Online Regenerator Placement*

George B. Mertzios[1], Mordechai Shalom[2],
Prudence W.H. Wong[3], and Shmuel Zaks[4]

[1] School of Engineering and Computing Sciences, Durham University, UK
george.mertzios@durham.ac.uk
[2] TelHai College, Upper Galilee, 12210, Israel
cmshalom@telhai.ac.il
[3] Department of Computer Science, University of Liverpool, Liverpool, UK
pwong@liverpool.ac.uk
[4] Department of Computer Science, Technion, Haifa, Israel
zaks@cs.technion.ac.il

Abstract. Connections between nodes in optical networks are realized by lightpaths. Due to the decay of the signal, a regenerator has to be placed on every lightpath after at most d hops, for some given positive integer d. A regenerator can serve only one lightpath. The placement of regenerators has become an active area of research during recent years, and various optimization problems have been studied. The first such problem is the Regeneration Location Problem (RLP), where the goal is to place the regenerators so as to minimize the total number of nodes containing them. We consider two extreme cases of online RLP regarding the value of d and the number k of regenerators that can be used in any single node. (1) d is arbitrary and k unbounded. In this case a feasible solution always exists. We show an $O(\log |X| \cdot \log d)$-competitive randomized algorithm for any network topology, where X is the set of paths of length d. The algorithm can be made deterministic in some cases. We show a deterministic lower bound of $\Omega\left(\frac{\log(|E|/d)\cdot\log d}{\log(\log(|E|/d)\cdot\log d)}\right)$, where E is the edge set. (2) $d = 2$ and $k = 1$. In this case there is not necessarily a solution for a given input. We distinguish between feasible inputs (for which there is a solution) and infeasible ones. In the latter case, the objective is to satisfy the maximum number of lightpaths. For a path topology we show a lower bound of $\sqrt{l}/2$ for the competitive ratio (where l is the number of internal nodes of the longest lightpath) on infeasible inputs, and a tight bound of 3 for the competitive ratio on feasible inputs.

Keywords: online algorithms, optical networks.

1 Introduction

Background. Optical wavelength-division multiplexing (WDM) is the most promising technology today that enables us to deal with the enormous growth of

* This work was supported in part by the Israel Science Foundation grant No. 1249/08 and British Council Grant UKTELHAI09.

A. Fernández Anta, G. Lipari, and M. Roy (Eds.): OPODIS 2011, LNCS 7109, pp. 4–17, 2011.

traffic in communication networks, like the Internet. Optical fibers using WDM technology can carry around 80 wavelengths (colors) in real networks and up to few hundreds in testbeds. As satisfactory solutions have been found for various coloring problems, the focus of studies shifts from the number of colors to the hardware cost. These new measures provide better understanding for designing and routing in optical networks.

A communication between a pair of nodes is done via a *lightpath*. The energy of the signal along a lightpath decreases and thus amplifiers are used every fixed distance. Yet, as the amplifiers introduce noise into the signal there is a need to place a regenerator every at most d hops.

There is a limit imposed by the technology on the number of regenerators that can be placed in a network node [3,5]. We denote this limit by k and refer to the case where this limit is not likely to be reached by any regenerator placement as $k = \infty$.

The problems. Given a network G, a set of lightpaths in G, and integers d and k, we need to place regenerators at the nodes of the network, such that a) for each lightpath there is a regenerator in at least one of each d consecutive internal nodes, and b) at most k regenerators are placed at any node. When $k = \infty$ we consider the *regenerator location problem* (RLP) where the objective is to minimize the number of nodes that are assigned regenerators. When k is bounded there are inputs for which there is no feasible regenerator placement that satify both conditions. For example, consider the case $d = 2$ and $k = 1$, and three identical lightpaths $u-v-w-x$. Each of these lightpaths must have a regenerator either at v or w, and this is clearly impossible). In this case we consider the *Path Maximization Problem* (PMP) that seeks for regenerator placements that serve as many lightpaths as possible. We consider online algorithms (see [2]) for these problems.

Online algorithms. In the online setting the lightpaths are given one at a time, the algorithm has to decide on the locations of the regenerators and cannot change the decision later. An algorithm is *c-competitive* for RLP if for every input the number of locations used is no more than c times the locations used by an optimal offline algorithm. An online algorithm is *c-competitive* for PMP if the number of lightpaths that it satisfies is at least $1/c$ times the number of lightpaths that could be satisfied by an optimal offline algorithm.

Related Work. Placement of regenerators in optical networks has become an active area in recent years. Most of the researches have focused on the technological aspects of the problems, heuristics and simulations in order to reduce the number of regenerators, (e.g., [3,4,7,9,10,11,12]). The regenerator location problem (RLP) was shown to be NP-complete in [3], followed by heuristics and simulations. In [5] theoretical results for the offline version of RLP are presented. The authors study four variants of the problem, depending on whether the number k of regenerators per node is bounded, and whether the routings of the requests are given. Regarding the complexity of the problem, they present polynomial-time algorithms and NP-completeness results for a variety of special cases.

We note that while considering the path topology, RLP has implications for the following scheduling problem: Assume a company has n cars and that car i needs to be serviced within every at most d days between day a_i and b_i. Furthermore, assume that the garage can serve at most k cars per day and charges a certain cost each time the garage is used. The objective is to service the cars in the fewest number of days and hence minimizing the number of times the garage is used.

Other objective functions have also been considered in the context of regenerator placement. E.g., in [8] the problem of minimizing the total number of regenerators is studied under other settings.

Our Contribution. In this paper we study the online version of the regenerator location problem, and consider two extreme cases regarding the value of d and the value k of the number of regenerators that can be used in any single node.

- RLP: $k = \infty$, G and d are arbitrary (in this case there is a solution for every input, and the measurement is the number of locations in which regenerators are placed). We show:
  - an $O(\log |X| \cdot \log d)$-competitive randomized algorithm for any network topology, that can be made deterministic (with the same competitive ratio) for some cases including tree topology networks, where X is the set of all paths of length d in G.
  - a deterministic lower bound of $\Omega\left(\frac{\log(|E|/d)\cdot\log d}{\log(\log(|E|/d)\cdot\log d)}\right)$, where E is the edge set of G.
- PMP: G is a path, $k = 1$ and $d = 2$ (in this case there is not necessarily a solution, and the measurement is the number of satisfied lightpaths). We distinguish between feasible inputs (for which there is a solution) and infeasible ones, on a path topology, and show:
  - a lower bound of $\sqrt{l}/2$ for the competitive ratio for general instances which may be infeasible (where l is the number of internal nodes of the longest lightpath).
  - a tight bound of 3 for the competitive ratio of deterministic online algorithms for feasible instances.

Organization of the paper. In Section 2 we present some preliminaries. In Section 3 we consider general topology and analyze the first extreme case (k unbounded). In Section 4 we analyze the other extreme case ($k = 1$) for a path topology. In Section 5 we present further research directions.

2 Preliminaries

Given an undirected underlying graph $G = (V, E)$ that corresponds to the network topology, a *lightpath* is a simple path in G. We are given a set $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ of simple paths in G that represent the lightpaths. The *length* of a lightpath is the number of edges it contains. The *internal vertices*

(resp. *edges*) of a path P are the vertices (resp. edges) in P except the first and the last ones.

A regenerator assignment is a function $reg : V \times \mathcal{P} \mapsto \{0,1\}$. For any $v \in V, P \in \mathcal{P}$, $reg(v,P) = 1$ if a regenerator is assigned to P at node v. Note that $reg(v,P) = 1$ only if v is an internal node of P. We denote by $reg(v)$ the number of regenerators located at node v, i.e., $reg(v) = \sum_{P\in\mathcal{P}} reg(v,P)$. Denote by $cost(reg)$ the cost of the assignment reg, measured by the total number of locations where regenerators have been placed. Let $R(reg) = \{v \in V | reg(v) \geq 1\}$, then $cost(reg) = |R(reg)|$.

Given an integer d, a lightpath P is *d-satisfied* by the regenerator assignment reg if it does not contain d consecutive internal vertices without a regenerator, in other words, for any d consecutive internal vertices of P, $v_1, v_2, \cdots, v_d$, $\sum_{i=1}^{d} reg(v_i, P) \geq 1$. A set of lightpaths is *d-satisfied* if each of its lightpaths is d-satisfied. Note that a path with at most d edges is d-satisfied regardless of reg, therefore we assume without loss of generality that every path $P \in \mathcal{P}$ has at least $d+1$ edges. For the sake of the analysis we assume, without loss of generality, that every edge of the graph is used by at least one path $P \in \mathcal{P}$. We want to emphasize that this is not assumed by the online algorithms, (what would be a loss of generality).

The Regenerator Location Problem (RLP): given a graph $G = (V,E)$, a set $\mathcal{P}$ of paths in G, a distance $d \geq 1$, determine the smallest number of nodes $R \subseteq V$ to place regenerators so that all the paths in $\mathcal{P}$ are d-satisfied. Formally:

REGENERATOR LOCATION PROBLEM (RLP)[a]

Input: An undirected graph $G = (V,E)$, a set $\mathcal{P}$ of paths in G, $d \geq 1$
Output: A regenerator assignment reg such that every path $P \in \mathcal{P}$ is d-satisfied.
Objective: Minimize $cost(reg)$.

[a] The offline version of this problem is denoted as RPP/∞/+ in [5].

reg^* denotes an optimal regenerator assignment and $cost^*$ denotes its cost $cost(reg^*)$. We consider the online version of the problem in which G and d are given in advance and the paths $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ arrive in an online manner, one at a time in this order. An online algorithm finds a regenerator assignment as the input arrives and once $reg(v,P)$ is set to 1 it cannot be reverted to 0. An online algorithm ALG for RLP is *c-competitive*, for $c \geq 1$, if its cost is at most $c \cdot cost^*$. Clearly, when $d = 1$, $cost(reg) = |V_I|$ for any regenerator assignment reg where V_I is the set of nodes that are internal nodes of some lightpaths, therefore any algorithm is 1-competitive. Hence we consider the case $d \geq 2$.

When k is finite, we study the Path Maximization Problem (PMP): given a graph $G = (V,E)$, a set $\mathcal{P}$ of paths in G, a distance $d \geq 1$, place regenerators so that the number of d-satisfied paths in $\mathcal{P}$ is maximized. Formally:

PATH MAXIMIZATION PROBLEM (PMP)

Input: An undirected graph $G = (V, E)$, a set $\mathcal{P}$ of paths in G, $d, k \geq 1$
Output: A regenerator assignment *reg* for which $reg(v) \leq k$ for every node $v \in V$.
Objective: Maximize the number of d-satisfied paths in $\mathcal{P}$.

An online algorithm ALG for PMP is *c-competitive*, for $c \geq 1$, if the number of paths it satisfies is at least $1/c$ times the number of paths satisfied by an optimal offline algorithm.

3 The Regenerator Location Problem

In this section we consider the case where the technological limit imposed on the number of regenerators at a node is unlikely to be reached by any regenerator assignment. In this case we can assume without loss of generality that whenever there is a node v and a path P with $reg(v, P) = 1$ then $reg(v, P') = 1$ for every other path $P' \in \mathcal{P}$, because this does not affect $cost(reg)$. In other words for any given node v and any two paths $P, P' \in \mathcal{P}$ we assume $reg(v, P) = reg(v, P')$, thus $reg(v) = \sum_{P \in \mathcal{P}} reg(v, P) \in \{0, |\mathcal{P}|\}$. In this section we divide the objective function by $|\mathcal{P}|$ and denote by $reg(v)$ the value $reg(v, P_1) = reg(v, P_2), = \cdots$.

3.1 Upper Bound for Path Topology

Lemma 1. *There is a 2-competitive deterministic online algorithm in path topologies for* RLP.

Proof. Let $V = \{v_1, v_2, \ldots, v_n\}$ be the nodes of the path and $E = \{\{v_i, v_{i+1}\} \,|1 \leq i < n\}$ be its edge set. We set $R = \{v_d, v_{2d}, \ldots\} \subseteq V$ and start with the empty assignment, i.e. $reg(v) = 0$ for every node $v \in V$. When a path P is presented to the algorithm we set $reg(v) = 1$ for every $v \in R \cap P$. This strategy clearly d satisfies all the paths.

We show that this algorithm is 2-competitive. Consider the union $\cup\mathcal{P}$ of all the paths in the input. $\cup\mathcal{P}$ is a disjoint union of maximal sub-paths of G. Consider such a maximal sub-path, and let ℓ be its length. Clearly, our algorithm uses at most $\lceil \frac{\ell}{d} \rceil$ locations among the nodes of this sub-path. Note that $\ell > d$, because otherwise there is at least one path in the input with at most d edges. Using these fact one can show by induction on $m = \lceil \frac{\ell}{2d} \rceil$ that any solution uses at least m locations among the nodes of this sub-path. □

3.2 Upper Bound for General Topologies

In this section we use the randomized algorithm presented in [1] for the online set-cover problem. For completeness, we provide brief descriptions of the problem and the algorithm.

An instance of the set cover problem is a pair $(X, \mathcal{S})$ where $X = \{x_1, x_2, \ldots\}$ is a ground set of elements, and $\mathcal{S} = \{S_1, S_2, \ldots\}$ is a collection of subsets of X. Given such an instance, one has to find a subset $\mathcal{C} \subseteq \mathcal{S}$ that covers X, i.e. $\cup_{S_i \in \mathcal{C}} S_i = X$. In [1] an online variant of the set cover problem is considered. An instance of the online set cover problem is a triple $(X, \mathcal{S}, X')$ where X and $\mathcal{S}$ are as before, and $X' \subseteq X$ is presented in an online manner, one element at a time. At any given time one has to provide a cover $\mathcal{C}' \subseteq \mathcal{S}$ of X', i.e. $X' \subseteq \cup_{S_i \in \mathcal{C}'} S_i$. Once a set is included in the cover $\mathcal{C}'$ this decision can not be changed when subsequent input is received. In other words, whenever an element is presented an online algorithm has to cover it by at least one set from $\mathcal{S}$ if it is not already covered. It is important to note that X and $\mathcal{S}$ are known in advance but X' is given online.

We proceed with a description of the online algorithm in [1]. We denote by $S^{(i)}$ the set of all sets containing x_i, i.e. $S^{(i)} \stackrel{def}{=} \{S_j \in \mathcal{S} | x_i \in S_j\}$. Let f an upper bound for the frequencies of the elements, i.e. $\forall x_i \in X, \left|S^{(i)}\right| \leq f$. The algorithm associates a weight w_j with each set S_j which is initiated to $1/f$. The weight $w^{(i)}$ of each element $x_i \in X$ is the sum of the weights of the sets containing it, i.e. $w^{(i)} = \sum_{S_j \in S^{(i)}} w_j$. See pseudo-code in Algorithm OnLineSetCover below for a description of the algorithm.

Algorithm 1. OnLineSetCover

When a non-covered element $x_i \in X$ is presented:
Find the smallest non-negative integer q such that $2^q \cdot w^{(i)} \geq 1$;
for each set $S_j \in S^{(i)}$ **do**
    $\delta_j = 2^q \cdot w_j - w_j$;
    $w_j += \delta_j$;
end for
do $4 \log |X|$ times
    Choose at most one set (from $S^{(i)}$) to the cover
        where each set S_j is chosen with probability $\delta_j/2$;

From an instance $(G, \mathcal{P}, d)$ of Rlp we build an instance $(X, \mathcal{S}, X')$ of the online set cover problem. X is the set of all possible paths of length d in G and $|\mathcal{S}| = |V|$. Each set $S_j \in \mathcal{S}$ consists of all the paths in X containing the node v_j. For a path P, let $P^{(d)}$ be the set of all its sub-paths of length d. X' is $\cup_{P \in \mathcal{P}} P^{(d)}$. Now we observe that for any feasible regenerator assignment *reg*, $R(reg)$ is a set cover, and vice versa, i.e. any set cover $\mathcal{C}$ corresponds to a feasible regenerator assignment *reg* such that $R(reg) = \mathcal{C}$. Indeed, a path P is d-satisfied if and only if every path of $P^{(d)} \subseteq X'$ contains a node v_j with regenerators, that corresponds to a set $S_j \in \mathcal{C}$ containing this path. Therefore all the paths $P \in \mathcal{P}$ are d-satisfied if and only if $\mathcal{C}$ constitutes a set cover of X'. Moreover the cost of the set cover is equal to the number of regenerator locations, i.e. $|\mathcal{C}| = \sum_{v_j} reg(v_j) = cost(reg)$.

When a path P is presented, we present to OnLineSetCover all the paths of $P^{(d)}$ one at a time. For each set S_j added to the cover by OnLineSetCover, we set $reg(v_j) = 1$.

We first note that although the number of sets in X is exponential in terms of the input size of our problem, for every path P the set $P^{(d)}$ contains only a polynomial number of paths, therefore the first loop of Algorithm OnLineSetCover runs only a polynomial number of times. The second loop is executed $\log|X|$ times, which is also polynomial in terms of our input size.

Algorithm OnLineSetCover is proven to be $O(\log|X| \cdot \log f)$-competitive. Note that a path of length d contains $d+1$ nodes, thus $f = d+1$. As the cost of a cover is equal to the cost of a solution of $(G, \mathcal{P}, d)$ we conclude

Lemma 2. *There is an $O(\log|X| \cdot \log d)$-competitive polynomial-time randomized online algorithm for instances $(G, \mathcal{P}, d)$ of* Rlp *where X is the set of all the paths of length d in G.*

In [1] algorithm OnLineSetCover is de-randomized using the method of conditional expectation. However in this method, in order to calculate the conditional expectancies, one has to consider all the elements of X. In our case X is the set of all paths of length d in G which is, in general, exponential in d, thus applying the technique in [1] directly to our case leads to an exponential algorithm. Although the definition of competitive ratio does not require polynomial running-time, for practical purposes we would like to have polynomial-time algorithms. The following theorem states some cases for which this condition is satisfied.

Theorem 1. *There is an $O(\log|X| \cdot \log d)$-competitive polynomial-time deterministic online algorithm for instances $(G, \mathcal{P}, d)$ of* Rlp *in each one of the following cases where X is the set of all the paths of length d in G.*

- *Both d and the maximum degree $\Delta(G)$ of G are bounded by two constants.*
- *The number of cycles in G is bounded, in particular G is a ring.*
- *G has bounded treewidth, in particular G is a tree.*

3.3 Lower Bound for General Topologies

In this section we show a lower bound nearly matching the upper bound in the previous subsection, by using the online version of a reduction in [5] of set cover to Rlp. Given an instance $(X, \mathcal{S}, X')$ of online set cover we build an instance $(G, \mathcal{P}, d)$ of Rlp as follows (see Figure 1).

We set $d = |\mathcal{S}|$. The node set $V(G)$ of G is $\mathcal{S} \cup V_1 \cup V_2$ where $V_1 = \{s_i, t_i | 1 \leq i \leq |X|\}$ and $V_2 = \{v_{ij} | 1 \leq i \leq |X|, 1 \leq j \leq |\mathcal{S}|\}$. We proceed with a description of the paths $\mathcal{P}$. The edge set of G will be all the edges induced by the paths of $\mathcal{P}$. For each element x_i there is a path P_i in $\mathcal{P}$ between s_i and t_i. If $x_i \in S_j$ then $S_j \in V(G)$ is an internal node of the P_i, otherwise v_{ij} is an internal node of P_i. The internal nodes are ordered within the path P_i by their j index, i.e. the path x_i is of the form $(s_i - u_1 - u_2 - \cdots - u_{|\mathcal{S}|} - t_i)$ where u_j is either S_j or v_{ij} as described before.

By this construction every path x_i has exactly $|\mathcal{S}| = d$ internal nodes. Therefore a regenerator assignment is feasible if and only if it assigns at least one regenerator to one of the internal nodes of every path. Without loss of generality every element x_i is contained in at least one set S_j, otherwise no set

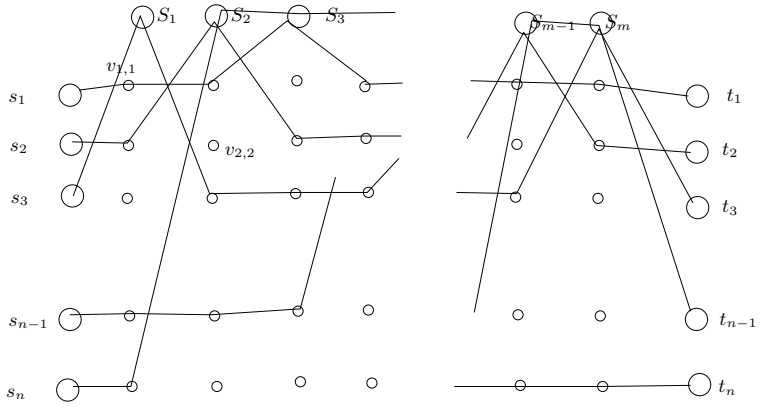

Fig. 1. Reduction from online set cover to RLP

cover exists. A feasible regenerator assignment *reg* corresponds to a set cover, in the following way. We first obtain a regenerator assignment *reg*′ such that $reg'(v_{ij}) = 0$ for every $v_{ij} \in V_2$ and $cost(reg') \leq cost(reg)$. For every node with $reg(v_{ij}) = 1$ we set $reg'(v_{ij}) = 0$, and if P_i is not d-satisfied in *reg*′ we choose arbitrarily a node S_j on P_i and set $reg'(S_j) = 1$. Now $R(reg') \subseteq \mathcal{S}$ is a set cover of cardinality at most $cost(reg)$.

Lemma 3. *There is no* $O(\frac{\log(|E|/d)\cdot\log d}{\log(\log(|E|/d)\cdot\log d)})$*-competitive online algorithm for* RLP.

Sketch of proof: Assume by contradiction that there is an $O(\frac{\log(|E|/d)\cdot\log d}{\log(\log(|E|/d)\cdot\log d)})$-competitive randomized algorithm ALG for RLP. From an instance $(X, \mathcal{S}, X')$ of online set cover we build an instance of RLP as described in the above discussion, and whenever we are presented an element $x_i \in X' \subseteq X$ we present the path P_i to ALG. We transform the regenerator assignment returned by ALG to a set cover $\mathcal{C}$ as described above. Note that the transformation does not exclude a set S_j from $\mathcal{C}$ if is was already in $\mathcal{C}$ before x_i was presented, thus $\mathcal{C}$ is an online set cover. We note that $|V| = \Theta(|X| \cdot |\mathcal{S}|), |E| = \Theta(|V|), d = \Theta(|\mathcal{S}|)$. This implies an $O(\frac{\log|X|\cdot\log|\mathcal{S}|}{\log(\log|X|\cdot\log|\mathcal{S}|)})$-competitive algorithm for the online set cover problem, which is proven to be impossible in [1]. □

4 Path Maximization in Path Topology. ($k = 1, d = 2$)

In this section we consider possibly the simplest instances of the PMP problem, i.e. the case where the network is a path, and $k = 1$, $d = 2$.

We say that an instance is feasible, if there is a regenerator assignment that d-satisfies all the paths in $\mathcal{P}$, and infeasible otherwise. We first show in Section 4.1 that if the input instance is infeasible, no online algorithm (for PMP) has a small

competitive ratio; precisely, we show that no online algorithm is better than $\sqrt{l}$-competitive, where l is the length of the longest path in the input. We then focus on feasible instances in Section 4.2.

4.1 Infeasible Instances

We show that there is a lower bound in terms of the length of the longest path if the input instance is infeasible, as follows:

Lemma 4. *Consider the path topology. For $k = 1$ and $d = 2$, any deterministic online algorithm for* PMP *has a competitive ratio at least $\sqrt{l}/2$, where l is the number of internal vertices of the longest path.*

Proof. The adversary first releases a path of length $l+1$ with l internal vertices. The online algorithm has to satisfy this path, otherwise, the competitive ratio is unbounded. Then the adversary releases $\sqrt{l}$ paths along the first path each with $\sqrt{l}$ (disjoint) internal vertices. If the online algorithm does not satisfy any of these paths, the competitive ratio is at least $\sqrt{l}$ and we are done. Suppose x of these paths are satisfied. In order to make the first path and these x paths 2-satisfied, there is one regenerator placed in each node along these x paths. For each of these x paths P, the adversary releases $\sqrt{l}/2$ paths along P each with two (disjoint) internal vertices. The online algorithm is not able to satisfy any of these short paths and the total number of 2-satisfied paths is $x + 1$. On the other hand, the optimal offline algorithm satisfies all the paths except the first path of length l, i.e., $\sqrt{l} + x\sqrt{l}/2$ paths. As a result, the competitive ratio of the online algorithm is $\frac{(x+2)\sqrt{l}}{2(x+1)} > \sqrt{l}/2$. □

4.2 Feasible Instances

We now consider feasible instances, that is, instances, where there exists a placement of regenerators such that all paths are satisfied. We will prove that, for feasible instances, there is a tight bound of 3 for the competitive ratio. That is, we provide an online algorithm Algorithm 2 with competitive ratio 3, and we show a lower bound of 3 for the competitive ratio of every deterministic online algorithm for feasible instances.

Algorithm 2 adopts a greedy approach and satisfies a newly presented path whenever possible. When a path P_i is presented, it checks whether there exist two consecutive internal vertices of P_i that are already assigned regenerators for previous paths. If yes, this means it is impossible (under the current assignment) to satisfy P_i. Otherwise, the algorithm satisfies P_i, as follows. There are two possible locations for the leftmost regenerator of P_i, namely, either its leftmost internal node, or the internal node adjacent to it. Among these two alternatives we choose the alternative that uses the smaller number of regenerators by trying the following regenerator allocation process. Suppose we put a regenerator at a certain internal node v of P_i. We check whether the node at distance 2 from v already has a regenerator; if no, we put a regenerator there and continue; if yes,

we put a regenerator at the node at distance 1 from v[1]. This continues until P_i is 2-satisfied.

Algorithm 2. Online algorithm for a path-topology, $k = 1$ and $d = 2$.

When the path P_i is presented:
if it is not possible to place regenerators to completely satisfy P_i **then**
    leave P_i unsatisfied;
else
    using the procedure described in the preamble of Algorithm 2, satisfy P_i using the smallest possible number of new regenerators
end if

Theorem 2. *Algorithm 2 is* 3*-competitive for* Pmp *for feasible inputs in path topologies, when* $k = 1$ *and* $d = 2$.

Proof. Let S and U denote the sets of paths that have been satisfied and unsatisfied by the algorithm, respectively. We prove the theorem by showing that $|U| \leq 2|S|$. Then, the competitive ratio of Algorithm 2 is $\frac{|\mathcal{P}|}{|S|} = \frac{|U|+|S|}{|S|} \leq \frac{2|S|+|S|}{|S|} = 3$, i.e., Algorithm 2 is 3-competitive. In the sequel we prove that $|U| \leq 2|S|$ by associating with every path in U some paths of S, and showing that each path in S is associated with at most two paths in U.

Note that for $d = 2$ a feasible solution can be described as follows: Remove the first and last edges of every path $P \in \mathcal{P}$ presented, and return a vertex cover of the remaining edges. Therefore, in this proof, when we refer to a path P_i, we mean the path that the leftmost and rightmost edges have been removed.

Note also that, since the instance is assumed to be feasible, for every edge uv there exist *at most* two paths P_i, P_j, such that $uv \in P_i$ and $uv \in P_j$ (indeed, otherwise there would exist at least one path that is unsatisfied on the edge uv). Suppose that a path P_i presented at iteration i is unsatisfied, i.e., when P_i arrives, it cannot be satisfied by placing new regenerators. Then, there exists an edge $ab \in P_i$, where both a and b already have regenerators of paths that have been previously satisfied by the algorithm. We distinguish now two cases regarding the regenerators on vertices a and b.

Case 1: $reg(a, P_j) = reg(b, P_h) = 1$, with $j, h < i$ and $j \neq h$, where the paths P_j, P_h have been satisfied previously by the algorithm.

We first consider the cases where $ab \in P_j$ or $ab \in P_h$. Suppose that $ab \in P_j$. Then, since also $ab \in P_i$ by assumption, it follows that $ab \notin P_h$, since the instance is feasible. That is, b is an endpoint of P_h. In this case, associate the unsatisfied path P_i to the satisfied path P_h. Suppose now that $ab \in P_h$. Then it follows similarly that $ab \notin P_j$, and thus a is an endpoint of P_j. In this case, associate the unsatisfied path P_i to the satisfied path P_j.

[1] The node at distance 1 must have no regenerator, else there are two consecutive internal nodes with regenerators and the algorithm would have rejected the path.

Suppose now that $ab \notin P_j$ and $ab \notin P_h$, i.e., a is an endpoint of P_j and b is an endpoint of P_h. If there exists another path P_ℓ that is left unsatisfied by the algorithm, such that $ab \in P_\ell$, then associate the unsatisfied paths $\{P_i, P_\ell\}$ to the satisfied paths $\{P_j, P_h\}$. Otherwise, if no such path P_ℓ exists, then associate the path P_i to either P_j or P_h.

Case 2: $reg(a, P_j) = reg(b, P_j) = 1$, where $j < i$ and the path P_j has been satisfied previously by the algorithm.

The edge $ab \in P_j$. Furthermore, neither a nor b is an endpoint of path P_j, since otherwise Algorithm 2 would not place a regenerator on both vertices a and b of path P_j. That is, there exist two vertices d, c of P_j, such that (d, a, b, c) is a subpath of P_j. Moreover, since a and b are consecutive vertices of P_j, according to the algorithm there must exist two other satisfied paths P_h, P_ℓ, such that $reg(d, P_h) = reg(c, P_\ell) = 1$.[2] Note also that $ab \notin P_h$ and $ab \notin P_\ell$, since the instance is feasible, and since $ab \in P_i$ and $ab \in P_j$. That is, d or a is an endpoint of P_h, while b or c is an endpoint of P_ℓ.

We claim that there exist *at most* two different unsatisfied paths P_i and $P_{i'}$ that include at least one of the edges da, ab, bc. Suppose otherwise that there exist three such unsatisfied paths P_i, $P_{i'}$, $P_{i''}$. Recall that $ab \in P_i$ and that $da, ab, bc \in P_j$. Therefore, since the instance is assumed to be feasible, it follows that, either $da \in P_{i'}$ and $bc \in P_{i''}$, or $bc \in P_{i'}$ and $da \in P_{i''}$. Since these cases are symmetric, we assume without loss of generality that $da \in P_{i'}$ and $bc \in P_{i''}$. In any optimal (i.e., offline) solution, at least one of $\{a, b\}$ has a regenerator for path P_j; assume without loss of generality that $reg(b, P_j) = 1$ (the other case $reg(a, P_j) = 1$ is symmetric). Then, it follows that $reg(a, P_i) = 1$. Then, since the edge da must be satisfied for both paths P_j and $P_{i'}$, it follows that $reg(d, P_j) = reg(d, P_{i'}) = 1$. This is a contradiction, since every vertex can have at most one regenerator. Therefore there exist at most two different unsatisfied paths P_i, $P_{i'}$ that include at least one of the edges da, ab, bc.

In the case that P_i is the only unsatisfied path that includes at least one of the edges da, ab, bc, associate the unsatisfied path P_i to either the satisfied path P_h or to the satisfied path P_ℓ. Otherwise, if there exist two different unsatisfied paths P_i, $P_{i'}$ that include at least one of the edges da, ab, bc, associate the unsatisfied paths $\{P_i, P_{i'}\}$ to the satisfied paths $\{P_h, P_\ell\}$.

We observe that by the above associations of unsatisfied paths to satisfied ones, that at most two unsatisfied paths are associated to every satisfied path P (i.e., at most one to the left side and one to the right side of P, respectively). This gives $|U| \leq 2|S|$ and the theorem follows. □

[2] Here we simplify the discussion slightly by assuming that the path P_i does not contain a chain of two internal edges that both do not belong to any other paths because the algorithm can simply assign regenerators to alternate internal nodes without conflicting any other paths and this would not affect the number of paths that can be satisfied by the algorithm.

Lemma 5. *Any deterministic online algorithm for* PMP *has a competitive ratio at least* 3 *even when the instance is restricted to feasible ones path topologies and* $k = 1, d = 2$.

Proof. We will prove that, for every $\varepsilon > 0$, there exists an input such that every algorithm has competitive ratio at least $3-\varepsilon$. Choose n, such that $\frac{2}{n+1} < \varepsilon$. The adversary provides initially a path P_0 with $13n - 2$ edges. The algorithm must satisfy the path P_0, since otherwise the adversary stops and the competitive ratio is infinite. We divide P_0 into n subpaths P_i, $i = 1, 2, \ldots, n$, with 11 edges each, where between two consecutive subpaths there exist two edges.

Consider any such subpath P_i, $i = 1, 2, \ldots, n$. Suppose that there exist two edges ab and cd of P_i, where $\{a, b\} \cap \{c, d\} = \emptyset$, such that $reg(a, P_0) = reg(b, P_0) = 1$ and $reg(c, P_0) = reg(d, P_0) = 1$. Then the adversary provides next the paths $P_{i,1} = (a, b)$ and $P_{i,2} = (c, d)$. These two paths $P_{i,1}$ and $P_{i,2}$ can not be satisfied, since each of the vertices a, b, c, d has a regenerator for path P_0.

Suppose that there do not exist two such edges ab and cd of P_i. That is, there exist *at most* three consecutive vertices u_1, u_2, u_3 of P_i, such that $reg(u_1, P_0) = reg(u_2, P_0) = reg(u_3, P_0) = 1$, while for every other edge uu' of P_i, there exists a regenerator for P_0 either on vertex u or on vertex u'. Then, it is easy to check that there always exist five consecutive vertices v_1, v_2, v_3, v_4, v_5 of P_i, such that $reg(v_1, P_0) = reg(v_3, P_0) = reg(v_5, P_0) = 1$ and $reg(v_2, P_0) = reg(v_4, P_0) = 0$.

The adversary now provides the path $P'_i = (v_2, v_3, v_4)$. Thus, since $reg(v_3, P_0) = 1$ and $reg(v_2, P_0) = reg(v_4, P_0) = 0$, the only way that the algorithm can satisfy P'_i is to place regenerators for P'_i at the vertices v_2 and v_4 (that is, $reg(v_2, P'_i) = reg(v_4, P'_i) = 1$).

The adversary proceeds as follows. In the case where the algorithm chooses not to satisfy the path P'_i, the adversary does not provide any other path that shares edges with P_i. Otherwise, if the algorithm satisfies P'_i, then the adversary provides the paths $P''_i = (v_1, v_2)$ and $P'''_i = (v_4, v_5)$. In this case, $reg(v_2, P'_i) = reg(v_4, P'_i) = 1$ and $reg(v_1, P_0) = reg(v_5, P_0) = 1$, and thus the paths P''_i and P'''_i remain unsatisfied by the algorithm. In the sequel we show that the instance constructed in the proof is feasible. We prove that the instance delivered by the adversary is indeed a feasible instance. To this end, we provide a placement of the regenerators such that the path P_0, as well as all paths $P_{i,1}$, $P_{i,2}$, P'_i, P''_i, and P'''_i are satisfied. First, we place a regenerator for P_0 on the vertex that lies between every two consecutive subpaths P_i and P_{i+1} of P_0. Inside the subpaths P_i of P_0, we place regenerators for P_0 on vertices with distance two between two regenerators. Then, we can assign appropriately regenerators to the paths $P_{i,1}$, $P_{i,2}$, P'_i, P''_i, and P'''_i. In particular, for every subpath P_i of P_0, for which the opponent provides the path P'_i, we have $reg(v_3, P'_i) = 1$ and $reg(v_2, P_0) = reg(v_4, P_0) = 1$. Furthermore, for the subpaths P_i of P_0, for which the opponent provides also the paths P''_i and P'''_i, we have $reg(v_1, P''_i) = reg(v_5, P'''_i) = 1$. Therefore, there exists a placement of regenerators on the vertices of the paths of the instance that the opponent delivers, such that all paths are satisfied. That is, the instance is feasible.

Denote now by h_1 the number of subpaths P_i, for which the algorithm adds the paths $P_{i,1}$ and $P_{i,2}$. Furthermore, denote by h_2 the number of subpaths P_i, for which the algorithm adds the path P'_i, but not the paths P''_i and P'''_i. Finally, denote by h_3 the number of subpaths P_i, for which the algorithm adds the three paths P'_i, P''_i, and P'''_i. Clearly, $h_1 + h_2 + h_3 = n$. The total number of paths that the adversary provided equals $1 + 2h_1 + h_2 + 3h_3$, while the number of satisfied paths equals $1 + h_3$. That is, the competitive ratio of the algorithm is $\frac{1+2h_1+h_2+3h_3}{1+h_3} \geq \frac{1+h_1+h_2+3h_3}{1+h_3} = 3 + \frac{n-h_3-2}{1+h_3}$. Therefore, since $h_3 \geq n$, it follows that the competitive ratio of the algorithm is at least $3 - \frac{2}{1+n} > 3 - \varepsilon$. Since this holds for every $\varepsilon > 0$, it follows that any deterministic online algorithm has competitive ratio at least 3. This completes the proof of the lemma. □

5 Future Work

We list some open problems and research directions:

- Close the gap between the bounds shown in this paper. In particular, we used in Section 3 a known approximation result of set cover and modified it for our problem. It might be of interest to improve the upper bound by developing a better algorithm for these special instances of the set cover problem. However we note that OnLineSetCover does not use the set of all potential elements but only its size. Therefore if the algorithm is supplied with an a priori information about the total length of the paths to be received, the algorithm can use it to get an upper bound which is logarithmic in terms of this bound, instead of the number of all possible paths of size d which can be much bigger.
- Extend the results for other values of the parameters d and k.
- Consider the regenerator location problem when also traffic grooming is allowed (that is, when up to g (the *grooming factor*) paths that share an edge can be assigned the same wavelength and can then share regenerators). In [6] optimizing the use of regenerators in the presence of traffic grooming is studied, but with two fundamental differences: (1) the cost function there is the number of locations where regenerators are used rather than the total number of regenerators suggested here, and (2) the authors consider the online case, where the requests for connection are not known a-priori, while here all requests are given in advance.
- Consider other objective functions (some of them are discussed in Section 1).

References

1. Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., Naor, S.: The online set cover problem. SIAM J. Computing 39(2), 361–370 (2009)
2. Borodin, A., El-Yaniv, R.: Online Computation and Competitive Analysis. Cambridge University Press, Cambridge (1998)
3. Chen, S., Ljubic, I., Raghavan, S.: The regenerator location problem. Networks 55(3), 205–220 (2010)

4. Fedrizzi, R., Galimberti, G.M., Gerstel, O., Martinelli, G., Salvadori, E., Saradhi, C.V., Tanzi, A., Zanardi, A.: Traffic independent heuristics for regenerator site selection for providing any-to-any optical connectivity. In: Proceedings of IEEE/OSA Conference on Optical Fiber Communications, OFC (2010)
5. Flammini, M., Marchetti-Spaccamela, A., Monaco, G., Moscardelli, L., Zaks, S.: On the complexity of the regenerator placement problem in optical networks. IEEE-TON 19(2), 498–511 (2011)
6. Flammini, M., Monaco, G., Moscardelli, L., Shalom, M., Zaks, S.: Optimizing Regenerator Cost in Traffic Grooming. In: Lu, C., Masuzawa, T., Mosbah, M. (eds.) OPODIS 2010. LNCS, vol. 6490, pp. 443–458. Springer, Heidelberg (2010)
7. Kim, S.W., Seo, S.W.: Regenerator placement algorithms for connection establishment in all-optical networks. IEE Proceedings Communications 148(1), 25–30 (2001)
8. 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)
9. Pachnicke, S., Paschenda, T., Krummrich, P.M.: Physical impairment based regenerator placement and routing in translucent optical networks. In: Optical Fiber Communication Conference and Exposition and The National Fiber Optic Engineers Conference, page OWA2. Optical Society of America (2008)
10. Sriram, K., Griffith, D., Su, R., Golmie, N.: Static vs. dynamic regenerator assignment in optical switches: models and cost trade-offs. In: Proceedings of the IEEE Workshop on High Performance Switching and Routing (HPSR), pp. 151–155 (2004)
11. Yang, X., Ramamurthy, B.: Dynamic routing in translucent WDM optical networks. In: Proceedings of the IEEE International Conference on Communications (ICC), pp. 955–971 (2002)
12. 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)