Distributed Partial Information Management (DPIM) Schemes for Survivable Networks - Part II

Dahai Xu Chunming Qiao
Department of Computer Science and Engineering
State University of New York at Buffalo
{dahaixu, qiao}@cse.buffalo.edu

Abstract— **A major challenge in shared path protection is to select a jointly optimized pair of active and backup paths. While Integer Linear Programming (ILP) based approaches are notoriously time consuming, existing heuristics such as active path first (APF) can only achieve sub-optimal results.**

In this paper, we propose a novel heuristic called APF with *potential backup cost* or APF-PBC under the distributed partial information management framework described in [1] for ultra-fast processing of on-line requests. We show that APF-PBC can *outperform* existing schemes based on Integer Linear Programming (ILP) with exactly the same input (either partial or complete information) and constraint. We also show how an intuitive function to compute the potential backup cost is derived mathematically based on the statistical analysis of experimental data.

Keywords— **bandwidth sharing, distributed routing, on-line heuristic, potential cost, statistical analysis.**

I. INTRODUCTION

Many emerging applications including nation-wide collaborative science and engineering projects require that reliable, high-bandwidth connections be dynamically set up and released between large computing resources (e.g., storage with terabytes to petabytes of data, clustered supercomputers and visualization displays). To meet the requirements of these emerging applications in an economical way, a network must quickly and dynamically provision bandwidth-guaranteed connections, each with sufficient protection against possible failures of network components (e.g., links or nodes).

In this work, we will introduce an ultra-fast and efficient algorithm to process *on-line* requests for either establishing or releasing a connection. Driven by the demand for more flexible services from applications mentioned above (the "push" effect), and service providers' desire to generate more revenues (the "pull" effect), much frequent requests for connections establishment and release are expected in the near future. This, in conjunction with the large (and increasing) size of the Internet, makes distributed control and signaling extremely attractive for the purpose of achieving a good scalability.

While approaches other than path protection may be adopted to achieve a good degree of survivability, this work considers shared path protection for its quick restoration speed (even without fast fault localization) and high bandwidth efficiency. In the following, we will limit our discussions to protection against a single link failure as the problem of protection against a single node failure can be transformed to that of protection against a single link failure by splitting each node into two halves interconnected with a "virtual" directed link.

With path protection (but no bandwidth sharing), a link-disjoint pair of paths, called active path (AP) and backup path (BP), respectively, from an ingress node to an egress node is used to satisfy each connection. More specifically, assume that the connection requires w units of bandwidth. The same amount of bandwidth (called active bandwidth or ABW) will be used on each link along the AP to establish the connection. Meanwhile, w units of bandwidth (called backup bandwidth or BBW) is also reserved on each link along the BP. This BP will be used to (re-)establish the connection to re-route the information to be carried after the AP breaks due to a link failure along the AP (and before the AP can be restored).

The concept of *shared protection* is illustrated using the following example. Assume that two connections, requiring w_1 and w_2 units of bandwidth, respectively, are established using two link disjoint APs. Note that under such an assumption, the two APs cannot break at the same time *provided that* at most one link in the network can fail at any given time, or more precisely, no additional failures may occur before an existing failure is repaired (which is a fairly practical and reasonable assumption). Accordingly, their corresponding BPs need not be used to (re)-establish the two connections, respectively, at the same time either. Hence, if the two corresponding BPs use the same link e, they can share the backup bandwidth (BBW) without affecting the survivability of either connection. More specifically, with shared protection, the total BBW that needs be allocated on link e (for the two BPs) is $\max\{w_1, w_2\}$ (instead of $w_1 + w_2$).

The problem of minimizing the total bandwidth or TBW (which is the sum of ABW and BBW) needed to satisfy a given set of requests for establishing connections is NP-hard. Many *off-line* approaches with applications to SONET, ATM and WDM networks case have been proposed (see for example the work in [2], [3], [4] and the references contained therein).

In the on-line case to be studied, not all requests arrive at the same time, and a decision as to how to satisfy a request for connection establishment (if possible at all) has to be made without knowing which requests will arrive in the future, and, for the sake of guaranteed QoS, *without* being able to rearrange the way existing connections are established. In such an case, a sensible approach is to try to allocate minimal TBW when satisfying *each* request for connection establishment, and de-allocate a maximal BBW when satisfying each request for connection release (as the ABW to be de-allocated is always a fixed amount).

The above approach *may* lead to either the minimization of TBW needed to support a given number of requests, or the maximization of revenues for a given network capacity, or both (these two performance metrics will be described in more detail in Sec. IV). In fact, several ILP formulations have been proposed to optimize for *each* request in the online case. However, even if one can allocate minimal TBW (and deallocate maximal TBW) for *each* request, one *cannot guarantee* the best overall performance in terms of minimization of TBW needed to support *a given number* of requests that have arrived over time, or maximization of revenues for a given network capacity.

The above reason is why an ILP formulation for the online case may not yield the best overall performance (unlike in the off-line case). In fact, one of the pleasantly surprising results of this paper is that our proposed heuristic algorithm can perform *better* than an ILP formulation (with the same input).

Clearly, reducing the additional BBW needed for a request for connection establishment through BBW sharing will help reduce TBW needed to satisfy the request (but not necessarily minimize TBW unless the AP, and thus the total ABW allocated, is already fixed as in the case where the APF heuristic is used). However, in order to determine whether or not a new BP chosen to satisfy the request for connection establishment (or release) can share (or have shared) BBW with an existing BP on a given link, and consequently, how much BBW on the link needs be allocated (or de-allocated) to satisfy the request, one needs to first determine whether or not their corresponding APs are link disjoint. Such decisions can be made if a controller (either in centralized or decentralized control implementation) maintains *complete per-flow* information as in the SCI scheme (which uses an ILP formulation) [5] or *complete aggregated* information as in the SR scheme (which uses the APF heuristic) [6]. In both cases, the amount of information to be maintained by a controller is at least $O(E^2)$, where E is the number of links in a network.

Several approaches have been proposed which require only $O(E)$ partial (and aggregated) information. They include the SPI scheme (which uses ILP) in [5] and the two DPIM schemes in [1]. While all these approaches achieve a lower degree of BBW sharing than SCI and SR, as a price paid to maintain at least an order of magnitude less information, the DPIM schemes perform remarkably better than SPI with the same order of magnitude information.

Part I of our work describes the DPIM framework and the two schemes in detail [1]. It also specifies, among others, how the partial information is exchanged, how to establish or release a connection, and in particular, how to allocate or deallocate bandwidth through distributed signaling.

In this paper, which is Part II of a document, we describes a novel DPIM scheme, which uses a heuristic based on APF but with an intuitive potential backup cost function derived mathematically based on the statistical analysis of experimental data. Such a heuristic, named APF-PBC, combines the joint optimization capability of ILP and the ultra-fast speed of APF (which may be considered as a special case of APF-PBC where the potential cost is set to 0). In fact, it can achieve a better performance than an ILP based DPIM scheme described in [1] with the exactly same input (partial information). It can also be applied to the case with complete (per flow or aggregated) information, and achieve a better performance than SCI (also based on ILP).

The rest of the paper is organized as follows. Section II contains the notations to be used throughout the paper. Section III describes the main idea and motivation behind the proposed APF-PBC, and its application to existing schemes with complete or partial information. Section IV presents the performance evaluation model used, followed by numerical results of the comparison between the APF-PBC heuristic and ILP formulation as well as other APF heuristics. Sections V through VII describe in detail the APF-PBC heuristic, including its mathematical derivation, approximation and simplification, respectively, based on the analysis of experimental data. Section VIII concludes the paper.

II. NOTATIONS

In this section, we present the notations to be used throughout the paper. Most of these notations, except the last two, have also been defined and used in Part I of the

document [1] but have to be included here in order to describe previous work as well as the proposed scheme.

We consider a network G with E directed links (represented by set $\mathcal{E}$ and V nodes, which can be classified into two categories: *edge* nodes (ingress or egress), to which users or terminal devices are connected, and *core* nodes (which are nodes other than an edge node).

To facilitate our presentation, we will use a tuple $(s \rightarrow d, w)$ to represent a new request for connection establishment (or release), where s and d are the source (or ingress) and destination (or egress) of the connection, respectively, and w is the bandwidth (in units) requested by the connection.

The following additional notations will be used, where a calligraphic font style (e.g., $\mathcal{A}$ is used to denote a set or a vector while a non-calligraphic style (e.g., A) is used to denote a scalar value:

- $\mathcal{OUT}(n), \mathcal{IN}(n) \subset \mathcal{E}$: Set of links going from and coming into node $n \in \mathcal{V}$, respectively.
- $\mathcal{AP}$ and $\mathcal{BP}$: Set of links along an AP and BP, respectively.
- $\mathcal{A}_e$: Set of connections whose APs traverse link *e*.
- $A_e = \sum_{k \in \mathcal{A}_e} w_k$: Total (i.e., aggregated) ABW on link *e* dedicated to the connections in $\mathcal{A}_e$.
- $\mathcal{B}_e$: Set of connections whose BPs traverse link *e*.
- B_e: Total BBW allocated on link *e* for $\mathcal{B}_e$. Due to BBW sharing, $B_e < \sum_{k \in \mathcal{B}_e} w_k$.
- R_e: Residue bandwidth of link e. Its initial value is equal to the capacity of link e, C_e. $R_e = C_e - A_e - B_e$ (with only protected connections).
- $\mathcal{S}_a^b = \mathcal{A}_a \bigcap \mathcal{B}_b$: Set of connections whose APs traverse link *a* and whose BPs traverse link *b*, where $a, b \in \mathcal{E}$.
- $S_a^b = \sum_{k \in \mathcal{S}_a^b} w_k$: Total amount of bandwidth required by the connections in $\mathcal{S}_a^b$. It is a fraction of $\mathcal{A}_a$ as well as $\mathcal{B}_b$ that is used by the APs and BPs, respectively, of the connections in $\mathcal{S}_a^b$.
- BC_a^b: *Additional BBW* needed on link b in order to use it as a part of a BP for a new connection whose AP traverses link *a*. Its value depends on which BBW estimation method is used.

While most of the above notations are similar to those used in [5], the following notations are *specific* to the proposed DPIM schemes:

- BC_e: Estimated BBW needed on link e along a new BP. Assuming that its corresponding AP is known, $BC_e = \max_{\forall a \in \mathcal{AP}} B_a^e$. Whether this value is the minimum BBW needed on link b or not depends on which BBW estimation method is used to derive B_a^e. In addition, this is equal to the actual BBW allocated on link b in SCI, SR and SPI but not in the DPIM schemes (which may result in an overestimation but always allocates the minimal BBW).
- $\mathcal{P}_{\mathcal{B}}(e) = \{S_a^e | a \in \mathcal{E}\}$: *Profile* of BBW on a given link e. This is a *vector* consisting of a list of S_a^e values, one for each link a. Basically, it specifies the amount of BBW on link e that is used to protect against the failure of every other link (e.g., a_1, $a_2 \cdots a_E \in \mathcal{E}$) in the network.
- $P_{Be} = \max_{\forall a} S_a^e$: This is the maximum value over all the components in $\mathcal{P}_{\mathcal{B}}(e)$. It is also the minimum (or *necessary*) amount of BBW needed on link *e* to backup all active paths. If a BBW allocation scheme (such as the DPIM schemes to be described) always allocates minimum BBW on link e, then $B_e = P_{Be}$.
- $\mathcal{P}_{\mathcal{A}}(e) = \{S_e^b | b \in \mathcal{E}\}$: *Profile* of ABW on a given link e. This is a *vector* consisting of a list (or set) of S_e^b values, one for each link b. It complements $\mathcal{P}_{\mathcal{B}}(e)$, and specifies the amount of ABW on link e that is protected by every link (e.g., b_1, $b_2 \cdots b_E \in \mathcal{E}$) in the network.
- $P_{Ae} = \max_{\forall b} S_e^b$: This is the maximum value over all the components in $\mathcal{P}_{\mathcal{A}}(e)$. It is also the *sufficient* amount of bandwidth that needs be reserved on any link in the network in order to protect against the failure of link e.
- $\overline{P_{Ae}}$: This is the *average* value over all the components of a given ABW profile on link e. It is only useful to describe the APF-PBC heuristic.
- $M = \max_{\forall a,b} S_a^b$: This is also equal to $\max_{\forall e} P_{Ae}$ or $\max_{\forall e} P_{Be}$.

III. APF-PBC and Its Extensions to Prior Solutions

In this section, we first describe the main idea behind APF-PBC, then briefly summarize recently proposed schemes for on-line shared path protection under distributed control, and finally describe their APF-PBC extensions, whenever applicable.

In the following discussion, we consider a request for connection establishment, $(s \rightarrow d, w)$.

A. Main Motivation and Idea of APF-PBC

As discussed in the Sec. I, a major challenge in achieving efficient shared path protection in an on-line case is that, while on each link used by an AP, a fixed amount of ABW (i.e., w units) is to be allocated, the amount of BBW to be allocated on each link along a BP depends on many factors including which links are used by the corresponding AP. This is why APF is not ideal as it does not consider (nor cares about) the *potential* cost along the BP yet to be chosen when selecting the AP. The *shortest pair of path* (or SPP) algorithm such as the one in [7] does not take possible BBW sharing into consideration either in that it

essentially assumes that the cost of each link on a BP is also equal to w. Therefore, such an SPP algorithm may be useful only when each edge node cannot obtain a better (i.e., more accurate) estimation of the additional BBW needed (than w).

On the other hand, on-line approaches based on ILP formulations guarantee minimal allocation of TBW for each request by optimize the joint selection of both AP and BP. However, they do not guarantee an optimal result for all requests that arrive over time. In addition, their computational complexity is usually too high to be scalable.

The main motivation for our work on APF-PBC is to overcome the disadvantages of the APF and ILP based approaches, while trying to combine the best of the two. More specifically, the path determination algorithms based on APF-PBC can run as fast as those based on APF. Yet, they can also take into consideration the BBW to be allocated when determining the AP as in ILP-based approaches. This is accomplished by assigning a *potential cost* to each link, based on which the AP is selected using a shortest-path algorithm.

In this paper, we will use

$$\beta_e(w) = c\frac{w \cdot P_{Ae}}{M} \tag{1}$$

as a function to estimate the potential cost of link e, where c is a constant between 0 and 1. This function β_e is derived mathematically based on the statistical analysis of experimental data, as to be described in later sections. Note that many other forms of the potential function derived from intuitions may be used. We have tested some of them and have found that they do not perform as well as that given above. Also, APF may be considered as a special case of APF-PBC where the value of $\beta_e(w)$ is always set to zero.

The main idea of an approach based on APF-PBC is as follows. Given a network G with some existing connections, any link e whose $R_e < w$ will be (logically) removed initially as such a link cannot be used by the AP. Each remaining link a will be assigned a cost which is equal to $w + \beta_a(w)$. A cheapest path is then found for use as the AP. Afterwards, the links along the AP are then removed, but the other links removed initially should be put back as they may have enough residual bandwidth for the corresponding BP yet to be chosen.

The BP is chosen next, also using a cheapest path algorithm but the outcome depends on how much information on existing APs and BPs the algorithm has. More information leads to more accurate estimation of the additional BBW (or backup cost) for each link, and accordingly a better selection of BP. It is worth noting that once a BP is chosen, the actual amount of additional BBW allocated on a link e along the BP *may not* be equal to the estimated amount at the time of determining the BP. In fact, the unique feature of the DPIM schemes is that, while an edge node will obtain a (slightly) over-estimated backup cost based only on the partial information it has, the actual amount to be allocated is always minimal [1].

We now turn to previous solutions and their APF-PBC extensions. We classify all the schemes into two main categories, one with complete information (per flow or aggregated) and the other with partial (and aggregated) information. In each category, either ILP or some heuristics such as APF and APF-PBC may be used.

B. With Complete Information

We first review two schemes requiring at least $O(E^2)$ complete information, with an emphasis on the main ideas behind path determination, before describing their APF-PBC extension.

The first, called SCI in [5], uses complete per flow information and an ILP formulation to determine a pair of paths. The second, called SR [6], uses complete and aggregated information and APF instead. Hereafter, these two will be named as SCI-I, where I is short for ILP, and SCI-A, where A is short for APF, respectively.

In SCI-I, a controller maintains, for every link $e \in \mathcal{E}$, both $\mathcal{A}_e$ and $\mathcal{B}_e$, and based on which, all other parameters described in Sec. II and in particular S_a^b for every link a and link b (i.e., all combinations) can be derived. In SCI-A, each edge node (a distributed controller) maintains, for every pair of links $a, b \in \mathcal{E}$, the complete and aggregated information on S_a^b (which turns out to be what SCI-I really needs).

In both schemes, if link b is a candidate link for the yet-to-be chosen BP, whose corresponding AP is already determined (i.e., $\mathcal{AP}$ is known), the additional BBW (or backup cost) to be allocated on link b can be calculated as

$$BC_b = \max\{\max_{\forall a \in \mathcal{AP}}(S_a^b + w - B_b), 0\} \tag{2}$$

In SCI-I, a pair of paths (whose links have sufficient residue bandwidth) with a minimal TBW is determined using ILP. Note that since BC_b is the *minimum* amount of additional BBW needed on link b, it is also the actual amount of additional BBW to be allocated to link b in SCI-I.

In SCI-A, an AP (with a minimal number of links whose residual bandwidth is larger than w) is found first using a shortest path algorithm. This is equivalent to APF-PBC with $\beta_a(w) = 0$. Then, the links used by the AP is removed, and each remaining link b is assigned a cost equal to BC_b given above. Thereafter, each link b with

$R_b < BC_b$ is removed and a cheapest path found next is used as the BP. As in SCI-I, the actual amount of additional BBW to be allocated to link b is also equal to BC_b (the minimum) in SCI-A.

We now describe a scheme, to be called SCI-P where P is short for PBC, which is basically an APF-PBC extension of SCI-I (and SCI-A). The proposed SCI-P scheme requires exactly the same information to be maintained as SCI-A. The only difference between SCI-P and SCI-A is that the former uses APF-PBC (with $\beta_a(w)$ given in Eq. 1). A surprising result is that SCI-P can perform better than SCI-I.

A scheme similar to SR but is mainly for link-based restoration was described in [8], which could also use the proposed APF-PBC heuristic to find a better AP.

As mentioned earlier, the main deficiency of these schemes is the huge amount of information they require to maintain.

C. With Partial Information

There are two sub-categories, one containing schemes based on SPI [5], and the other containing schemes based on DPIM [1]. All these schemes use $O(E)$ partial information but those based on DPIM perform much better than those based on SPI.

C.1 SPI Based Schemes

In SPI, also to be named SPI-I for consistency, only the values of A_e and B_e (in addition to R_e) for every link e are maintained by a controller (at each edge node). Its ILP formulation is the same as that used by SCI-I except that Eq. 2 is replaced by

$$BC_b = \max\{\max_{\forall a \in \mathcal{AP}} (A_a + w - B_b), 0\} \qquad (3)$$

Note that, this can clearly be an over-estimation. Unfortunately, without any further information, the additional amount of BBW so overly estimated will also be the *actual* amount of BBW to be allocated to link b.

One can apply APF and APF-PBC to SPI and obtain schemes to be named as SPI-A and SPI-P, respectively. SPI-A and SPI-P will be the same as their SCI counterparts except in terms of how additional BBW needed is estimated and allocated. In addition, since the information on P_{Ae} and M are not available in SPI-P, one has to replace them with A_e and $\max_{\forall e \in \mathcal{E}} A_e$ respectively, in Eq. 1 in order to calculate the potential cost.

Our results, though not shown in this paper, indicate that SPI-P can also outperform SPI-I. In addition, all SPI based schemes performs significantly worse than their DPIM counterparts, which agree with the results shown in [1]. Later, we will show the performance of SPI-I only for comparison.

C.2 DPIM-based Schemes

In one DPIM scheme described in [1], to be named DPIM-SAM-I, every node (i.e., edge or core) maintains the following two vectors and four scalars for each local outgoing link $e \in \mathcal{OUT}(n)$: $\mathcal{P}_\mathcal{A}(e)$, $\mathcal{P}_\mathcal{B}(e)$, and B_e, R_e, P_{Ae}, and P_{Be}. The last two scalars are for convenience only as they can be derived from the two vectors. The total amount of information maintained at each node is practically limited to $O(E)$ as the number of local links at each node is typically bounded by a small constant.

An edge (ingress) node will also maintain the three scalars, B_e, R_e, and P_{Ae} for each *remote* (i.e., non-local) link e. The last one is useful for obtaining a good estimate of the additional BBW needed on the remote link e. It is also used to determine M. Hence, even if every node is an edge node, the amount of information to be maintained at each node is still limited to $O(E)$. How the local information is updated and remote/non-local information is exchanged has been discussed in [1].

DPIM-SAM-I uses an ILP formulation similar to that described for SPI in [5] with several improvements. For example, a more accurate backup cost estimation than that given by Eq. 3 can be obtained as follows:

$$BC_b = \max\{\max_{\forall a \in \mathcal{AP}} \min\{(P_{Aa} + w - B_b), w\}, 0\} \quad (4)$$

Another major improvement made in DPIM-SAM-I is to allocate only minimal BBW on each link along a chosen BP. More specifically, a signaling packet to reserve BBW along the BP will contain $\mathcal{AP}$, and based on which, each node along the BP updates their locally maintained information on the profiles of BBW (i.e., $\mathcal{P}_{\mathcal{B}e}$). Thereafter, each node calculates $bw = \max\{P_{Be} - B_e, 0\}$, which is the minimal BBW to be allocated on link e (see [1] for detailed explanations).

Based on DPIM-SAM-I, one can also apply APF and APF-PBC to obtained two extended schemes, to be named DPIM-SAM-A and DPIM-SAM-P, respectively. These two schemes will work in the same way as their SCI counterparts, except that each DPIM-SAM based scheme may select a different BP than that selected by its SCI counterpart (due to different estimations of the back up cost).

Another DPIM scheme described in [1], called DPIM-M-A, maintains only R_e for any remote link e. As a result, it cannot take advantage of APF-PBC (but can use, and indeed uses APF as a special case of APF-PBC). This is also true in several other distributed schemes with partial information described in [9], [2], [10]. where only R_e

for any remote link is available at the edge. According to our results obtained from this study (to be shown next), and those from [1], both DPIM-SAM-P and its special instance, DPIM-SAM-A, outperform DPIM-M-A. Since they only require the edge nodes to maintain two more scalars (namely, B_e and P_{Ae}) for each remote link e than DPIM-M-A does, the cost-effectiveness of the proposed APF-PBC (and its special case APF) under the DPIM framework is quite obvious.

IV. Performance Evaluation

We are primarily interested in comparing the performance of the schemes based on APF-PBC, namely SCI-P, and DPIM-SAM-P, with their counterparts based on ILP, namely SCI-I and DPIM-SAM-I, as well as those based on APF, namely, SCI-A (=SR) and DPIM-SAM-A. Processing and signaling overheads are ignored in this quantitative comparison study. When simulating APF-PBC based schemes, c in Eq. 1 is set to 0.5.

In the rest of the section, we describe the network topology assumed, traffic types considered, and performance metrics used before presenting the results.

A. Network Topology

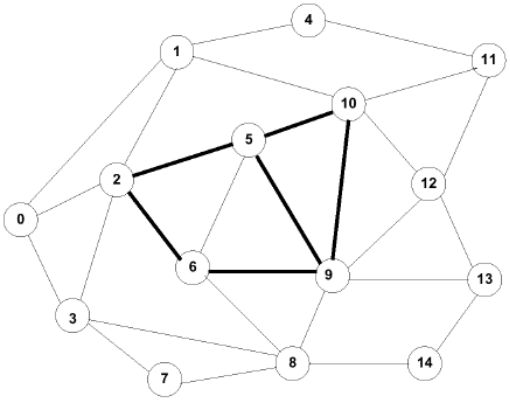

Fig. 1. A 15-node network

To facilitate a fair comparison between various approaches, we consider the topology shown in Fig. 1, which is the same as that used in [5] and has 15 nodes and 28 bi-directed edges (for a total of 56 links). The capacity of each link is assumed to be either infinite or limited as to be discussed in the next subsection. Another network with 70 nodes and 264 links is also considered, and relatively consistent performance results are obtained.

B. Traffic Types

We consider two types of traffic, one in which an established connection lasts forever (i.e., incremental traffic) as in [5], [6], and the other in which it may terminate after a certain duration (i.e., dynamic traffic).

In both cases, the ingress and egress of a connection establishment request is evenly distributed among all nodes, and requests arrive in an on-line fashion. For the case with incremental traffic, the bandwidth required by the connections is uniformly distributed between 1 and 10 units as in [5]. Note that, any request arrival process may be assumed.

For the case with dynamic traffic, the bandwidth required varies from 1, 2, 3, 4, 6 and 12 units with probability being 20%, 10%, 30%, 10%, 10%, 20%, respectively. In addition, requests are assumed to arrive according to a Poisson process, and the connection duration has a Pareto distribution. This is just an attempt to model realistic traffic (which may be self-similar and whose bandwidth requirements range from OC-1 to OC-12). Other possibilities, including uniformly distributed bandwidth requirements and exponentially distributed connection durations, have also been examined, and we have found that they have no significant impact on the performance of various schemes studied in this paper.

C. Performance Metrics

The following two performance metrics are used, one for each traffic type considered.

C.1 Bandwidth Saving (Ratio)

To obtain this metric, it is assumed that the capacity of each link is infinite (and hence all requests will be satisfied), and the traffic is incremental. After an appreciable number of requests have been satisfied, TBW consumed (i.e., sum of ABW and BBW on APs and BPs, respectively) for each of the schemes is evaluated and consequently, bandwidth saving, in terms of TBW consumption ratio over the NS scheme, is determined as similarly done in [5]. Note that, for a given request, the BBW needed will be no less than the ABW needed in NS. Hence, even if an ideal scheme that achieves maximum BBW sharing is used, the bandwidth saving ratio will be upper-bounded by 50% (achievable only if no BBW is needed at all).

C.2 Total Earning (Ratio)

The bandwidth saving measure may not mean much since in a practical case, all links have a finite capacity and thus not all requests can be satisfied.

Accordingly, in this set of experiments (simulation), we assume that each link has a finite capacity and dynamic traffic is considered. For example, in the Fig. 1 above, each dark (bold) link (consisting of two unidirectional links) is assumed to have a capacity of 192 units in each direction (to model an OC-192 link), and each of the other

links has a capacity of 48 units in each direction (to model an OC-48 link). As a result, some requests will be rejected under a heavy traffic load.

The total number of rejected connection establishment requests (after an initial set of requests are satisfied) using each scheme has been used as a performance measure (e.g., in [5]). However, comparison between different schemes based on such a measure (or equivalently blocking probability) may not be fair as it does not differentiate one request from another (see related discussions in [1]).

This motivates us to use the total earning (or revenue) as a metric. To this end, a scheme-independent *Earning Rate* matrix for the entire network is used. An entry at (i, j) represents earnings per bandwidth unit and time unit by a connection from ingress node i to egress node j. The earnings from a connection from i to j is thus the product of the earning rate, requested units of bandwidth, and the connection duration.

In this study, for lack of a better alternative, the earning rate is based on the cost of using the cheapest (or shortest) pair of AP and BP in the network from i to j (assuming there were infinite capacity in the network), and hence is independent of the current load in the network [1]. An important and desirable consequence of using the assumed earning rate (along with the earnings from a connection) is that it tends to discourage an algorithm that tries to maximize earnings from choosing an unnecessarily expensive (or long) path to establish the connection. Because choosing an expensive/long path under such a model may prevent other (future) connections from being established and thus resulting in lost revenues.

We compare the total earnings of each scheme and in particular, the improvement ratio over the NS scheme.

D. Simulation Results

Fig. 2 shows the total bandwidth consumed after satisfying 200 connection establishment requests (or demands) in the 15-node network from 10 experiments (simulation runs). It can be seen from the figure that, given the same $O(E)$ partial information, DPIM-SAM-P consistently outperforms DPIM-SAM-I (which in turn consistently outperforms SPI-I). In addition, SCI-P also outperforms SCI-I slightly. The difference between SCI-P and DPIM-SAM-P may not be big enough to warrant the additional overhead involved in maintaining $O(E^2)$ complete information as required by SCI-P.

Table I shows the average bandwidth saving ratio (vs. NS) (over the 10 experiments) in the 15-node and 70-node networks. For each network, the first row is for schemes using ILP, and the second and third rows are for their corresponding schemes using APF and APF-PBC, respectively.

[1] If the earning rate is load-dependent, it will become scheme-dependent also

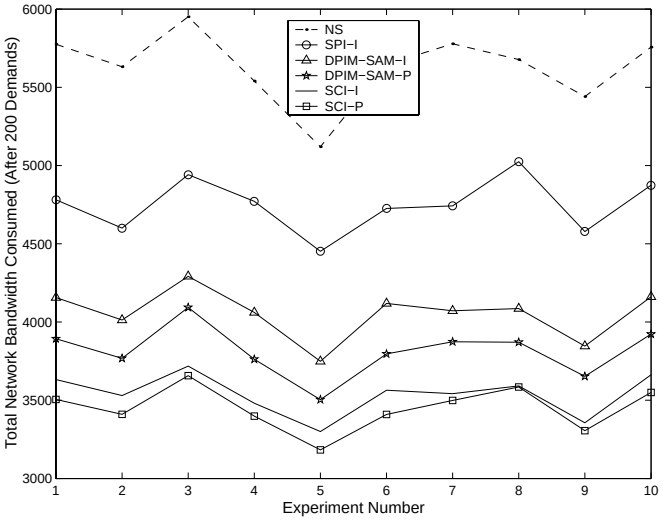

Fig. 2. Bandwidth consumed (from 10 experiments)

TABLE I
AVERAGE BANDWIDTH SAVING RATIO

15-node network	
37.2%(SCI-I)	28.0%(DPIM-SAM-I)
36.4%(SCI-A)	29.3%(DPIM-SAM-A)
38.7%(SCI-P)	32.1%(DPIM-SAM-P)
70-node network	
35.5%(SCI-I)	26.4%(DPIM-SAM-I)
34.3%(SCI-A)	27.0%(DPIM-SAM-A)
36.4%(SCI-P)	29.7%(DPIM-SAM-P)

An interesting observation is that the proposed DPIM-SAM-A also performs slightly better than DPIM-SAM-I, while SCI-A performs slightly worse than SCI-I.

In order to evaluate the performance of various schemes in networks with limited link capacity and dynamic traffic, another 10 experiments have been conducted in which each network is loaded with heavy (dynamic) traffic (under a light load, the differences between various schemes are insignificant).

Fig. 3 shows the total earnings after processing 500 demands (not all 500 requests are satisfied).

An interesting observation is that while the SCI and DPIM based schemes perform well, SPI-I performed poorly and in fact, worse than NS in some experiments. This is because BC_b given by Eq. 3 may be larger than w (which is needed by NS), that is, SPI-I may overly estimate the BBW needed, and allocate excessive BBW. This is especially problematic in a network with limited capacity as

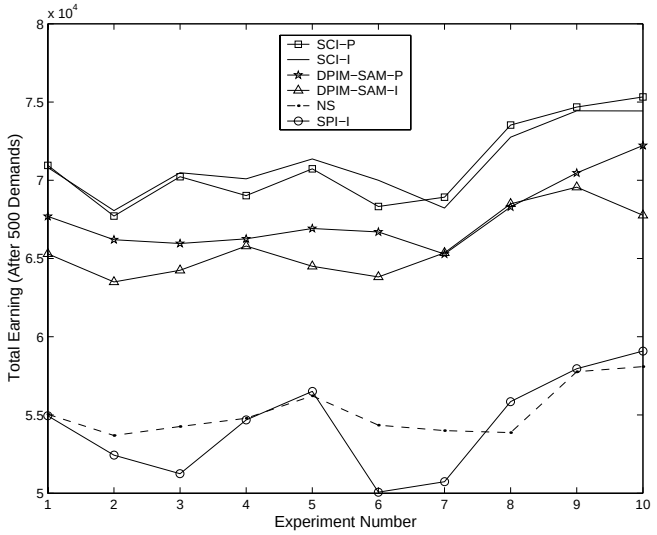

Fig. 3. Total earning after 500 demands (from 10 experiments)

many connection requests that could have been satisfied even in NS will now be rejected.

Table II shows the average total earning ratio (vs. NS) for SCI and DPIM based schemes. These results also show that APF-PBC outperform their ILP counterparts. An interesting observation is that even the APF schemes perform very well and in fact the differences between APF and APF-PBC are not much. Also, these results suggest that the trade-offs between additional overhead and performance gains (for dynamic traffic) may be worthwhile in a large network with limited capacity and heavy load.

TABLE II
AVERAGE TOTAL EARNING RATIO

15-node network	
28.7%(SCI-I)	19.3%(DPIM-SAM-I)
27.6%(SCI-A)	21.1%(DPIM-SAM-A)
29.6%(SCI-P)	23.1%(DPIM-SAM-P)
70-node network	
30.1%(SCI-I)	12.3%(DPIM-SAM-I)
31.4%(SCI-A)	17.6%(DPIM-SAM-A)
31.2%(SCI-P)	17.1%(DPIM-SAM-P)

V. POTENTIAL BACKUP COST - DERIVATION

In this section, we will describe how the PBC function $\beta_a(w)$ is derived based on the statistical analysis of experimental data. All experimental data is based on the traces collected from the simulation runs in which SCI-I is evaluated for the 15-node network shown in Fig. 1 assuming infinite link capacity (see Sec. IV for additional assumptions made in those experiments). The basic idea behind choosing the traces collected from SCI-I is that we want to mimic this almost "ideal" approach.

In APF-PBC, a major challenge is that when trying to determine PBC for link a, the ingress node responsible for path determination does not even know which link has been or will be used by BPs to protect against the failure of link a. If link b were to be used by a new BP (whose corresponding AP is going to use link a), and the amount of the BBW already allocated to link b (which is B_b) were known to be x, the additional BBW needed on link b (or backup cost) is

$$BC_a^b = \max\{s + w - x, 0\} \tag{5}$$

where $s = S_a^b$, which should be no greater than x.

The first step towards "guessing" the potential backup cost is to assume that s is known for the time being (this assumption will be relaxed later), and proceed to calculate an "expected" backup cost using a *weighted average* over all possible values of x (i.e., B_b of all $b \in \mathcal{E}$). More specifically, we will treat $\frac{x}{M}$, for an arbitrary link b, (where $M = \max_{\forall b} B_b$), as a random variable U between 0 and 1 whose cumulative distribution function (CDF) $F(U)$, obtained experimentally, is illustrated in Fig. 4. As can be seen, $F(U = \frac{B_b}{M})$ can be approximated by a normal distribution function $N(\mu, \delta)$, where μ is the mean value, and δ is its standard deviation. Let its corresponding probability density function be $f(U)$.

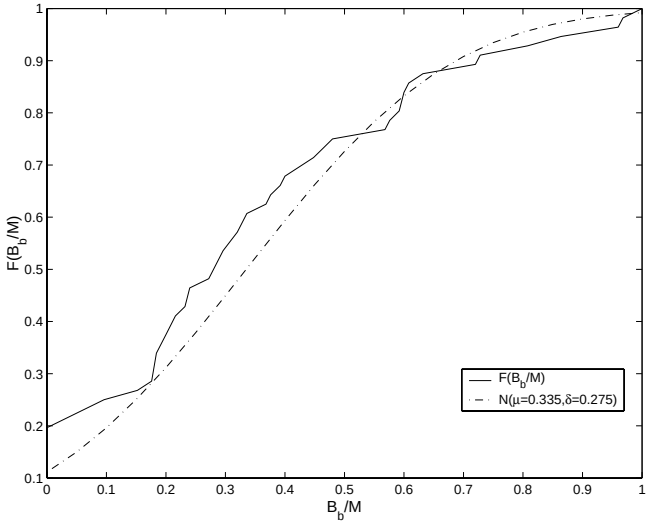

Fig. 4. Distribution of $\frac{B_b}{M}$

Based on Eq. 5 and the definition of a density function, the expected backup cost of using any link b by the BP to back up an AP using link a, provided that the value of S_a^b is equal to s, becomes

$$\zeta(w,s,M) = \int_0^{\min(\frac{s+w}{M},1)} (s+w-x)\frac{f(\frac{x}{M})}{P(M\geq y\geq s)}d(\frac{x}{M}) \quad (6)$$

where $\min(\frac{s+w}{M},1)$ is used as the upper bound for $\frac{x}{M}$ due to the fact that $BC_a^b = 0$ when $x \geq s+w$. In addition, $P(M \geq y \geq s)$, which is equal to $F(1) - F(\frac{s}{M})$, denotes the probability that a variable y (representing the BBW on a link) is between s and M, and thus needs be included since only those links whose $M \geq y \geq s$ are valid.

Note that $\zeta(w,s,M)$ is in fact independent of any particular link. The solid lines in Fig. 5 shows a typical graph for $\zeta(w,s,M)$, where the horizontal and vertical axis are normalized to M and w, respectively. The value of $\zeta(w,s,M)$ for a given w, s and M is obtained by using the adaptive Lobatto quadrature [11] method to evaluate the integral function in Eq. 6, and those values form the curves shown in Fig. 5. In Fig. 5, M is chosen to be 125 units, which is a typical value after a large number (e.g. 500) of connections have been established in each of the several experiments conducted.

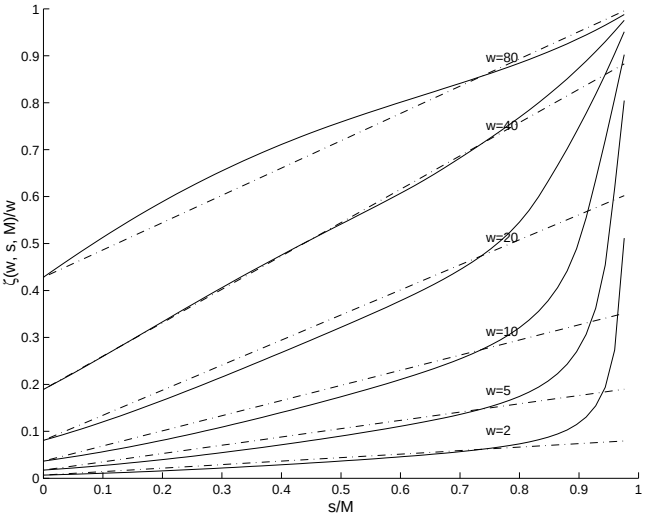

Fig. 5. Graph of $\zeta(w,s,M)$

The second step towards guessing the potential backup cost is to relax the assumption that s is known by calculating a weighted average value of $\zeta(w,s,M)$ over all possible values of s on link a. More specifically, let us treat $\frac{s}{P_{Aa}}$ as a random variable V between 0 and 1 with a CDF $G(V)$. Fig. 6 shows $G(V = \frac{s}{P_{Aa}})$, obtained experimentally, for five randomly selected links in the 15-node network (specifically, links $0 \to 1, 2 \to 6, 8 \to 3, 9 \to 5, 9 \to 12$).

From Fig. 6, it seems that the CDF $G(V)$ may be approximated by an exponential function (which matches our intuition/expectation), but such an approximation is not necessary to derive the potential cost function and hence will not be made.

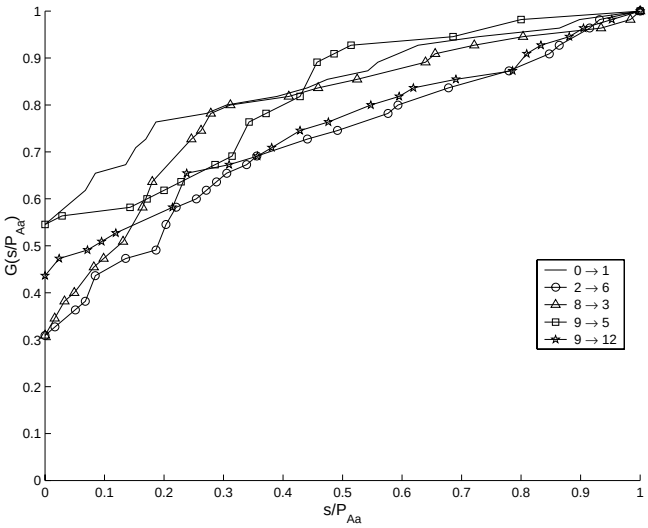

Fig. 6. Cumulative distribution function of $\frac{s}{P_{A\,a}}$

Let the corresponding density function be $g(V)$ (whose actual distribution or its approximation is not important). Based on the above discussion, the potential backup cost is:

$$\beta_a(w) = \int_0^1 \zeta(w,s,M) g(\frac{s}{P_{Aa}}) d(\frac{s}{P_{Aa}}) \quad (7)$$

VI. Potential Backup Cost - Approximation

In this section, we will simplify Eq. 7 with several reasonable approximations.

First, from the dashed lines in Fig. 5, it seems that each curve (or rather the points on each curve), which corresponds to a given value of w can be approximated (or fit) by a line of the form $Y = c_1 \cdot X + c_2$, where $Y = \frac{\zeta(w,s,M)}{w}$, $X = \frac{s}{M}$ (which happens to be the same as x defined earlier), and the values of c_1 and c_2 depend on the given w and M (as to be discussed later). Although such an line-fitting approximation is good only up to the point where X reaches about 0.75, it will not have much affect on the value of $\beta_a(w)$ given by Eq. 7 above. This is because as can be seen from Fig. 6, the probability that $\frac{s}{P_{Aa}}$ is larger than 0.75 is already very small. Moreover, since $P_{Aa} \leq M$, the probability that $\frac{s}{M}$ is larger than 0.75 is even smaller, and in fact, almost negligible.

Furthermore, as can be seen from Fig. 5, we may fit the group of curves with lines as follows. First, the slopes of these fitting lines, to be denoted by $\Phi(w,M)$ (instead of c_1 to more clearly indicate their dependency on w and M), may be obtained as $\frac{dY}{dX}$ at a reasonable value of X (or $\frac{s}{M}$). Fig. 7 shows the values of $\Phi(w,M)$ (obtained when

$\frac{s}{M} = 0.75$) as a function of w normalized with M.

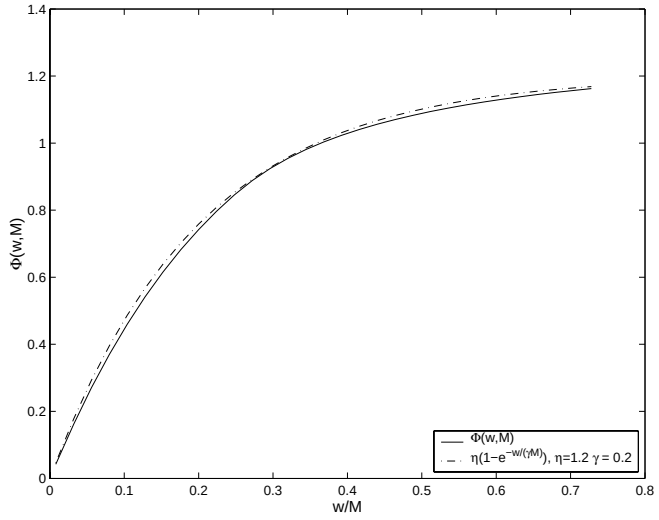

Fig. 7. Graph of $\Phi(w, M)$

It is clear from Fig. 7 that $\Phi(w, M)$ can be simply approximated by an exponential function $\eta(1 - e^{-\frac{w}{\gamma M}})$, where $\eta = 1.2$ and $\gamma = 0.2$.

In addition, as can be seen from Fig. 5, $c_2 \approx 0.2$ if $w = 40$, and $c_2 \approx 0.1$ if $w = 20$ and so on. In other words, $c_2 \approx 0.05w$. But since in all these cases, $M = 125$, we may set $c_2 = \frac{\theta w}{M}$, where $\theta \approx 0.6$.

In short, we have the following approximation for $\zeta(w, s, M)$:

$$\zeta(w, s, M) \approx \Phi(w, M) \cdot \frac{ws}{M} + \theta \cdot \frac{w^2}{M} \qquad (8)$$

(where the right hand and left hand sides of Eq. 8, normalized with w, are plotted using the curves and lines, respectively, in Fig. 5).

We now plug in this approximation for $\zeta(w, s, M)$ into Eq. 7. Since we have:

$$\int_0^1 s \cdot g(\frac{s}{P_{Aa}}) d(\frac{s}{P_{Aa}}) = \int_0^\infty s \cdot g(\frac{s}{P_{Aa}}) d(\frac{s}{P_{Aa}}) = \overline{P_{Aa}}$$

(this is because $\frac{s}{P_{Aa}} \leq 1$ and the expected value of s on link a, is, by definition, $\overline{P_{Aa}}$), and in addition

$$\int_0^1 g(\frac{s}{P_{Aa}}) d(\frac{s}{P_{Aa}}) = 1$$

,

we have the following approximation for $\beta_a(w)$:

$$\beta_a(w) \approx \frac{\Phi(w, M) \cdot w \cdot \overline{P_{Aa}}}{M} + \frac{\theta \cdot w^2}{M} \qquad (9)$$

VII. Potential Back Cost - Simplification

While one may use the potential cost function given in Eq. 9, it can be further simplified without affecting its usefulness or the performance of a scheme that adopts it.

First, we note that if Eq. 9 is used, each edge node needs to maintain $\overline{P_{Aa}}$ which is not needed in the DPIM-SAM-I nor in DPIM-SAM-A. Every time an edge node satisfies a connection establishment (or release) request whose AP uses link a, it can simply increase (or decrease) $\overline{P_{Aa}}$ by $\frac{wq}{E}$, where q is the number of links along the chosen BP, and multicast the updated value to all other edge nodes.

To avoid the need to maintain $\overline{P_{Aa}}$ at the edge nodes, one may replace $\overline{P_{Aa}}$ with $\mu_a \cdot P_{Aa}$, where $\mu_a = \frac{\overline{P_{Aa}}}{P_{Aa}}$ is almost a constant as can be seen from its CDF shown in Figure 8, which is obtained experimentally. From the figure, it is clear that the CDF can be fit by a curve with a normal distribution (whose mean is around 0.18 and whose standard deviation is only 0.043).

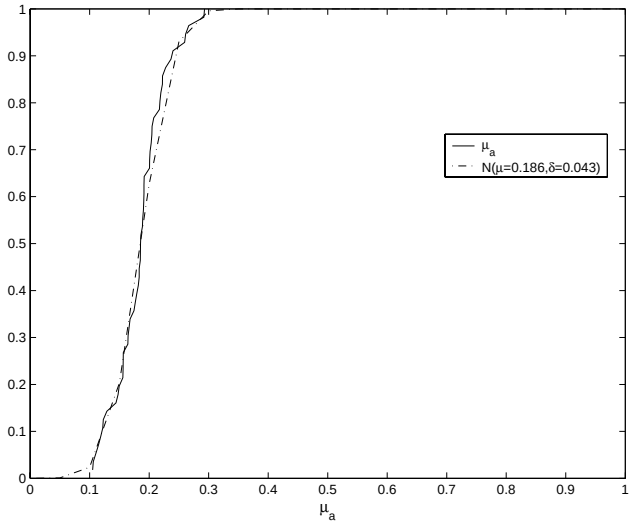

Fig. 8. Distribution of μ_a

To further simplify the PBC function, one may omit the second term in Eq. 9, that is, $\frac{\theta \cdot w^2}{M}$, as this term is not only small, but most importantly, independent of all links as well (as to be shown, this omission will not affect the usefulness of the PBC).

Finally, we note that for any request with $w > 0$, the value of $\Phi(w, M)$ is larger than 0 but no greater than 1.2 (see Fig. 7). In particular, it is independent of the link for which a PBC is being estimated. Although it cannot be eliminated as it is a part of the first term in Eq. 9 which is link dependent, the results shown in Fig 9, for example, indicate that any reasonable value of $\mu_a \cdot \Phi(w, M)$ between 0 and 1 may be used to estimate the PBC without any significant impact on the performance of the APF-PBC heuristic.

In other words, we may replace $\mu_a \cdot \Phi(w, M)$ with a constant c to arrive at Eq. 1, which is rewritten here as

$$\beta_a(w) = c \cdot \frac{w \cdot P_{Aa}}{M} \tag{10}$$

As can be seen from Fig 9 which shows the bandwidth saving ratio of DPIM-SAM-P over NS in the 15-node network, the value of c does not matter much as long as $c \leq 1$. The figure also shows that with or without the second term in Eq. 9, (i.e., with $\theta = 0.6$ or $\theta = 0$), the performance of DPIM-SAM-P is pretty much the same.

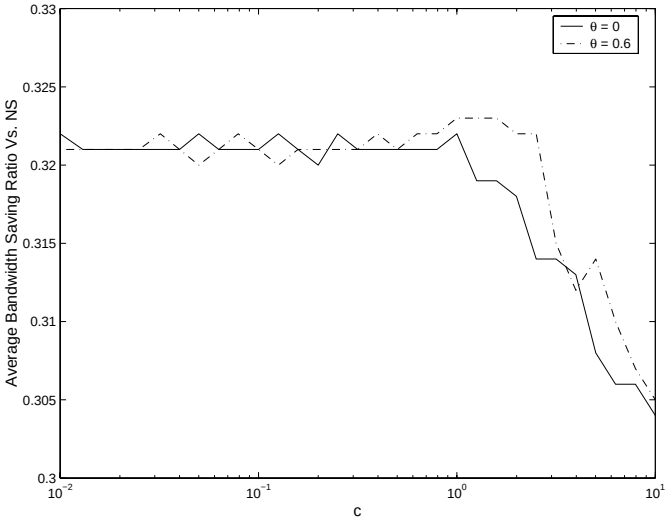

Fig. 9. The effect of constants c and θ on the performance of APF-PBC

Note that such a PBC function, though derived mathematically, is quite intuitive as the larger the w, the higher the PBC. In addition, for a given w, the larger the P_{Aa}, the more likely that a larger amount of additional BBW needs be allocated on BP (and hence a larger PBC).

VIII. Conclusion

In this paper, we have proposed a novel heuristic called *APF-PBC* to determine a pair of active and backup paths for each on-line request to establish a connection using shared path protection. Its basic idea is to attach a potential backup cost (PBC) to each link so that one may select an active path first (APF), while still taking into consideration the impact of bandwidth sharing along a yet-to-be-chosen backup path.

We have mathematically derived an intuitive potential backup cost (PBC) function based on the statistical analysis of experimental data, which can be used to minimize the total bandwidth or maximize the total earnings. Although the APF-PBC heuristic is developed primarily for the distributed partial information management (DPIM) framework, it can also be applied to other schemes with distributed or centralized routing. In fact, its basic idea may also be extended to other joint optimization problems.

One of the interesting results obtained from this study is that the proposed APF-PBC, of which APF is a special case, can not only run much faster than Integer Linear Programming (ILP) based approaches, but also *outperform* them. Our performance evaluation results have revealed that this is true in all the on-line cases we have considered in this paper, as long as APF-PBC and ILP have the same information (either complete or partial), traffic load (either incremental or dynamic) and constraints (such as no rearrangements of existing connections). We have explained the reason for this pleasantly surprising result, which should encourage people to look for more practical and efficient heuristics when tackling similar on-line optimization problems instead of simply resorting to ILP.

References

[1] Chunming Qiao and Dahai Xu, "Distributed partial information management (dpim) schemes for survivable networks - part i," in *submitted to INFOCOM 2002*, 2001.

[2] B. Doshi et. al, "optical network design and restoration," *Bell Labs Technical Journal*, pp. 58–84, Jan-Mar 1999.

[3] Yijun Xiong and Lorne G. Mason, "Restoration strategies and spare capacity requirements in self-healing atm networks," in *IEEE/ACM Trans. on Networking, vol. 7, no. 1*, 1999, pp. 98–110.

[4] Ramu Ramamurthy et. al, "Capacity performance of dynamic provisioning in optical networks," *Journal of Lightwave Technology*, vol. 19, no. 1, pp. 40–48, 2001.

[5] Murali Kodialam and T V. Lakshman, "Dynamic routing of bandwidth guaranteed tunnels with restoration," in *Proceedings - INFOCOM*, 2000, pp. 902–911.

[6] Yu Liu, D. Tipper, and P. Siripongwutikorn, "Approximating optimal spare capacity allocation by successive survivable routing"," in *Proceedings - INFOCOM*, 2001, pp. 699–708.

[7] J.W. Suurballe and R.E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, pp. 325–336., 1984.

[8] Ching-Fong Su and Xun Su, "An online distributed protection algorithm in wdm networks," in *Proceedings - ICC*, 2001.

[9] C. Dovrolis and P. Ramanathan, "Resource aggregation for fault tolerance in integrated service networks," in *ACM Computer Communication Review, vol. 28, no. 2*, 1998, pp. 39–53.

[10] Ramu Ramamurthy, Sudipta Sengupta, and Sid Chaudhuri, "Comparison of centralized and distributed provisioning of lightpaths in optical networks," in *Optical Fiber Communication Conference - OFC 2001*, 2001, pp. MH4–1.

[11] W. Gander and W. Gautschi, "Adaptive quadrature - revisited," in *BIT, Vol. 40*, 2000, This document is also available at http://www.inf.ethz.ch/personal/gander, pp. 84–101.