

Near-Optimal Approaches for Shared-Path Protection in WDM Mesh Networks

Canhui (Sam) Ou[†], Jing Zhang[†], Hui Zang[‡], Laxman H. Sahasrabudhe^{*}, and Biswanath Mukherjee[†]

[†]Dept. of Computer Science, Univ. of California, Davis, CA 95616, USA

Tel: +1.530.752.5129, Fax: +1.530.752.4767, Email: {ouc, zhangj, mukherje}@cs.ucdavis.edu

[‡]Sprint Advanced Technology Labs., Burlingame, CA 94010, USA. Email: hzang@sprintlabs.com

^{*}SBC Services, Inc., San Ramon, CA 94583, USA. Email: ls9526@sbc.com

Abstract— This paper investigates the problem of dynamic shared-path-protected lightpath provisioning in optical mesh networks employing wavelength-division multiplexing (WDM). We prove that the problem of finding an eligible pair of working and backup paths for a new lightpath request requiring shared-path protection under the current network state is NP-complete. We develop a heuristic, called CAFES, to compute a feasible solution and an algorithm, called OPT, to optimize resource consumption for a given solution. The merits of our approaches are that they capture the essence of shared-path protection and approach to optimal solutions without enumerating paths. We evaluate the effectiveness of our heuristics and the results are found to be promising.

Index Terms—Optical network, WDM, lightpath, provisioning, fault management, shared-path protection.

I. INTRODUCTION

In a wavelength-routed WDM network, the failure of a network element can cause the failure of several lightpaths, thereby leading to large data and revenue loss. Protection [1], [2], [3], a proactive procedure in which spare capacity is reserved during lightpath setup, can be employed to combat such failures. Protection schemes can be classified by the type of routing used (link-based versus path-based) and by the type of resource sharing (dedicated versus shared). A path that carries traffic during normal operation is known as a *working* path. When a working path fails, the lightpath is rerouted over a *backup* path.

We consider the problem of dynamic shared-path-protected lightpath provisioning against single-fiber failures¹. Under such a scenario, a network management system needs to compute two link-disjoint paths—a dedicated working lightpath and a shared backup lightpath—based on the current network state for an incoming lightpath request. We concentrate on computing link-disjoint paths for each incoming lightpath request with the assumptions that existing lightpaths cannot be disturbed and future arrivals are unknown. While we consider full wavelength-conversion networks here, the extension to the wavelength-continuous case is straightforward.

A significant amount of research has been conducted on dynamic lightpath provisioning with protection in WDM networks (e.g., [3], [4], [5], [6]) and in multiprotocol label switching

(MPLS) networks (e.g., [7], [8], [9]). (Please see [10] for an extensive overview.) Most existing approaches for computing two link-disjoint paths under shared-path constraints are mainly based on enumerating paths. In such approaches, for provisioning an incoming lightpath request, K minimum-cost paths l_w^j ($j = 1, 2, \dots, K$) are computed as working candidates; another K paths l_b^j ($j = 1, 2, \dots, K$; l_b^j is link disjoint to l_w^j) are computed as backup candidates; and then the link-disjoint paths l_w^j and l_b^j (if any) of minimum total cost is selected as the working and backup paths. When $K = 1$, this method becomes the well-known two-step approach, in which a minimum-cost path is first computed as the working path, and then a link-disjoint minimum-cost path is computed as the backup path.

In our present study, we prove that the problem of finding an eligible pair of working and backup paths under shared-path-protection constraints for a lightpath request with respect to existing lightpaths is NP-complete. We develop a backtracking-based heuristic, CAFES, to compute a feasible solution (i.e., two link-disjoint paths). We also design an algorithm, OPT, to optimize the resource consumption for a given solution. While our focus is on dynamic lightpath provisioning, we remark that the two approaches can be readily applied to static lightpath provisioning in which all the lightpaths are known a priori.

II. PROBLEM STATEMENT AND COMPLEXITY ANALYSIS

We first define the notations. A network is represented as a weighted, directed graph $G = (V, E, C, \lambda)$, where V is the set of nodes, E is the set of unidirectional fibers (referred to as links), $C : E \rightarrow R^+$ is the cost function for each link (where R^+ denotes the set of positive real numbers), and $\lambda : E \rightarrow Z^+$ specifies the number of wavelengths on each link (where Z^+ denotes the set of positive integers).

We use λ_f^e to denote the number of free wavelengths on link $e \in E$. We denote the set of existing lightpaths by $\mathcal{L} = \{(l_w^i, l_b^i, t_a^i, t_h^i)\}$, where the quadruple $(l_w^i, l_b^i, t_a^i, t_h^i)$ specifies the working path, the backup path, the arrival time, and the holding time, in order, for the i^{th} lightpath. We denote the current lightpath request by (l_w, l_b, t_a, t_h) . We represent the cost of l_w and l_b using $C_w(l_w)$ and $C_b(l_w, l_b)$, respectively.

We associate a conflict set with a link² to identify the sharing potential between backup lightpaths. The conflict set ν_e for

This work has been supported by NSF Grant No. ANI-98-05285 and Sprint Advanced Technology Labs.

¹We focus on single-fiber failures because they are the predominant form of failures in communication networks.

²In the wavelength-continuous case, we would associate a conflict set to a wavelength. The conflict set defined here is similar to the conflict vector in [3], the aggregated square matrix in [8], and the “bucket” link metric in [6], but

link e defines the set of links used by those working lightpaths whose backup lightpaths utilize wavelengths on link e . The conflict set ν_e for link e can be represented as an integer set, $\{\nu_e^{e'} \mid \forall e' \in E, 0 \leq \nu_e^{e'} \leq \lambda(e')\}$, where $\nu_e^{e'}$ specifies the number of working lightpaths that traverse link e' and are protected by link e (and their corresponding backup lightpaths traverse link e). The number of wavelengths reserved for backup lightpaths on link e is thus $\nu_e^* = \max_{\forall e'} \{\nu_e^{e'}\}$. Clearly, the union of the conflict sets for all the links aggregates the per-lightpath-based information, and the size of the conflict set depends only on the number of links, not on the number of lightpaths. In the absence of such a mechanism, per-lightpath-based information is necessary for identifying shareable backup channels [4]. It is, thus, advantageous to use conflict set since the number of lightpaths can be significantly more than the number of links.

The working and backup lightpaths l_w and l_b satisfy the *shared-path protection constraints* with respect to the existing lightpaths as follows:

- C.1 l_w and l_b are link disjoint.
- C.2 l_w and l_b^i , $1 \leq i \leq |\mathcal{L}|$, do not utilize the same wavelength on any common link they traverse.
- C.3 l_w does not share any wavelength with l_b^i , $1 \leq i \leq |\mathcal{L}|$, on any common link they traverse.
- C.4 l_b and l_b^i can share a wavelength on a common link only if l_w and l_w^i are link disjoint.

We now state the dynamic shared-path-protected lightpath-provisioning (DSPPLP) problem as follows: Given a WDM network as $G = (V, E, C, \lambda)$ and the set of existing lightpaths (or the associated conflict sets $\{\nu_e \mid e \in E\}$), route each incoming lightpath request under shared-path-protection constraints while minimizing the total cost of the working and backup paths. We formally state the decision version of the above DSPPLP problem and prove that it is NP-complete.

Instance: A graph $G = (V, E, C, \lambda)$, the set of existing lightpaths \mathcal{L} (or the set of conflict sets $\{\nu_e \mid e \in E\}$), and a lightpath request from s to d ($s, d \in V$).

Question: Do there exist from s to d two lightpaths, l_w and l_b , such that they satisfy the shared-path-protection constraints with respect to the existing lightpaths?

Theorem 1: DSPPLP is NP-complete.

Proof: Please see [11]. ■

III. HEURISTIC ALGORITHMS

As the existence version of DSPPLP is NP-complete, we resort to heuristics. In Section III-A, we design a backtracking-based heuristic, CAFES, to compute an eligible pair of working and backup paths for a lightpath request. In Section III-B, we develop a general optimization procedure, OPT, to jointly optimize the resource consumption of the working and backup paths for a given solution (i.e., two link-disjoint paths).

A. Compute A Feasible Solution (CAFES)

The two-step approach cannot find a solution in a trap topology [12] even though a solution exists. As the K shortest paths

it is more general in the sense that the conflict set can model wavelength-continuous networks, wavelength-convertible networks, and networks of sparse wavelength-conversion capability.

are likely to share some common links, enumerating paths (with finite K) may be susceptible to a trap topology as well. Furthermore, due to backup sharing, trap situations can arise in a non-trap topology. A possible drawback of enumerating paths is the lack of backtracking, i.e., the information gathered from enumerating the first i paths is not utilized in enumerating the $(i + 1)^{th}$ path. We analyze the characteristics of the two types of trap situations—trap topology and backup-sharing-caused trap—and propose a backtracking-based solution.

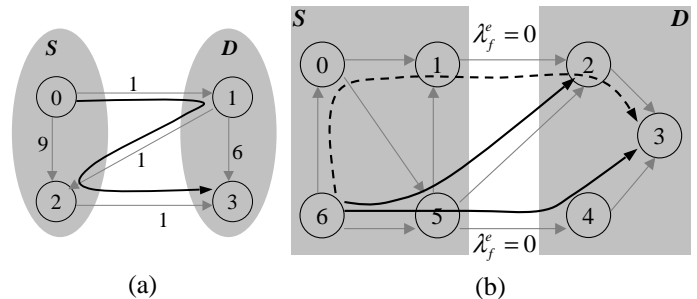


Fig. 1. Trap situations: (a) trap topology; (b) backup-sharing-caused trap. Solid gray lines represent links; solid black lines denote working paths; dashed black lines denote backup paths.

1) *Trap topology:* A two-step approach cannot find two link-disjoint paths from node 0 to node 3 in the network in Fig. 1a (even though they exist) because the graph is disconnected after the removal of the first minimum-cost path $\langle 0, 1, 2, 3 \rangle$.

We introduce backtracking based on network flow to overcome the trap situation. Let S be the set of nodes reachable from the source node after removing the links which are not link disjoint to the first minimal-cost path. Let D be the complement of S . (S, D) is referred to as a *cut*. We refer to a link as a *backhaul* link with respect to cut (S, D) if its source node is in D and its destination node is in S . For example, link $\langle 1, 2 \rangle$ in Fig. 1a is a backhaul link.

If the second minimal-cost path is not found and backhaul links exist, CAFES increases the cost of the backhaul links to some large value and restarts the two-step process. This way, the first minimal-cost path will avoid, if possible, these backhaul links, and the second minimal-cost path will have a chance to reach nodes in D . For example, if we increase the cost of the backhaul link $\langle 1, 2 \rangle$ to 1000 and recompute the first minimal-cost path, which turns out to be $\langle 0, 1, 3 \rangle$, we are able to compute a link-disjoint minimal-cost path $\langle 0, 2, 3 \rangle$.

2) *Backup-sharing-caused trap:* Consider the example network state in Fig. 1b. One existing lightpath with working path $\langle 6, 5, 4, 3 \rangle$ and backup path $\langle 6, 0, 1, 2, 3 \rangle$ is shown (other existing lightpaths are not shown). Suppose, to protect one more working path traversing link $\langle 6, 5 \rangle$, links $\langle 5, 4 \rangle$ and $\langle 1, 2 \rangle$ both need to allocate one more free wavelength (i.e., $\nu_{\langle 5,4 \rangle}^{(6,5)} = \nu_{\langle 5,4 \rangle}^*$ and $\nu_{\langle 1,2 \rangle}^{(6,5)} = \nu_{\langle 1,2 \rangle}^*$) and these two links have no free wavelength. Assume other links have free wavelengths and the cost of each link is unity. When a new lightpath request $6 \rightarrow 2$ comes, a two-step approach may compute path $\langle 6, 5, 2 \rangle$ as the working path. As a result, no backup can be found because no more link-disjoint flow can be pushed from S to D (please note:

$\langle 6, 5, 2 \rangle$ and $\langle 6, 5, 4, 3 \rangle$ are not link disjoint; $\nu_{\langle 5,4 \rangle}^{\langle 6,5 \rangle} = \nu_{\langle 5,4 \rangle}^*$; $\lambda_f^{\langle 5,4 \rangle} = 0$; $\nu_{\langle 1,2 \rangle}^{\langle 6,5 \rangle} = \nu_{\langle 1,2 \rangle}^*$; and $\lambda_f^{\langle 1,2 \rangle} = 0$.

Again, we use backtracking to overcome this situation. Define a link $\langle m, n \rangle$ as a *conflicting* link with respect to cut (S, D) if there exists link $\langle p, q \rangle$ ($p \in S, q \in D$) such that $\nu_{\langle p,q \rangle}^{\langle m,n \rangle} = \nu_{\langle p,q \rangle}^*$ and $\lambda_f^{\langle p,q \rangle} = 0$. For example, link $\langle 6, 5 \rangle$ in Fig. 1b is a conflicting link as $\nu_{\langle 5,4 \rangle}^{\langle 6,5 \rangle} = \nu_{\langle 5,4 \rangle}^*$ and $\lambda_f^{\langle 5,4 \rangle} = 0$.

If the second minimal-cost path is not found and conflicting links exist, CAFES increases the cost of the conflicting links to some large value and restarts the two-step process. For example, if we increase the cost of the conflicting link $\langle 6, 5 \rangle$ to 1000 and recompute the first minimal-cost (working) path, which turns out to be $\langle 6, 0, 5, 2 \rangle$, we are able to compute a link-disjoint minimal-cost backup path $\langle 6, 5, 1, 2 \rangle$ as it can share the wavelength-link $\langle 1, 2 \rangle$ with the existing backup $\langle 6, 0, 1, 2, 3 \rangle$.

In case there are chained trap situations, in which some traps do not appear until some others are processed, we can recursively apply this procedure. We introduce a parameter k to limit the number of recursions. The parameter k can be considered as the maximum number of trap situations we want to deal with.

A formal specification of our heuristic, CAFES, is in Algorithm 1. In the algorithm, ϵ is a small number, e.g., $10^{-4} \times C(e)$. The backup cost function C_1 is used to meet the shared-path-protection constraints C.1-C.4 (the first and last cases in C_1 's definition) and to increase backup sharing (the second case). The last case of C_1 's definition, $C(e) + \epsilon \cdot (\lambda(e) - \lambda_f^e)$, is used for load balancing: when there are two eligible backup paths of the same cost, the less loaded backup will be chosen.

Algorithm 1 CAFES

Input: $G = (V, E, C, \lambda)$, $\nu = \{\nu_e \mid e \in E\}$, $s, d \in V$, k

Output: Two paths l_p and l_b satisfying constraints C.1-C.4, or NULL if no such paths are found.

- 1) $l'_w \leftarrow \text{NULL}$,
- 2) compute a minimal-cost path l_w on G from node s to node d ; return NULL if l_w is not found or if $l'_w = l_w$,
- 3) compute a minimal-cost path l_b from node s to node d using cost function:

$$C_1(e) := \begin{cases} +\infty & \text{if } e \in l_w \vee (\lambda_f^e = 0 \wedge \\ & (\exists e' \in l_w, \nu_{e'}^e = \nu_{e'}^*)) \\ \epsilon & \text{if } \forall e' \in l_w, \nu_{e'}^e < \nu_{e'}^* \\ C(e) + \epsilon \cdot (\lambda(e) - \lambda_f^e) & \text{otherwise} \end{cases}$$

return $\langle l_w, l_b \rangle$ if l_b is found; return NULL if l_b is not found and $k = 0$,

- 4) compute the set of backhaul links L_b and the set of conflicting links L_c ,
 - 5) increase the cost of any link in L_b and L_c to some large value, and
 - 6) $k \leftarrow k - 1$, $l'_w \leftarrow l_w$; go to Step 2.
-

The computational complexity of CAFES is $O(k \times |E|^2)$. In particular, the complexities of Steps 1-6 are $O(1)$, $O(|V|^2)$, $O(|E|^2)$, $O(|E|^2)$, $O(|E|)$, and $O(1)$, respectively; Steps 2-6 repeat for at most $k + 1$ times.

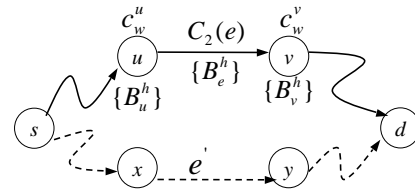


Fig. 2. Illustration of OPT (dashed line is the fixed backup).

B. Optimization (OPT)

Given a feasible solution l_w and l_b for a lightpath from node s to node d , we develop an algorithm, called OPT, to minimize the total cost of l_w and l_b , $C_w(l_w) + C_b(l_w, l_b)$. Similar to [2], OPT iteratively refines l_w and l_b ; different from [2], OPT jointly optimizes l_w and l_b . In one iteration, OPT first recomputes l_w with l_b fixed, and then it recomputes l_b with l_w fixed. The iteration continues as long as there are improvements. How to recompute l_b with l_w fixed is straightforward. We show how to recompute l_w with l_b fixed while jointly optimizing the cost of both. The challenge here is that *the cost of the backup path l_b changes as the working path l_w changes*.

The basic idea is to consider the changes in backup cost when recomputing the working path l_w . To recompute l_w , we use a standard shortest-path algorithm (such as Dijkstra's algorithm) with a modified *relaxation step* [13] outlined below.

Let H_b be the number of hops l_b has. For any link $e \in E$, associate a backup cost vector $B_e = \{B_e^h \mid 1 \leq h \leq H_b\}$. B_e^h denotes the cost of l_b 's h^{th} hop if e is used by l_w . B_e^h is defined as follows (e' denotes the h^{th} hop of l_b in the definition below):

$$B_e^h := \begin{cases} +\infty & \text{if } \nu_{e'}^e = \nu_{e'}^* \wedge \lambda_f^e = 0 \\ C(e') & \text{if } \nu_{e'}^e = \nu_{e'}^* \wedge \lambda_f^e > 0 \\ \epsilon & \text{otherwise} \end{cases}$$

The first case ($\nu_{e'}^e = \nu_{e'}^* \wedge \lambda_f^e = 0$) indicates that link e' needs to allocate a free wavelength to protect a new working path traversing link e . As there is no free wavelength on link e' , link e cannot be used by the working path l_w (l_b is fixed). The second case ($\nu_{e'}^e = \nu_{e'}^* \wedge \lambda_f^e > 0$) implies that link e' needs to allocate a free wavelength (and it is available) to protect a new working path traversing link e . The cost of link e' in this case is the original cost. The last case implies that link e' does not need to allocate any more free wavelength to protect one more working path traversing link e . Thus, link e' is used for "free" if link e is the working path. (Please refer to Fig. 2.)

We redefine the link-cost function for computing the working path as follows:

$$C_2(e) := \begin{cases} +\infty & \text{if } e \in l_b \vee \lambda_f^e = 0 \\ C(e) & \text{otherwise} \end{cases}$$

If l_b traverses link e or there is no free wavelength on link e , then the working path cannot utilize link e and its cost is infinite; otherwise, the cost of link e is its original cost.

For any node $v \in V$, associate a cost variable c_w^v and a backup cost vector $B_v = \{B_v^h \mid 1 \leq h \leq H_b\}$. c_w^v denotes the cost of the working path from s to v . $\sum_h B_v^h$ indicates the backup cost if the minimum-cost path from s to v is used

by the working path l_w . By definition, $C_w(l_w) = c_w^d$ and $C_b(l_w, l_b) = \sum_h B_d^h$.

We then employ a standard shortest-path algorithm to minimize $C_w(l_w) + C_b(l_w, l_b) = c_w^d + \sum_h B_d^h$ with the following relaxation step (please refer to Fig. 2):

```
RELAX( $u, v$ )
  LET  $B^h = \max\{B_u^h, B_e^h\}, 1 \leq h \leq H_b$ 
  IF  $c_w^u + C_2(e) + \sum_h B^h < c_w^v + \sum_h B_v^h$  THEN
     $c_w^v \leftarrow c_w^u + C_2(e)$ 
     $B_v^h \leftarrow B^h, 1 \leq h \leq H_b$ 
  SET NODE  $u$  AS NODE  $v$ 'S PREVIOUS HOP
```

The idea of the above relaxation step is to “relax” along the path which leads to the minimum total cost of the working and the backup paths. To decide whether to use node u as node v 's previous hop along the working path (if node v is traversed by the working path), the relaxation step compares the total cost of the working [$c_w^u + C_w(w)$ vs. c_w^v] and the backup ($\sum_h B^h$ vs. $\sum_h B_v^h$). Thus, the changes in backup cost when the working path changes are correctly captured.

A formal specification of OPT is in Algorithm 2. Clearly, if l_w or l_b hits optimum in one iteration, OPT will stop at the next iteration and output the joint optimal l_w and l_b .

Algorithm 2 OPT

Input: $G = (V, E, C_2, \lambda), \nu = \{\nu_e \mid e \in E\}, s, d \in V, l_w, l_b$
Output: optimized l_w and l_b .

- 1) compute B_e for $e \in E$ with l_b fixed,
- 2) $c_w^s \leftarrow 0, c_w^v \leftarrow +\infty (v \in V \wedge v \neq s);$
 $B_s^h \leftarrow 0, B_v^h \leftarrow +\infty (1 \leq h \leq H_b, v \in V \wedge v \neq s),$
- 3) compute a minimum-cost path as l'_w using a standard shortest-path algorithm with the new relaxation step,
- 4) compute a minimum-cost path as l'_b with l'_w fixed using cost function C_1 in Algorithm 1, and
- 5) if $C_w(l'_w) + C_b(l'_w, l'_b) < C_w(l_w) + C_b(l_w, l_b)$ then $l_w \leftarrow l'_w,$
 $l_b \leftarrow l'_b,$ go to Step 1; otherwise return l_w and l_b .

The computational complexity of OPT is $O(|E|^2)$. In particular, the complexities of Steps 1-5 are $O(|E|^2), O(1), O(|V|^2), O(|E|^2),$ and $O(1),$ respectively; and the number of iterations is typically a small constant for practical-sized networks.

IV. ILLUSTRATIVE NUMERICAL RESULTS

We now quantitatively evaluate our heuristic algorithms. We simulate a dynamic network environment with the assumptions that the lightpath-arrival process is Poisson and the lightpath-holding time follows a negative exponential distribution. For the illustrative results shown here, in every experiment, 10^6 lightpath requests are simulated; they are uniformly distributed among all node pairs; average lightpath-holding time is normalized to unity; the cost of any link is unity; and the topology with 16 wavelengths per fiber is shown in Fig. 3.

We compare CAFES to an effective two-step approach FIR [9] in terms of the percentage of unreachable blocking. We compare the blocking probability and average hop distance of OPT to those of FIR. The feasible solution to OPT is the solution of CAFES (with $k = 1$ since the performance improvement is marginal if we increase k to any larger value).

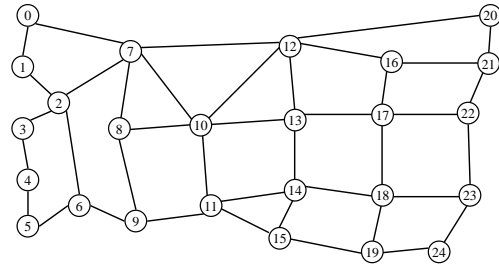


Fig. 3. An example network used in this study.

A. Percentage of Unreachable Blocking

The percentage of unreachable blocking (PUB) is defined as the percentage of blocking due to the fact that a pair of link-disjoint paths does not exist for a lightpath under the current network state³. Intuitively, this metric measures how “unlikely” it is that a lightpath request will be dropped when an eligible route exists. Due to the NP-completeness of the problem, there is no polynomial-time algorithm to compute this metric. Here, we use an approximation. The approximate PUB is defined as the percentage of blocking we are sure that no link-disjoint paths exist for a lightpath under the current network state. (Clearly, the approximate PUB is slightly less than the exact PUB.) For example, in FIR, this metric is the percentage of blocking due to working paths not found. In CAFES, for $k > 0$, it is the percentage of blocking that happens at Step 2 in Algorithm 1, i.e., the percentage of blocking due to working paths not found and due to the situation that the l_w in the $(k - 1)^{th}$ iteration is the same as the l_w in the k^{th} iteration. As shown in Fig. 4, with only one iteration ($k = 1$), more than 95% blocking is due to unreachability, compared to less than 55% in FIR. The reason FIR has such a low PUB is that there exists a trap situation in the topology: FIR cannot find two link-disjoint paths from node 9 to node 3. However, in a non-trap topology, our results (not shown here) also show that CAFES has significantly higher PUB compared to FIR, though the difference is not as drastic as here. If we increase k , then the PUB in CAFES will further approach towards 100%.

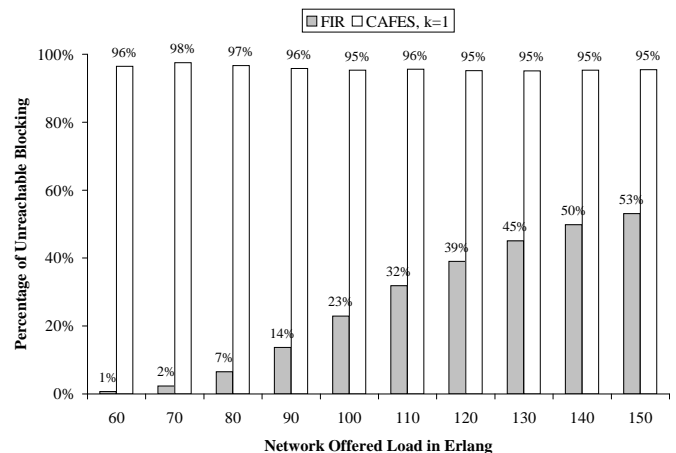


Fig. 4. Approximate percentage of unreachable blocking.

³The other type of blocking is that, due to the NP-completeness of the problem, a heuristic may not find two link-disjoint paths even though they exist.

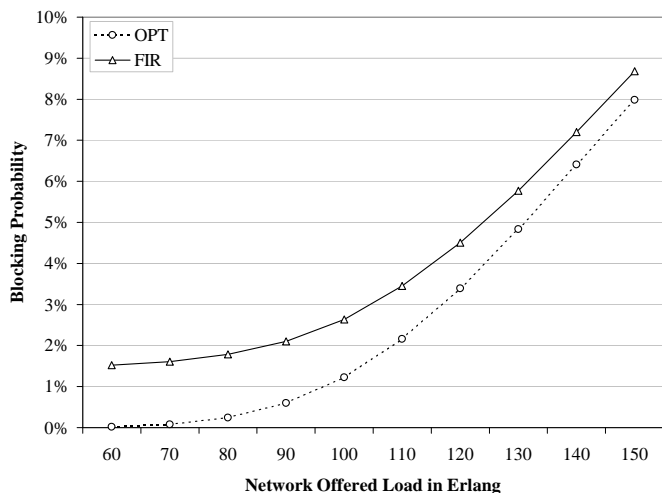


Fig. 5. Blocking probability vs. network offered load.

B. Blocking Probability

Figure 5 compares the blocking probability of OPT to FIR⁴. We observe that OPT has lower blocking probability. This is because of the backtracking in CAFES and the joint optimization in OPT. The reason that the difference between the two curves decreases as the network offered load increases is as follows. When the network offered load is modest or low, a lightpath is unlikely to be blocked because of resource limitation. Blocking happens mainly because a heuristic cannot find two eligible link-disjoint paths. Since FIR fails in trap situations while CAFES does not, FIR has a much higher blocking probability. When the network offered load is high, a lightpath is more likely to be blocked because there are no available wavelengths on some links. While some lightpaths still get blocked due to trap situations in FIR, other lightpaths (which might be blocked in CAFES) can utilize the resources these lightpaths are supposed to use (if they are accepted). Although CAFES can find a solution for a lightpath request with much higher probability than FIR, accepting a lightpath with a “detoured” route (as in the case when FIR cannot find a route while CAFES can) might interfere or even block future arrivals. Thus, the difference between the two curves decreases (but they will not cross each other because of the joint optimization in OPT). Further results (not shown here) from a non-trap topology show that our heuristic always has lower blocking probability as well.

C. Average Hop Distance

The advantages of our OPT algorithm come for a slight increase in the average backup-path hop distance. Figure 6 shows that both OPT and FIR have similar average working-path hop distance while OPT has slight longer average backup-path hop distance. The load balancing in CAFES and the joint optimization in OPT (which tends to maximize backup shareability) lead to the increase in backup hop distance.

⁴While we use CAFES to generate feasible solutions as input for OPT here, we can use any shared-path-protected routing heuristic, e.g., FIR. Thus, OPT is complementary to existing shared-path-protected routing approaches.

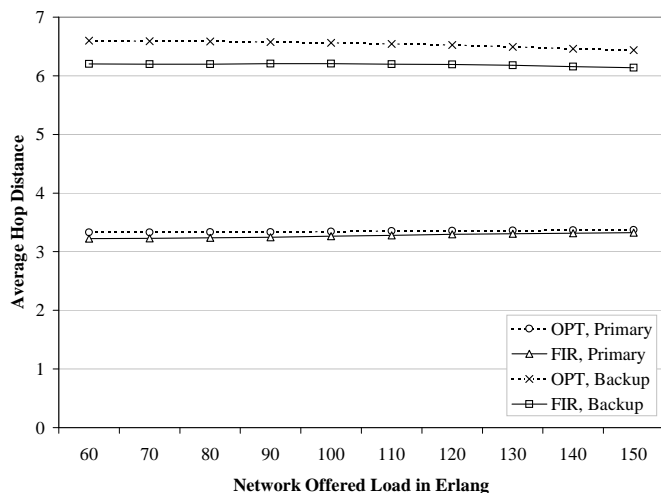


Fig. 6. Average hop distance vs. network offered load.

V. CONCLUSION

We proved that the problem of finding an eligible pair of working and backup paths under shared-path-protection constraints in a WDM mesh network for a new lightpath arrival with respect to existing lightpaths is NP-complete. We presented a heuristic—CAFES—to compute a feasible solution and an algorithm—OPT—to optimize the resource consumption for a given solution. The core ideas of how to compute a feasible solution and how to jointly optimize a given solution can be applied to lightpath provisioning with shared protection in MPLS networks as well. We showed via numerical examples that our heuristic algorithms are efficient and effective.

REFERENCES

- [1] S. Ramamurthy and B. Mukherjee, “Survivable WDM mesh networks, Part I – protection,” in *Proc. IEEE INFOCOM*, pp. 744–751, Mar. 1999.
- [2] B. T. Doshi et al., “Optical network design and restoration,” *Bell Labs Technical Journal*, vol. 4, pp. 58–84, Jan.-Mar. 1999.
- [3] G. Mohan, C. S. R. Murthy, and A. K. Somani, “Efficient algorithms for routing dependable connections in WDM optical networks,” *IEEE/ACM Trans. Networking*, vol. 9, pp. 553–566, Oct. 2001.
- [4] E. Bouillet et al., “Stochastic approaches to compute shared mesh restored lightpaths in optical network architectures,” in *Proc. IEEE INFOCOM*, pp. 801–807, Jun. 2002.
- [5] V. Anand and C. Qiao, “Dynamic establishment of protection paths in WDM networks, part I,” in *Proc. IEEE ICC*, pp. 198–204, 2000.
- [6] X. Su and C. Su, “An online distributed protection algorithm in WDM networks,” in *Proc. IEEE ICC*, vol. 5, pp. 1571–1575, June 2001.
- [7] M. Kodialam and T. V. Lakshman, “Dynamic routing of bandwidth guaranteed tunnels with restoration,” in *Proc. IEEE INFOCOM*, vol. 2, pp. 902–911, Mar. 2000.
- [8] Y. Liu, D. Tipper, and P. Siripongwutikorn, “Approximating optimal spare capacity allocation by successive survivable routing,” in *Proc. IEEE INFOCOM*, vol. 2, pp. 699–708, Apr. 2001.
- [9] G. Li, D. Wang, C. Kalmanek, and R. Doverspike, “Efficient distributed path selection for shared restoration connections,” in *Proc. IEEE INFOCOM*, pp. 140–149, Jun. 2002.
- [10] G. Ellinas et al., “Routing and restoration architectures in mesh optical networks,” *SPIE Optical Networks Magazine*, in press.
- [11] C. Ou et al., “Online algorithms for shared-path protection in optical WDM mesh networks,” Tech. Report CSE-2002-6, Mar. 2002, Dept. of Computer Science, University of California, Davis. Available at http://networks.cs.ucdavis.edu/~ouc/publications/ou_tr02.pdf.
- [12] D. Dunn, W. Grover, and M. MacGregor, “Comparison of k-shortest paths and maximum flow routing for network facility restoration,” *IEEE J. Selected Areas in Communications*, vol. 12, pp. 88–99, Jan. 1994.
- [13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. New York City, NY: McGraw-Hill, 1990.