ($B = 0.1C$ and $0.2C$), and $B = 0.05C$ and $f = 2$ for the exponential class based update policy (the number of classes is 4). We adopt $n = 2$ and 3 for our proposed link state update policy and $\tau = 0.8$ (threshold) for the threshold based update policy in the simulations.

It should be noted that the threshold update policy is different from the class based ones in the sense that the number of classes in the class based update policies determines the accuracy of link state information, while the threshold update policy achieves different accuracy level of link state information by adjusting the threshold value, which is a continuous variable. Naturally, the larger the threshold value is, the coarser the link state information is and the lower the protocol overhead is. As such, in order to claim that a class based update policy to be "strictly" better than the threshold one, we need to make sure that for every single threshold value, we can find a corresponding number of classes such that the class based update policy outperforms its contender in terms of both the update rate and false blocking probability. We have verified that our proposal is capable of achieving better performance than the threshold update policy through

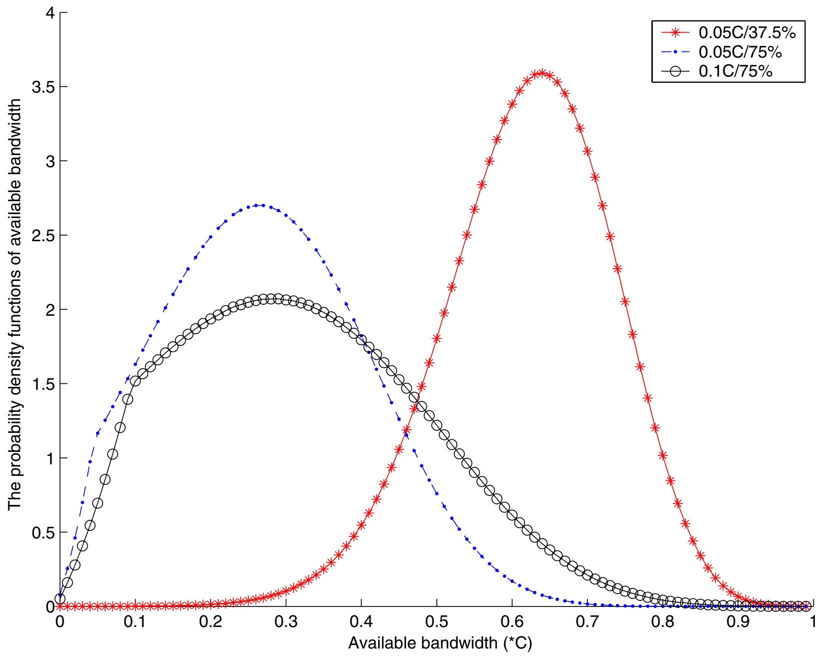

Fig. 6. The distributions of the available bandwidth.

extensive simulations. Owing to the page limit, we only selectively present the performance evaluation of the threshold update policy when $\tau = 0.8$.

As the first step of our proposed link state update policy, we compute the probability density function of the available bandwidth as shown in Fig. 6, where the traffic loads are 37.5% and 75%, and $B_{\max} = 0.05C$ and $0.1C$, respectively (we denote $0.1C/75\%$ as the case that $B_{\max} = 0.1C$ and the traffic load is 75%). Note that the probability that the

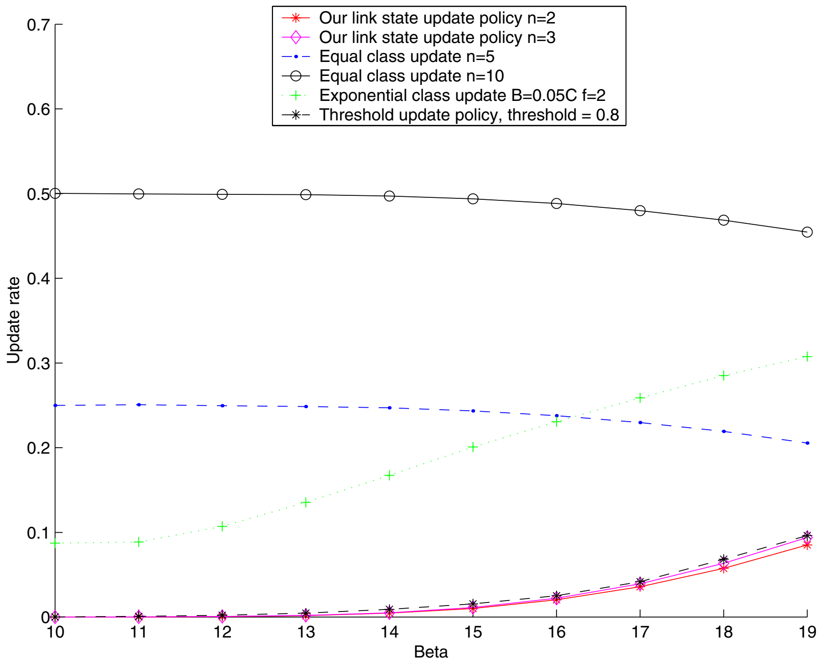

Fig. 7. Update rates of policies when $B_{\max}$ =0.05C.

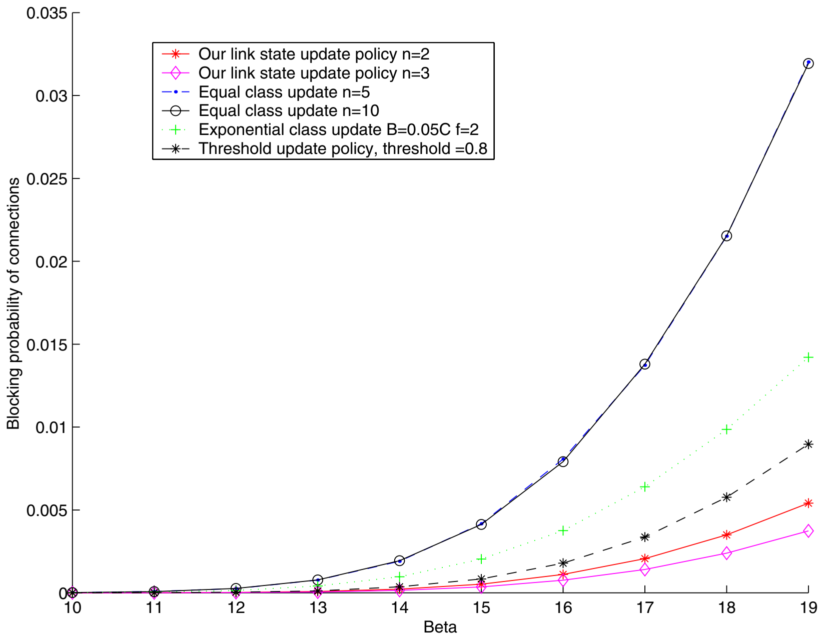

Fig. 8. False blocking probabilities of policies when $B_{\max} = 0.05C$.

available bandwidth is less than B_{max} is very small (the integration of the probability density function of the available bandwidth from 0 to B_{max}, i.e., the probability that the available bandwidth is no larger than B_{max}, is less than 5%). Hence, the update rate of our proposed link state update policy is low.

Figs. 7–10 illustrate our simulation results, in which n denotes the number of classes and *Beta* denotes the parameter β. In both simulations, our

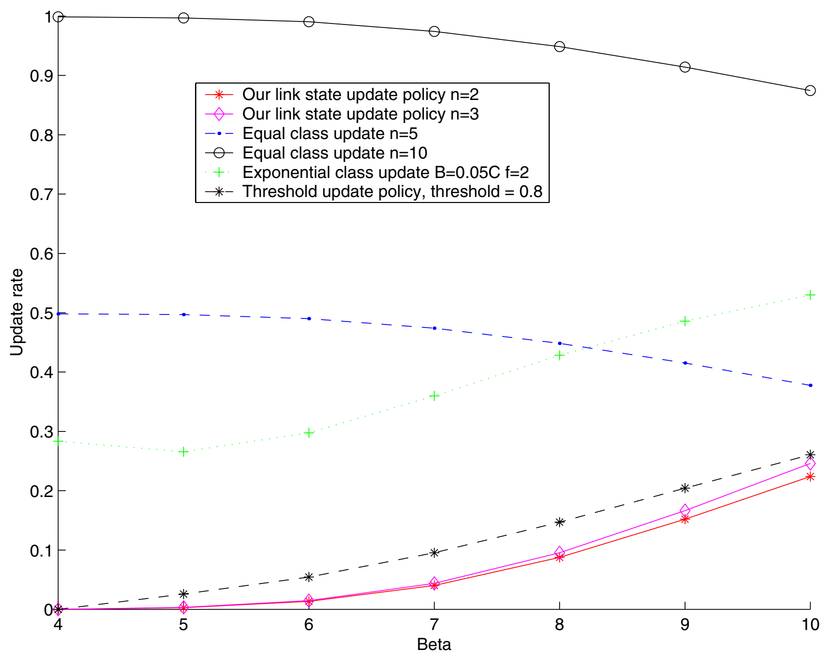

Fig. 9. Update rates of policies when $B_{max} = 0.1C$.

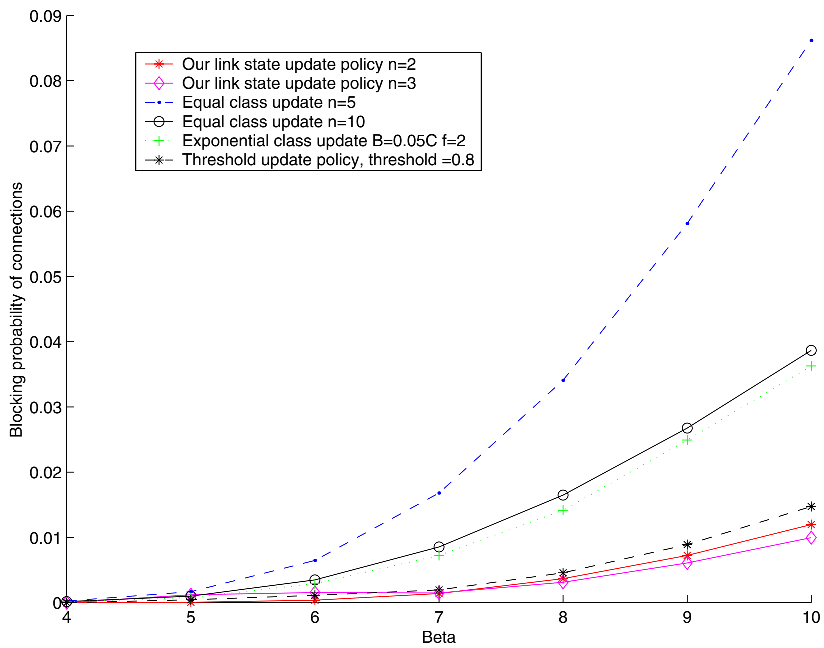

Fig. 10. False blocking probability of connections when $B_{max} = 0.1C$.

proposed link state update policy can achieve much better performance than the three update policies, i.e., our proposed link state update policy achieves lower blocking probabilities with lower update rates than others, implying that our proposed link state update is more practical than the equal and exponential class based link state update policies in terms of the update rate and false blocking probability of connections.

Finally, it should be noted again that although our simulations are conducted on a single link, as long as no cooperation between links is used for updating the link state information, we can equally claim that our proposed link state information update scheme outperforms the three policies from the perspective of a whole network.

6. Conclusions

In this paper, we have proposed an efficient link state update policy. Through theoretical analysis and extensive simulations, we have shown that it greatly outperforms its contenders, i.e., it achieves a much lower false blocking probability with a very low update rate. As a result, we can increase the performance of QoS routing using the proposed link state update policy.

References

[1] S. Chen, K. Nahsted, An overview of quality of service routing for next-generation high-speed network: problems and solutions, IEEE Network 12 (6) (1998) 64–79.
[2] G. Cheng, N. Ansari, On multiple additively constrained path selection, IEE Proceedings on Communications 149 (5) (2002) 237–241.
[3] R. Guerin, A. Orda, QoS based routing in networks with inaccurate information: theory and algorithms, in: Proceedings of the IEEE INFOCOM'97, 1997, pp. 75–83.
[4] T. Korkmaz., M. Krunz, S. Tragoudas, An efficient algorithm for finding a path subject to two additive constraints, in: Proceedings of the ACM SIGMETRICS'2000, 2000, pp. 318–327.
[5] L. Gang, K.G. Ramakrishnan, A prune: an algorithm for finding k shortest paths subject to multiple constraints, in: Proceedings of IEEE INFOCOM'2001, vol. 2, 2001, pp. 743–749.
[6] S. Chen, K. Nahrstedt, Distributed QoS routing with imprecise state information, in: Proceedings of 7th International Conference on Computer Communications and Networks, 1998, pp. 614–621.
[7] Z. Wang, J. Crowcroft, Quality of Service routing for supporting multimedia applications, IEEE Journal on Selected Areas on Communications 14 (7) (1996) 1228–1234.
[8] P.A. Humblet, S.R. Soloway, Topology broadcast algorithms, Computer Networks and ISDN Systems 16 (1988–1989) 179–186.
[9] B. Bellur, R.G. Ogier, A reliable, efficient topology broadcast protocol for dynamic networks, in: Proceedings of IEEE INFOCOM'99, vol. 1, 1999, pp. 178–186.
[10] A. Zinin, M. Shand, Flooding optimizations in link-state routing protocols, IETF, draft-ietf-ospf-isis-flood-opt-01.txt, 2001.
[11] J. Moy, Flooding over a subset topology, IETF, draft-ietf-ospf-subset-flood-00.txt, 2001.
[12] J. Moy, OSPF version 2, RFC2328, IETF, 1998.
[13] G. Apostolopoulos, R. Guerin, S. Kamat, S. Tripathi, Quality-of-service based routing: a performance perspective, in: Proceedings of ACM SIGCOMM 1998, vol. 28, 1998, pp. 17–28.
[14] A. Shaikh, J. Rexford, K.G. Shin, Evaluating the impact of stale link state on quality-of-service routing, IEEE/ACM Transactions on Networking 9 (2) (2001) 162–176.
[15] Q. Ma, P. Steenkiste, Quality-of-service routing for traffic with performance guarantees, in: Proceedings of IFIP International Workshop on Quality of Service 1997, 1997, pp. 115–126.
[16] L. Breslau, D. Estrin, L. Zhang, A simulation study of adaptive source routing in integrated services networks, Computer Science Department, University of Southern California, Technical Report 93-551, 1993.
[17] M. Peyravian, R. Onvural, Algorithm for efficient generation of link-state updates in ATM networks, Computer Networks and ISDN System 29 (1997) 237–247.
[18] X. Li, L.K. Shan, W. Jun, N. Klara, QoS extension to BGP, in: Proceedings of IEEE ICNP'02, 2002, pp. 100–109.
[19] S. Floyd, V. Paxson, Difficulties in simulating the internet, IEEE/ACM Transactions on Networking 9 (2001) 392–403.
[20] W.E. Leland, W. Willinger, M.S. Taqqu, D.V. Willson, On the self-similar nature of Ethernet traffic, IEEE/ACM Transactions on Networking 2 (1994) 1–15.
[21] B.A. Mah, An empirical model of HTTP network traffic, in: Proceedings of the IEEE INFOCOM'97, vol. 2, 1997, pp. 592–600.
[22] M.E. Crovella, A. Bestavros, Self-similarity in World Wide Web traffic: evidence and possible causes, in: Proceedings of ACM SIGMETRICS'96, 1996, pp. 160–169.

Gang Cheng received both B.E. and M.E. degrees in Information Engineering from the Beijing University of Posts and Telecommunications, Beijing, PR China, in 1997 and 2000, respectively, and Ph.D. in Electrical Engineering from the New Jersey Institute of Technology in 2005. His research focuses on QoS guarantee issues in high speed networks, Internet routing protocols and service architectures, and modeling and performance evaluation of computer and communication systems, especially on QoS routing, traffic scheduling, queue management, and congestion control in high speed networks. He is currently employed with VPISystems Inc.

Nirwan Ansari received B.S.E.E. (summa cum laude), M.S.E.E., and Ph.D. degrees from NJIT, University of Michigan, and Purdue University in 1982, 1983, and 1988, respectively.

Since 1997, he has been working as a full-time professor in the Department of Electrical and Computer Engineering at NJIT. He authored with E.S.H. Hou Computational Intelligence for Optimization (Kluwer, 1997, translated into Chinese in 2000), and edited with B. Yuhas Neural Networks in Telecommunications (Kluwer, 1994). He is a technical editor of the IEEE Communications Magazine, Computer Communications, the ETRI Journal, as well as the Journal of Computing and Information Technology. His current research focuses on various aspects of broadband networks and multimedia communications. His research has been supported by various federal and state agencies, and private industries. He has also contributed over 80 refereed journal articles, and numerous conference papers and book chapters.

He organized (as General Chair) the First IEEE International Conference on Information Technology: Research and Education (ITRE2003), and was instrumental, while serving as its Chapter Chair, in rejuvenating the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award and a 2003 Chapter Achievement Award, served as Chair of the IEEE North Jersey Section and in the IEEE Region 1 Board of Governors during 2001–2002, and currently serves in various IEEE committees including as TPC Co-chair/Vice-chair of several conferences. He was the 1998 recipient of the NJIT Excellence Teaching Award in Graduate Instruction, and a 1999 IEEE Region 1 Award. He is frequently invited to deliver keynote addresses, tutorials, and talks.