Cost-based QoS Routing

Jian Chu Chin-Tau Lea Albert Wong
chujian@ee.ust.hk eelea@ee.ust.hk eealbert@ee.ust.hk
EEE Dept, HKUST, Clear Water Bay, Hong Kong

***Abstract*— This paper presents cost-based QoS routing and compares its performance with the commonly used widest-shortest-path routing. The advantages of cost-based routing are demonstrated in two major areas. First, it is stable under heavy-loaded conditions. Second, it requires a lower state-update overhead and is more robust to the inaccuracy of state information. The concept presented in the paper can be applied to any reservation-based QoS approach – like IntServ or MPLS type networks.**

I. Introduction

Constraint-based routing in IntServ or MPLS type networks have been extensively studied [1]-[10]. In addition to achieving high throughput and meeting the QoS constraints (like bandwidth requirement), there are still some open issues to be considered. One is admission control. When multiple paths are selectable for a source-destination pair, some are direct and are shorter, and the others are considered alternate paths and are longer. Under a normal load, the direct paths are used more often. However, when the traffic load exceeds a certain limit, the network will suddenly go through a state change likened to the phase transition in material and the alternate paths are used more often than the direct path(s). As a result, the network throughput will drop even as the load increases [11], [12]. Admission control through trunk reservation should be deployed to protect the network from entering the unstable state [1]. However, how much bandwidth to reserve and how to do it effectively is still an open issue.

Another open issue is the frequency of link state update. Efficiency of a routing scheme depends, to a large extent, on the accuracy of link state information [1], [2], [9]. The trade-off is between the volume of updates and the accuracy of state information. A sensitive update policy provides more accurate state information with a big communication overhead, while a coarse update policy controls the volume of updates at the expense of introducing more inaccuracy [4], [6]. How to reduce the state update overhead without significantly degrading performance is still an open research issue.

This paper proposes cost-based routing the main characteristic of which is that a flow may sometimes be rejected even when a feasible path is available. We study two schemes, the least-cost-path (LCP) and the constrained-cost minimum-hop path (CCMHP). We compare their performance with other routing schemes in terms of throughput, stability under heavy loads, and state update overheads. With small modifications to the existing routing protocols or their extensions, the proposed schemes can be applied to traffic engineering in MPLS networks [13]. The rest of the paper is organized as follows. Section II describes two cost-based routing schemes and their implementation complexity. Section III compares their performances with other schemes based on several criteria: throughput (under even and uneven traffic conditions), blocking probability, and the amount of updating activities. Finally, Section IV concludes and discusses the findings in this paper.

II. Cost-based QoS Routing

The cost-based routing proposed in this paper differs from others in two aspects. First, the state information of a link is a "cost" function, not an available bandwidth as in other routing schemes. This leads to a low state update overhead. Second, a flow may be rejected even if a path with sufficient resources is available. This makes the routing scheme stable even under a heavy-loaded condition. Trunk reservation and the associated issue of deciding how much bandwidth to reserve are not required.

The decision of accepting a flow is based on the concept of "reward" and "cost" of adding a flow. If the objective function is throughput, then the reward of adding a flow is the throughput increase of the network. However, adding a flow will increase the blocking probabilities of future flows. The throughput loss due to the added blocking probabilities of future flows is the price (cost) we pay for accepting the new flow. To maximize the performance of the network, we should reject a flow if its cost exceeds its reward. This is the gist of cost-based routing.

The center issue in cost-based routing is computing the cost function. Although computation intensive, it is done off-line. Once the cost function is set, implementing the routing scheme is rather straightforward.

A. Cost Computation

We assume flows generated from outside follow a Poisson stream. Although links are not entirely independent, it is customary to assume that the independence holds to make the analysis tractable. With this assumption, we can focus on one link in cost computation. Assume that there are K types of traffic flows. Flow arrivals are Poisson and their service times follow a negative exponential distribution. (These assumptions are used for analytical purposes only.) We define the following parameters.

- λ_k : The arrival rate of type k traffic.
- b_k : The bandwidth requested of type k traffic.
- $(1/\mu_k)$: The average connection time of type k traffic.
- $X = [n_1, n_2, , n_K]$: The state variable, where n_k represents the number of type k flows existing on the link.

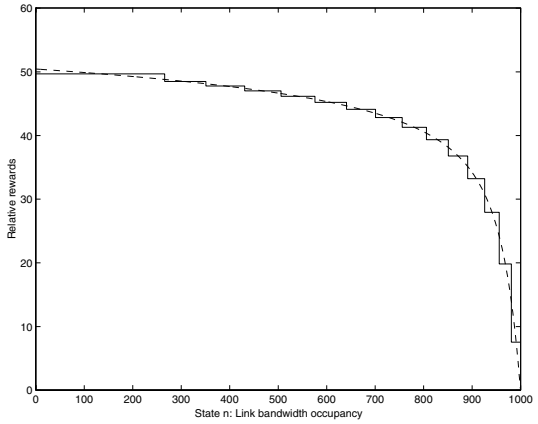

Fig. 1. The cost for $C = 1000$, $b_1 = 1$, $b_2 = 4$. $\lambda_1 = \lambda_2 = 180$, $\mu_1 = \mu_2 = 1$. Normalized offered load = 0.9. The solid-line represents the quantization of the cost into a small number of levels.

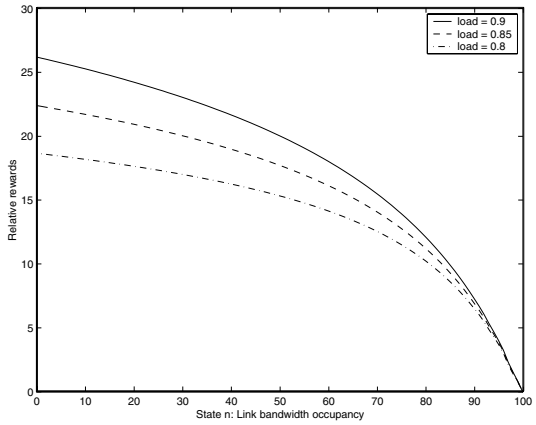

Fig. 2. The relative rewards for different offered loads with the same proportion of requests of each type of flow. $C = 100$, $b_1 = 1$, $b_2 = 4$.

- $r_{XX_k^+} = (b_k w_k)/\mu_k$: The reward (throughput gain) by adding a type k flow in state X, where $X_k^+ = [n_1, n_2, , n_k + 1, , n_K]$, w_k is the weighting factor and is set to 1 in this paper.
- v_X : The *Relative reward* function for a given state X. It is used to compute the cost function D_X^k.
- D_X^k : The cost for adding a type k flow and $D_X^k = v_X - v_{X_k^+}$. It is the throughput loss due to additional blocking caused by adding a type k flow in state X.

Once v_X is known, the cost function D_X^k can be derived and v_X can be computed with the MDP (Markov decision process) theory [14], [15]. The number of equations to be solved for v_X is the same as the number of states of the Markov process (see Appendix). Assume the capacity of a link is C units/sec. Then the state space consists of all vectors X such that $\sum_{k=1}^{K} b_k n_k \leq C$. For a high-speed, multi-rate network, the computation complexity for finding v_X is prohibitively high.

We can reduce the computation complexity significantly by using two approximations (see Appendix). The first step is to reduce the multi-rate problem into a single-rate problem [16] by grouping all states of the same occupied capacity $\sum_k b_k n_k$ into one state. The resulting v_u will be a function of the link utilization u only, where $u = \sum_k b_k n_k$. The corresponding relative reward function is shown in Fig. 1. Routing a type k flow on this link will change the link state from u to $u + b_k$ and its cost is simplified to $D_u^k = v_u - v_{u+b_k}$. The second step is to quantize v_u into a small number of levels. The difference between v_u and v_{u+1} is very small over a wide region until u is close to full utilization. We can set an arbitrary number of quantization levels and use the formula in [17] to derive the approximated v_u. Through this technique, we will reduce the number of states in the one-dimensional problem. More details are given in the Appendix.

B. Cost for Multi-hop Flow

Given the cost function of a link, we can compute the cost for adding a flow traversing multiple hops. Suppose a two-hop flow is blocked. The throughput loss due to this connection is counted on both links. However, as far as the network is concerned, only one connection is lost. Thus the total cost of adding a type k flow traversing j hops should be adjusted to $(\sum_j D_{u(j)}^k)/\bar{L}$, where $D_{u(j)}^k$ is the cost on the j_{th} hop, and $\bar{L}$ is the average number of hops for all flows (or connections). Since most connections use the direct paths (minimum-hops) most of the time, we compute $\bar{L}$ as the average number of minimum hops for all source destination pairs. This can be derived from the topology of the network.

C. Robustness in Cost Estimation

Another parameter that affects the accuracy of the cost function is the offered load of each link. Even the offered loads from outside for each (source, destination) pair are known, the load on each link cannot be easily derived as it is affected by both the routing policy and the network topology. It is therefore highly desirable to do away with the need for prior load information. Although the link offered load can be obtained through the on-line measurement techniques [18], we use a much simpler approach to this problem. Consider the v_u shown in Fig. 2. Near the range of full occupancy, we find that cost functions differ little for different link loads. It is this range (near full capacity) that will most affect the decision of accepting a flow or not.

Thus, given the prior information of the proportion of each class of requests, instead of using on-line measurements to obtain more accurate link load estimations, we just assume the link offered load = 0.9 and compute the cost function for all links. We will show that this approach offers the same good performance as that using the link load estimations derived from on-line measurements.

D. Two Cost-based Routing Algorithms

In cost-based routing, the link state information is the current relative reward value of the link. As shown in Section II-A, from relative rewards, we can derive the cost for adding a flow. We develop two cost-based routing schemes, namely the Least-Cost-Path (LCP), and the Constrained-Cost Minimum-Hop Path (CCMHP).

- LCP (*Least-Cost-Path*) algorithm: This is the standard shortest path algorithm except that a flow will be rejected if the cost (throughput loss) for routing the flow along the least-cost path exceeds its throughput reward.
- CCMHP (*Constrained-Cost Minimum-Hop Path*) algorithm: Instead of selecting the least-cost path, CCMHP selects the minimum-hop path among all paths that satisfy the "reward $>$ cost" constraint. If there are several paths with the same minimum hop count, "least cost" is used to break the tie. The flow is rejected if we can not find a path to meet the constraint. This algorithm can be derived from the Bellman-Ford algorithm [19].

We compare the two with the conventional WSP (*Widest-Shortest-Path*) algorithm that selects the path with the minimum number of hops. If there are several such paths, the one with the maximum available bandwidth is selected [2], [7].

III. Performance Evaluation

In our study, we assume that the source node is responsible for selecting a path on demand of a request. Signaling is then issued to set up forwarding state hop-by-hop over the path, and the destination node initiates bandwidth reservation backward on each link in the path. The states of adjacent links of a node are updated immediately. The states of other links are updated when *link state advertisements* (LSA) are received. The flow will be rejected if either a link does not have enough bandwidth or the cost of adding the flow exceeds its throughput gain (reward). The major characteristics and parameters of our simulations are given below.

1) The simulations are performed on randomly generated network topologies using the GT-ITM software [20]. All topologies are generated using the "flat random graph" model and each topology has 20 nodes, with an average connectivity degree in the range [5.5, 6]. Each link has the same capacity of 100 Mbits/sec.
2) Flows generated at each node follow a Poisson stream with the arrival rate $\lambda = 1$. The flow holding time is exponentially distributed and can be adjusted to represent different offered loads. The requested bandwidth is randomly selected from the set of values $\{1, 2, 3, 4, 5\}$ Mbits/sec.
3) The simulation includes *balanced* and *unbalanced* loads. For balanced load, flows are evenly distributed among all source destination pairs. For unbalanced load, several node pairs are randomly selected to represent the hot spot nodes. The traffic between these hot spot nodes is much larger than the traffic between other node pairs, e.g., the traffic between these hot spot nodes may account for 20% of the total traffic to the network. During the simulation, the hot spot nodes are periodically reselected to reduce the dependencies on the network topology.

A. Comparative Study with Accurate State Information

As indicated previously, when traffic load exceeds a certain limit, the network will be unstable if it uses conventional routing algorithms, like WSP. Trunk reservation is required for WSP to stabilize the network under a heavy-loaded condition. However, deciding how much bandwidth to reserve for direct paths is not easy and can only be done empirically. From our own setting, we found that reserving 5% of the total capacity leads to the best performance. For cost-based CCMHP and LCP, there is no need for trunk reservation. The instability issue is prevented automatically as the cost of an alternate path can easily exceed its reward under heavy-loaded conditions. This advantage is clearly demonstrated in Fig. 3.

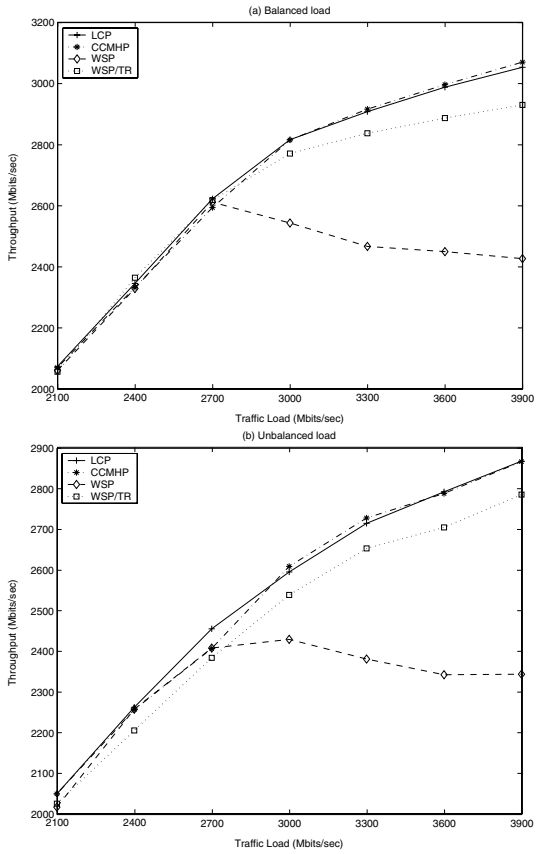

Fig. 3. Network throughput for LCP, CCMHP, WSP and WSP with trunk reservation (WSP/TR) schemes: (a) balanced load; (b) unbalanced load.

We also study the impact of using approximate cost functions. As pointed out previously, we use the cost function of offered load = 0.9 for all links. We want to find out if this simplification causes any performance degradation in cost-based routing. Fig. 4 compares the throughput difference between the two cost functions: One is based on the approximation cost function of offered load = 0.9, and the other is based on the cost function of using link load derived from the online measurement technique [18]. As shown in the figure, the difference is not significant. The difference is not even noticeable under the balanced load condition when load is light.

B. Impact of Inaccurate State Information

Link state information cannot be updated instantaneously and routing is often based on inaccurate state information.

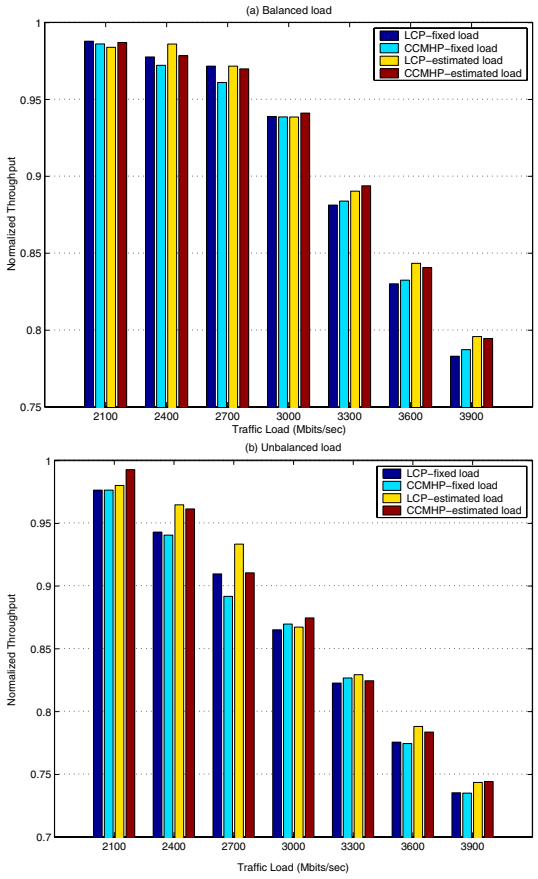

Fig. 4. Normalized throughput with respect to the offered load for fixed load and estimated load in cost-based routing: (a) balanced load; (b) unbalanced load.

Below we compare the three routing schemes when frequent link state updates are not possible. To highlight the effect of inaccurate state information, we select a moderate traffic load. The reason is that under such a load the three routing schemes have similar flow rejecting rates (blocking probabilities) if accurate state information is available. We then see how their performances change as we reduce the frequency of link state updates. The state update policy can be threshold based, class based, or timer based [1]. In the first experiment, we compare the performance with a periodic link state update policy. This results in a predictable and the same level of update overhead for all the routing schemes. In the second experiment, we perform a threshold trigger in which a new update is triggered if the relative change between the current and the previously advertised link state has exceeded a predetermined value. For the lack of space, here we only report the results for the balanced load condition. The results of the unbalanced load condition just show the same trends.

We compare the three in terms of (1) bandwidth blocking probability and (2) the link-state update overhead. The latter is measured by the total number of update messages. The bandwidth blocking probability, denoted by P_B, is defined as the percentage of the total requested bandwidth that is blocked, i.e., $P_B = \sum rejected\ bandwidth / \sum request\ bandwidth$.

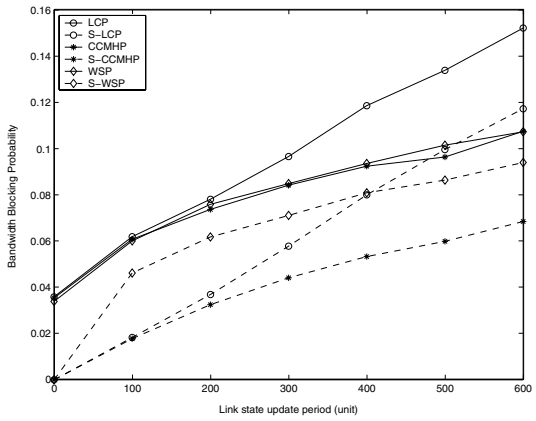

Fig. 5. Bandwidth blocking probability and set-up failure probability (S-) for LCP, CCMHP, and WSP schemes with periodic updates.

Conventionally, P_B can be further divided into two parts [6]: the routing failure probability P_r, and the set-up failure probability P_s. *Routing failures* refer to the flow rejection caused by the source node failing to find a feasible path for the flow. Due to inaccurate state information, routing failures can happen when in fact a feasible path is still available. *Set-up failures* are the opposite: the source node selects a path when in fact the path is not available due to inaccurate state information. A set-up failure is only known after the set-up procedure is started and signaling messages are sent out. It is thus more costly than a routing failure that involves no exchange of set-up messages between nodes.

Fig. 5 shows the bandwidth blocking probability and the corresponding set-up failure probability as a function of the state update period. LCP is affected the most by the inaccurate link state information. CCMHP and WSP give priority to shorter paths and are more robust to the lack of accurate state information. However, although the overall bandwidth blocking probabilities of CCMHP and WSP are about the same, CCMHP has lower set-up failure probabilities. As mentioned previously, set-up failures are more costly than routing failures. In this sense, CCMHP clearly outperforms the other schemes.

Fig. 6 plots the bandwidth blocking probability and set-up failure probability for a range of triggers. A small hold-down timer is also used in this experiment to enforce a minimum spacing between two consecutive updates. Unlike the experiments with periodic link state updates, the overall bandwidth blocking probability does not change a lot as the triggering value increases. Again, we observe that both LCP and CCMHP have lower set-up failure probability compared to WSP. This is because set-up failure is likely to occur when links are close to full utilization. However, since the cost of these links is high, the source will be unlikely to route flows along these links. Although this may increase the routing failure probability, it saves network resources and results in less message exchange overhead.

In cost-based routing, the link state information exchanged is the value of the relative reward function of the link. From the relative rewards, we can derive the cost function for adding a

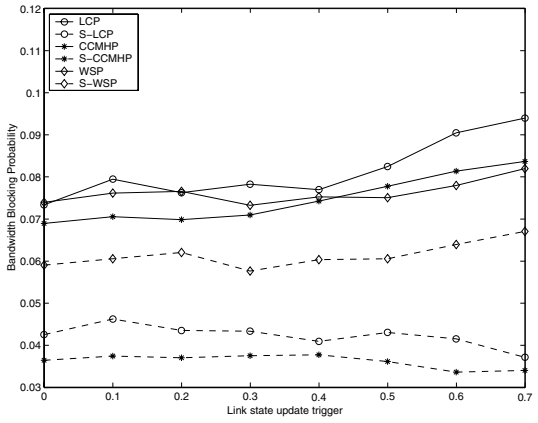

Fig. 6. Bandwidth blocking probability and set-up failure probability (S-) for LCP, CCMHP, and WSP schemes with triggered updates.

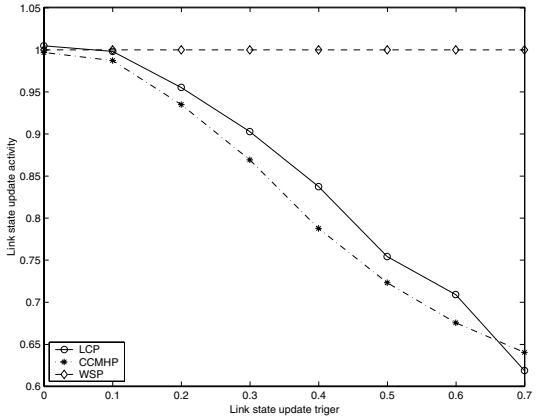

Fig. 7. Link state update overhead for LCP, CCMHP, and WSP with triggered updates.

flow. The function is flat for a wide capacity range. Thus only a significant change of link bandwidth utilization will trigger a state update. Fig. 7 shows the link state update overhead of the three routing schemes, normalized with respect to the number of update messages triggered by the WSP scheme for a range of triggers. Both LCP and CCMHP reduce the link state update overhead significantly, especially for larger trigger values. Additional experiments (not shown) illustrate that if a large hold-down timer is used, the updating overhead is dominated by the hold-down timer value. Thus the overhead difference between these three schemes is small. Even in that case, cost-based routing schemes still have an edge in terms of update overhead over the WSP scheme.

IV. Conclusions

In the paper, we proposed two cost-based QoS routing schemes and compared them with the commonly used WSP routing scheme under balanced and unbalanced traffic loads. We demonstrated the advantages of cost-based routing in two major areas. First, it is stable under heavy-loaded conditions. Second, it requires a lower state-update overhead. In addition, CCMHP scheme that gives priority to shorter paths is more robust to the inaccuracy of state information. The concept can be applied to any connection-oriented network – like IntServ, or MPLS type networks.

References

[1] G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi, "Quality of service based routing: A performance perspective," *ACM SIGCOMM'98 Vancouver, B.C.*, 1998.

[2] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick, "A framework for QoS-based routing in the internet," *Internet RFC 2386*, Aug. 1998.

[3] Zheng Wang and Jon Crowcroft, "Quality of service routing for supporting multimedia applications," *IEEE JSAC*, pp. 1288–1234, Sep. 1996.

[4] G. Apostolopoulos, R. Guerin, S. Kamat, A. Orda, and S. K. Tripathi, "Intradomain QoS routing in IP networks: A feasibility and Cost/Benefit analysis," *IEEE Networks*, pp. 42–54, Sep./Oct. 1999.

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

[6] A. Shaikh, J. Rexford, and K. G. Shin, "Evaluating the impact of stale link state on quality-of-service routing," *IEEE/ACM Transactions on Networking*, vol. 9, April 2001.

[7] Q. Ma and P. Steenkiste, "Routing traffic with quality-of-service guarantees in integrated services networks," *Proceedings of NOSSDAV'98*, July 1998.

[8] A. Orda, "Routing with end-to-end QoS guarantees in broadband networks," *INFOCOM'98*, March 1998.

[9] R. Guerin and A. Orda, "QoS routing in networks with inaccurate information: Theory and algorithms," *IEEE/ACM Transactions on Networking*, vol. 7, June 1999.

[10] S. Nelakuditi, Z.-L. Zhang, and R. P. Tsang, "Adaptive proportional routing: A localized QoS routing approach," *INFOCOM'2000*, March 2000.

[11] R. J. Gibbens, F. P. Kelly, and P. B. Key, "Dynamic alternate routing - modeling and behaviour," *ITC-12*, 1989.

[12] F. P. Kelly, "Routing and capacity allocation in network with trunk reservation," *Mathematics of Operations Research*, pp. 771–793, 1990.

[13] Awduche, D. J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for traffic engineering over MPLS," *RFC 2702*, Sep. 1999.

[14] R. Howard, *Dynamic Programming and Markov Processes*, MIT Press, 1960.

[15] K. R. Krishnan and T. J. Ott, "State-dependent routing of telephone traffic: Theory and results," *Proceeding of the 25th IEEE Control and Decision Conference*, Dec. 1986.

[16] K. R. Krishnan and F. Huebner-Szabo de Bucs, "Admission control and state-dependent routing for multirate circuit-switched traffic," *ITC-15*, 1997.

[17] Chin-Tau Lea and Kai-Wei Ke, "Quantization and cost computation of MDP-based admission and routing," *INFOCOM'96*, pp. 1004–1011, 1996.

[18] Lisa Zhang, Matthew Andrews, William Aiello, Sandeep Bhatt, and K. R. Krishnan, "A performance comparison of competitive on-line routing and state-dependent routing," *GLOBECOM '97*, vol. 3, pp. 1813–1819, 1997.

[19] D. Cavendish and M. Gerla, "Internet QoS routing using the bellman-ford algorithm," *IFIP Conference on High Performance Networking - HPN98*, 1998.

[20] GT-ITM, *Georgia Teach-Internetwork Topology Models*, URL: http://www.cc.gatech.edu/projects/gtitm.

Appendix

Given that the system is initially in state i, we denote $V_i(t)$ the expected total future reward at time t. For an ergodic system, when t is large enough, $V_i(t)$ has been shown to be [14]

$$V_i(t) = gt + v_i \tag{1}$$

where

v_i = The relative reward (or the relative value) of state i.
$g = \lim_{t\to\infty}(V_i(t)/t)$
= The long term average reward rate of the system.

The g and v_i can be solved from the following value determination equations (VDE) [14]

$$\begin{aligned} g &= q_i + \textstyle\sum_{j=1}^{N} \alpha_{ij} v_j, \quad 1 \le i \le N \\ q_i &= r_{ii} + \textstyle\sum_{j\neq i} \alpha_{ij} r_{ij} \end{aligned} \tag{2}$$

where

N = The total number of states,
q_i = The average earning rate of the system in state i,
r_{ij} = Reward of making the transition from i to j.
r_{ii} = The rate of reward when system is in state i, and
α_{ij} = The transition rate from state i to j.

When the system is in the steady-state, $g = \sum_i \pi_i q_i = \sum_i \pi_i (r_{ii} + \sum_{j\neq i} \alpha_{ij} r_{ij})$, where π_i denotes the steady-state probability of state i. The best policy is the one that maximizes g. Based on N equation (2), we can calculate the relative rewards, v_j. The cost of adding a type k flow can thus be computed as $D_X^k = v_X - v_{X_k^+}$.

The number of states in equation (2) determines the number of equations that should be solved simultaneously. To make the cost-based routing a viable approach, we can use the following techniques to reduce the computation complexity.

1) Dimension Reduction: The first step is to reduce the number of dimensions (rates) in the system. We use the idea in [16] by aggregating all the states with the same bandwidth occupancy $\sum_i b_i n_i$ to one state. Then the original multi-dimensional Markov chains are reduced to be a one-dimensional chain and can be solved by $(C+1)$ linear equations, where $(C+1)$ represents the number of states of the aggregated Markov process. For the new process created by the state aggregation, we can write down the transition rates for the aggregated states as:

$$\begin{aligned} &(a)\ \alpha_{i,i+b_k} = \lambda_k, && \text{for } i \le C - b_k, 1 \le k \le K \\ &(b)\ \alpha_{i,i-b_k} = \mu_k E(m_k|i), && \text{for } i \ge b_k, 1 \le k \le K \\ &(c)\ \alpha_{ij} = 0, && \text{for all other } j (j \neq i) \\ &(d)\ \alpha_{ii} = -\textstyle\sum_{j\neq i} a_{ij} \end{aligned} \tag{3}$$

where $E(m_k|i)$ denotes the conditional expected number of class k flows in progress, given that i is the total occupancy of the link. It can be determined by the following recursion formula:

$$\begin{aligned} \lambda_k \pi(i - b_k) = \mu_k E(m_k|i)\pi(i), \\ i = b_k, b_k + 1, \ldots, C, k = 1, \ldots, K \end{aligned} \tag{4}$$

where $\pi(i)$ represents the probability of the aggregated link occupancy i.

2) Cost Quantization: The next step is to quantize the relative reward v_j into a small number of levels. If we quantize the cost function into m levels, we only need to solve m linear equations to compute the relative rewards, which is a considerable reduction in terms of the complexity of cost computation. We use a simple quantization scheme in which the m quantization states are uniformly selected among the last $q(m-1)$ states, where q is the distance between two quantized states. The remaining $[N - q(m-1)]$ states are grouped into one. This scheme is called the Backward Uniform Quantization (BUQ) in [17].

Once the states are grouped, we have constructed a new Markov model to approximate the original one. Let us suppose the original state space is $N = \{1, 2, ..., n\}$ and there are k states with the same relative reward. Without loss of generality, we assume they are states 1 to k and let $K = \{1, 2, ..., k\}$. Now we can group the states 1 to k into a single state called 0, then we construct an new Markov chain with the state space $N' = \{0, k+1, ...n\}$. The transition rates for this new Markov chain should be

$$\begin{aligned} \alpha_{0j} &= \frac{\sum_{m=1}^{k} \pi_m \alpha_{mj}}{\sum_{m=1}^{k} \pi_m}, \quad \text{for } j = k+1, ..., n, \text{ and} \\ \alpha_{j0} &= \sum_{m=1}^{k} \alpha_{jm}. \end{aligned} \tag{5}$$

The transition rewards r_{0j} and r_{j0} are also modified accordingly as

$$\begin{aligned} r_{0j} &= \frac{\sum_{m=1}^{k} \pi_m \alpha_{mj} r_{mj}}{\sum_{m=1}^{k} \pi_m \alpha_{mj}}, \quad \text{for } j = k+1, ..., n, \text{ and} \\ r_{j0} &= \frac{\sum_{m=1}^{k} \alpha_{jm} r_{jm}}{\sum_{m=1}^{k} \alpha_{jm}}. \end{aligned} \tag{6}$$

Finally, the reward rate of state 0 is also changed to

$$r_{00} = \frac{\sum_{i=1}^{k} \pi_i \left[r_{ii} + \sum_{\substack{j=1 \\ j\neq i}}^{k} \alpha_{ij} r_{ij} \right]}{\sum_{i=1}^{k} \pi_i}. \tag{7}$$