

PFR: A Distributed Preemption Strategy for Improved QoS in Multi-Class Networks

Fahad Rafique Dogar, Zartash Afzal Uzmi, and Shahab Munir Baqai

Department of Computer Science and Engineering
Lahore University of Management Sciences, Pakistan
Email: {fahad,zartash,baqai}@lums.edu.pk

Abstract—In this paper, we propose a distributed preemption strategy, *Preemption for Future Requests (PFR)*, that results in improved QoS for future traffic requests. We use the notion of *critical links* to select those connections for preemption that are traversing congested links. Thus, we define critical links as those links that are approaching their capacity and are likely to block future requests. Releasing bandwidth on such links, while making the preemption decision, improves the chances of accepting future requests. We formulate this strategy as an Integer Linear Programming (ILP) problem, which can also incorporate other preemption objectives, such as minimizing the preempted bandwidth and minimizing the number of preempted connections etc. Our formulation allows the network service provider to assign appropriate weights to each preemption objective, based on the network management requirements.

We also propose a simple heuristic algorithm that illustrates the use of PFR with currently used preemption objectives. Our heuristic algorithm has a complexity of $O(k \log k)$, where k is the number of preemptable connections. The low complexity makes it suitable for use in large networks having thousands of connections. We compare the performance of PFR heuristic algorithm with an existing preemption scheme. The simulation results show that the use of PFR heuristic algorithm results in approximately 20% less rejected requests compared to the preemption scheme that does not take into account future requests. Moreover, it also decreases the overall preemptions in the network by approximately 5%.

I. INTRODUCTION

Emerging multimedia applications have created new requirements of Quality of Service (QoS) provisioning over the Internet. The stringent delay and bandwidth requirements of applications such as VoIP and video streaming are different from those of data traffic. Therefore, Internet is evolving from the traditional best effort service to a multi-class QoS aware environment which could possibly satisfy the challenges posed by these applications. One major challenge is to satisfy the QoS

requirements of various traffic types while judiciously managing the scarce network resources, such as bandwidth. This requires efficient bandwidth reservation and management mechanisms that account for the bandwidth requirements of every request, keeping in view available network resources.

Bandwidth reservation and management is easier in a centralized scenario which has up-to-date information about network state. However, considering the distributed nature of current Internet protocols, a centralized management mechanism would require significant changes to routing and resource reservation mechanisms [1]. In a distributed scenario, propagating complete network state information creates scalability concerns. Therefore, the challenge is to efficiently manage the network with little or no extensions to current Internet protocols.

This problem of bandwidth reservation and management in a distributed multi-class environment becomes even more challenging if requests come in an *online* manner. In online routing, no information about future requests is available. This can lead to sub-optimal routing decisions where a decision to admit a request may result in insufficient available resources for future high priority requests [2]. It is also undesirable to reject requests, in anticipation of future higher priority requests. *Preemption* allows the flexibility to accept a request and to later retract it, in case a future higher priority request needs resources. This allows efficient use of network resources since bandwidth can always be made available through preempting less important connections. Preemption can also be used in case of a network failure to provide quick restoration to high priority traffic by preempting less important connections.

Preemption has therefore become an important component of multi-class QoS provisioning proposals such as DiffServ based Traffic Engineering (DS-TE) [3] [4]. DS-TE combines the benefits of DiffServ [5] and Multi-Protocol Label Switching (MPLS) [6], thereby allow-

ing differentiation of traffic into multiple classes and providing bandwidth guaranteed paths based on the requirements of each traffic class. The provisioning of bandwidth guaranteed paths in a multi-class environment creates a need to use preemption for efficient network management. While DS-TE allows preemption strategies to be incorporated in resource reservation protocols such as RSVP-TE, it leaves open the choice of *preemption algorithm* that should be used [4]. The preemption algorithm determines the connection(s) that should be preempted in order to free up enough resources to accept the new request.

Several preemption algorithms have been proposed that try to optimize different criteria while selecting the connections to be preempted [7] [2] [8] [9] [10]. Common objectives of these preemption algorithms include: minimizing the priority of preempted connections, minimizing the preempted bandwidth, and minimizing the number of preempted connections. These preemption criteria reflect the requirements of networks service providers, giving them the flexibility to select a preemption algorithm that satisfies the network management objectives. This decision depends on several factors such as the cost of rerouting a preempted connection or loss in revenue associated with preempting bandwidth. Therefore, in addition to the priority of preempted connections, other factors such as the number of preempted connections and the preempted bandwidth also become important. Another important consideration from a networks service provider's perspective is the impact of preemption on future requests. Most routing architectures for traffic engineering try to maximize the number of accepted requests and the bandwidth associated with these requests [11] [13]. However, this consideration has been restricted to routing of connections and has not been considered in earlier preemption schemes.

In this paper, we propose a preemption strategy, *Preemption for Future Requests (PFR)*, that results in higher QoS for future traffic requests. We use the notion of 'critical links' to select those connections for preemption that are traversing congested links. Similar approach has been adopted in MIRA [11] and shortest widest path routing [12] to route the requests such that they avoid 'critical links'. We use this idea in preemption to select those paths for preemption that are likely to block future requests. This strategy ensures that more bandwidth is available on congested links for future requests, thereby decreasing their blocking probability. We formulate this strategy as an Integer Linear Programming (ILP) problem which can incorporate other

preemption objectives such as minimizing the preempted bandwidth and minimizing the number of preempted connections etc. Our formulation allows flexibility in assigning weights to each preemption objective, based on the network management requirements.

We also propose a simple heuristic algorithm that illustrates the use of PFR with currently used preemption objectives. We use the objective of minimizing the number of preempted connections in our heuristic algorithm. Our proposed algorithm has a complexity of $O(k \log k)$, where k is the number of preemptable connections. Therefore, it is suitable for use in large networks having thousands of connections. We compare the performance of PFR heuristic algorithm with an existing preemption scheme. The simulation results show that the use of PFR heuristic algorithm results in approximately 20% less rejected requests compared to the preemption scheme that does not take into account future requests. Moreover, it also decreases the overall preemptions in the network by approximately 5%.

The rest of the paper is organized as follows: In Section II, we define the preemption problem in a DS-TE environment and discuss earlier work related to our specific problem. In Section III, we present our proposed strategy, PFR, while simulation experiments comparing the performance of PFR heuristic algorithm with an existing preemption algorithm are presented in Section IV. Finally, we give our conclusions in Section V.

II. PROBLEM BACKGROUND

In this section, we formally define the general preemption problem in a DS-TE environment and discuss earlier work related to our specific problem.

A. Problem Definition

We consider a DS-TE aware multi-class network which provides bandwidth guaranteed paths, such as CAIP [13]. Requests arrive in an online manner and no information about future requests is assumed. A request arrives at its source node, which calculates the constraint based route based on the available bandwidth for each class. To this end, we assume that a bandwidth constraint model is used to specify the *bandwidth constraints* for each traffic class. The bandwidth constraint for a given link specifies the maximum unreserved bandwidth (UB) for each traffic class. After a new bandwidth reservation, the UB values are updated, according to the bandwidth constraint model [14], and are then propagated throughout the network.

Based on the unreserved bandwidth for each traffic class, the source calculates a constraint based shortest path, from the source to destination, that satisfies the bandwidth requirement. Note that the unreserved bandwidth is not the same as amount of residual bandwidth available on a given link. Therefore, the actual node while reserving resources may need to preempt bandwidth of lower priority classes in order to free up adequate resources to accept the new request. However, the source while calculating the path is oblivious of the number of preemptions (if any) that would be needed at each link. The source accepts a request if a feasible path is found, otherwise the request is rejected. The accepted path is signalled through resource reservation protocols such as RSVP-TE [15]. Each node along this path has to make bandwidth reservations for its adjacent link. If the residual bandwidth on the link is greater than the bandwidth request then preemption is not required. Otherwise a preemption algorithm is used to free up enough resources so that the residual bandwidth on the link becomes greater than the bandwidth request. The preempted connections are retracted and bandwidth reservations are made for the new request. The preempted connections may or may not be rerouted which depends on the choice of the network service provider.

B. Related Work

The above mentioned distributed approach, where each node has to make preemption decision for its adjacent node, is more suitable for integration with distributed protocols such as OSPF-TE [16] and RSVP-TE [15]. However, we can also have a centralized scenario where a central entity having complete information about the network state makes the preemption decision. The preemption decision has to account for changes in the whole network as a result of preempting any connection. Therefore, the preemption decision must be based on a global view of the whole network. This makes optimization of objectives, such as minimizing the preempted bandwidth or minimizing the number of preempted connections, an NP-complete problem in a centralized scenario [2]. Considering the distributed nature of Internet protocols, a distributed preemption approach is preferred, and has therefore been the subject of several recent studies [7] [2] [8] [9]. These distributed preemption algorithms try to achieve various objectives such as: minimizing the preempted bandwidth, minimizing the number of preempted connections, minimizing the priority of preempted connections or a combination of the above. Oliveira et. al [7] provides the network

service provider the option of optimizing any of these criterions by minimizing an objective function which is a weighted sum of these criterions. Therefore, appropriate weights can be assigned, reflecting the choices of the network service provider. Note that in a distributed preemption mechanism, each node tries to optimize the preemption decision for its outgoing link only, regardless of how this decision would affect other preemptions in the networks. Therefore, preemption schemes can be independently deployed and need not be the same across all routers.

The above preemption schemes only optimize criterions keeping in view the current connections in the network. It is an important objective of a network service provider to accommodate maximum number of requests on the network. This objective has driven many routing schemes which try to maximize the overall throughput of the network: routing paths in a manner which results in greater amount of traffic and bandwidth to be placed on the network. Since in online routing, no information about future requests is known, these schemes work on heuristics that give good results under practical scenarios [11].

In shortest widest path routing [12], the cost of each link is set to be the inverse of the residual bandwidth on that link. Therefore, links that have a greater residual bandwidth have a lower cost and are thus preferred for inclusion in a new path. On the other hand, links that are approaching their capacity have a higher link cost and are thus less preferred for selection. Similar concept is used in MIRA [11], which calculates the maximum flow between each source-destination pair. Links on which the rate of change of maximum flow, as a result of an additional unit of bandwidth reservation, is greater have a higher cost and are thus less preferred for routing. Therefore, both widest path routing and MIRA are based on the idea of avoiding 'critical links', where the definition of critical links is different in each case.

III. PFR: PREEMPTION FOR FUTURE REQUESTS

In this section, we propose a new strategy for preemption, PFR, that results in higher QoS for future demands. We initially present the basic idea of PFR, its advantages and some important considerations regarding the use of PFR. This is followed by an Integer Linear Programming (ILP) formulation of PFR and finally, we present a heuristic algorithm for PFR which also minimizes the number of preempted connections.

A. Basic Idea, Advantages, and Considerations

We use the idea of 'critical links' in our preemption strategy. We base our strategy on the objective of a network service provider to maximize the efficiency of the network by accepting greater number of connection requests. Earlier work, such as [11] and [12], has shown that clever decisions of avoiding critical links while routing a path can help achieve this objective. However, without a priori information about future requests such strategies cannot guarantee optimal results. Preemption provides us with a powerful tool to select those connections for preemption that are likely to interfere more with future requests. Therefore, we propose PFR that selects those connections for preemption that are using critical links. We define critical links as those links which are approaching their capacity. Therefore, the cost of any link is taken to be the residual bandwidth of the link; the total cost associated with preempting a connection is the sum of all link costs along the connection's path. Minimizing the cost ensures that those connections which have a lower cost are selected for preemption. Consequently the bandwidth that would be released as a result of preempting such connections is likely to benefit future requests.

PFR also results in less preemptions in the network since links which are approaching capacity, thereby requiring preemption, are able to free up bandwidth. The impact of PFR on preemption is twofold: on current request as well as on future requests. Since preemption decisions are made in a distributed scenario, each node reserves the required bandwidth and this may require preempting connections. The downstream nodes get the reservation request after the preemption and reservation decisions earlier made by the upstream nodes. Therefore, the selection of connections for preemption made by PFR also reduces the preemption chances at the downstream nodes for the given request.

In addition to reducing preemptions for the current request, PFR also reduces the chances of preemption for future requests. As discussed earlier, the routing decision takes into account the unreserved bandwidth for the given class and not the residual bandwidth on a given link. Thus, it is possible that the residual bandwidth on a given link is zero while there is enough unreserved bandwidth for a certain class to accept the new request. This is possible in Russian Doll bandwidth constraint model [14] where lower classes can occupy the bandwidth proportion allocated for a higher class traffic but that bandwidth is preemptable. So the residual

bandwidth on a given link does give an indication of the amount of preemption that may be required on that link. PFR, by preempting connections from heavily congested links, increases the residual bandwidth on such links and thereby reduces the chances of future preemptions.

The objective of freeing bandwidth from congested links, if formulated as an optimization problem, would result in preempting much more bandwidth and connections than the actual requirement. It is undesirable to preempt connections unnecessarily since this reduces network utilization. Therefore, the use of PFR must be complemented with objectives such as minimizing the number of preempted connections or minimizing the priority of preempted connections. This ensures that the objective of accepting greater number of future requests is not achieved at the cost of preempting more connections for preemption. We therefore formulate PFR in a manner which allows other objectives to be incorporated in the objective function.

B. Integer Linear Programming (ILP) Formulation

We formulate PFR as an integer optimization problem. We consider a request for a new connection with bandwidth b and class c . We assume that the route has been calculated and is signalled through RSVP-TE. Now each node along the path has to make preemption decisions for its adjacent link l . All previous connections traversing link l with class lower than c are included in the preemptable set of connections ϵ which has a cardinality of k . The residual bandwidth on any link l is defined as $A_{bw}l$, thus the bandwidth required for preemption on link l is: $t = b - A_{bw}l$.¹ Total links in the network are n and a vector R with cardinality n represents the residual bandwidth on each link in the network.² The path of any connection, j , is also represented in a vector P with cardinality n . This vector comprises binary values where $P_j[l] = 1$ implying that the connection passes through link l while a zero value represents otherwise. The bandwidth of any request j is given by B_j where the cardinality of B is again equal to k .

We now define a preemption cost function for any connection j as:

$$C_j := P_j \cdot R^T \quad (1)$$

¹Without loss of generality, we assume that bandwidth is available in bandwidth modules and thus variables like b and t are integers.

²This information is propagated in the network through OSPF-TE [16].

This represents the cost of preempting connection j . Recall that the total number of connections in ϵ is k . We define a total cost vector X with cardinality k that is composed of C_j 's with j ranging from 1 to k .

We now define an objective function that minimizes the cost of preempting connections. The output of this integer program is a vector Z with cardinality k having binary values. A value of 1 at index j represents that connection j is selected for preemption while a value of 0 represents otherwise. As discussed earlier, PFR should be used with other criteria in order to ensure that the number of preempted connections remain low. In our objective function we also minimize the number of preempted connections. We therefore have two individual cost functions, each having an associated weight. These weights can be adjusted according to the requirements of network service provider. The optimization formulation for PFR along with the objective of minimizing the number of preempted connections is as follows:

GIVEN $\epsilon, b, B, X, t, \alpha, \beta$
 FIND Z (k binary variables)
 MINIMIZING $F(Z) = \alpha(Z.I^T) + \beta(X.I^T)$
 SUBJECT TO $Z.B^T \geq t$

In the above objective function, $Z.I^T$ represents the number of preempted connections where I is a unit vector with cardinality k . Similarly, the cost function for PFR is represented by $X.I^T$. The weights α and β can be used to give appropriate importance to each criterion. Our objective function can also be incorporated in the problem formulation of Oliveira et al. [7] which includes other objectives such as minimizing the priority of preempted connections and minimizing the preempted bandwidth.

C. Algorithm

The above ILP formulation can be used in small and medium sized networks with limited connections. For networks with thousands of connection, running an integer optimization problem would require significant time. Therefore, Oliveira et al. [7] have proposed heuristic algorithm to complement their ILP formulation so that their policy can be used in practical scenarios. We also propose a heuristic algorithm which minimizes the number of preempted connections and then uses the strategy of PFR. Since our first objective in this case is to minimize the number of preempted connections, we select the h largest connections that minimizes the number of preempted connections. Subsequently the last

connection is replaced with another connection that has the lowest cost (as per PFR strategy) but still makes the bandwidth sum of selected connections exceed the required threshold t .

Fig. 1. PFR Heuristic Algorithm (B, P, R, ϵ, t)

```

1)  $PreemptedBW := 0$ 
2)  $h := 0$ 
3) Sort  $B$  in decreasing order of bandwidth
4) do
5)   Add current value of  $B$  to  $PreemptedBW$ 
6)   Remove current element of  $B$ 
7)   increment  $h$ 
8) until  $PreemptedBW \geq t$ 
9) for every connection  $i$  in  $\epsilon$ 
10)   $C_i := P_i.R$ 
11) select  $h - 1$  largest connections for preemption
12) sort  $\epsilon$  in increasing order of  $C$ 
13) for every connection  $i$  in  $\epsilon$ 
14)  temporarily include it in preemption set
15)  if BW of preemption set exceeds  $t$ 
16)    break //preemption set complete
17)  else
18)    remove the temporary connection
19) preempt all connections in preemption set

```

Figure 1 lists the steps of our heuristic algorithm. In steps 4-8 we find the minimum number of connections that exceeds the threshold t . This value is represented as h . In steps 9-10 we calculate the cost of preempting any connection, as per the strategy of PFR. We then select $h - 1$ largest connections in step 11. The last connection is selected in steps 12-18 such that this is the connection with minimum cost (as per PFR) that also makes the preemption set exceed the bandwidth requirement t . Finally, the selected connections are preempted in step 19. Since the number of connections is k , the complexity of this algorithm is dictated by the sorting of these connections which can take $O(k \log k)$ steps.

IV. SIMULATION EXPERIMENTS

In this section, we describe the simulation experiments that were used to evaluate the performance of PFR strategy. We compare its performance with another preemption algorithm that does not take into account future requests while making the preemption decision. To this end, we select the preemption algorithm, *Minn_Conn*, proposed by Peyravian et al. [1], that minimizes the number of connections and in case of multiple feasible options it then tries to minimize the preempted bandwidth and the priority of preempted connections. This algorithm presents an ideal case for comparison since our PFR heuristic algorithm also minimizes the number of connections and then applies the PFR heuristic.

Therefore, for any preemption situation, the number of connections preempted by both *Minn_Conn* and PFR would be the same. Moreover, if h connections need to be preempted, $h - 1$ preempted connections would be the same and the only difference would be in the last preempted connection. PFR heuristic algorithm would preempt that connection that would release more bandwidth on congested links which would not be the case in *Minn_Conn*.

We compare the performance of PFR heuristic algorithm with *Minn_Conn* in terms of the number of rejected requests and the preempted connections. Note that for the first preemption in the network, both the algorithms would preempt the same number of connections. However, the difference in selection would result in different routing and preemption decisions in future. Since PFR releases bandwidth from congested links, it is able to accommodate more future requests and therefore results in less connection rejections compared to *Minn_Conn*. Similarly the availability of bandwidth on otherwise congested links also results in less preemptions in the network in case of PFR compared to *Minn_Conn*.

The simulation experiments are conducted on a homogeneous network topology which is adapted from the network used in [17]. It represents the Delaunay triangulation for the twenty largest metros in continental United States [17]. All links in the network are uni-directional having a capacity of 48 units. The bandwidth constraint model used for link advertisement and reservation is the Russian Doll bandwidth constraint model [14]. There are two QoS classes: high priority and low priority classes with bandwidth constraints 50% and 100% respectively.

The traffic matrix consists of 10,000 connections, where 20% of the requests are of high priority class while the rest are of low priority. Requests arrive with a constant inter-arrival time of one unit³ and are characterized by a source, destination, traffic class, the associated bandwidth, and the holding time of the request. The source and destination nodes are chosen randomly from amongst all source-destination pairs. The bandwidth demand for a request is uniformly distributed between 15 and 20 units for the high priority class and between 1 and 5 units for lower priority class. Finally, the call holding time for each request is uniformly distributed between 300 and 800 units.

With the above topology and traffic matrix, we generate results to compare the performance of PFR heuristic algorithm and *Minn_Conn*. All the results are presented

³Total simulation time is 10,000 units.

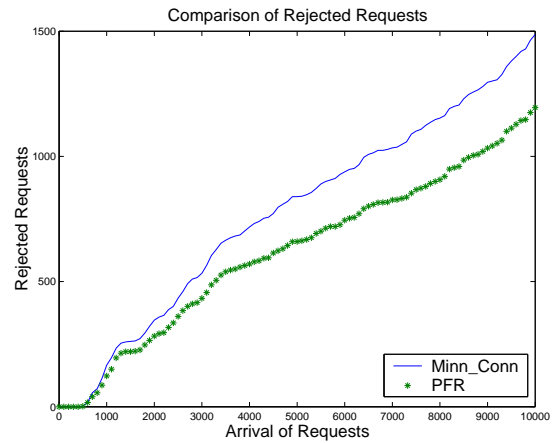


Fig. 2. Comparison of Rejected Requests

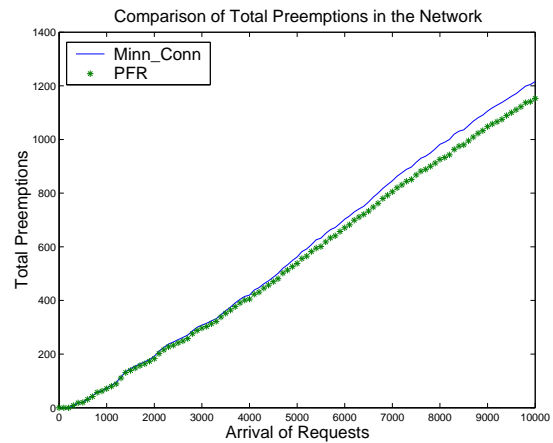


Fig. 3. Comparison of Preemptions in the Network

as a function of the arrival of requests on the network. Figure 2 shows the total number of rejected requests in both algorithms. Initially the network is being loaded and there are few connection requests that are rejected. Once the network gets loaded, the number of rejections start to increase; with the *Minn_Conn* algorithm rejecting more requests compared to the PFR algorithm. On average, the PFR heuristic algorithm rejects 20% less connections compared to the *Minn_Conn* algorithm. Note that only lower priority class traffic is being preempted; therefore the bandwidth that is made available after preemption was earlier preemptable for the higher priority class traffic while it was not available for use by lower priority class traffic. Since we release bandwidth on congested links, more lower priority requests, which were earlier being rejected, are now placed on the network.

As noted above, the bandwidth that is being released as a result of preemption was one that was earlier preemptable for higher priority class traffic. Our decision

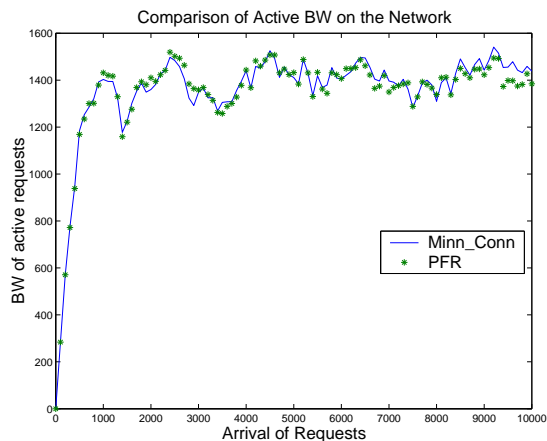


Fig. 4. Comparison of Active BW Placed on the Network

to prefer those connections that are traversing congested links ensure that future requests of high priority class traffic require less preemption since more residual bandwidth is available on that link. Figure 3 depicts this comparison which shows that in PFR heuristic algorithm approximately 5% less connections are preempted compared to the *Minn_Conn* algorithm.

Finally, we compare the two algorithms in the amount of *active bandwidth* that is being placed on the network. We define active bandwidth as bandwidth occupied by requests currently placed on the network. For the last connection, the PFR heuristic algorithm tends to select a higher bandwidth connection compared to the *Minn_Conn* algorithm which minimizes the preempted bandwidth once the number of preempted connections are minimized. This results in less active bandwidth which remains on the network after preemption in case of PFT heuristic algorithm. However, the decision to free bandwidth from critical links ensures that future requests that would otherwise be rejected are now placed. This offsets the earlier impact of selecting a heavy bandwidth connection as the last connection for preemption. Figure 4 shows the active bandwidth placed on the network which is the same on average for both *Minn_Conn* and PFR.

V. CONCLUSIONS

In this paper, we proposed a distributed preemption strategy, *Preemption for Future Requests (PFR)*, that results in higher QoS for future traffic requests. PFR is a truly distributed preemption scheme that can be used with current routing and resources reservation protocols, such as OSPF-TE and RSVP-TE. In PFR, we used the concept of critical links which has been used in earlier work on shortest widest path routing [12] and MIRA

[11]. We used this concept of critical links to select those connections for preemption that are traversing congested links. Such a preemption policy ensures that more bandwidth is available on congested links for future requests, thereby decreasing their blocking probability. We provided an Integer Linear Programming (ILP) formulation for this problem which can incorporate other preemption objectives. Our formulation allows flexible implementation of network service provider's priorities through assigning appropriate weights to each preemption objective.

We also provide a simple heuristic algorithm that illustrates the use of PFR with existing preemption objectives. The heuristic algorithm, although not optimal, has a low complexity and can be used in large networks having thousands of connections. We conducted simulations to compare the performance of *PFR* heuristic algorithm with an existing preemption scheme, *Minn_Conn*, that does not take into account future requests while making the preemption decision. The simulation results show that the use of *PFR* heuristic algorithm results in approximately 20% less connection rejections compared to *Minn_Conn*. Moreover, it also decreases the overall preemptions in the network by approximately 5%.

REFERENCES

- [1] M. Peyravian and A. D. Kshemkalyani, "Connection Preemption: Issues, Algorithms, and a Simulation Study," in *Proceedings of Infocom*, 1997, pp. 143–151.
- [2] J. A. Garay and I. S. Gopal, "Call Preemption in Communication Networks," in *Proceedings of Infocom*, 1992, pp. 1043–1050.
- [3] F. L. Faucheur, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, P. Cheval, and J. Heinanen, "RFC 3270: Multi-protocol Label Switching (MPLS) Support for Differentiated Services," May 2002.
- [4] —, "RFC 3564: Requirements for support of differentiated services-aware MPLS traffic engineering," July 2003.
- [5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "RFC 2475: An Architecture for Differentiated Services," December 1998.
- [6] E. Rosen, A. Viswanathan, and R. Callon, "RFC 3031: Multi-protocol Label Switching Architecture," January 2001.
- [7] J. D. Oliveira, C. Scoglio, I. Akyildiz, and G. Uhl, "New Preemption Policies for DiffServ-Aware Traffic Engineering to Minimize Rerouting in MPLS networks," in *Transactions on Networking*, vol. 12, no. 4, August 2004, pp. 733–745.
- [8] S. Jeon, R. T. Abler, and A. E. Goulart, "The Optimal Connection Preemption Algorithm in a Multi-Class Network," in *Proceedings of ICC*, 2002, pp. 2294–2298.
- [9] L. Lei and S. Sampalli, "Backward Connection Preemption in Multi-class QoS-aware Networks," in *Proceedings of ICC*, 2004, pp. 153–157.
- [10] S. Poretzky and T. Ganon, "An Algorithm for Connection Precedence and Preemption in Asynchronous Transfer Mode (ATM) Networks," in *Proceedings of ICC*, 1998, pp. 299–303.

- [11] K. Kar, M. Kodialam, and T. V. Laskhman, "Minimum interference routing of bandwidth guaranteed tunnels with mpls traffic engineering applications," in *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 12, December 2000, pp. 2566–2579.
- [12] R. Guerin, A. Orda, and D. Williams, "QoS Routing Mechanisms and OSPF Extensions," in *Proceedings of IEEE Global Internet*, November 1997, pp. 1903–1908.
- [13] F. R. Dogar, Z. A. Uzmi, and S. M. Baqai, "CAIP: A Restoration Routing Architecture for DiffServ Aware MPLS Traffic Engineering," in *Proceedings of Design of Reliable Communication Networks (DRCN)*, October 2005, pp. 55–60.
- [14] F. L. Faucheur, "RFC 4127: Russian Doll Bandwidth Constraint Model for DiffServ-aware MPLS Traffic Engineering," June 2005.
- [15] D. Awduche, L. Berger, D. Gan, T. Li, G. Swallow, and V. Srinivasan, "RSVP-TE: Extensions to RSVP for LSP Tunnels," RFC 3209, December 2001.
- [16] D. Katz, K. Kompella, and D. Yeung, "RFC 3630: Traffic Engineering (TE) Extensions to OSPF Version 2," September 2003.
- [17] S. Norden, M. M. Buddhikot, M. Waldvogel, and S. Suri, "Routing Bandwidth Guaranteed Paths with Restoration in Label Switched Networks," in *Proceedings of ICNP*, November 2001, pp. 71–79.