Diverse Routing of Scheduled Lightpath Demands in an Optical Transport Network

Josué Kuri, *Student member, IEEE,* Nicolas Puech and Maurice Gagnaire, *Member, IEEE*
École Nationale Supérieure des Télécommunications, Department of Computer Science and Networks.
46, rue Barrault. F-75634 Paris CEDEX 13, France.
E-mail: {kuri,npuech,gagnaire}@enst.fr

***Abstract*—This article addresses the problem of defining working and protection paths for Scheduled Lightpath Demands (SLDs) in an optical transport network. An SLD is a demand for a set of lightpaths (connections), defined by a tuple $(s, d, n, \alpha, \omega)$, where s and d are the source and destination nodes of the lightpaths, n is the number of requested lightpaths and α, ω are the set-up and tear-down dates of the lightpaths. The problem is formulated as a combinatorial optimization problem where the objective is to minimize the number of channels required to instantiate the lightpaths. Two techniques are used to achieve this goal: *channel reuse* and *backup-multiplexing*. The former consists of assigning the same channel (either working or spare) to several lightpaths, provided that these lightpaths are not simultaneous in time. The latter consists of sharing a spare channel among multiple lightpaths. A spare channel *cannot* be shared if two conditions hold: a) the working paths of these lightpaths have at least one span in common and b) these lightpaths are simultaneous in time. In the other cases, the spare channel can be shared. We propose a Simulated Annealing (SA) based algorithm to find approximate solutions to this optimization problem since finding exact solutions is computationally intractable. The results show that backup-multiplexing improves the utilization of channels but requires significant computing capacity. Under a fixed computing capacity budget, the technique is useful in cases where there is little time disjointness among SLDs.**

***Index Terms*—Scheduled demands, optimization, protection, simulated annealing.**

I. Introduction

The ITU has developed standards that define the architecture of WDM Optical Transport Networks (OTN) [1]. These networks are expected to provide functionality to set-up, maintain and tear-down optical channels (OCh)[1] using either a management plane [2] or a control plane [3], [4]. Based on this functionality, OTNs can provide services such as Optical Virtual Private Networks (OVPNs). As an example of this service, consider the case of an OVPN client company that requests a set of static lightpaths from the OTN operator to satisfy its minimal connectivity and capacity requirements. The client may also request some *scheduled* lightpaths to increase the capacity of its OVPN between specific sites during certain periods, for example, between headquarters and production sites or between R&D sites during office hours and between data centers during the night, for the backup of databases. Finally, the unexpected peaks of traffic could be borne by dynamically established lightpaths. This example shows that the services offered by the OTN operator can lead to three different types of lightpath demands: static, scheduled and unexpected. We believe that when OTNs will become a reality, at least for some years, most of the demands will be for static and scheduled lightpaths. The reason is that the traffic load in a transport network is fairly predictable because of its cyclic nature. Figure 1 gives an indication of this phenomenon. The figure shows the traffic on the link New York - Washington of the Abilene Backbone Network [5] for a typical week. A similar cyclic pattern was observed on all the other links of the network in the same period. It can be argued that what is observed on a link is not necessarily an indication of the end-to-end traffic load profile and that the traffic load on a research network may be very different from the traffic load on a commercial network. However, the figure is an evidence of the link between communication among humans using the network (more intense during working hours), and the network traffic load.

In this paper we deal with Scheduled Lightpath Demands (SLDs). An SLD is a demand for a set of lightpaths, represented by a tuple $(s, d, n, \alpha, \omega)$ where s and d are the source and destination nodes of the lightpaths, n is the number of requested lightpaths and α, ω are the set-up and tear-down dates of the lightpaths. Table I shows an example of a set of SLDs. Note that a demand for $n > 1$ lightpaths exists if the requested rate is higher than the nominal rate of a single lightpath[2].

It may happen that some of the demands in a set of SLDs are not simultaneous in time (for example, SLDs δ_1 and δ_3 in

[1]Optical Channel (OCh) is the name given by ITU to what is commonly called a connection or a *lightpath* in the literature. In this paper we adopt the term lightpath to refer to an ITU OCh.

[2]G.872 defines OCh signals (lightpaths) of 2.5, 10 and 40 Gbit/s.

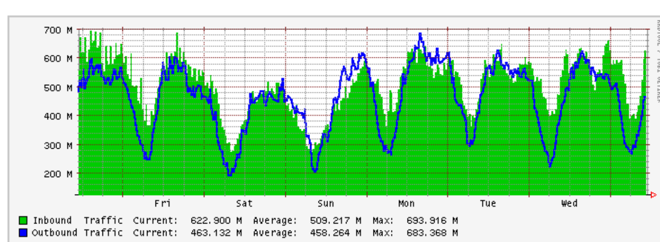


Fig. 1. Traffic on link New York - Washington of the Abilene Backbone Network from 4/3/03 to 4/10/03.

TABLE I

AN EXAMPLE OF THREE SCHEDULED LIGHTPATH DEMANDS (SLDS) AND THEIR ASSOCIATED TIME DIAGRAM

No.	s	d	n	α	ω
δ_1	2	8	2	08:00	14:40
δ_2	3	7	3	11:00	13:00
δ_3	1	6	2	17:00	19:30

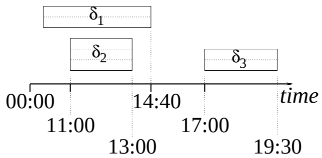


Table I). We will show in this paper that the *a priori* knowledge of the SLDs' time-disjointness can be exploited to minimize the amount of channels[3] required to instantiate the demanded lightpaths. Indeed, a same channel can be allocated to multiple lightpaths, provided that they are time-disjoint. Clearly, the more a channel is shared, the smaller the total amount of required channels is.

Survivability is a critical aspect of transport networks because of the inherent vulnerability of wire-based transmission systems and because of the increasing reliance of society on telecommunications services. Network survivability mechanisms may be classified into two main categories: restoration and protection. The former includes methods that compute backup paths and allocate spare resources *a posteriori* for working traffic affected by a network failure. These methods are potentially efficient in terms of network resource utilization since spare resources are allocated only in case of a network failure. However, it is usually difficult to guarantee *bounded* restoration times with them. On the other hand, protection mechanisms compute backup paths and allocate spare resources *a priori*, which is essential for rapid reconfiguration and, ultimately, to assure bounded restoration times. Protection is in general suitable for transport networks, where this type of guarantees is mandatory.

Figure 2 shows the different types of protection methods. Span protection methods provide a replacement to a failing span by allocating resources on a path connecting the endpoints of this span. Path protection methods define a backup path for each connection affected by the failure between the endpoints of the connection. Spare resources can be either dedicated to the protection of a link or span, or shared among multiple links or paths to protect. Dedicated spare resource protection methods are simpler than shared protection in terms of implementation, but less efficient in terms of resource utilization, which depends on the extent to which spare resources can be shared. In this paper we propose a path protection method with shared spare resources for SLDs.

The next section describes the channel reuse and backup-multiplexing techniques. The section also presents two versions of the SLD Diverse Routing problem (SLD DR): SLD DR without backup-multiplexing (SLD DR_A) and SLD DR with backup-multiplexing (SLD DR_B). In Section III we formalize both versions of the problem using two combinatorial optimization problem formulations. We also introduce a metric used to characterize the time correlation within a set of SLDs. Section IV describes the problem-specific elements needed to solve the problems using the Simulated Annealing (SA) metaheuristic algorithm. In Section V we experimentally evaluate the proposed algorithm. Section VI presents our conclusions and ongoing work.

[3]We define a *channel* as a logical pipe between two physically adjacent OTN switches. Physically, a channel corresponds to the use of a wavelength on an optical fiber connecting two switches. A channel is directional, *i.e.*, information "flows" only in one direction. An optical fiber is also directional in that all its channels are in the same direction. A *span* is a collection of fibers connecting two switches. Fibers in an span may have opposite directions. Moreover, all the fibers in a span are contained into a single cable or duct and share the same fate in the event of a cut of the cable.

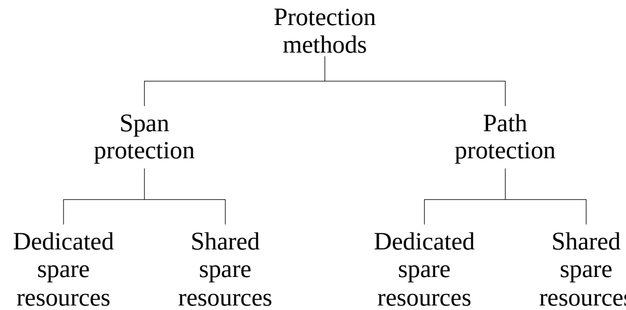


Fig. 2. Classification of protection methods.

II. PROBLEM DESCRIPTION

The SLD Diverse Routing problem (SLD DR) is the following: given a network and a set of SLDs, define for each SLD a pair of arc-disjoint paths to be used as working and protection paths, such that the number of channels (both working and spare) required in the network to instantiate the demanded lightpaths is minimized. The problem belongs to the general class of Routing and Spare Capacity Allocation (RSCA) problems in connection-oriented networks investigated in numerous papers [6]–[12]. The main difference of SLD DR with respect to those works is the use of a traffic model based on SLDs, rather than on static connection demands. In fact, static connections can be seen as a particular case of SLDs where all the demands have the same set-up and the same tear-down dates (*i.e.*, there is no time-disjointness that can be exploited to reuse channels). From a practical point of view, SLD DR is different from other RSCA problems in that the former is relevant in the context of the day-to-day operation of the network (remember the OVPN example of the previous section) whereas RSCA with static traffic model is related to the design and (long-term) planning of the network. To the best of our knowledge, this is the first time that a RSCA problem with SLD traffic is investigated.

To simplify the problem, we assume that full wavelength conversion exists in the switches at the network nodes. Thus, the wavelength assignment problem becomes trivial since a lightpath may be assigned any available wavelength on each arc (defined later) spanned by its defined path.

A first technique that can be used to reduce the number of working channels consists of assigning a same channel to as many lightpaths as possible, provided that these lightpaths

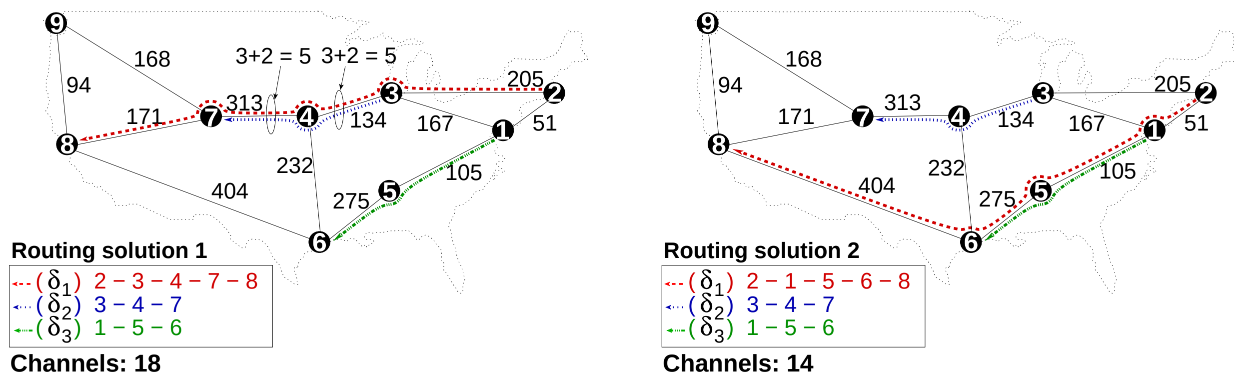


Fig. 3. Two possible routing solutions for the SLDs of Table I.

are not simultaneous in time. We call this technique *channel reuse*. Figure 3 illustrates how two different sets of working paths for the three SLDs of Table I lead to different amounts of required working channels depending on whether channel reuse is exploited or not. A channel is required for a lightpath on each arc of its path. Thus, in routing solution 1 (left side), $2 \times 4 = 8$ channels are required for SLD δ_1, $3 \times 2 = 6$ for SLD δ_2 and $2 \times 2 = 4$ for SLD δ_3, which totals 18 channels. In routing solution 2 (right side) only 14 channels are required because the channels on arcs 1-5 and 5-6 used from 08:00 to 14:00 by the lightpaths of SLD δ_1 are reused by the lightpaths of SLD δ_3 from 17:00 to 19:30. The four channels saved in solution 2 can then be used as part of the pool of resources used to support unexpected lightpath demands. Besides, clients that request services in the form of SLDs can be rewarded with service price discounts.

Channel reuse can also be used to reduce the number of spare channels required for restoration in case of failure. However, a more resource-efficient technique called *backup-multiplexing* can be used for spare channels. In this technique, a same spare channel may serve to protect multiple lightpaths provided that two conditions do not hold *simultaneously*: the involved lightpaths are simultaneous in time and their working paths share at least one common span. The point is illustrated in Figure 4: with channel reuse, spare channels can be shared for protection among multiple lightpaths in cases 1 and 3; with backup multiplexing, the channels can be shared in cases 1, 2 and 3. Backup-multiplexing is basically a form of channel reuse but, as we will see in Section V, it is more resource-efficient than mere channel reuse. The technique is inspired on an idea originally described in [7]. Depending on whether backup-multiplexing is used or not, we have two versions of the SLD DR problem: SLD DR_B and SLD DR_A, respectively. In SLD DR_A, channel reuse is used to share spare channels.

Note that a channel may serve either as a working channel or as a spare channel, but not as both.

III. Mathematical model

We first present the notations used to formally define the SLD DR_A and SLD DR_B problems as combinatorial optimization problems.

$G = (V, E, w)$
is an arc-weighted symmetrical directed graph with vertex set $V = \{v_1, v_2, \ldots, v_N\}$, arc set $E = \{e_1, e_2, \ldots, e_L\}$ and weight function $w : E \to \mathbb{R}^+$. The graph represents a telecommunications network. The set V corresponds to the network nodes. An arc $e \in E$ represents the set of optical fibers in one direction of an span. Function w corresponds to the geographical length or to the cost of the spans (*e.g.*, defined by the network operator).

$U = \{(e, e') \mid e, e' \in E, s(e) = d(e'), d(e) = s(e')\}$
is the set of spans in the network. A span is represented by a pair of arcs $e, e' \in E$ such that the source of one of the arcs is the destination of the other and *vice versa*. Hereafter, it is important to keep in mind the difference between an *arc* and a *span*.

$N = |V|$, $L = |E|, S = |U| = L/2$
are, respectively, the number of vertices, arcs and spans in G. Note that there are exactly $L/2$ spans in U.

$\Delta = \{\delta_1, \delta_2, \ldots, \delta_M\}$
is the set of M SLDs, where

$\delta_i = (s_i,\ d_i,\ n_i,\ \alpha_i,\ \omega_i)$
is a tuple representing the SLD number i; $s_i, d_i \in V$ are the source and destination nodes of the demand, n_i is the number of requested lightpaths, and α_i and ω_i are the set-up and tear-down dates of the demand, respectively.

(G, Δ)
is a pair representing an instance of the SLD DR problem.

$P = \{(x_0, x_1), (x_1, x_2), \ldots, (x_{z-1}, x_z)\}$
is an ordered set of z arcs representing a *path* from x_0 to x_z. The $(x_{i-1}, x_i) \in E$ arcs of P are all distinct (the paths are loop-free).

$P_{k,i}$, $1 \leq k \leq K$, $1 \leq i \leq M$
represents the k^{th} alternate *working path* in G from s_i to d_i.

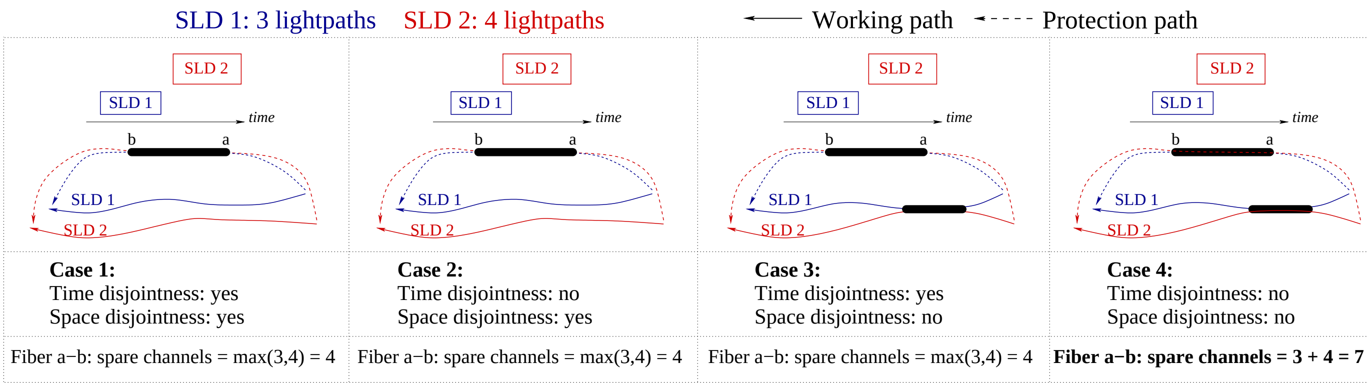


Fig. 4. The four possible cases of time and space disjointness between demands. Channel reuse on the arc a-b is possible in cases 1 and 3. Backup-multiplexing is possible in cases 1, 2 and 3.

For the purposes of this paper, we compute the K physically shortest paths for each demand using the algorithm defined in [13]. However, the paths might be defined according to any other criterion.

$P'_{k,i}$, $1 \leq k \leq K$, $1 \leq i \leq M$
represents the k^{th} alternate *backup path* in G from s_i to d_i. The pair $(P_{k,i}, P'_{k,i})$ represents the k^{th} couple of arc-disjoint paths between s_i and d_i. Two paths P_{k_a,i_a} and P'_{k_b,i_b} must be arc-disjoint only if $(k_a, i_a) = (k_b, i_b)$. In this paper, $P'_{k,i}$ is the k^{th} path computed with the algorithm defined in [13] on the graph $G' = (V, E')$, where $E' = E \backslash P_{k,i}$.

$\pi^a_{\rho,\Delta} = (P_{\rho_1,1} \;\; P_{\rho_2,2} \;\; \ldots \;\; P_{\rho_M,M})$, $\rho \in \{1, \ldots, K\}^M$
is called an *admissible routing solution* for Δ. ρ is an M-dimensional vector whose elements can take a value between 1 and K. An admissible routing solution is fully characterized by ρ.

$\pi^b_{\rho,\Delta} = (P'_{\rho_1,1} \;\; P'_{\rho_2,2} \;\; \ldots \;\; P'_{\rho_M,M})$, $\rho \in \{1, \ldots, K\}^M$
is called the *backup solution* associated to $\pi^a_{\rho,\Delta}$. An admissible routing solution has only one associated backup solution.

$\Pi_\Delta = \{(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta}), \rho \in \{1, \ldots, K\}^M\}$
is the set of solution pairs $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$ for Δ. There are $|\Pi_\Delta| = K^M$ solution pairs in the set. Hereafter we use the generic term *solution* to refer to either an admissible routing solution $\pi^a_{\rho,\Delta}$ or a backup solution $\pi^b_{\rho,\Delta}$. We denote this solution by $\pi_{\rho,\Delta}$.

$C : \Pi_\Delta \rightarrow \mathbb{N}$
is the cost function that computes the number of required channels for a given solution $\pi_{\rho,\Delta}$. If $\pi_{\rho,\Delta}$ represents an admissible routing solution, C computes the number of required working channels. If $\pi_{\rho,\Delta}$ represents a backup solution, C computes the number of required spare channels when backup multiplexing is not used. To formalize the function C, we define the following additional notations:

$\theta = (\theta_{ij})$
is a $\{0,1\}^{M \times M}$ upper triangular matrix; θ_{ij}, $i \leq j$, indicates whether the SLDs δ_i and δ_j overlap in time ($\theta_{ij} = 1$) or not ($\theta_{ij} = 0$). By definition $\theta_{ii} = 1$, $1 \leq i \leq M$, and $\theta_{ij} = 0$ for $i > j$. This matrix expresses the temporal relationship between the SLDs.

$\beta = (\beta_{ij}) = diag(n_i)$
is the diagonal matrix where $\beta_{ii} = n_i$, $1 \leq i \leq M$, *i.e.*, β_{ii} is the number of lightpaths required by the SLD δ_i.

$\gamma^{\pi_{\rho,\Delta}} = (\gamma^{\pi_{\rho,\Delta}}_{ij})$
is a $\{0,1\}^{L \times M}$ arc-path incidence matrix; $\gamma^{\pi_{\rho,\Delta}}_{ij}$ indicates whether arc $i \in E$ is part of path $P_{\rho_j,j}$ in solution $\pi_{\rho,\Delta}$ ($\gamma^{\pi_{\rho,\Delta}}_{ij} = 1$) or not ($\gamma^{\pi_{\rho,\Delta}}_{ij} = 0$). For the sake of simplicity, we note γ instead of $\gamma^{\pi_{\rho,\Delta}}$. This matrix describes the physical routing of the SLDs for a given solution $\pi_{\rho,\Delta}$.

$\eta = \theta \cdot \beta \cdot \gamma^T = (\eta_{ij})$
is a $\mathbb{N}^{M \times L}$ matrix; η_{ij} indicates the number of time-overlapping lightpaths on arc e_j between SLD δ_i and SLDs δ_k, $\forall\, k > i$ for a given solution $\pi_{\rho,\Delta}$.

Thus, the cost function C is defined as:

$$C(\pi_{\rho,\Delta}) = \sum_{j=1}^{L} \max_{1 \leq i \leq M} \eta_{ij}. \tag{1}$$

In Figure 3, the value of function C is 18 for routing solution 1 (left-side) and 14 for routing solution 2 (right-side).

A. SLD DR_A optimization problem

When backup-multiplexing is not used, we can use function C to compute both the number of working channels in an admissible solution $\pi^a_{\rho,\Delta}$ and the number of spare channels in its associated backup solution $\pi^b_{\rho,\Delta}$. Thus, the cost of a given solution pair $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$ when backup-multiplexing is not used is given by:

$$C(\pi^a_{\rho,\Delta}) + C(\pi^b_{\rho,\Delta}). \quad (2)$$

The SLD DR_A problem is formally defined by the following combinatorial optimization problem:

$$\textbf{Minimize:} \quad C(\pi^a_{\rho,\Delta}) + C(\pi^b_{\rho,\Delta}), \quad (3)$$

subject to:

$$(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta}) \in \Pi_\Delta, \quad (4)$$

that is, we want to find a solution pair $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$ in Π_Δ that minimizes the number of required working and spare channels for the set of demands Δ. Due to constraint (4), the working/backup couple of paths for a demand δ_i has to be selected among the K couples of paths $(P_{k,i}, P'_{k,i})$ precomputed for this demand. A practical advantage of using precomputed paths is that their properties (*e.g.*, geographical length, number of hops, etc.) can be under direct engineering and jurisdictional control.

B. SLD DR_B optimization problem

We need to define additional notations to describe the SLD DR_B problem.

$B : \Pi_\Delta \rightarrow \mathbb{N}$
is the cost function that computes the number of spare channels required for backup solution $\pi^b_{\rho,\Delta}$ when backup-multiplexing is used. The following notations are necessary to describe this function:

$\Delta_e^{\pi^b_{\rho,\Delta}} = \{\delta_i \mid e \in P'_{\rho_i,i}\}$
is the subset of SLDs, $\Delta_e^{\pi^b_{\rho,\Delta}} \subseteq \Delta$, whose backup path contains the arc $e \in E$ in backup solution $\pi^b_{\rho,\Delta}$. For the sake of simplicity, we note Δ_e instead of $\Delta_e^{\pi^b_{\rho,\Delta}}$.

$\Gamma^{\pi^a_{\rho,\Delta}} = (\Gamma_{ij}^{\pi^a_{\rho,\Delta}})$
is a $\{0,1\}^{S\times M}$ span-path incidence matrix similar to $\gamma^{\pi_{\rho,\Delta}}$. $\Gamma_{ij}^{\pi^a_{\rho,\Delta}}$ indicates whether at least one of the arcs of span $i \in U$ is part of path $P_{\rho_j,j}$ ($\Gamma_{ij}^{\pi^a_{\rho,\Delta}} = 1$) or not ($\Gamma_{ij}^{\pi^a_{\rho,\Delta}} = 0$) in admissible routing solution $\pi^a_{\rho,\Delta}$. As for γ, we note Γ instead of $\Gamma^{\pi^a_{\rho,\Delta}}$.

$\phi = \Gamma^T \cdot \Gamma = (\phi_{ij})$
is an $\mathbb{N}^{M\times M}$ matrix; ϕ_{ij} is the number of spans on admissible routing solution $\pi^a_{\rho,\Delta}$ where SLD δ_i overlaps with SLD δ_j.

$G_e = (\Delta_e, E_e)$
is a *conflict graph* associated to arc $e \in E$ for solution pair $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$. Each vertex of G_e represents an SLD in Δ_e. The edge set is defined as $E_e = \{(\delta_i, \delta_j) \mid \delta_i, \delta_j \in \Delta_e, \theta_{ij} = 1, \phi_{ij} > 0\}$, that is, there is an edge between two vertices if the respective SLDs overlap in time ($\theta_{ij} = 1$) and the working paths of these SLDs in solution $\pi^a_{\rho,\Delta}$ have at least one common span ($\phi_{ij} > 0$). The idea of using a conflict graph that integrates the time correlation among demands as part of the "conflict" between demands was initially used in [14] to solve the Wavelength Assignment problem for routed SLDs.

$\mathcal{A}$
is a deterministic algorithm that finds a *proper coloring* for G_e. A proper coloring of a graph $G = (V, E)$ is a partition of V such that any two vertices of a same class are not connected by an edge. For the purposes of this paper, we use a polynomial-time sequential algorithm called Largest-First First-Fit (LFFF). The algorithm defines a proper coloring with a number of colors $\chi'(G)$ that approximates the chromatic number[4] $\chi(G)$: $\chi'(G) \geq \chi(G)$. Finding a proper coloring with exactly $\chi(G)$ colors in an NP-complete problem.

$\Delta_{e,i}, \quad 1 \leq i \leq \chi'(G_e)$
is the subset $\Delta_{e,i} \subseteq \Delta_e$ of SLDs whose vertices in G_e have been colored with color i using $\mathcal{A}$. $\chi'(G_e)$ is the number of colors used by $\mathcal{A}$ to color G_e.

The cost function B is defined as:

$$B((\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})) = \sum_{e\in E} \kappa(G_e), \quad (5)$$

where,

$$\kappa(G_e) = \sum_{i=1}^{\chi'(G_e)} \max_{\delta_j \in \Delta_{e,i}} n_j \quad (6)$$

is the cost of G_e when colored with $\mathcal{A}$. In fact, $\kappa(G_e)$ corresponds to the number of spare channels required on arc e for a given backup solution $\pi^b_{\rho,\Delta}$ when backup-multiplexing is used.

The cost of a given solution pair $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$ when backup-multiplexing is used is given by:

$$C(\pi^a_{\rho,\Delta}) + B((\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})). \quad (7)$$

Thus, the SLD DR_B problem is formally defined by the following combinatorial optimization problem:

$$\textbf{Minimize:} \quad C(\pi^a_{\rho,\Delta}) + B((\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})), \quad (8)$$

subject to:

$$(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta}) \in \Pi_\Delta. \quad (9)$$

Ideally, we would like $\mathcal{A}$ to minimize $\kappa(G_e)$. On the other hand, the LFFF algorithm aims at minimizing $\chi'(G_e)$. Though these two objectives are not necessarily equivalent, we selected the LFFF algorithm because of practical reasons. The Simulated Annealing (SA) algorithm described in the next

[4] The minimum number of colors required for a proper coloring of G.

section evaluates tens of thousands of different solution pairs $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$. When solving the SLD DR_B problem, a conflict graph G_e must be built and colored for each arc $e \in E$ of a backup solution $\pi^b_{\rho,\Delta}$. Consequently, algorithm $\mathcal{A}$ must have a low time complexity (which is the case of LFFF) in order to the SA algorithm to be usable on problem instances of large size.

C. Characterization of time correlation among SLDs

The time correlation among the SLDs of a set Δ has a significant impact on the reachable degree of channel sharing of instances (G, Δ) including this set. Indeed, the smaller the time correlation, the greater the time disjointness and the greater the possibility of channel sharing. It is thus important to characterize the time correlation among the SLDs of a set Δ as a way to estimate the degree of channel sharing. We define a normalized metric of time correlation for this purpose. Let:

$$\varepsilon = (\bigcup_{i=1}^{M} \{\alpha_i\}) \cup (\bigcup_{i=1}^{M} \{\omega_i\}) \tag{10}$$

be an ordered set of $T = |\varepsilon|$ values $\varepsilon_1 < \varepsilon_2 < \ldots < \varepsilon_T$ ($T \leq 2M$ since $(\cup\{\alpha_i\}) \cap (\cup\{\omega_i\}) \neq \emptyset$) and

$$B_i = \{j \in \{1, \ldots, M\} \mid [\varepsilon_i, \varepsilon_{i+1}] \subseteq [\alpha_j, \omega_j]\} \tag{11}$$

be the set of SLD indices j such that SLD δ_j is active (at least) over time period $[\varepsilon_i, \varepsilon_{i+1}]$.

The normalized time correlation of a set Δ is given by the formula[5]:

$$\tau(\Delta) = \frac{\displaystyle\sum_{i=1}^{T-1} \sum_{j \in B_i : |B_i| > 1} n_j (\varepsilon_{i+1} - \varepsilon_i)}{\displaystyle\sum_{i=1}^{M} n_i \cdot (\omega_i - \alpha_i)}. \tag{12}$$

A value $0 \leq \tau(\Delta) \leq 1$ close to 0 indicates weak time correlation among SLDs and a value close to 1, strong time correlation.

IV. Simulated Annealing algorithm

A possibility to solve the SLD DR_A and SLD DR_B problems is to use a Branch & Bound (B&B) algorithm similar to the one proposed in [15] to solve the SLD routing (without protection) problem. However, the exponential time complexity of the algorithm renders this possibility inapplicable when dealing with problem instances of large size (in the worst-case, the K^M solutions in Π_Δ are explored). Another possibility consists of using a meta-heuristic algorithm, such as the Tabu Search (TS) algorithm proposed in [16] or SA. Meta-heuristics find *approximate* solutions to optimization problems in the sense that the result is not guaranteed to satisfy the optimality criterion of the problem under consideration. However, meta-heuristics are widely used in practice because, in general, they are able to deal with problem instances of realistic size and, at the same time, provide solutions with a cost "close enough" to the optimal one. We observed in [16] that the cost of the solutions computed with the TS algorithm was, in the worst case, within 10.08% of the optimal ones computed with B&B for problem instances with $K < 5$ and $M = 30$. In this paper we propose to use a Simulated Annealing [17] (SA) algorithm to find approximate solutions to the SLD DR_A and SLD DR_B problems. We call SA DR_A and SA DR_B the versions of the SA algorithm that solve the SLD DR_A and SLD DR_B problems, respectively. We choose SA instead of TS since we had the opportunity of using parSA [18], a parallel implementation of SA. The parallelisation provides a computing capacity that increases (sub)linearly with the number of used computers. This is an important practical consideration given the higher complexity of the SLD DR_A and SLD DR_B problems when compared to the complexity of the problem addressed in [14], [16].

SA is a generic algorithm that can be applied to a variety of optimization problems provided that one can supply three problem-specific elements: a) an initial solution, b) a cost function to evaluate the solutions generated by the algorithm and c) a perturbation procedure to generate a new solution from a current one. We define these three elements for SA DR_A and SA DR_B in the following paragraphs.

Remember from Section III that a solution pair $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$ is fully characterized by a vector ρ. Hereafter, we use vector ρ to refer to the $(\pi^a_{\rho,\Delta}, \pi^b_{\rho,\Delta})$ solution pair.

We select as initial solution the vector ρ whose components are all equal to 1. The cost functions for SA DR_A and SA DR_B are defined in (2) and (7), respectively. Finally, the perturbation procedure is defined by the following steps:

1) Generate a pseudo-random number i, uniformly distributed in the interval $[1, M]$.
2) Generate a pseudo-random number j, uniformly distributed among the elements of the set $\{1, \ldots, K\} \backslash \{\rho_i\}$.
3) Generate a new vector ρ' by replacing ρ_i by j in ρ.

For example, vector $\rho = (2, \boxed{1}, 1, 3)$ can be "perturbed" according to this procedure and lead to vector $\rho' = (2, \boxed{3}, 1, 3)$.

An SA algorithm iteratively explores the solution space until a stop condition is satisfied. ParSA provides a set of control parameters to implement various stop conditions. For example, stopping after a given number of iterations or temperature steps without significant cost improvement or stopping after a given CPU-time budget is exhausted. We denote the former condition by IWIS (Iterations Without Improvement Stop) and the latter by ETBS (Exhausted Time Budget Stop).

A. Time complexity

The only difference between SA DR_A and SA DR_B is the cost function. Therefore, if we want to compare the time

[5] Note that only index sets with cardinality $|B_i| > 1$ must be considered in the formula, since they correspond to time intervals $[\varepsilon_i, \varepsilon_{i+1}]$ where least 2 overlapping SLDs.

complexity of both algorithms we only need to focus on the complexity of the implementations of (2) and (7). Let us call these implementations $\mathcal{C}_a$ and $\mathcal{C}_b$. The time complexity is $O(LM(\log_2(M)+1))$ for the former and $O(LM(\log_2(M)+M))$ for the latter[6]. The complexity of $\mathcal{C}_b$ is greater because of a conflict graph must be built and colored for each arc of the network.

V. Experimental evaluation

The purpose of the experimental evaluation is to compare algorithms SA DR_A and SA DR_B in order to characterize the trade-off between the gain provided by backup-multiplexing and the computational cost of this technique.

We first describe the parameters common to all the experiments. Figure 5 shows the graph G used for all the problem instances (G, Δ) investigated in this section. The graph represents a hypothetical US backbone network with $N = 29$ nodes and $S = 44$ spans. For the sets Δ, the source/destination nodes, the number of lightpaths and the set-up/tear-down dates of the SLDs were drawn from uniform distributions in the intervals $[1, N]$, $[1, 10]$ and $[1, 1440]$, respectively. The set-up/tear-down dates were constrained to satisfy a target time correlation value $\tau(\Delta)$. We used the sequential implementation on Linux (kernel v2.4.18) and the parallel implementation on Solaris 5.8 of the ParSA library (v2.2). The former was executed on a PC with an AMD 266 MHz processor and 128 MB of RAM and the latter on a cluster of 10 Sun Ultra-SPARC 5 computers with 128 MB of RAM each. The sequential implementation was used in experiments involving the measurement of execution time (see below). To avoid interference from uncontrollable resource-consuming processes, the PC executing this implementation was configured as a single-user system and was disconnected from the network.

A first way to characterize the trade-off between gain and computational cost of backup-multiplexing is to execute algorithms SA DR_A and SA DR_B on a same problem instance using an IWIS condition, that is, running the algorithms until no significant cost improvement is obtained. With this condition, SA DR_B should compute a solution with cost (number of channels) lower than SA DR_A, but should take a longer time to complete. For this experiment, we used the sequential implementation of the ParSA library with a minimum acceptance ratio[7] of 20%, a frozen limit of 5 and a geometric temperature schedule $T_i = \vartheta T_{i-1}$ with constant $\vartheta = 0.9$. We averaged the CPU time and the cost over 5 runs on sets Δ with an increasing number M of SLDs and a time correlation of $\tau(\Delta) \approx 0.99$. Figure 6 shows the average CPU time and the average number of channels computed by SA DR_A and SA DR_B for sets Δ with different number M of SLDs using an IWIS condition. The difference of CPU-time growth rate between the two algorithms reflects their respective difference of time complexity. On the other hand, the difference in number of channels grows less abruptly because it depends on the gain provided by backup-multiplexing with respect to mere channel reuse of spare channels, as used in SA DR_A.

[6]Time complexity analysis is not developed here because of limited space.

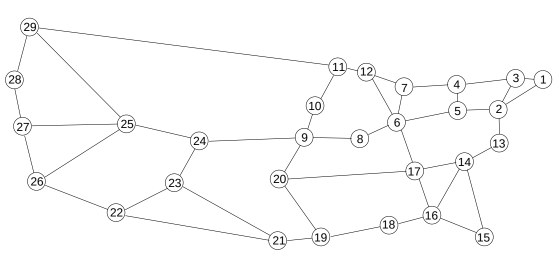
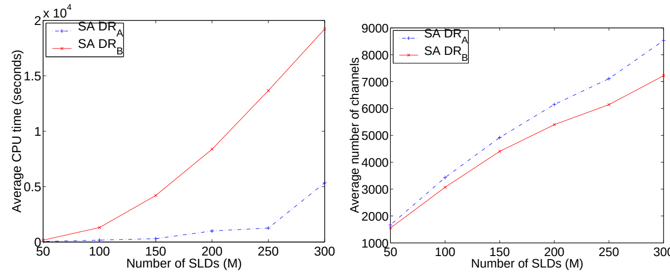


Fig. 5. The graph G considered in the experiments.



(a) Average CPU time.

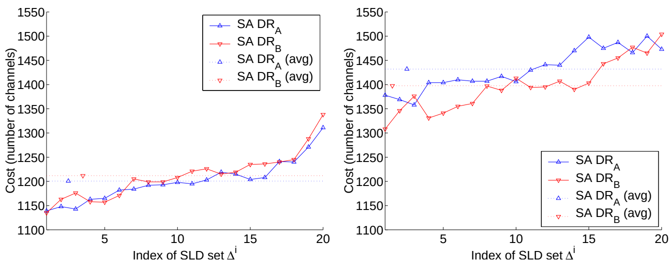
(b) Average number of channels.

Fig. 6. Average CPU time and number of channels computed by SA DR_A and SA DR_B for different values of M with an IWIS condition.

A typical characteristic of SA (and other local descent meta-heuristic algorithms) is that most of the improvements in cost occur during the first iterations of the algorithm and improvements after this initial phase are relatively seldom. This sort of Pareto's law suggests that the algorithm may be stopped after a small number of iterations and still compute a solution whose cost is not far from the cost of the best solution that the algorithm can potentially find. Based on this idea, we carried out an experiment to compare the cost of solutions computed with SA DR_A and SA DR_B using an ETBS condition, that is, running the algorithms with a limited CPU-time budget. We generated 40 sets Δ of $M = 150$ SLDs, 20 of them with a time correlation of $\tau(\Delta) \approx 0.1$ and the other 20 with $\tau(\Delta) \approx 0.9$. We fixed the CPU-time budget to 600 seconds (in the previous experiment, the average CPU-time for the problem instance with $M = 150$ was of 301.44s for SA DR_A and of 4197.85s for SA DR_B).

Figure 7 shows the number of channels computed by SA DR_A and SA DR_B for each set Δ and the average number of channels (dotted lines in the figure) for $\tau(\Delta) \approx 0.1$ and $\tau(\Delta) \approx 0.9$. We observe that the number of channels is in general smaller for sets Δ with weak time correlation (around 1200 for $\tau(\Delta) \approx 0.1$) than for sets Δ with strong time correlation (around 1400 for $\tau(\Delta) \approx 0.9$). Indeed, the greater

[7]See the ParSA library documentation [18] for the explanation of these parameters.



(a) $\tau(\Delta^i) \approx 0.1$

(b) $\tau(\Delta^i) \approx 0.9$

Fig. 7. Number of channels computed by SA DR_A and SA DR_B in 600 seconds on example sets Δ^i of $M = 150$ SLDs with weak and strong time correlation.

time disjointness of the former allows a more efficient use of resources by means of channel reuse. For $\tau(\Delta) \approx 0.1$, the average number of channels computed with SA DR_A and by SA DR_B are almost the same because, for a same CPU-time budget, SA DR_A explores more solutions than SA DR_B and thus, has more opportunities to improve the solutions cost. SA DR_B compensates this disadvantage with the gain provided by backup-multiplexing. For $\tau(\Delta) \approx 0.9$, the average number of channels computed with SA DR_B is smaller than the corresponding value of SA DR_A because the gain provided by backup-multiplexing becomes more significant under limited time disjointness conditions.

VI. Conclusions

We investigated the problem of defining diverse working and protection paths for SLDs so that the number of required channels is minimized. The versions of the problem with and without backup-multiplexing were described. The problem was formulated as a combinatorial optimization problem and an SA based algorithm was proposed to find approximate solutions. The results show that the gain in cost of SA DR_B with respect to SA DR_A increases with the number of SLDs but the growth rate of this gain is lower than the CPU-time gap growth rate. For a limited CPU-time budget, the performance of both algorithms is almost the same. Moreover, the same relative performance is observed in problem instances whose sets Δ have different levels of time correlation.

Several aspects of the proposed algorithm need to be investigated. For example, the gain of the proposed SA algorithms with respect to simple greedy heuristics or the ratio of spare to working capacity and the incidence of the time correlation $\tau(\Delta)$ and the precomputed paths $(P_{k,i}, P'_{k,i})$ on this ratio.

Another aspect that must be further investigated is the definition of a search-efficient combination of algorithm's parameters so that the algorithms can find a solution of lowest possible cost (hopefully the optimal one) using a limited CPU-time budget. A first step to achieve this goal is to understand the structural properties of the solution space of a problem instance. The solution space can be dominated, for example, by plateaus of near equal-cost solutions or by a multitude of attractor basins. Moreover, some properties can be either particular to a class of instances or general to all the instances of a problem. In the case of the SLD DR problem considered in this paper, we need to investigate the incidence of both the time correlation and the precomputed paths of an instance (G, Δ) on the structure of the instance's solution space. The incidence of backup-multiplexing needs to be investigated as well. Understanding structural properties provides valuable insight to determine a search-efficient combination of algorithm's control parameters, in particular, the type of scheduler, its parameters and the parallelisation strategy. This understanding can be also useful to determine if other local descent meta-heuristic algorithms such as TS or population based algorithms such as Genetic Algorithms are more suitable to this problem.

References

[1] "ITU-T Recommendation G.872, Architecture of optical transport networks," November 2001.

[2] "ITU-T Recommendation G.852, Management of the transport network," March 1999.

[3] "ITU-T Recommendation G.807, Requirements for the automatic switched transport network (ASTN)," July 2001.

[4] E. M. (Editor), "Generalized Multi-Protocol Label Switching (GMPLS) architecture." IETF ccamp group, March 2002. Work in progress.

[5] Abilene Backbone Network, "Advanced networking for research and education," (n.d.) [Online]. http://abilene.internet2.edu/, July 29th 2003 [date accessed].

[6] M. Herzberg and S. Bye, "An optimal spare-capacity assignment model for survivable networks with hop limits," in *Procs. of IEEE GLOBECOM 1994*, pp. 1601–1607, 1994.

[7] R. Kawamura, K. Sato, and I. Tokizawa, "Self-healing ATM networks based on virtual path concept," *IEEE JSAC*, vol. 12, no. 1, pp. 120–127, 1994.

[8] R. Iraschko, M. MacGregor, and W. Grover, "Optimal capacity placement for path restoration in STM or ATM mesh-survivable networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 3, pp. 325–336, 1998.

[9] B. V. Caenegem, W. V. Parys, F. D. Turck, and P. Demeester, "Dimensioning of survivable WDM networks," *IEEE JSAC*, vol. 16, no. 7, pp. 1146–1157, 1998.

[10] W. Grover, R. Iraschko, and Y. Zheng, "Comparative methods and issues in design of mesh-restorable STM and ATM networks," in *Telecommunication Network Planning* (B. Sanso and P. Soriano, eds.), ch. 10, pp. 169–200, Kluwer Academic Publishers, 1999.

[11] Y. Xiong and L. Mason, "Restoration strategies and spare capacity requirements in self-healing ATM networks," *IEEE/ACM Transactions on Networking*, vol. 7, no. 1, pp. 98–110, 1999.

[12] W. Grover and J. Doucette, "Topological design of survivable mesh-based transport networks," *Annals of Operations Research*, vol. 106, pp. 79–125, 2001.

[13] D. Eppstein, "Finding the k shortest paths," *SIAM Journal of Computing*, vol. 28, no. 2, pp. 652–673, 1998.

[14] J. Kuri, N. Puech, M. Gagnaire, and E. Dotaro, "Routing and wavelength assignment of scheduled lightpath demands in a WDM optical transport network," in *Procs. of ICOCN 2002*, (Singapore), pp. 270–273, IEEE Communications Society, November 2002.

[15] J. Kuri, N. Puech, M. Gagnaire, E. Dotaro, and R. Douville, "Routing and wavelength assignment of scheduled lightpath demands," *IEEE JSAC Optical Communications and Networking Series*, 2003. In press.

[16] J. Kuri, N. Puech, M. Gagnaire, and E. Dotaro, "Routing foreseeable lightpath demands using a tabu search meta-heuristic," in *Procs. of GLOBECOM 2002*, (Taipei, Taiwan), IEEE Communications Society, November 2002. Paper OPNT-04-7.

[17] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[18] G. Kliewer and P. Unruh, "Parallele simulated annealing bibliothek: Entwicklung und einsatz in der flugplanoptimierung," Master's thesis, University of Paderborn, Paderborn, Germany, February 1998.