# On the Complexity of the Regenerator Cost Problem in General Networks with Traffic Grooming*

Michele Flammini[1], Gianpiero Monaco[1], Luca Moscardelli[2], Mordechai Shalom[3], and Shmuel Zaks[4]

[1] Department of Computer Science, University of L'Aquila, L'Aquila, Italy
{flammini,gianpiero.monaco}@di.univaq.it
[2] Department of Science, University of Chieti-Pescara, Pescara, Italy
moscardelli@sci.unich.it
[3] Tel Hai Academic College, Upper Galilee, 12210, Israel
cmshalom@telhai.ac.il
[4] Department of Computer Science, Technion, Haifa, Israel
zaks@cs.technion.ac.il

**Abstract.** We consider the problem of minimizing the number of regenerators in optical networks with traffic grooming. In this problem we are given a network with an underlying topology of a graph $G$, a set of requests that correspond to paths in $G$ and two positive integers $g$ and $d$. There is a need to put a regenerator every $d$ edges of every path, because of a degradation in the quality of the signal. Each regenerator can be shared by at most $g$ paths, $g$ being the grooming factor. On the one hand, we show that even in the case of $d = 1$ the problem is $APX-hard$, i.e. a polynomial time approximation scheme for it does not exist (unless $P = NP$). On the other hand, we solve such a problem for general $G$ and any $d$ and $g$, by providing an $O(\log g)$-approximation algorithm and thus extending previous results holding only for specific topologies and specific values of $d$ or $g$.

**Keywords:** Optical Networks, Wavelength Division Multiplexing (WDM), Regenerators, Traffic Grooming, Approximation Algorithms and Complexity.

## 1 Introduction

In modern optical networks, high-speed signals are sent through optical fibers using WDM (Wavelength Division Multiplexing) technology. The decrease in the energy of the signal with the traveled distance necessitates optical amplification at every (almost) fixed distance. However this amplification introduces

* This work was partially supported by the Israel Science Foundation grant No. 1249/08, and by the PRIN 2008 research project COGENT (COmputational and GamE-theoretic aspects of uncoordinated NeTworks), funded by the Italian Ministry of University and Research.

A. Fernández Anta, G. Lipari, and M. Roy (Eds.): OPODIS 2011, LNCS 7109, pp. 96–111, 2011.

noise into the signal, so that it has to be regenerated after a certain number of amplifications. The signal is regenerated by first using a ROADM (Reconfigurable Optical Add-Drop Multiplexer) to extract a set of wavelengths from the optical fiber. Then, for each extracted wavelength, an optical regenerator regenerates the signal carried by that wavelength. That is, at a given optical node, one needs one ROADM if any regeneration will take place, and as one regenerators per wavelength to be regenerated.

The dominant part of the regeneration cost is the cost of the regenerators, because they are (a) expensive and (b) needed one per wavelength. Therefore the *total* number of regenerators is an important cost parameter to be minimized [15].

A logical path formed by a signal traveling from its source to its destination using a unique wavelength is termed a *lightpath*. Let $d$ be the maximum number of hops a lightpath can make without meeting a regenerator. Then an optimal solution can be found by simply placing one regenerator for every $d$ consecutive vertices of each lightpath $\ell$. However the problem becomes harder when the *traffic grooming* enters the picture.

*Traffic grooming*: The network usually supports traffic that is at rates lower than the full wavelength capacity. Therefore the network operator puts together (= grooms) low-capacity connection requests into high capacity lightpaths. In graph-theoretic terms, we associate a path in the graph with each connection, and the problem can viewed as assigning wavelengths to these paths so that at most $g$ of them using the same wavelength ($g$ being the *grooming factor*) can share one edge. Thus, all paths (i.e. connections) that get the same color (i.e., the same wavelength) and form a connected subgraph correspond to grooming of these connections into one lightpath. The optical signal is routed in the intermediate nodes, based on wavelength only, therefore connection requests assigned the same wavelength can not *split* from each other, i.e. they might not induce a graph with a node with degree 3 or higher. In other words a set of path assigned the same wavelength induces a graph with maximum degree two.

## 1.1 Related Work

Various variants of regenerator placement problems were studied in [3,8,16,19]. Most of these results concentrate in heuristics and simulations and do not consider traffic grooming.

In the literature, two different scenarios have been studied, depending on whether or not it is allowed to split the paths in order, for instance, to reduce the number of used wavelengths or the cost of hardware components. In particular, [5,7] assume that no splitting is allowed, while [6] allows to split paths and [14] considers both scenarios. In this work, we focus on the case in which splitting lightpaths is not allowed.

In [9] theoretical results (upper bounds and lower bounds) are presented for a family of related problems. The objective in that work is to minimize the number of regenerator locations (as opposed to the total number of regenerators), and traffic grooming is not considered. On the other hand, [15] consider the same cost measure as this work but still does not consider traffic grooming.

The problem we study is shown to be NP-hard in other contexts such as fiber minimization in [18] and its NP-hardness is also implied by the proof of a similar result in [11] holding even for path topology and $g = 2$.

When the underlying graph is a path the problem is equivalent to a machine scheduling problem studied in [10], where several approximation algorithms are presented for it and for some of its special cases. In [12] and [13] these results have been extended to the tree topology, and also an algorithm for general networks has been provided. Unfortunately, the worst case approximation ratio of that algorithm is very high for general topologies, namely as the order of the number of lightpaths.

### 1.2 Our Contribution

In this work we consider the problem of minimizing the number of regenerators in optical networks with traffic grooming, extending the results in [10,12,13] for general settings.

We first show that even in the case of $d = 1$, $G$ being a bipartite graph, the problem is $APX - hard$ for any $g \geq 2$, i.e. a polynomial time approximation scheme (PTAS) for it does not exist (unless $P = NP$). We then provide an $O(\log g)$-approximation algorithm for the most general version of this problem in which general topologies are admitted and both $d$ and $g$ can be arbitrary.

The paper is organized as follows. In Section 2 we define our problem. On the one hand, in Section 3 we show that our problem is $APX$-Hard even in the case of $d = 1$, $G$ being a bipartite graph and $g \geq 2$. On the other hand, in Section 4 we provide a polynomial time approximation algorithm solving the problem for general topologies and any value of $g$ and $d$, with an approximation ratio logarithmic in $g$. We conclude by suggesting open research directions in Section 5.

## 2 Definitions and Problem Statement

We consider instances $(G, \mathcal{P}, g, d)$ where $G = (V, E)$ is a graph modeling the optical network, $\mathcal{P}$ is a set of simple paths in $G$, $g \in \mathbb{N}^+$ is the grooming factor and $d$ is the maximum number of hops a lightpath can travel without meeting a regenerator.

A coloring (or wavelength assignment) of $(G, \mathcal{P})$ is a function $w : \mathcal{P} \mapsto \mathbb{N}$. For a coloring $w$ and color $\lambda$, $\mathcal{P}^w_\lambda$ denotes the subset of paths from $\mathcal{P}$ colored $\lambda$ by $w$, i.e. $\mathcal{P}^w_\lambda \stackrel{def}{=} \{P \in \mathcal{P} | w(P) = \lambda\}$. When there is no ambiguity on the coloring $w$ under consideration, we omit the superscript $w$ and use $\mathcal{P}_\lambda$.

For a node $v$, $\mathcal{P}_v$ denotes the subset of paths of $\mathcal{P}$ having $v$ as an intermediate node, and similarly for an edge $e$, $\mathcal{P}_e$ denotes the subset of paths of $\mathcal{P}$ using the edge $e$. For every $e \in E$ we define $load(\mathcal{P}, e) \stackrel{def}{=} |\mathcal{P}_e|$ and $load(\mathcal{P}) \stackrel{def}{=} \max_{e \in E} load(\mathcal{P}, e)$.

A set of paths is called a *no-split instance* or shortly an NSI if the union of its paths (as sets of edges) induces a graph of maximum degree at most 2. In particular, if also the minimum degree of such a graph is 2 (i.e., such a graph is a vertex disjoint union of rings), we call every connected component of it a *ring*-NSI, otherwise we call every connected component a *path*-NSI.

In this work we assume (as in [5,7,14]) that splitting of paths is not allowed, i.e. paths using the same wavelength and going through the same edge of the network can be routed only to another unique edge. Moreover, since we do not consider bounds on the number of colors and our goal is independent of it, without loss of generality we assume that, in any solution, paths belonging to different NSIs are assigned different colors, and therefore every set of paths with the same color has to be an NSI.

A *valid* coloring (or wavelength assignment) $w$ of $(G, \mathcal{P}, g, d)$ is a coloring of $\mathcal{P}$ such that for every $\lambda$, $P^w_\lambda$ satisfies the following two conditions:

- *The load condition*: For any edge $e$ at most $g$ paths using $e$ are colored with $\lambda$, i.e. $load(\mathcal{P}^w_\lambda) \leq g$.
- *The no-splitting condition*: $P^w_\lambda$ is an NSI.

Given a valid coloring $w$ of $(G, \mathcal{P}, g, d)$, a regenerator assignment is a boolean function $r_w : V \times \mathbb{N} \mapsto \{0, 1\}$; in particular, $r_w(u, \lambda) = 1$ if and only if a regenerator operating at wavelength $\lambda$ is placed at node $u$.

We are now ready to give a formal definition of our problem.

Total Regenerators with Grooming (Trg)

`Input:` A quadruple $(G, \mathcal{P}, g, d)$, where $G = (V, E)$ is a graph, $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ is a set of simple paths in $G$, $g$ is an integer, namely the grooming factor, and $d$ is the maximum number of hops a lightpath can go through without needing a regenerator.
`Output:` A valid coloring $w : \mathcal{P} \mapsto \mathbb{N}$ and a regenerator assignment $r_w$ such that, $r_w$ satisfies the constraint that every lightpath has a regenerator every at most $d$ hops (we will refer to this condition as the *regeneration condition* through this work).
`Objective:` The cost of a solution is given by the total number of regenerators $REG^w \stackrel{def}{=} \sum_\lambda \sum_{u \in V} r_w(u, \lambda)$. The goal is to minimize the total number of regenerators $REG^w$.

$OPT(G, \mathcal{P}, g, d)$ denotes the cost of any optimal coloring.

## 3 Hardness of Approximation

In this section we show that the problem Trg is $APX$-hard even if restricted to instances $(G, \mathcal{P}, g, 1)$, with $g$ at least 2.

Notice that, if $d = 1$, the coloring $w$ univocally identifies the regenerator assignment $r_w$; in fact, given an NSI $\mathcal{N}$ colored $\lambda$ by $w$, a regenerator is needed at each node being an internal node of some path in $\mathcal{N}$, i.e., $r_w(u,\lambda) = 1$ if and only if $u$ is an internal node of some path in $\mathcal{N}$.

We first define a problem that will be used in our proof.

B-Bounded Edge Partition into Triangles and Minimum Paths (MEPTP-B):
`Input:` A graph $G = (V, E)$ with $\Delta(G) \leq B$, where $\Delta(G)$ is the maximum degree of a node of $G$.
`Output:` A partition of $E$ into connected graphs with at most 3 edges.
`Measure of a solution:` The number of paths of the returned partition.
`Objective:` Minimizing the measure of the returned solution.

By exploiting a reduction similar to the one used in [1] we will prove that this problem is APX-Hard. Finally we will reduce this problem to the $(G, \mathcal{P}, g, 1)$ problem with $g \geq 2$ in order to show the APX-Hardness of the latter.

**Definition 1.** *Given a tripartite graph $G = (V_0 \cup V_1 \cup V_2, E)$ we can obtain a directed graph, by directing the edges from nodes of $V_i$ nodes of $V_{(i+1) \bmod 3}$. We will say that $G$ is directed Eulerian if the directed graph obtained in this way is directed Eulerian.*

**Lemma 1.** *The* MEPTP-B *problem is* APX-*Hard for any fixed $B \geq 12$ even when the graph is directed Eulerian tripartite and the optimum is at least $|E|/10$.*

**Theorem 1.** *The set of $(G, \mathcal{P}, g, 1)$ instances of the* Trg *problem is APX-hard for any $g \geq 2$ and even when $G$ is a bipartite graph.*

*Proof.* We will give an approximation ratio preserving reduction from the MEPTP-B problem in graphs satisfying the conditions of Lemma 1 to the $(G, \mathcal{P}, g, 1)$ instances of Trg, with $g \geq 2$.

Let $G' = (V_0' \cup V_1' \cup V_2', E')$ be an instance of MEPTP-B. We build an instance $(G = (V, E), \mathcal{P}, g, 1)$ of our problem as follows (see Figure 1): For each $i \in \{1, 2, 3\}$ and for every node $v \in V_i'$, $G$ contains a path with three nodes $v^-, v, v^+$ and two edges $(v^-, v), (v, v^+)$. $G$ contains 3 special nodes $u_{01}, u_{12}, u_{20}$. Node $u_{ij}$ is connected to all the $v^+$ nodes corresponding to any $v \in V_i'$ and to all the $v^-$ nodes corresponding to any $v \in V_j'$.

For each edge $(v, w) \in E'$ where $v \in V_i'$ and $w \in V_j'$ ($j \equiv (i+1) \mod 3$), $\mathcal{P}$ contains the path $(v^-, v, v^+, u_{ij}, w^-, w, w^+)$.

The constructed instance has the following properties:

- Any two distinct edges of $G'$ both connecting nodes from $V_i'$ and $V_j'$ correspond to two paths in $\mathcal{P}$ that induce a graph with degree 3 or 4 at node $u_{ij}$. Therefore these two paths cannot be part of an NSI. We conclude an NSI can contain at most 3 paths, i.e., one corresponding to an edge of $G'$ from $V_0'$ to $V_1'$, another one corresponding to an edge of $G'$ from $V_1'$ to $V_2'$, and another one corresponding to an edge of $G'$ from $V_2'$ to $V_0'$.
- The two paths corresponding to any pair of adjacent edges in $G'$ are compatible (i.e., form together an NSI) and have one intermediate node in common.

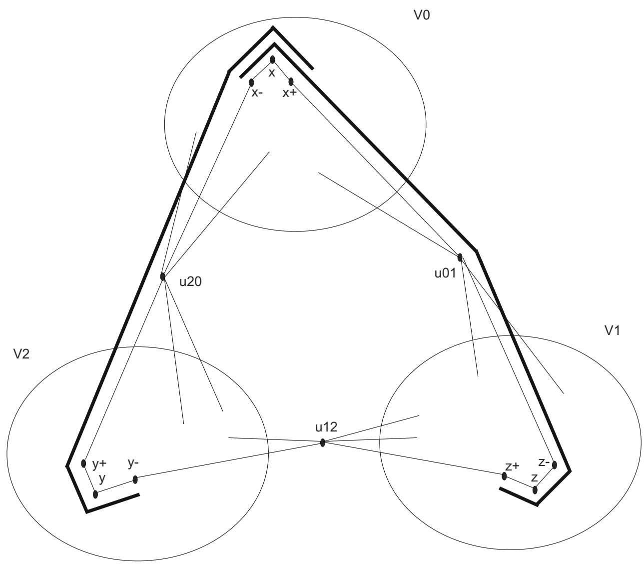


**Fig. 1.** The bipartite graph in the proof of Theorem 1

We conclude that the edges in $G'$ corresponding to the paths of an NSI form either a triangle, or a path with 1, 2 or 3 edges, and conversely, every such subgraph of $G'$ corresponds to an NSI. Let $T$ be the number of NSIs corresponding to triangles of $G'$, and $l_k$ be the number of NSIs corresponding to paths of length $k$ of $G'$ for $k \in \{1,2,3\}$. Note that each path of $\mathcal{P}$ has 5 intermediate nodes. Therefore the number $Reg$ of regenerators used by such a solution is $Reg = 5|E'| - 3T - 2l_3 - l_2$. As $G'$ is directed Eulerian tripartite, for any $T$ there is a solution with $l_2 = l_1 = 0$, with cost $Reg = 5|E'| - 3T - 2l_3 = 5|E'| - (3T + 2l_3) = 5|E'| - (|E'| - l_3) = 4|E'| + l_3$. Therefore the minimum number of regenerators is obtained at the optimum of the MEPTP-B instance. Let $Reg^*$ be the optimum of instance $(G, \mathcal{P}, g, 1)$, and consider a $\rho$-approximate solution of it with $Reg = \rho \cdot Reg^*$. Moreover, let $l_k^*$, for $k \in \{1,2,3\}$, be the number of paths of length $k$ in an optimal solution of the corresponding instance of the MEPTP-B problem. Then

$$\begin{aligned} l_3 &= Reg - 4|E'| = \rho \cdot Reg^* - 4|E'| = \rho \cdot (l_3^* + 4|E'|) - 4|E'| \\ &= \rho \cdot l_3^* + 4(\rho - 1)|E'| \le \rho \cdot |l_3^*| + 40(\rho - 1)l_3^* \\ &= (\rho + 40(\rho - 1))l_3^*, \end{aligned} \tag{1}$$

where 1 holds because by Lemma 1 we can assume that the optimum of $G'$ is at least $\frac{|E'|}{10}$.

Assume that our problem admits a PTAS. For any $\epsilon > 0$ we run the PTAS with the parameter $\epsilon' = \epsilon/41$ to obtain a $\rho = 1 + \epsilon/41$ approximated solution.

This corresponds to a solution of the MEPTP-B with $l_3 \leq (1+\epsilon)l_3^*$. A contradiction to the fact that MEPTP-B does not admit a PTAS unless P=NP. □

## 4 Approximation Algorithm

In this section we provide an approximation algorithm for the TRG problem for general topologies, guaranteeing an $O(\log g)$ approximation ratio in polynomial time.

A *proper* set $\bar{\mathcal{P}}$ of paths, is a set of paths that constitute and independent set with respect to inclusion. In other words no paths of $\bar{\mathcal{P}}$ is included in another. An instance is said to be proper if its set of paths $\mathcal{P}$ is proper.

This section is organized as follows: We first provide an $O(\log g)$-approximation algorithm for the case of proper instances with $d = 1$. We then extend this result to the more general case in which $d = 1$ but the instance is not necessarily proper. Finally we extend the result to any value of $d$. Each time we extend the previous result, we lose only a constant factor in the approximation ratio, therefore achieving an $O(\log g)$-approximation ratio for the general case.

We introduce some definitions that will be useful in the proofs contained in this section. We denote by $INT(P)$ the set of intermediate nodes, i.e. of all the nodes not being endpoints, of a path $P$ in $G$, and $int(P) \stackrel{def}{=} |INT(P)|$. For a set $\mathcal{P}$ of paths we define:

$$SPAN(\mathcal{P}) \stackrel{def}{=} \bigcup_{P\in\mathcal{P}} INT(P),\, span(\mathcal{P}) \stackrel{def}{=} |SPAN(\mathcal{P})|\,,\, len(\mathcal{P}) \stackrel{def}{=} \sum_{P\in\mathcal{P}} int(P).$$

Notice that, if $d = 1$, the number of regenerators operating at wavelength $\lambda$ is $span(\mathcal{P}_\lambda^w)$; in fact, at each node being an intermediate node of some path in $\mathcal{P}_\lambda^w$ a regenerator operating at this wavelength is needed. Moreover, when $d = 1$, we have the following trivial lower bound (the grooming bound) for the cost of any coloring $w$ (in particular for an optimal coloring), holding because a regenerator can be used by at most $g$ intermediate nodes of paths: $REG^w \geq \frac{len(\mathcal{P})}{g}$.

### 4.1 Proper Instances with $d = 1$

In this section, we focus on the case $d = 1$, i.e., a regenerator is needed at every internal node of a path, and the set of paths constitute a proper set. In particular, we provide Algorithm 2 working for $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ instances, with $\bar{\mathcal{P}}$ being a proper set of paths. It exploits the greedy set cover approximation algorithm *GreedySetCover* for the minimum weight set cover problem presented in [4]. Such an algorithm guarantees an $H_k$ approximation ratio, where $k$ is the maximum cardinality of a subset in the input and $H_k$ is the $k$-th harmonic number $\sum_{i=1}^{k} \frac{1}{i}$.

More formally, the Set Cover problem is defined as follows:
MINIMUM WEIGHTED SET COVER
`Output:` A subcollection $\mathcal{SC} \subseteq \mathcal{S}$ of subsets covering the elements in $A$, i.e. such that $\cup\mathcal{SC} = A$.
`Measure of a solution:` $\sum_{S \in \mathcal{SC}} weight[S]$, i.e. the sum of the weights of the selected subsets.
`Objective:` Minimizing the measure of the returned solution.

We present here the *GreedySetCover* Algorithm of [4] because we will slightly modify it in the sequel, to improve the time complexity of our algorithm.

**Algorithm 1.** [4] $GreedySetCover(A, \mathcal{S}, weight)$

$\mathcal{SC} \leftarrow \emptyset$
$Covered \leftarrow \emptyset$
**while** $Covered \neq A$ **do**
    **for** $i = 1$ to $m$ **do**
        $eff[S_i] \leftarrow \frac{weight[S_i]}{|S_i \setminus Covered|}$
    **end for**
    $min \leftarrow \text{argmin}_{i=1}^{m} eff[S_i]$
    $\mathcal{SC} \leftarrow \mathcal{SC} \cup \{S_{min}\}$
    $Covered \leftarrow Covered \cup S_{min}$
**end while**
**return** $\mathcal{SC}$

**Definition 2.** *Given a set $\mathcal{Q}$ of paths, and a path $P \in \mathcal{Q}$, we say that $P$* dominates $\mathcal{Q}$ *if $\forall P' \in \mathcal{Q}$, $E(P') \cap E(P) \neq \emptyset$. A set $\mathcal{Q}$ of paths is said to be dominated if there exists a path $P \in \mathcal{Q}$ that dominates $\mathcal{Q}$.*

We term an NSI $\mathcal{N}$ such that $load(\mathcal{N}) \leq g$ as a $g$-NSI. Our algorithm is based on the following basic lemma.

**Lemma 2.** *Let $\mathcal{N}$ be a proper $g$-NSI. $\mathcal{N}$ can be covered with proper, dominated $g$-NSI's $\mathcal{Q}_0^{\mathcal{N}}, \mathcal{Q}_1^{\mathcal{N}}, \ldots$, such that $\sum_i span(\mathcal{Q}_i^{\mathcal{N}}) \leq 2 \cdot span(\mathcal{N})$.*

*Proof.* Let $\widehat{\mathcal{N}} \subseteq \mathcal{N}$ be a maximal subset of pairwise edge-disjoint paths in $\mathcal{N}$. It follows from the maximality, that every path $P \in \mathcal{N}$ edge-intersects with at least one path of $\widehat{\mathcal{N}}$. Let $\widehat{\mathcal{N}} = \{P_0, P_1, \ldots, P_{|\widehat{\mathcal{N}}|-1}\}$; if $\mathcal{N}$ is a path-NSI we assume without loss of generality that $P_0, P_1, \ldots$ are ordered from left to right, otherwise (i.e., if $\mathcal{N}$ is a ring-NSI) we assume that they are ordered clockwise. In the following the terms *before* and *after* refer to this order, *right* and *clockwise* are used interchangeably, and when $\mathcal{N}$ is a ring-NSI index arithmetic is done modulo $|\widehat{\mathcal{N}}|$.

We observe that a path $P \in \mathcal{N}$ intersects either exactly one path $P_i$, or two consecutive paths $P_i, P_{i+1}$, because otherwise there would exist a path $P_j$ included in $P$, contradicting the properness of $\mathcal{N}$. We partition $\mathcal{N}$ into $\mathcal{Q}_0^{\mathcal{N}}, \ldots, \mathcal{Q}_{|\widehat{\mathcal{N}}|-1}^{\mathcal{N}}$ such that, for $i = 0, \ldots, |\widehat{\mathcal{N}}| - 1$, $\mathcal{Q}_i$ consists of the paths of

$\mathcal{N}$ intersecting only $P_i$, or both $P_i$ and $P_{i+1}$. Clearly each $\mathcal{Q}_i^{\mathcal{N}}$ is a proper $g$-NSI dominated by $P_i$. It remains to show that the last condition in the statement of the Lemma holds. We will show that every node $v \in SPAN(\mathcal{N})$ is in at most two sets $SPAN(\mathcal{Q}_i^{\mathcal{N}})$ and $SPAN(\mathcal{Q}_{i+1}^{\mathcal{N}})$.

Clearly, the claim holds when $\left|\widehat{\mathcal{N}}\right| \leq 2$. Assume that $\left|\widehat{\mathcal{N}}\right| \geq 3$ and that there is a node $v \in SPAN(\mathcal{Q}_i^{\mathcal{N}}) \cap SPAN(\mathcal{Q}_{i+j}^{\mathcal{N}})$ where $j > 1$. If $v$ is not before the right endpoint of $P_{i+1}$ then there is a path $P \in \mathcal{Q}_i^{\mathcal{N}}$ whose right endpoint is not before the right endpoint of $P_{i+1}$, thus including $P_{i+1}$, contradicting the properness of $\mathcal{N}$. If $v$ is before the right endpoint of $P_{i+1}$, then there is a path $P \in \mathcal{Q}_{i+j}^{\mathcal{N}}$ that intersects $P_{i+1}$ contradicting the way we partitioned $\mathcal{N}$. □

**Algorithm 2.** $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$

1: ▷ Prepare the input for *GreedySetCover*
2: $\mathcal{S} \leftarrow \emptyset$
3: **for** each $\mathcal{Q} \subseteq \bar{\mathcal{P}}$ such that $|\mathcal{Q}| \leq 2g-1$ **do**
4:     **if** $load(\mathcal{Q}) \leq g$ **and** $\mathcal{Q}$ is dominated **and** $\mathcal{Q}$ is an NSI **then**
5:         $\mathcal{S} \leftarrow \mathcal{S} \cup \{\mathcal{Q}\}$
6:         $weight[\mathcal{Q}] \leftarrow span(\mathcal{Q})$
7:     **end if**
8: **end for**
9: $\mathcal{SC} \leftarrow GreedySetCover(\bar{\mathcal{P}}, \mathcal{S}, weight)$
10: ▷ Eliminate inclusions
11: **while** there exist $S, S' \in \mathcal{SC}$ such that $S \cap S' \neq \emptyset$ **do**
12:     $S \leftarrow S \setminus S'$
13: **end while**
14: ▷ Assign colors to paths
15: **for** each $S_i \in \mathcal{SC}$ **do**
16:     **for** each $P \in S$ **do**
17:         $w(P) \leftarrow i$
18:     **end for**
19: **end for**
20: **return** $w$

**Theorem 2.** *Algorithm 2 is a* $2H_{2g-1}$*-approximation algorithm for* $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ *instances, where* $\bar{\mathcal{P}}$ *is a proper set of paths. Its running time is not polynomial in* $g$*.*

*Proof.* The cost of the solution is at most the cost of the set cover returned by the greedy algorithm, because the elimination of inclusions (lines 11–13 of Algorithm 2) can only reduce the cost of the cover.

Since the maximum cardinality of a subset in the collection $\mathcal{S}$ given in input to *GreedySetCover* is $2g-1$ and therefore, by [4], *GreedySetCover* guarantees an $H_{2g-1}$ approximation ratio, in order to prove the claim, it remains to show that there exists a subcollection $\overline{\mathcal{SC}} \subseteq \mathcal{S}$ such that $\sum_{S \in \overline{\mathcal{SC}}} weight[S] \leq 2 \cdot OPT(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$.

Let $\mathcal{N}_1^*, \mathcal{N}_2^*, \ldots, \mathcal{N}_{W^*}^*$ be the NSIs of an optimal solution. For each $1 \leq i \leq W^*$, let $\mathcal{Q}_0^{\mathcal{N}_i^*}, \ldots, \mathcal{Q}_{|\widehat{\mathcal{N}_i^*}|-1}^{\mathcal{N}_i^*}$ be the proper, dominated $g$-NSIs whose existence is guaranteed by Lemma 2, and let $\overline{\mathcal{SC}} = \left\{ \mathcal{Q}_j^{\mathcal{N}_i^*} | 1 \leq i \leq W^*, 0 \leq j \leq |\widehat{\mathcal{N}_i^*}| - 1 \right\}$. It holds that:

$$\sum_{S \in \overline{\mathcal{SC}}} weight[S] = \sum_{S \in \overline{\mathcal{SC}}} span(S) = \sum_{i=1}^{W^*} \sum_{j=0}^{|\widehat{\mathcal{N}_i^*}|-1} span(\mathcal{Q}_j^{\mathcal{N}_i^*})$$
$$\leq \sum_{i=1}^{W^*} 2 \cdot span(\mathcal{N}_i^*) = 2 \cdot OPT(\mathcal{G}, \bar{\mathcal{P}}, g, 1).$$

To conclude the proof we note that a proper, dominated $g$-NSI contains at most $2g-1$ paths. This is because each path of such a set must use one of the extremal edges of the dominating path, otherwise such a path either does not intersect the dominating path, or it is included in it, both of which contradict the definition of dominated, proper set. There can be at most $g-1$ paths except the dominating path using each extremal edge, therefore at most $2(g-1)+1 = 2g-1$ paths. We thus conclude $\overline{\mathcal{SC}} \subseteq \mathcal{S}$. □

Though Algorithm 2 is not polynomial in $g$ (as it considers all subset of paths of cardinality at most $2g-1$) we are able to provide a polynomial time algorithm preserving the same approximation ratio up to a constant factor. First of all, we relax the greedy choice (line 7) of algorithm *GreedySetCover* as the following algorithm does.

**Algorithm 3.** $GreedySetCover_2(A, \mathcal{S}, weight, \rho)$

1: $\mathcal{SC} \leftarrow \emptyset$
2: $Covered \leftarrow \emptyset$
3: **while** $Covered \neq A$ **do**
4:     **for** $i = 1$ to $m$ **do**
5:         $eff[S_i] \leftarrow \frac{weight[S_i]}{|S_i \setminus Covered|}$
6:     **end for**
7:     Let $j$ such that $eff[S_j] \leq \rho \cdot \min_{i=1}^{m} eff[S_i]$
8:     $\mathcal{SC} \leftarrow \mathcal{SC} \cup \{S_j\}$
9:     $Covered \leftarrow Covered \cup S_j$
10: **end while**
11: **return** $\mathcal{SC}$

The following lemma states a well known fundamental result on the approximation ratio guaranteed by Algorithm $GreedySetCover_2$.

**Lemma 3.** *Algorithm* $GreedySetCover_2$ *guarantees a* $(\rho \cdot H_k)$*-approximation for the set cover problem, where* $k$ *is the size of the biggest set in the input.*

An immediate consequence of the above lemma and Theorem 2 is

**Corollary 1.** *Algorithm 2 in which, at line 9, $GreedySetCover_2$ is invoked with $\rho = 2$ instead of GreedySetCover is a $4H_{2g-1}$-approximation algorithm for $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ instances, where $\bar{\mathcal{P}}$ is a proper set of paths. Its running time is not polynomial in $g$.*

We are now ready to provide and analyze the following algorithm.

**Algorithm 4.** $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$

```
1: SC ← ∅
2: Covered ← ∅
3: while Covered ≠ P̄ do
4:     for each Q_i ⊆ P̄ \ Covered such that |Q_i| ≤ 3 and Q_i is a dominated NSI do
5:         for each P ∈ P̄ \ Covered do
6:             if Q_i ∪ {P} is a dominated g-NSI and SPAN(Q_i ∪ {P}) = SPAN(Q_i) then
7:                 Q_i ← Q_i ∪ {P}
8:             end if
9:         end for
10:        eff[Q_i] ← span(Q_i) / |Q_i|
11:    end for
12:    Choose Q_min that minimizes eff[Q_min]
13:    SC ← SC ∪ {Q_min}
14:    Covered ← Covered ∪ Q_min
15: end while
16:                                                        ▷ Assign colors to paths
17: for each Q_i ∈ SC do
18:     for each P ∈ Q_i do
19:         w(P) ← i
20:     end for
21: end for
22: return w
```

**Lemma 4.** *Algorithm 4 is a polynomial time $4H_{2g-1}$-approximation algorithm for $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ instances, with $\bar{\mathcal{P}}$ being a proper set of paths.*

*Proof.* In this proof Algorithm 2 refers to the variant in which, at line 9, instead of $GreedySetCover$, $GreedySetCover_2$ is invoked with $\rho = 2$. Recall that Corollary 1 holds for this variant. Actually Algorithm 4 is equivalent to Algorithm 2 in the following sense: Instead of preparing an exponential number of sets (lines 2–8 of Algorithm 2) and passing it to $GreedySetCover_2$ (line 9 of Algorithm 2), it actually simulates it, and each time calculates the greedy choice of $GreedySetCover_2$, by iterating over a polynomial number of sets.

In particular, in the following we show that the subcollection $\mathcal{SC}$ computed at the end of line 15 of Algorithm 4 is one of the possible subcollections that Algorithm $GreedySetCover_2$, executed at line 9 of Algorithm 2 on input $(\bar{\mathcal{P}}, \mathcal{S}, weight)$ (where $\mathcal{S}$ and $weight$ are those computed at lines 2–8 of Algorithm 2), could return as output.

Let without loss of generality the subcollection $\mathcal{SC}$ computed at lines 1–15 of Algorithm 4 be $\{\mathcal{B}_1, \mathcal{B}_2, \ldots\}$ in the order they are chosen in the *while loop*. We prove that $\mathcal{SC}$ is a possible subcollection that can be returned by the $GreedySetCover_2$ algorithm invoked at line 9 of Algorithm 2. It is enough to show that for every $k$, $\mathcal{B}_k$ is a possible outcome of iteration $k$ of the while loop of $GreedySetCover_2$. In the following discussion we confine ourselves to the $k$-th iteration of both algorithms and to the values of $\mathcal{SC}$ and $Covered$ in the beginning of this iteration. Specifically we have to show that for every $\mathcal{D} \in \mathcal{S}$, $eff[\mathcal{B}_k] \leq 2 \cdot eff[\mathcal{D}]$ that means that $\mathcal{B}_k$ can be chosen by $GreedySetCover_2$ algorithm at iteration $k$. Consider an arbitrary set $\mathcal{D} \in \mathcal{S}$. Since $\mathcal{D}$ is a dominated, proper $g$-NSI then $|\mathcal{D}| \leq 2g - 1$ and there exists a set of three[1] paths $P_1, P_2, P_3 \in \mathcal{D}$ dominated by $P_1$, such that $SPAN(\mathcal{D}) = SPAN(\{P_1, P_2, P_3\})$. The set $\{P_1, P_2, P_3\}$ is considered by Algorithm 4 at line 4 and therefore a corresponding set of paths $\mathcal{Q}_i$ with at least $\min(g, |\mathcal{D}|)$ paths is built, such that $SPAN(\mathcal{Q}_i) = SPAN(\{P_1, P_2, P_3\})$. We get $|\mathcal{Q}_i| \geq \frac{|\mathcal{D}|}{2}$ and therefore

$$eff[\mathcal{Q}_i] = \frac{SPAN(\{P_1, P_2, P_3\})}{|\mathcal{Q}_i|} = \frac{SPAN(\mathcal{D})}{|\mathcal{Q}_i|} \leq \frac{2 \cdot SPAN(\mathcal{D})}{|\mathcal{D}|} = 2 \cdot eff[\mathcal{D}].$$

Since $\mathcal{B}_k$ is chosen as the subset of paths with minimum cost-effectiveness at line 12 of Algorithm 4, $eff[\mathcal{B}_k] \leq eff[\mathcal{Q}_i] \leq 2 \cdot eff[\mathcal{D}]$, as required.

As a final remark, notice that the algorithm puts each path in exactly one subset of the subcollection $\mathcal{SC}$, and therefore there is no need to eliminate inclusions as in lines 11–13 of Algorithm 2. □

### 4.2 Case $d = 1$

In this section we deal with the case $d = 1$ and general instances, by reducing the problem to the special case of proper instances.

In order to show such a reduction, we exploit Algorithm "FirstFit" of [13] and we also need a lemma of [13].

Algorithm $FirstFit$ colors the paths greedily by considering them one after the other, from longest to shortest. Each path is assigned the lowest possible color for it. $FirstFit$ uses colors starting from $\lambda_{start}$.

**Lemma 5.** *[13] Let $w$ be the coloring returned by $FirstFit$ and $W \geq 1$ be the number of colors used by $w$; for any $\lambda_{start} < \lambda \leq \lambda_{start} + W$, $len(\mathcal{P}_{\lambda-1}) \geq \frac{g}{3} span(\mathcal{P}_\lambda)$.*

We are now ready to prove the following lemma.

[1] Actually, the number of such paths could be less than three, but the proof easily extends to these cases.

**Algorithm 5.** [13] $FirstFit(G, \mathcal{P}, g, \lambda_{start})$ where $G$ is a path or a ring

1: Sort the paths in non-increasing order of length, i.e., $int(P_1) \geq int(P_2) \geq \ldots \geq int(P_n)$.
2: Consider the paths by the above order and, for any path $P_j$, $j \in \{1, \ldots, n\}$, let $w(P_j)$ be the first possible color $\lambda \geq \lambda_{start}$ that will not violate the load condition. Namely, find the minimum value $\lambda \geq \lambda_{start}$ such that, for every edge $e$ of $P_j$, $load(\mathcal{P}_\lambda, e) \leq g - 1$ and set $w(P_j) \leftarrow \lambda$.
3: **return** $w$

**Lemma 6.** *Given a polynomial time $\rho$-approximation algorithm $\mathcal{A}$ for $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ such that $\bar{\mathcal{P}}$ is a proper set of paths, it is possible to obtain a polynomial time $(2\rho + 3)$-approximation algorithm $\mathcal{A}'$ for $(\mathcal{G}, \mathcal{P}, g, 1)$, where $\mathcal{P}$ is not necessarily proper.*

*Proof.* As a first step we calculate a maximal subset $\bar{\mathcal{P}} \subseteq \mathcal{P}$ of proper paths, by picking up all the paths $P \in \mathcal{P}$ and adding $P$ to $\bar{\mathcal{P}}$ as long as $P$ is not included in any other path of $\bar{\mathcal{P}}$. Clearly, $\bar{\mathcal{P}}$ is proper and every path $P \in \mathcal{P} \setminus \bar{\mathcal{P}}$ is included in a path of $\bar{\mathcal{P}}$. Consider Algorithm 6 executed on input $((\mathcal{G}, \mathcal{P} = \mathcal{P}' \cup \bar{\mathcal{P}}, g, 1), \bar{w})$, where $\bar{w} = \mathcal{A}(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$. For any $1 \leq \lambda \leq \bar{W}$, let $first_\lambda$ and $last_\lambda$ be the minimum and the maximum color used by $w_\lambda$, respectively. By Lemma 5, we obtain

$$\sum_{\lambda'=first_\lambda+1}^{last_\lambda} span(\mathcal{P}_{\lambda'}) = \sum_{\lambda'=first_\lambda}^{last_\lambda-1} span(\mathcal{P}_{\lambda'+1}) \leq \frac{3}{g} \sum_{\lambda'=first_\lambda}^{last_\lambda-1} len(\mathcal{P}_{\lambda'}) < \frac{3}{g} len(P'_\lambda). \tag{2}$$

Moreover, Algorithm 6 guarantees that $span(\mathcal{P}_{first_\lambda}) \leq span(\bar{\mathcal{N}}_\lambda)$, because all the paths in $\mathcal{P}_{first_\lambda}$ are included in some path of $\bar{\mathcal{N}}_\lambda$.

The number of regenerators used by the coloring $w$ returned by Algorithm 6 is

$$\begin{aligned}
&\sum_{\lambda=1}^{\bar{W}} span(\bar{\mathcal{N}}_\lambda) + \sum_{\lambda=1}^{\bar{W}} \sum_{\lambda'=first_\lambda}^{last_\lambda} span(\mathcal{P}_{\lambda'}) \\
&= \sum_{\lambda=1}^{\bar{W}} span(\bar{\mathcal{N}}_\lambda) + \sum_{\lambda=1}^{\bar{W}} \left( span(\mathcal{P}_{first_\lambda}) + \sum_{\lambda'=first_\lambda+1}^{last_\lambda} span(\mathcal{P}_{\lambda'}) \right) \\
&\leq \sum_{\lambda=1}^{\bar{W}} span(\bar{\mathcal{N}}_\lambda) + \sum_{\lambda=1}^{\bar{W}} \left( span(\bar{\mathcal{N}}_\lambda) + \frac{3}{g} \cdot len(P'_\lambda) \right) \qquad (3) \\
&= 2 \sum_{\lambda=1}^{\bar{W}} span(\bar{\mathcal{N}}_\lambda) + \frac{3}{g} \sum_{\lambda=1}^{\bar{W}} len(P'_\lambda)
\end{aligned}$$

$$\leq 2\rho \cdot OPT(\mathcal{G}, \bar{\mathcal{P}}, g, 1) + 3 \cdot OPT(\mathcal{G}, P', g, 1) \quad (4)$$
$$\leq (2\rho + 3) \cdot OPT(\mathcal{G}, P' \cup \bar{\mathcal{P}}, g, 1), \quad (5)$$

where inequality 3 holds by (2); inequality 4 holds because A is $\rho$-approximation algorithm for $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ and by the grooming bound; finally, inequality 5 holds because both $OPT(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ and $OPT(\mathcal{G}, P', g, 1)$ are at most $OPT(\mathcal{G}, P' \cup \bar{\mathcal{P}}, g, 1)$. □

**Algorithm 6.** $((\mathcal{G}, \mathcal{P}, g, 1), \bar{w})$, where $\bar{w}$ is a valid coloring for the instance $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$ with $\bar{\mathcal{P}} = \mathcal{P} \setminus \mathcal{P}'$ being a maximal proper set of paths

Let $1, 2, \ldots, \bar{W}$ be the colors used by $\bar{w}$
$\lambda_{new} \leftarrow \bar{W} + 1$
**for** $\lambda = 1$ to $\bar{W}$ **do**
  $\bar{\mathcal{N}}_\lambda = \bar{\mathcal{P}}_\lambda$ ▷ $\bar{\mathcal{P}}_\lambda$ is the set of paths in $\bar{\mathcal{P}}$ colored $\lambda$ by $\bar{w}$
  $\mathcal{P}'_\lambda \leftarrow \emptyset$
  **for** each $P \in \mathcal{P}'$ such that $P$ is included in some path of $\bar{\mathcal{N}}_\lambda$ **do**
    $\mathcal{P}'_\lambda \leftarrow \mathcal{P}'_\lambda \cup \{P\}$
  **end for**
  $w_\lambda \leftarrow FirstFit(\bar{\mathcal{N}}_\lambda, \mathcal{P}'_\lambda, g, \lambda_{new})$
  $\lambda_{new} \leftarrow 1+$ the maximum color used in $w_\lambda$
**end for**
**return** $w = \cup_{\lambda=1}^{\bar{W}} w_\lambda$.

### 4.3 The General Case

The following lemma of [13] shows that, given a $\rho$-approximation algorithm for $(G, \mathcal{P}, g, 1)$, it is possible to obtain a $4\rho$-approximation algorithm for $(G, \mathcal{P}, g, d)$.

**Lemma 7 ([13]).** *Given a polynomial time $\rho$-approximation algorithm $\mathcal{A}$ for $(G, \mathcal{P}, g, 1)$, for any $d > 1$, it is possible to obtain a polynomial time algorithm $\mathcal{A}'$ guaranteeing a $(4 \cdot \rho)$-approximation for $(G, \mathcal{P}, g, d)$.*

Given an NSI $\mathcal{N}$, let $G(\mathcal{N}) = (V(\mathcal{N}), E(\mathcal{N}))$ be the graph corresponding to the NSI $\mathcal{N}$, i.e. such that $V(\mathcal{N}) = \bigcup_{P \in \mathcal{N}} V(P)$ and $E(\mathcal{N}) = \bigcup_{P \in \mathcal{N}} E(P)$. We now provide a polynomial time algorithm (Algorithm 7) whose existence is shown in Lemma 7, transforming a feasible solution for $(G, \mathcal{P}, g, 1)$ into a feasible solution for $(G, \mathcal{P}, g, d)$.

By combining Algorithms 4, 6 and 7, we are finally able to provide a polynomial time approximation algorithm (Algorithm 8) working for any $(G, \mathcal{P}, g, d)$ instance of the general problem.

By exploiting Lemmata 7, 6 and 4, we obtain the following theorem.

**Theorem 3.** *Algorithm 8 is a $(32H_{2g-1} + 12)$-approximation polynomial time algorithm for general instances.*

**Algorithm 7.** $((G, \mathcal{P}, g, d), w)$, $w$ being a valid coloring such that $\mathcal{P}_\lambda$ is an NSI for any $\lambda$

Let $1, 2, \ldots, W$ be the colors used by $w$
**for** $\lambda = 1$ to $W$ **do**
  $\mathcal{N}_\lambda = \mathcal{P}_\lambda$
  **for** each connected component $G'$ in the subgraph of $G(\mathcal{N}_\lambda)$ induced by the nodes in $SPAN(\mathcal{N}_\lambda)$ **do** ▷ $G'$ is either a path or a cycle
    Build $r_w$ such that in $G'$ there is a regenerator every $d$ nodes
  **end for**
**end for**
**return** $(w, r_w)$

**Algorithm 8.** $(\mathcal{G}, \mathcal{P}, g, d)$

$\mathcal{P}' \leftarrow \emptyset$ ▷ Partition $\mathcal{P}$ into $\bar{\mathcal{P}}$ and $\mathcal{P}'$ such that $\bar{\mathcal{P}}$ is a maximal proper set of paths.
$\bar{\mathcal{P}} \leftarrow \mathcal{P}$
**while** there exist $P, \bar{P} \in \bar{\mathcal{P}}$ such that $P$ is included in $\bar{P}$ **do**
  $\mathcal{P}' \leftarrow \mathcal{P}' \cup \{P\}$
  $\bar{\mathcal{P}} \leftarrow \bar{\mathcal{P}} \setminus \{P\}$
**end while**
$w' \leftarrow$ **Algorithm 4** $(\mathcal{G}, \bar{\mathcal{P}}, g, 1)$
$w'' \leftarrow$ **Algorithm 6** $((\mathcal{G}, \mathcal{P}, g, 1), w')$
$w \leftarrow$ **Algorithm 7** $((\mathcal{G}, \mathcal{P}, g, d), w'')$
**return** $w$

## 5 Open Problems

The main open problem is that of closing the gap between the hardness result of Section 3 and the approximation ratio guaranteed by the Algorithm 8 provided in Section 4. In particular, determining whether the problem is in APX constitutes a very interesting research direction.

Interesting research directions are that of modeling the network by means of a edge-weighted graph and also that of considering lightpaths requiring a bandwidth being $\frac{b}{g}$, with $1 \leq b \leq g$; notice that in this paper we have dealt with the case $b = 1$.

It would be also interesting to extend our result by considering more involved cost functions taking into account other switching parameters (e.g., the ADMs - Add-Drop-Multiplexers - used at the endpoints of the lightpath) or the possibility of splitting paths.

## References

1. Amini, O., Pérennes, S., Sau, I.: Hardness and approximation of traffic grooming. Theor. Comput. Sci. 410(38-40), 3751–3760 (2009)
2. Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., Protasi, M.: Complexity and Approximation, Combinatorial Optimization Problems and Their Approximability Properties. Springer, Heidelberg (1999)

3. Chen, S., Ljubic, I., Raghavan, S.: The regenerator location problem, vol. 55, pp. 205–220 (2010)
4. Chvátal, V.: A greedy heuristic for the set covering problem. Mathematics of Operation Research 4, 233–235 (1979)
5. Călinescu, G., Frieder, O., Wan, P.-J.: Minimizing electronic line terminals for automatic ring protection in general wdm optical networks. IEEE Journal of Selected Area on Communications 20(1), 183–189 (2002)
6. Călinescu, G., Wan, P.-J.: Splitable traffic partition in wdm/sonet rings to minimize sonet adms. Theoretical Computer Science 276(1-2), 33–50 (2002)
7. Călinescu, G., Wan, P.-J.: Traffic partition in wdm/sonet rings to minimize sonet adms. Journal of Combinatorial Optimization 6(4), 425–453 (2002)
8. Fedrizzi, R., Galimberti, G.M., Gerstel, O., Martinelli, G., Salvadori, E., Saradhi, C.V., Tanzi, A., Zanardi, A.: A Framework for Regenerator Site Selection Based on Multiple Paths. In: Prooceedings of IEEE/OSA Conference on Optical Fiber Communications, OFC (2010)
9. Flammini, M., Marchetti-Spaccamela, A., Monaco, G., Moscardelli, L., Zaks, S.: On the complexity of the regenerator placement problem in optical networks. IEEE/ACM Transactions on Networking 19(2), 498–511 (2011)
10. Flammini, M., Monaco, G., Moscardelli, L., Shachnai, H., Shalom, M., Tamir, T., Zaks, S.: Minimizing total busy time in parallel scheduling with application to optical networks. Theor. Comput. Sci. 411(40-42), 3553–3562 (2010)
11. Flammini, M., Monaco, G., Moscardelli, L., Shalom, M., Zaks, S.: Approximating the Traffic Grooming Problem with Respect to ADMs and OADMs. In: Luque, E., Margalef, T., Benítez, D. (eds.) Euro-Par 2008. LNCS, vol. 5168, pp. 920–929. Springer, Heidelberg (2008)
12. Flammini, M., Monaco, G., Moscardelli, L., Shalom, M., Zaks, S.: Optimizing Regenerator Cost in Traffic Grooming (extended abstract). In: Lu, C., Masuzawa, T., Mosbah, M. (eds.) OPODIS 2010. LNCS, vol. 6490, pp. 443–458. Springer, Heidelberg (2010)
13. Flammini, M., Monaco, G., Moscardelli, L., Shalom, M., Zaks, S.: Optimizing regenerator cost in traffic grooming. Technical Report CS-2011-07, Technion, Department of Computer Science (2011)
14. Gerstel, O., Lin, P., Sasaki, G.: Wavelength assignment in a wdm ring to minimize cost of embedded sonet rings. In: INFOCOM 1998, Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 69–77 (1998)
15. Mertzios, G.B., Sau, I., Shalom, M., Zaks, S.: Placing Regenerators in Optical Networks to Satisfy Multiple Sets of Requests. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010. LNCS, vol. 6199, pp. 333–344. Springer, Heidelberg (2010)
16. Sriram, K., Griffith, D., Su, R., Golmie, N.: Static vs. dynamic regenerator assignment in optical switches: models and cost trade-offs. In: Workshop on High Performance Switching and Routing (HPSR), pp. 151–155 (2004)
17. Vazirani, V.V.: Approximation Algorithms. Springer, Heidelberg (2004)
18. Winkler, P., Zhang, L.: Wavelength assignment and generalized interval graph coloring. In: SODA, pp. 830–831 (2003)
19. Yang, X., Ramamurthy, B.: Sparse Regeneration in Translucent Wavelength-Routed Optical Networks: Architecture, Network Design and Wavelength Routing. Photonic Network Communications 10(1), 39–53 (2005)