LSP and Back up Path Setup in MPLS Networks Based on Path Criticality Index

Ali Tizghadam, Alberto Leon-Garcia
Dept. of Electrical and Computer Engineering
University of Toronto
40 St. George St., Toronto, ON, M5S 2E4, Canada
{ali.tizghadam, alberto.leongarcia}@utoronto.ca

Abstract— **This paper reports on a promising approach for solving problems found when Multi Protocol Label Switching (MPLS), soon to be a dominant protocol, is used in core network systems. Difficulty is found largely in LSP routing and traffic engineering approaches. While there are a number of online and offline proposals to establish the LSPs but no one is a complete solution considering all the aspects of routing plan from traffic engineering point of view. Our research takes a viewpoint inspired by the concept of "between-ness" from graph theory, from which we introduce notions of link and path criticality indexes. The basis of the work is finding the most critical paths which are mathematically defined based on the algebra of routing. We try to avoid running aggregated flows or commodities on the most critical paths for the short term, and plan increasing the bandwidth of the critical paths for future if possible. This approach shows promise in simulations have run on benchmark networks available from research literature.**

Keywords- Flow Assignment; Graph Theory; MPLS; Quality of Service (QoS); Routing; Traffic Engineering

I. Introduction

The infrastructure of the traditional service provider is undergoing a fundamental transition from a telephone-service-focused circuit-switching architecture to a multi-service packet-switching architecture based on Internet Protocol (IP) and enhancements that enable Quality of Service and traffic engineering. IP transport networks that can transfer packets according to differentiated levels of QoS, availability and price are a key element to generating revenue through a rich offering of services and applications.

In this shared infrastructure MPLS has a key role. In addition to providing a transparent intermediate layer to hide the complexity of the different data layer technologies from the higher layers, MPLS provides a flexible routing mechanism by assigning traffic flows to the end-to-end label switched paths (LSP). MPLS can be used to engineer the traffic among different paths of the network by intelligent matching of the path capacities with flows to avoid network congestion.

An abundance of work has been done in the research community and industry to address the routing and flow assignment problem and traffic engineering issues in MPLS networks [1], [2], [3]. The main goal of the research reported in this paper is to look at the problem from another standpoint. Motivated by the definition of "between-ness" from graph theory [4], [5] we introduce the notion of link and path criticality indexes. The essence of the approach is to identify the most critical paths to avoid running the flows on these paths and try to set up LSPs and back up LSPs on less critical paths in the short-term as well as to plan to increase the bandwidth of the paths with high criticality index when possible. Our simulations on some benchmark networks discussed in the research literature show that the approach is promising.

The paper organized as follows. Section II reviews the state of the art in MPLS routing and flow assignment problems. We describe the issues with existing methods and the associated research challenges. In section III, we introduce the mathematics and notions of algebraic routing based on [6] that allow us, in section IV, propose our path-criticality index-based routing scheme. Section V provides an extension to compute maximally disjoint back up paths. Section VI provides validation for our proposal. We assess our proposed method on some benchmark networks and present encouraging results. Finally we conclude with a discussion of open issues and future work.

II. Current State of The Art

The most popular algorithm used in the research community for routing LSPs is the *shortest path routing* (SP*)*. In this method the path with the least total number of links between source and destination (or the one with minimum overall weight when the links are weighted) is chosen. If there is more than one shortest path between source and destination, the algorithm is flexible, and one can choose a shortest path at random. Typically, the LSP setup algorithm keeps information about the residual capacity of each link, and when a new request comes, only the links with enough residual capacity are considered by the shortest path algorithm. While the shortest path algorithm enjoys the benefit of simplicity, it can cause major problems to the network due to the lack of any load balancing mechanism. Some links might become saturated while some remain underutilized.

A variant of shortest path algorithm called widest shortest path (WSP) is introduced in [7]. The authors select the path with greater capacity in case where there is more than one shortest path between source and destination. WSP improves the performance of the shortest path algorithm but the possibility of having bottlenecks remains. Furthermore both SP and WSP do not impose any form of admission control. These algorithms always accept flows if a path with sufficient capacity exists and this may block a large number of future flows. We will shed more light on this problem when we describe the benchmark topology of Fig. 2.

Ref. [1] introduced the minimum interference routing algorithm (MIRA). In contrast to the prior methods, MIRA considers the effect of source-destination pairs on the routing plan. A number (max-flow) is assigned to every source-destination pair (S, D), showing the maximum amount of traffic that can be sent from S to D through the network. MIRA operates based on the notion that running traffic on some of the links may decrease the max-flow of (S, D). This process is called "interference". MIRA tries to find the links causing interference for a specific source-destination pair and avoid using them in the LSP construction phase. Indeed MIRA tries to build the paths in a way to minimize the interference. Introducing the notion of "interference" is significant, but some problems are still present with the algorithm introduced in [1]. MIRA concentrates on the effect of interference on just one source-destination pair, but there are situations where some links can cause bottleneck on a cluster of node pairs. Ref. [3] investigates three benchmark networks: parking-lot (Fig. 2), concentrator (Fig. 3) and distributor and shows that MIRA is unable to respond to the network flow requests correctly and causes blocking for a large number of incoming flows in these networks. Another issue with MIRA is its computational complexity in comparison to shortest path and widest shortest path algorithms. Finally MIRA is designed to provide bandwidth guaranteed paths. It does not account for any other QoS constraint in the network.

Profile-based routing (PBR) is another proposal for routing of bandwidth guaranteed flows in MPLS networks [3]. PBR assumes that the source-destination pair and the traffic-profile between them are known. According to PBR, a traffic profile is the aggregate bandwidth demand for a specific traffic class between a source-destination pair. PBR has two phases. In the offline phase a multicommodity flow assignment problem (an optimization problem) is solved with the goal of routing as much commodity as possible. Each profile is considered as a separate commodity. The result of this phase is used to assign the capacity of the links for each commodity. The online part will use these pre-allocated capacities to route the flows per class. In this phase PBR simply uses the shortest path algorithm and because of this the computational complexity of the online phase in PBR is less than MIRA and comparable with SP and WSP. But PBR also has some problems. Like MIRA, PBR just cares about bandwidth and does not consider multi constraint QoS problem. Furthermore in [8] the authors introduce a network called "Rainbow Topology" and show that the performance of PBR in this network is much worse than MIRA and WSP. The main reason behind this is that PBR relies on the results of the offline part which is not always correct.

III. Algebra for Routing

In this section we introduce some mathematical notions that are used in the rest of the paper. The methodology and notions that are being used in this section are mostly based on ref. [6]. We model a network as a directed graph $G(V, E)$. V is the set of nodes and E is the set of links or edges of the network. A *walk* in a network is a sequence of nodes $v_n v_{n-1} ... v_1 v_0$ such that (v_i, v_{i-1}) is a network link for $1 \le i \le n$ and a *path* is a walk where all nodes are distinct. The *order* of a walk is the number of links it contains. A path of order zero is called *trivial*. Given walks P and Q where the last node of P is the first node of Q, we denote their concatenations by $P \circ Q$. For the special case where uv is a path with only two nodes, we say that $uv \circ Q$ is the extension of walk Q to node u, or that $uv \circ Q$ is the extension of walk Q to link (u, v). An algebra for routing is defined as follows: An algebra for routing is an ordered septet $(W, \le, L, \sum, \phi, \oplus, f)$ comprising:

A set of weights W;

A set of labels L;

A set of signatures $\sum$;

A total order $\le$ on W; (the relation $a \prec b$ on W is defined such that $a \le b$ and $a \ne b$)

A binary operation $\oplus$ that maps pairs with a label and a signature into a signature;

A function f that maps signatures into weights;

The special signature ϕ;

Every algebra for routing has at least these two following properties:

Absorption For all $l \in L$, $l \oplus \phi = \phi$;

Maximality For all $a \in \sum - \{\phi\}$, $f(a) \prec f(\phi)$

An algebra for routing is finite if $\sum$ is finite, in which case the set of labels and weights can also be finite. The links of a network are assigned labels from the set L, with $L(l)$ denoting the label of link l. The walks of the network are assigned signatures from the set $\sum$, with $s(P)$ denoting the signature of the walk P. The signature of walk P is obtained from the labels of its constituent links through composition with operation $\oplus$. The special signature ϕ is reserved for *unusable* walks, which are those that cannot be used for packet transport. Any walk with signature different from ϕ is said to be *usable*. The mapping f from signatures to the totally ordered set of weights W results in an assignment of weights to walks with the weight of walk P being given by $f(s(P))$. This establishes a ranking among walks (in our case paths). The lower the weight of a walk, according to the order $\leq$, the "better." The "maximality" property implies that any usable walk is "better" than an unusable one.

An optimal path from u to d is a usable path from u to d of minimum weight, that is, whose weight is less than or equal, according to the order $\leq$, to that of any walk from u to d. In other words the optimal path is "better" or "as good as" any walk from u to d. We will use the routing algebra in the next section to develop the idea of less criticality based routing.

IV. Path Criticality Index Based Routing (PCIBR)

The routing and flow allocation problem in core networks can be formulated as an optimization problem [9], [1]. In this paper our goal is to find a robust routing plan for the core network that allows the network service provider to manage the assignment of flows to the paths primarily at the edge of the core network. By robustness we mean resiliency against failures, predicted changes in traffic demands and source-destination pairs. To achieve the goal first we need to identify the important factors affecting the routing plan and flow assignment. One can summarize these factors as:

1. Network topology and connectivity.
2. source-destination pairs.
3. Capacity of the links.
4. Traffic Matrix.

In order to have a robust routing plan we need to recognize the effect of link and node topology on network connectivity. Connectivity is a well studied subject in graph theory [4], [5], [10] allowing us to define some useful metrics to measure the sensitivity of the network to node or link failures. Capacity of a network is another key issue in flow assignment problem. Clearly the paths with more capacity are desired since the low capacity paths are prone to congestion. Hence an intelligent routing plan should avoid routing the flows onto the low capacity paths and should request for capacity increases for those paths if possible. Finally traffic demand directly affects the routing plan. The traffic demand profile or source-destination pairs may change from time to time (e.g. week-day traffic profile). Traffic changes might be predictable and periodical or chaotic. We assume in this paper that the demand matrix is stochastic but well-behaved and do not consider the case of catastrophes.

We now introduce two metrics to estimate the effect of the aforementioned characteristics: link criticality index (LCI) and path criticality index (PCI) which are built based on the theory of graphs [4], [5] and the algebra for routing. We will subsequently propose our routing algorithm based on PCI.

A. Link Criticality Index

Freeman [4] introduced a useful measure in graph theory called "between-ness centrality." Suppose that we are measuring the centrality of node k. The between-ness centrality is defined as the share of times a node i needs a node k in order to reach a node j via the shortest path. Specifically, if g_{ij} is the number of geodesic paths (shortest paths) from i to j, and g_{ikj} is the number of these geodesics that pass through node k, then the between-ness centrality of node k is given by: $\sum_{i,j} \frac{g_{ikj}}{g_{ij}} \quad i \neq j \neq k$

We can modify the definition of between-ness centrality to introduce a useful measure for criticality of links in a network. Suppose p_{sd} is the number of paths between source-destination pair (s,d) and p_{sld} is the number of paths between (s,d) containing the specific link l. Inspired by the definition of between-ness one can quantify the effect of network topology and source-destination pairs by dividing $\frac{p_{sld}}{p_{sd}}$ over all source-destination pairs. This gives an indication of how critical the link l is in the network topology. This criticality of link l is then:

$$LCI_{top}(l) = \sum_{s,d} \frac{p_{sld}}{p_{sd}} \tag{1-a}$$

The effect of link capacity and average demand for the source-destination (s,d) (provided by the traffic matrix) is accounted for in the residual bandwidth of the link. The residual bandwidth of link l is the capacity available after considering the flows already traversing the link, and is denoted by c_l. Obviously the link criticality has an inverse relation with available bandwidth, and so we can account for residual bandwidth by multiplying equation (1-a) by $1/c_l$.

The ability of a link to handle a given offered volume of data flow or a given level of QoS also has to be reflected in the link criticality. For example, suppose a new request offers flow γ_l to the link. Let the indicator function $I(x)$ and the modified link criticality be:

$$I(x) = 1 \;\; if \;\; x \rangle 0 \;\; otherwise \;\; 0 \tag{1-b}$$

$$LCI_{trc}(l) = \frac{1}{c_l} \times \frac{1}{I(c_l - \gamma_l)} \qquad \text{(1-c)}$$

In general, we can consider the effect of QoS constraints (if any) by defining link criticality index (LCI) as:

$$LCI(l) = \sum_{s,d} \frac{p_{sld}}{p_{sd}} \times \frac{1}{c_l} \times w_0(l) \qquad (1)$$

The indicator function can be incorporated in $w_0(l)$. The term $w_0(l)$ is the overall QoS weight of the link that is used in cases where we want to investigate multi-constraint routing (which is the subject of our next step of research and out of the scope of this paper). One can clearly see the effect of topology and connectivity $(\frac{p_{sld}}{p_{sd}})$ as well as capacity and traffic matrix (c_l) in the definition of LCI.

With reference to Algebraic Routing, we can say that the labels are given by the Link Criticality Indexes defined here.

B. Path Criticality Index (PCI)

We are now ready to define the path criticality index (PCI). In this section we assume that there are no constraints involved other that bandwidth, so $w_0(l) = 1$ assuming the link capacity is more than the demand. First we define the signature on each path by having the labels be defined to be $LCI(l)$ for every link. We define the binary operation $\oplus$ as the regular sum (+) of two real numbers. By a recursive operation of $\oplus$ over the constituent links of the path we can conclude that the signature of a path between a source-destination pair (s, d) consisting of the links $(l_1, l_2, ..., l_n)$ is given as follows. Considering $l_1, l_2, l_3, ..., l_n$ as the constituent links of the path P we build recursive paths P_i by starting from $P_1 = l_1$ and then the other paths by concatenation of the links:

$$P_i = l_i o P_{i-1} \Rightarrow s(P_i) = L(l_i) \oplus s(P_{i-1}) \;\; for\ i = 1,2,...,|P| = n$$

Recursive application of this formula will result in:

$$s(P) = L(l_1) \oplus L_2 \oplus .... \oplus L_{n-1} \oplus L_n$$

By replacing $L(l_i) = LCI(l_i)$ and we have that the signature of the path P is:

$$s(P) = \sum_{i=1}^{n} L(l_i) = \sum_{i=1}^{n} LCI(l_i) \qquad \text{(1-1)}$$

Now we define the function f in the algebraic routing plan that maps the signatures to the path weights by the following formula ($|P|$ is the order of path P assuming P is non-trivial):

$$PCI(s,d) = f(s(P)) = \frac{s(P)}{|P|} \qquad (2)$$

This function satisfies the conditions required for f to be a mapping function in our algebraic routing to map signatures to weights. Considering the definition of $s(P)$ in (1-1) and substituting the result of signature $s(P)$ in (2) one can see that $PCI(s,d)$ is the average of $LCI(l_i)$ of the constituent links of the path P:

$$PCI(s,d) = \frac{s(P)}{|P|} = \frac{\sum_{i=1}^{n} LCI(l_n)}{n} \qquad \text{(2-1)}$$

Where $|P| = n$ is the order of path n.

More generally, the path criticality index is a function of link criticality indexes l_1 to l_n:

$$PCI(s,d) = f(LCI(l_1), LCI(l_2), ..., LCI(l_n)) \quad (3)$$

The function has to meet the requirements of the algebra for routing defined in section III. Finding the best form for f is one of our ongoing research topics, but the average function (2) which is introduced here works well for all of the benchmark networks that we have examined.

C. Path Criticality Index Based Routing Algorithm (PCIBR)

The basic idea of our routing algorithm is to accommodate new requests for connections along LSPs that have a low PCI. This requires that we find the link criticality indexes. To do this, we need to obtain all possible label switched paths for each source-destination pair. This is not feasible since the number of paths grows rapidly with the number of network nodes and links. Although the shortest path is not necessarily the path with the lowest PCI, one can expect that the path or paths with lowest PCI are among the *k-shortest paths* of the network. Hence we use the k-shortest path method proposed by Eppstein with a modification to avoid loops [11].

Our algorithm begins with a predefined value of k, but the value may be increased during the course of running the routing algorithm if the desired number of paths to route the traffic cannot be found. We use thresholds tr_1 (the default value is infinity in case the threshold is not defined) and tr_2 (the default value is zero in case the threshold is not defined) for PCI. The first threshold defines the lower confidence boundary for the path criticality index. All the paths with path criticality index less than tr_1 are considered eligible to route traffic. On the other hand all the paths with the criticality index larger than tr_2 are considered too risky and may be identified to the (offline) core network management system for increased capacity assignment. The paths with criticality index in between the thresholds will share traffic based on their criticality index as long as they remain within the boundaries.

We note that when a path accepts traffic, the residual capacity of its links will decrease for the duration of the traffic flow. This means that the criticality index of this path must be increased. In other words a constant monitoring of the PCI for all the paths is necessary and in fact it is the main building block of our algorithm.

PCIBR:

Input: ***A network or more formally a graph $G(L,E)$, a set of capacities (residual capacities if we are not in the initial stage), a set of source-destination pairs (s,d), and traffic matrix D for these source-destination pairs.***

Output: ***A set of LSPs between all source-destination pairs meeting the demand requirements according to the traffic matrix.***

Algorithm: (By default we are in idle state)

```
1. Go to the initial state, Select $k$, $tr_1$ and $tr_2$ (use default values of $tr_1$ and $tr_2$ if we are not concerned about the thresholds).
2. Compute the k-shortest paths for all source-destination pairs to meet the demand requirements and measure their $PCI$.
3. If a path with $PCI \le tr_1$ exists, choose that path to route the demand (in case there are more than one path meeting the threshold requirements then choose the one with lowest PCI).
    a. Adjust residual capacities
    b. Adjust path criticality indexes accordingly
4. If there are paths with $PCI \ge tr_2$ send a message to the core management system requesting additional bandwidth for these paths.
5. If there is no path satisfying the condition of step 3 then increase $k$ by one and go to step 2.
6. In the (very rare) case that no path with criticality index less than $tr_1$ can be found (this happens when $k$ keeps increasing but no satisfactory path results) then use the paths with $tr_1 \le PCI \le tr_2$ in a round robin or random fashion.
```

The most time-consuming part of the algorithm is the k-shortest path calculation. According to [11] the complexity of the proposed k-shortest path algorithm is $O(m + n\log n + k)$ where, m is the number of links and n is the number of the nodes. The algorithm is polynomial time and as a result PCIBR is also a polynomial time algorithm if we set a maximum value for k such as $k_{\max}$. The complexity of the other parts of the algorithm (without k-shortest path) is $O(N^2)$. The problem in this case is that we might not get the desired path from the algorithm by $k_{\max}$ iteration. On the other hand if we do not place any upper bound for k then we can find a desired path (if one exists) but not necessarily in polynomial time. This comes from the fact that the problem we are trying to solve by nature is an NP-Complete one [12].
In MIRA the time complexity without considering the max-flow algorithm is also $O(N^2)$. But if we compare the time complexity of the Tarjan max-flow method [12] that is being used in MIRA ($O(n \times m \times \log(n))$ where n and m are node and link dimensions) with the Eppstein k-shortest path method [11] used in our algorithm ($O(m + n \times \log n + k)$) we notice that the complexity of the k-shortest path algorithm is less than max-flow one.

V. Back up Path

To set up the back up path for primary LSP, a greedy algorithm is to select the second least critical path as the back up path, however in many cases there are totally disjoint paths with higher path criticality but more suitable candidate for being the back up path. This simple example is clearly demonstrating the idea:

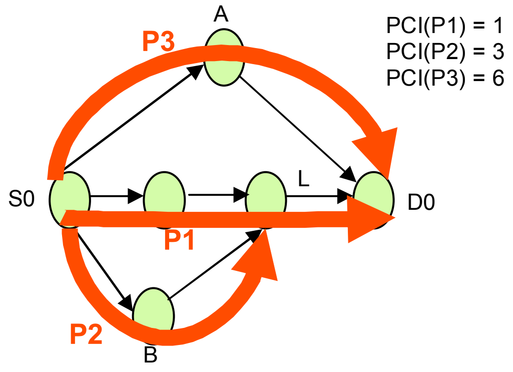


Figure 1

In Fig. 1 the primary path from S0 to D0 is P1with PCI=1. A greedy algorithm will choose P2 as the back up path since its path criticality is less than P3, but P1 and P2 are sharing link L while P3 is totally link-disjoint with P1, therefore P3 is a better choice for being back up path of P1. Hence the best approach to find the back up path is to first examine the sub-graph obtained by removing all the constituent links of path P1 and run PCIBR to obtain the least critical path from S0 to D0 if any, otherwise the next phase is to obtain the second least critical path from original graph.

VI. Experimental results (proof of concept)

The PCIBR algorithm has been implemented with C++ and tested for many network configurations. Among these we have chosen three benchmark networks to show that Path Criticality Index Routing is an effective method to find the best LSP routing plan for networks that have been found difficult to handle by previous proposals. We have also applied our algorithm on the network used in [1] to show its effectiveness in more realistic networks.

A. Parking-Lot Topology

The parking-lot network topology, shown in Fig. 2, is an interesting example. If one unit of bandwidth is requested to be sent from S_0 to D_0, all the previous routing approaches such as SP, WSP and MIRA will choose the straight path and run the flow resulting in the blocking of demands of one unit come from any other source such as S_i to the destination D_i [3]. A wiser decision is to block the first request from S_0 to D_0 so the network will be able to route the other n source-destination pair requests.

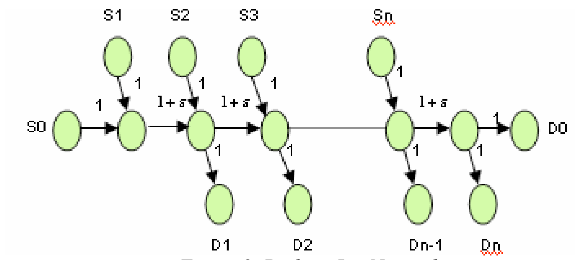


Figure 2. Parking Lot Network

To investigate the behavior of our *PCIBR* algorithm we suppose n = 3 and we have just four source-destinations. In Table 1 we summarized the results of calculating the path criticality index based on the formulas (1) and (2).

Path	Path Criticality Index (PCI)
S0 –> D0	1.490909
S1 –> D1	1.272727
S2 –> D2	1.272727
S3 –> D3	1.272727

Table 1. Parking Lot Network n=3

In this experiment we assumed that the commodities could be sent just from sources to destinations with the same index. One can see that the criticality index of path $(S0, D0)$ is more than the others. So by defining appropriate thresholds the proposed algorithm can be made to block the request from $S0$.

In more general case where different combinations of source-destination pairs are possible and the number of nodes is much more than 3, our experiment results show that the criticality of the path $S0 \rightarrow D0$ is much higher than the other combinations. In Table 1.1 the results of our experiment with n=10 is reflected and clearly shows that $S0 \rightarrow D0$ is the most critical path. .

Path	Path Criticality Index (PCI)
S0-D0	27.590902
S1-D1	11.333333
S2-D2	14.090907
S3-D3	15.909087
S4-D4	17.121209
S5-D5	17.727271
S6-D6	17.727271
S7-D7	17.121209
S8-D8	15.909087
S9-D9	14.090907
S10-D10	11.333333

Table 1.1 Parking Lot Network with different S-D combinations n=10

In general case of the parking-lot topology with n nodes the same approach can be followed and again the proposed routing plan will choose the straight path $S0 \rightarrow D0$ as the most critical one.

B. Concentrator Topology

In the concentrator topology, shown in Fig. 3, MIRA, SP and WSP all will have trouble with a request for transporting a flow of n units from $S0$ to D. They will all choose the shortest path (S0-C-D) which will then leave only one unit of bandwidth for link C-D. In other words only one more request (with 1 unit of bandwidth) can be concurrently handled.

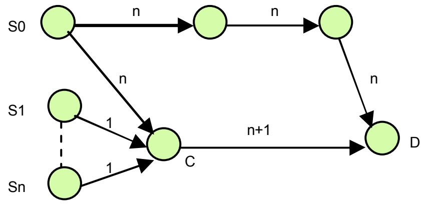


Figure. 3. Concentrator Network

Table 2 shows the result of our tests on the concentrator topology. The path (S0-C-D) (or the path with two links from $S0$ to D) has PCI much more than the one with three links. So the PCIBR algorithm will choose the longer path (the path with three links) to save the bandwidth of link C-D for other source-destination pairs.

In general case of the Concentrator network with n nodes the same approach can be followed and again the proposed routing will choose the path $S0 \rightarrow D$ (with three links) as the least critical one.

Path	Path Criticality Index (PCI)
S0 –> D (3links)	0.166667
S0 –> D (2links)	0.520833
S1 –> D	0.937500
S2 –> D	0.937500
S3 –> D	0.937500

Table 2. Concentrator Network n=3

C. Rainbow Topology

Fig. 4 shows the Rainbow network. Ref. [8] uses this topology to show the shortcomings of profile-based routing (PBR). The authors in [8] show that the performance of PBR in Rainbow network is much worse than MIRA, WSP and SP. They also show that PBR will be blocked after accepting 2 units of bandwidth for $(S1, D1)$ and 2 units for $(S2, D2)$.

We conducted our experiment in two phases to show the details of the PCIBR algorithm. In first two columns of Table 3, the PCI for different paths between source-destination pair $(S1, D1)$ is shown. One can observe that the longest path is the less critical one; hence routing will allocate two units of bandwidth.

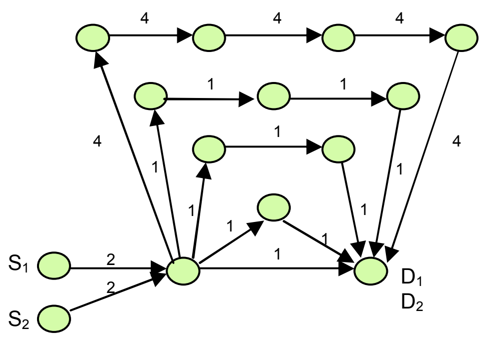


Figure 4. Rainbow Network

In the second step, the source-destination pair $(S2, D2)$ will ask for routing of two units of bandwidth. At this time the residual capacities have changed and the new PCIs have to be calculated.

The results are in columns 3 and 4 of Table 3. Again the longest path is still the less critical one and the best candidate for routing the requested flow.

Path	(PCI)	Path	(PCI)
S1 –> D1 (6links)	0.166667	S2 –> D2 (6links)	0.250000
S1 –> D1 (5links)	0.420000	S2 –> D2 (5links)	0.420000
S1 –> D1 (4links)	0.425000	S2 –> D2 (4links)	0.425000
S1 –> D1 (3links)	0.433333	S2 –> D2 (3links)	0.433333
S1 –> D1 (2links)	0.450000	S2 –> D2 (2links)	0.450000

Table 3. Rainbow Network. Two phases of flow assignment

D. Simulation Results for KL-Topology

We ran a set of simulations on the network of Ref. [1] that we refer to as the KL-topology (Fig. 5-a). We assume the bandwidth of the thin links is 1200 units and that the thick links have 4800 units of bandwidth. In order to compare the results with [1] we implemented exactly the same simulations as in [1].

In the first experiment the requests for bandwidth (which is our main QoS measure in these simulations) arrive with Poisson distribution and stay for ever (no departures). In our tests the bandwidth requests for LSP setups are taken to be uniformly distributed between 1 and 3 units.

In Fig. 5-b we show the number of rejected calls for the KL-topology and we compare the performance to that of shortest path, widest shortest path and PCI (with initial value $k = 1$ and possible subsequent increments based on PCIBR). We measured the number of blocked requests for LSP setup from $S1$ to $D1$. As one can see after about 1200 trials the SP algorithm starts to experience blocking while the PCI-based algorithm can adapt itself and still accept bandwidth requests

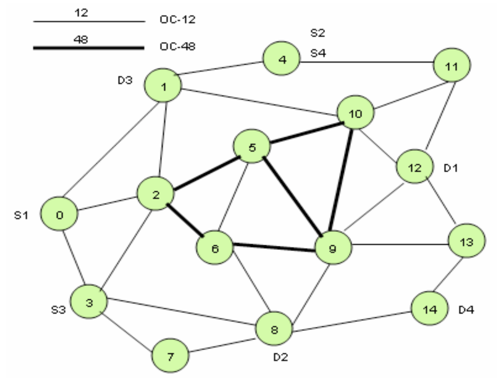
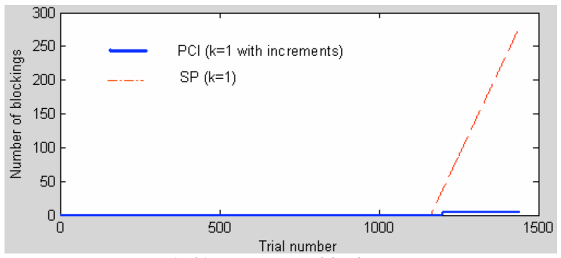


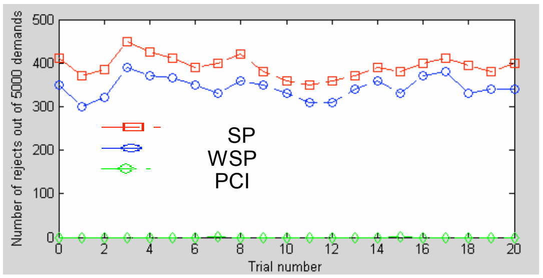
Figure 5. Network Topology from [1] (5-a) KL Network

without significant blockage. We observed that the PCIBR begins increasing the value k when the knee in the curve is reached.



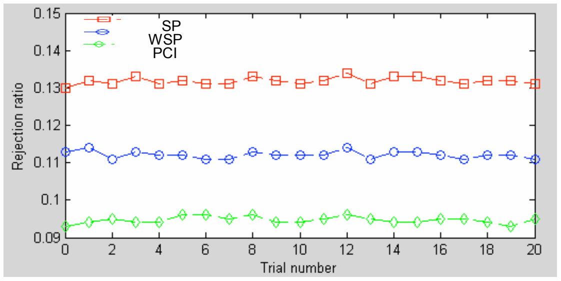
(5-b) Static case, blockage

In a second example we assessed the path acceptance level of the algorithms. At first we ran 5000 attempted LSP setups and counted the number of unsuccessful attempts. We conducted 20 trials and in each one counted the number of rejected paths (out of 5000). The results for three algorithms SP, WSP and our PCIBR are compared in Fig. 5-c. We note that the results for PCIBR are very close to those of MIRA in [1]. Thus PCIBR shows the same effectiveness as MIRA while maintaining good performance on benchmark networks where MIRA shows major blocking.



(5-c) Static, Path acceptance

In the last experiment we examined the behavior of the algorithms in the presence of dynamic traffic. Fig. 5-d shows the proportion of the LSP requests rejected in 20 experiments for the following scenario. LSP requests arrive between each source-destination point according to a Poisson process with an average rate λ, and the holding times are exponentially distributed with mean $1/\mu$. We assume $\lambda/\mu = 150$ in our experiments. In this scenario, we scale down the bandwidth of each link in the KL-network with the ratio of 10 to have bandwidths of 120 and 480 units for thin and thick links respectively. Next we generate about 1,000,000 requests for path setup and measure the rejection ratio for each one of the algorithms. The results are shown in Fig. 5-d and again the results are very close to MIRA.



(5-d) Dynamic

VII. Concluding Remarks

In this paper we have proposed a new approach for path setup and routing of flows in MPLS networks. The most important problem with the existing approaches is that each one of them solves a part of the overall problem but fails with other parts. We have tried to consider different aspects of the network (i.e. topology, capacity, and demand) and quantified these aspects using measures inspired by the mathematics of graphs. The essence of our work is based on determine a path criticality index for each path showing how critical that path is to the changes in the topology and traffic demand of a network. Our algorithm identifies the least critical paths for allocation of new traffic flow requests.

The results from applying the proposed algorithm to networks that are difficult to handle by existing approaches are very encouraging. These results confirm the validity of the notion of path criticality. The simulation results show that PCIBR matches the performance of MIRA in typical networks. We also showed that the complexity of PCIBR relative to MIRA shows improvement.

However there are many issues that remain to be investigated in the new approach. We need to investigate more on the effect of the threshold parameters. As we have seen in equation (3) the PCI is a function of link criticality indexes and in our first algorithm we used "average function" to obtain PCI but more elaboration is necessary. We need to add QoS constraints to the network and assess the behavior of our approach at the presence of different QoS constraints and modify it to accommodate all the situations if necessary.

References

[1] K. Kar, M. Kodialam, and T. V. Lakshman, "Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications," IEEE Journal on Selected Areas in Communications, Vol. 18, No. 12, Dec. 2000, pp. 2566-2579.

[2] D. O. Awduche, L. Berger, D. Gan, T. Li, G. Swallow, and V. Srinivasan, "Extensions to RSVP for LSP tunnels," Internet Draft draf-ietf-mpls-rsvp-lsp-tunnel-04.txt, Sep. 1999.

[3] S. Suri, M. Waldvogel, and P. R. Warkhede, "Profile-Based Routing: A New Framework for MPLS Traffic Engineering," In Quality of Future Internet Services, Lecture Notes in Computer Science 2156, Springle Verlag, Sep. 2001.

[4] L. C. Freeman, "Centrality in Networks: I. Conceptual Clarification," Social Netwoks, No. 1, 1978/79 ,pp. 215-239.

[5] S. P. Borgatti, "Centrality and Network Flow," Social Networks, Vol. 27, No. 1, 2005, pp. 55-71.

[6] J. L. Sobrino, "An Algebraic Theory of Dynamic Network Routing," in IEEE/ACM Transactions on Networks, Vol. 13, No. 5, October 2005, pp. 1160-1173

[7] R. Guerin, A. Orda, and D. Williams, "QoS routing mechanisms and OSPF extensions," In Proceedings of 2nd Global Internet Miniconference, Nov. 1997.

[8] S. Yilmaz, I Matta, "On the Scalability-Performance Tradeoffs in MPLS and IP Routing," Proceedings of SPIE ITCOM May 2002.

[9] T. Ott, T. Bogovic, T. Carpenter, K. R. Krishnan, and D. Shallcross, "Algorithms for Flow Allocation for Multi Protocol Label Switching," MPLS International Conference, Oct. 2000.

[10] A. H. Dekker, B. D. Colbert, "Network Robustness and Graph Topology," In Proceedings of 27th Australasian Computer Science Conference, Vol. 26, Jan 2004, pp. 359-368.

[11] D. Eppstein, "Finding the k Shortest Paths," Society for Industrial and Applied Mathematics (SIAM) Journal of Computing, Vol. 28, No. 2, 1998, pp. 652-673.

[12] R. K. Ahuja, T. L. Magnanti, J. B. Orlin "Network Flows: Theory, Algorithms, and Applications", Prentice Hall, 1993.

.