A Refined Analysis of Online Path Coloring in Trees

Astha Chauhan and N.S. Narayanaswamy(✉)

Department of Computer Science and Engineering,
Indian Institute of Technology Madras, Chennai, India
{astha,swamy}@cse.iitm.ac.in

Abstract. Our results are on the online version of path coloring in trees where each request is a path to be colored online, and two paths that share an edge must get different colors. For each T, we come up with a hierarchical partitioning of its edges with a minimum number of parts, denoted by $h(T)$, and design an $O(h(T))$-competitive online algorithm. We then use the lower bound technique of Bartal and Leonardi [1] along with a structural property of the hierarchical partitioning, to show a lower bound of $\Omega(h(T)/\log(4h(T)))$ for each tree T on the competitive ratio of any deterministic online algorithm for the problem. This gives us an insight into online coloring of paths on *each* tree T, whereas the current tight lower bound results are known only for special trees like paths and complete binary trees.

1 Introduction

The problem of path coloring in graphs has been motivated by the problem of wavelength allocation in communication networks that make use of Wavelength Division Multiplexing (WDM). In WDM, multiple optical signals are transmitted simultaneously through the same fibre link but at different wavelengths of light. Any two nodes in such a network communicate by establishing a path between them and assigning a wavelength to the path. Paths which use the same fibre link are assigned different wavelengths. A natural goal is to minimize the number of wavelengths used in such a network. This crucial problem in communication networks is known as the *wavelength allocation problem*: Given a network and a set of requests on the network, the problem is to assign distinct wavelengths to all requests that share a communication link. This problem may be viewed as the problem of coloring paths on a network graph such that two paths that share a link receive different colors (representing wavelengths in communication network). One of the most well-studied network topologies in this framework is the tree topology. Now, we formally define the *path coloring problem on trees*:

K. Jansen and M. Mastrolilli (Eds.): WAOA 2016, LNCS 10138, pp. 142–154, 2017.
DOI: 10.1007/978-3-319-51741-4_12

Path Coloring on Trees
Instance: A tree T and a set $\mathcal{P}$ of paths of T
Output: A coloring function $c : \mathcal{P} \mapsto \{1, \cdots, r\}$ for some integer $r \geq 1$ such that for every pair of distinct paths P_i and P_j in $\mathcal{P}$ with $P_i \cap P_j \neq \emptyset$, $c(P_i) \neq c(P_j)$
Goal: To obtain a function c which minimizes r.

Several researchers have extensively studied both offline and online versions of this problem on different network topologies. In the offline setting, the topology, and the entire request sequence are known in advance. However, in online wavelength allocation problems, though the topology is known in advance, requests arrive one at a time and a request has to be assigned a wavelength as soon as it is presented (this assignment cannot be changed later). In an online algorithm, the inputs arrive in a sequence and each input has to be processed depending only on already received requests and served as soon as it arrives with no knowledge of future requests. The performance of an online algorithm A is analyzed using the *competitive ratio*. It is the worst-case ratio between the cost of the solution found by the algorithm A to the cost of an optimal solution.

The path coloring problem for trees may be viewed as the vertex coloring problem for edge intersection graph of paths on a tree, called *EPT graphs*, such that no two adjacent vertices in the intersection graph get the same color. In the edge intersection graph of the given paths, two vertices are adjacent if and only if the corresponding paths have a common edge. The vertex intersection graphs of paths of a tree is the class of path graphs [5], which is a subclass of chordal graphs. Chordal graphs are the vertex intersection graphs of subtrees of a tree [4] and can be optimally colored in polynomial time [3]. Thus, in the offline setting, path graphs can be optimally colored in polynomial time. However, coloring of edge intersection graph of paths in an undirected tree (EPT graph) has been shown to be NP-complete [7]. Tarjan [16] gave a $\frac{3}{2}$-approximation algorithm for coloring EPT graphs. Erlebach and Jansen [2] showed that in the case of undirected trees of bounded degree, the path coloring problem can be solved in polynomial time. However, for undirected trees of arbitrary degree, the problem is NP-hard and approximation algorithm with absolute approximation ratio $\frac{4}{3}$ and asymptotic approximation ratio $\frac{11}{10}$ are known [2]. Path coloring is also proved to be NP-hard on undirected and bi-directed ring networks [2], bi-directed binary trees [2] and bi-directed binary caterpillars [15]. Several interesting approaches have been proposed in the literature([2,8,10,11,14,15]) for network topologies like rings, caterpillars, trees and trees of rings. If the tree itself is a path, then the edge intersection graph of subpaths of this path is an interval graph [6]. Therefore, in this case, coloring algorithms for the vertex intersection graphs can be used for optimally coloring the edge intersection graphs. The coloring of the vertex intersection graphs on the line topology (special case of tree topology, i.e., a tree in which no vertex has degree 3 or more) is very well studied. It is known to be optimally polynomial time solvable in the offline setting.

The path coloring problem is also extensively studied in the online framework. Bartal and Leonardi [1] proposed an $O(\log(n))$ competitive online algorithm

(described in Sect. 2 and subsequently called as the BL algorithm) for coloring edge intersection graphs of paths on a tree. They also [1] gave a lower bound of $\Omega(\frac{\log n}{\log \log n})$ for the online path coloring problem on complete binary trees. The path coloring problem on the line topology has been studied in the context of interval graph coloring. The edge intersection graph and vertex intersection graph of paths on line topology are both known to be interval graphs [6]. Thus, the problem of online coloring paths on the line topology is equivalent to the problem of online coloring vertices of interval graph which is a subclass of chordal graphs. One of the simplest strategies adopted for the online coloring of vertices in an interval graph is the First Fit strategy: allocate the color of least index permissible. Several researchers have analyzed this method and have come up with improved bounds over the years and 8 is the best known competitive ratio [12]. However, a 3-competitive recursive algorithm by Kierstead and Trotter [9] (subsequently called as the KT algorithm) for the online coloring of interval graphs, was known much earlier (a detailed presentation of the algorithm can be found in [13]). They showed a matching lower bound as well; that is, no deterministic online algorithm can achieve a competitive ratio better than 3.

1.1 Our Motivation and Results

We address the problem of online path coloring in trees and design algorithm that uses the Kierstead-Trotter approach [9] to color paths in the line topology. Throughout the paper, we consider the coloring problem on edge intersection graphs. For the online path coloring in the line topology, we use the optimal KT algorithm on the edge intersection graph, which is an interval graph. The starting point of our work is the BL Algorithm [1], which achieves a competitive ratio of $\log n$ for online path coloring in trees. We show that this algorithm can be forced to achieve $\Omega(\log(n))$ competitive ratio when applied to coloring paths in the line topology. However, this is far from the performance of the optimal 3-competitive algorithm by Kierstead and Trotter [9]. Our motivation is to understand this gap between performance of KT-algorithm and BL-algorithm for line topology. We present a simple online algorithm to solve the problem of online path coloring on caterpillars using at most $(5\omega - 3) < h(T)(5\omega - 3)$ colors, whereas the BL algorithm can be forced to use $\omega \log n$ colors by an adversary, details of which are given in Sect. 2.

For an arbitrary tree T, we define a hierarchical partition of the vertex set which we refer as the Hierarchical Path Partition (HPP). We associate a caterpillar with each part in the HPP, which results in a partition of the edge set of tree T. We call this edge partition a Hierarchical Caterpillar Partition (HCP) and we use an HCP in an online algorithm to color the path requests. In this online algorithm, we follow the template of Bartal and Leonardi [1]: each level in the HCP uses a distinct set of colors, and each coloring request is colored as a path coloring request at the highest level in which it intersects (has an edge in common) with a caterpillar. We denote by $h(T)$ the number of parts in an HPP with the minimum number of parts, and our algorithm uses at most $h(T)(5\omega-3)$ colors, where ω is the size of the maximum clique in the edge intersection graph

of the given set of path coloring requests. Since ω is a lower bound on the number of colors to be used, our algorithm is an $O(h(T))$ competitive algorithm. It also gives us a refined understanding of the performance of the algorithm based on structure of tree T. To the best of our knowledge, the concept of hierarchical path partitioning is new and we believe it could be of significant interest in designing online algorithms for different problems on trees.

We also present an algorithm that computes an HPP with $h(T)$ parts in polynomial time. This algorithm also serves as a characterization of the optimum HPP. We then show that the optimum HPP has a subtree that *looks* like a complete binary tree of depth $h(T) - 1$. We refer to this as a *complete pseudo binary tree*. We use this subtree in conjunction with the lower bound argument due to Bartal and Leonardi [1] to show a competitive ratio lower bound of $\Omega(\frac{h(T)}{\log(4h(T))})$ for any deterministic online algorithm. In particular, our results nicely generalize the results of Bartal and Leonardi [1]- we design a $h(T)$ competitive online algorithm for path coloring in T, and also show a lower bound of $\Omega(\frac{h(T)}{\log(4h(T))})$ on the competitive ratio of any deterministic algorithm.

Definitions and Notation: We use standard graph theoretic concepts like *graph* G, *vertex set* $V(G)$, *edge set* $E(G)$, *degree* $deg_G(v)$ of a vertex v, *neighborhood* $N_G(v)$ of a vertex v, *diameter* Δ, *path* $P = [v_1, v_2, \cdots, v_k]$ and *tree* T from the textbook by Douglas B. West [17]. The size of the largest clique in G is called its *clique number* and is denoted by $\omega(G)$. We use ω to denote the clique number of the edge intersection graph of input path requests of the underlying tree T. A caterpillar is a tree that has a dominating path. This dominating path is called the spine and an edge for which exactly one vertex is on the spine is called a hair. By definition all leaves other than the two on the spine are adjacent to a vertex on the spine. A *balanced tree separator* is a vertex whose removal splits the tree into multiple disjoint trees to form a forest such that each tree in that forest consists of at most $\frac{2}{3}n$ vertices. We denote path coloring requests by P and paths which are not path coloring requests by p.

Online Interval Coloring: Online path coloring in a tree T which is a path is the well-studied Online Interval Coloring problem. Kierstead and Trotter [9] gave a 3-competitive algorithm for the online interval coloring, which uses atmost $3\omega - 2$ colors where ω is the maximum number of pairwise intersecting intervals. They also showed a matching lower bound; that is, no deterministic online algorithm can achieve a competitive ratio better than 3. We refer to this online algorithm as the KT Algorithm.

2 Caterpillar Based Online Coloring of Paths in Trees

The main result in this section is our online algorithm for coloring paths in a tree T by partitioning the edges of T into parts, each of which is a set of vertex disjoint caterpillars. We show that this algorithm has a competitive ratio of $h(T)$, where $h(T)$ is a combinatorial parameter associated with T. The algorithm is based on our observation that the online path coloring algorithm (referred to

as the BL Algorithm) due to Bartal and Leonardi [1] essentially maintains a partition of the edges of T, such that each part in the partition is a set of vertex disjoint stars - a special type of caterpillar.

A Worst Case Instance for the BL Algorithm: The first observation we make is based on the construction of a sequence of coloring requests in an n-vertex path to show that the BL algorithm has an $\Omega(\log n)$ competitive ratio. The BL algorithm has two phases: a preprocessing phase and a coloring phase. The preprocessing phase partitions the edges of T into $\log n$ levels. We refer to this partition as the BL partition. The preprocessing phase and the coloring phase are standard throughout all the algorithms we present, and the preprocessing phase depends only on the tree T and not on the path coloring requests.

Algorithm 1. BL

1: *Preprocessing Phase:* The first level L_0 consists of a single vertex s which is a balanced separator of T. Iteratively, for $i \geq 1$, the i^{th} level L_i consists of balanced separators of all the subtrees in $T \setminus \bigcup_{0 \leq j \leq i-1} L_j$. In this way, vertices of T are partitioned into $\Theta(\log n)$ levels. The edge partition is obtained by associating with each vertex the set of incident edges whose other end point is at a higher level. Indeed, it is clear that each level is one set in a partition of the edges, and that the edges associated with a vertex at any level forms a star.

2: *Coloring Phase:* When a coloring request for a path P arrives, the algorithm first assigns a level identifier to it. This identifier is the minimum level number of a level that contains a vertex of P. Then, P is assigned the minimum color which is not assigned to any other previously colored path that has a common edge with P and has the same level identifier.

Theorem 1 (Bartal and Leonardi [1]). *The BL algorithm for online path coloring on a tree of n vertices uses at most $(2\omega - 1)\log n$ colors. Thus, the algorithm achieves a competitive ratio of $O(\log n)$.*

We now observe that BL algorithm has a competitive ratio of $\Omega(\log n)$ even if T is a path. On the other hand, the KT algorithm uses only $3\omega - 2$ colors to color paths from T if it is a path. We present an input instance generated by an adversary that forces the BL Algorithm to use $(\log n)OPT$ colors, where OPT is the number of colors in the optimal coloring. For an integer $k > 1$, consider the path T having $n = 2^k - 1$ vertices v_1 to v_{2^k-1}. Since there are $2^k - 1$ vertices, in the preprocessing phase, the vertices are partitioned into $L \geq k$ levels in any BL partition. Now, for each level $0 \leq l < L - 1$ in the BL partition, the adversary selects one edge of T with one end point at level l and the other end point at a level at least $l + 1$. Let these edges be $e_0, \ldots, e_{L-2}$. Further v be a vertex in level L (note that it has index $L - 1$) in the BL partition. In the path coloring sequence, there are $k(L - 1)$ paths consisting of exactly one edge and k paths consisting of exactly one vertex as follows: k paths consisting of e_0, followed by

k paths consisting e_1, and so on, ending with k paths consisting of e_{L-2} and, k single vertex paths consisting of v. Since the BL algorithm uses a distinct set of colors for each of the L levels, it uses at least $kL \geq (\log n)k = (\log n)OPT$.

Online Path Coloring in Caterpillars: The BL algorithm partitions the edges of T into $O(\log n)$ levels of stars. We explore if we can get a better competitive ratio by maintaining a set of caterpillars in each level. To achieve this, we first observe that paths in caterpillars can be colored by an online algorithm of competitive ratio 5. We refer to the algorithm below as Algorithm OPCC.

Algorithm 2. *OPCC- Online Path Coloring in Caterpillars*

1: *Preprocessing Phase:* In the preprocessing phase, we partition the edges of T into two sets: E_s is the set of all the edges on the spine and E_h is the set of all hairs.
2: *Coloring Phase:* We use two disjoint set of colors, C_s to color paths which have an edge in E_s and C_h to color paths whose edges are in E_h. When a coloring request for a path P arrives, it is colored as follows:
Case 1: P contains an edge in E_s- In this case, we consider the subpath P' of P formed by the edges in E_s as an online path coloring request on the spine. P' is colored using the KT algorithm [9] (see Sect. 1.1)and the color of P' is the color given to P.
Case 2: P does not contain any edge in E_s- In this case P must belong to some star in $E \setminus E_s = E_h$. P is greedily colored such that it gets a color different from paths colored earlier with which it shares an edge.

Lemma 1. *Let T be a caterpillar. Algorithm OPCC requires at most $5\omega - 3$ colors to color path coloring requests on T.*

Proof. The paths which fall into the first case are colored by the KT Algorithm [9] which uses at most $3\omega - 2$ colors. Secondly, the greedy algorithm on the paths colored in case 2 will use at most $2\omega - 1$ colors. The reason is that each such path has at most two edges, and if a path coloring request P cannot be colored by any of the already used $2\omega - 1$ colors, then one of the two edges in P is already in at least ω paths that have already been colored. This edge would then be common to $\omega + 1$ paths, contradicting the definition of ω. Hence, at most $5\omega - 3$ colors are required to color all input paths. □

2.1 A New Online Path Coloring Algorithm for Trees

While the best competitive ratio for online path coloring in caterpillars is at most 5, we have shown that the BL algorithm has an $\Omega(\log n)$ competitive ratio when T is a caterpillar (because a simple path is a special caterpillar). We have also observed that for each tree T, the BL algorithm maintains a partition of the edge set into levels such that in each level, the edges form a set of vertex disjoint stars. We now present our algorithm where in the preprocessing phase, we maintain a partition of the edge set of T into levels such that in each level,

the edges form a set of vertex disjoint caterpillars. In a way, this can be seen as a strengthening of the BL Algorithm. Then, by using the online coloring algorithm on caterpillars, we show that our online algorithm achieves a competitive ratio of $h(T)$, where $h(T)$ is defined as the number of levels in an optimal partitioning of the vertices that we call the Hierarchical Path Partition.

Hierarchical Path Partitioning: For a tree T, we call a partition $\{H_1, H_2 \ldots, H_h\}$ of $V(T)$ a *hierarchical path partitioning (HPP)* of T if the following properties hold:

- **P.1** Each set H_i induces a set of vertex disjoint paths in T. The paths in H_i are denoted by $\{p_{i,1}, p_{i,2}, \cdots, p_{i,l_i}\}$.
- **P.2** The sets are arranged hierarchically with one root set H_h. Note that at this place we have an important notational difference from Bartal and Leonardi [1] who use the level index 0 for the root.
- **P.3** Root set H_h contains a single path.
- **P.4** For each $i < h$, if p is a path in set H_i, then there is exactly one edge from exactly one of the vertices of path p to one vertex of a path $p' \in H_j$ where $j > i$. Further, this edge is incident on one of the end vertices of p. We refer to p' as the parent path of p.
- **P.5** For each $i > 1$, if p is a path in set H_i, then there are at least two edges from one endpoint of p to endpoints of paths p' and p'' in H_j where $j < i$. We refer to p', p'' as the children of p.

H_i in the partition is referred to as *level* i in the hierarchy. We use the HPP output by the following algorithm and the number of parts h output by this algorithm is referred to as $h(T)$.

Algorithm 3. OHPP: Optimum Hierarchical Path Partitioning

We define a sequence of non-empty subtrees $\{T_i\}$ of tree T where $T_1 = T$. The vertex set of T_i is $V(T_{i-1}) \setminus H_{i-1}$. For some $h \geq 1$, if T_h is a path, then H_h consists of the single path, and the algorithm stops and outputs $\{H_1, H_2 \ldots, H_h\}$. For each $i \geq 1$, $H_i = \{p_{i,1}, p_{i,2} \cdots, p_{i,l_i}\}$ is a set of vertex disjoint paths, such that for each $p_{i,j} \in H_i$, one endpoint is a leaf in T_i, the other endpoint of $p_{i,j}$ has a neighbor in T_i whose degree in T_i is at least 3, and all other vertices in $p_{i,j}$ have degree 2 in T_i. Clearly, $T_{i+1} = T_i \setminus H_i$ induces a subtree of T_i. The algorithm is illustrated in Fig. 1.

From an HPP we naturally obtain a unique hierarchical partition of the edge set of T into vertex disjoint sets of caterpillars as follows: For each path $p_{i,j} \in H_i$, $1 \leq i \leq h$, we associate a caterpillar $c_{i,j}$ by taking $p_{i,j}$ to be the spine and every other edge e incident on vertices of $p_{i,j}$ such that other vertex of e is on a path in H_k for some $k < i$. This produces a hierarchical partitioning of the edge set of T into caterpillars called *Hierarchical Caterpillar Partitioning (HCP)* of T. Let $C_1, \ldots, C_h$ denote this family of sets of caterpillars corresponding to HPP $H_1, \ldots, H_h$.

Online Path Coloring Using a HCP: We now describe our Algorithm OPCT to color path requests in a given tree T. As in the BL Algorithm 1, we have a preprocessing phase and a coloring phase.

Algorithm 4. *OPCT- Online Path Coloring in Trees*

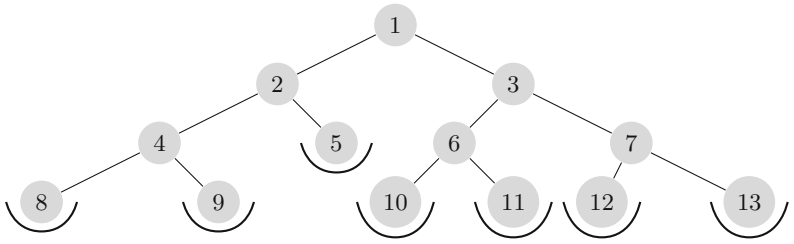
1: *Preprocessing phase:* Compute the natural (as described above) HCP $\{C_1, \ldots, C_h\}$ of the given tree T from an Optimum HPP computed using Algorithm OHPP described in Sect. 3. Here $h = h(T)$.
2: *Coloring Phase:* For each level in the HCP, a distinct set of colors is used. When a coloring request for a path P arrives, it is assigned a level number q which is the maximum among all levels with which path P intersects at some edge in the level. Let q be the largest level such that there is a caterpillar $c_{q,j} \in C_q$ with which P intersects at an edge in $c_{q,j}$ (In the proof below, we use q to denote this level associated with a coloring request P. For example, for a path P_1, q_1 is used to denote this level). Then, the subpath P' of P consisting of edges of $E(P) \cap E(c_{p,j})$ is considered as a path coloring request in $c_{p,j}$ (In the proof below, we use P' to denote the coloring request in $c_{q,j}$ associated with P. For example, for a path P_1, P_1' is used to denote the coloring request in $c_{q_1,j}$). It is colored using Algorithm OPCC, and the color given to P' is taken to be the color of P.

Theorem 2. *Let T be a tree, then online Algorithm OPCT requires at most $h(5\omega - 3)$ colors to compute a valid coloring on an online request sequence of paths from tree T such that any edge is present in at most ω paths.*

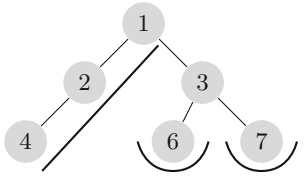
Proof. To prove that the coloring is valid, we first need to observe that any path P intersects at an edge with exactly one caterpillar in C_q. We know from Lemma 1 that the path coloring request sequence in each caterpillar in each level of the HCP uses at most $5\omega - 3$ colors. Since the HCP is an induced set of vertex disjoint caterpillars (therefore, edge disjoint), it follows that the path coloring request sequence to each level of the HCP uses at most $5\omega - 3$ colors. Since each level uses a distinct set of colors, it follows that Algorithm OPCT uses at most $h(5\omega - 3)$ colors. Let P_1 and P_2 be two path coloring requests that share an edge in T. We show that the color given to P_1 and P_2 are distinct. If q_1 and q_2 are different, then P_1' and P_2' get different colors and therefore P_1 and P_2 get different colors. In the case when $q_1 = q_2$, since P_1 and P_2 intersect at an edge, it follows that they intersect with the same caterpillar $c_{q_1,j} \in C_{q_1}$. Further, it is easy to see that they also share a common edge in $c_{q_1,j}$. Thus it follows that P_1' and P_2' get different colors, and consequently P_1 and P_2 get different colors. Hence the theorem. □

3 A Lower Bound for Deterministic Online Algorithms Using $h(T)$

In this section we prove a lower bound on the competitive ratio of deterministic online algorithms as a function of $h(T)$. We start by illustrating in Fig. 1 a HPP computed by Algorithm 3 and observing some bounds on $h(T)$.



(a) Tree T= T_1 with set $H_1 = \{[8], [9], [5], [10], [11], [12], [13]\}$



(b) Tree T_2 with set $H_2 = \{[4, 2, 1), [6], [7]\}$

(c) Tree T_3 with set H_3=[3]

Fig. 1. Tree T with partition $\{H_1, H_2, H_3\}$. (a) T_1 with H_1, (b) T_2 with H_2, (c) T_3 with H_3

Some Bounds on $h(T)$: We observe that, for a tree T having L leaves, $h(T) \leq \log_2 L$, as the number of leaves at each level reduce by a factor of at least 2. Formally, let l_i be the number of leaves in T_i. Then, $l_{i+1} \leq \frac{l_i}{2}$. Similarly, the diameter of T_{i+1} is at most diameter of T_i minus 2. Therefore, $h(T) \leq \frac{\Delta(T)}{2}$, where Δ is the diameter.
We next show that the HPP computed by Algorithm OHPP can be used by an adversary to ensure that any deterministic online algorithm has a *bad* competitive ratio.

3.1 A Lower Bound Based on $h(T)$

To prove our lower bound on deterministic online algorithms, we use the lower bound technique of Bartal and Leonardi [1] on path coloring requests on a complete binary tree of n nodes. To use this, we show that we can perform their adversarial lower bound argument on a complete pseudo binary tree which we show is present in the output of Algorithm $OHPP$. Intuitively, the subtree of T that we take will turn out to be a tree that looks like a complete binary tree whose vertices correspond to paths in which each internal vertex has degree 2. We now describe this Complete Pseudo Binary Tree. Consider the partitioning obtained after applying the algorithm $OHPP$ to T. We know that the output of Algorithm $OHPP$ is an optimum HPP $H_1, \cdots, H_h$. We call a family $\{S_1, S_2 \ldots, S_h\}$ of sets of paths a *Complete Pseudo Binary Tree* if the following properties hold.

- For each $1 \leq i \leq h$, $S_i \subseteq H_i$, S_i consists of 2^{h-i} paths, and $S_h = H_h$.
- Let v be one end point on the path $p \in H_h$. Then, S_{h-1} consists of exactly 2 paths, say p' and p'' such that v is adjacent to one end point each of p' and p''.

- For each $1 \leq i \leq h-1$, in every path p in S_{i+1}, there are two edges from one end point of p to the end points of two paths in S_i.
- For each $1 \leq i \leq h-1$, in every path p in S_i there is an edge from one end point of p to the end point of a path in S_{i+1}.

Lemma 2. *Let T be a tree, $h = h(T)$, and let $H_1, \ldots, H_h$ be the optimal HPP obtained from Algorithm OHPP. T has a subtree which is a complete pseudo binary tree of height h.*

Proof. We prove our claim by induction on the number of levels $h(T)$ in the partition of T produced by our algorithm. Let $h = h(T)$. Let $L(h)$ be the statement that there exists a complete pseudo binary tree in the hierarchical path partitioning of at most h levels. The base case is the statement $L(1)$. Then, $L(1)$ is true because there does exist a complete pseudo binary tree having 1 pseudo-node. Suppose the statement $L(h)$ is true. We now prove the statement $L(h+1)$. Let $\{H_1, \ldots, H_{h+1}\}$ be the hierarchical path partitioning having $h+1$ levels output by Algorithm OHPP. Now, consider the tree $T_2 = T \setminus V(H_1)$ with HPP $\{H_2, \ldots, H_{h+1}\}$ obtained after removing all the paths in H_1. By the induction hypothesis, there exists a complete pseudo binary tree for T_2 in $H_2, \ldots, H_{h+1}$. Let this complete pseudo binary tree be $S_2, \ldots, S_{h+1}$. Now, consider the paths in S_2. For each $p \in H_2$, let l_p be the leaf endpoint of p in T_2. By the definition of an HPP, l_p is incident on at least two edges, for each of which the second vertex (the one different from l_p) is an endpoint of a path in H_1. We construct $S_1 \subseteq H_1$ by taking any two such paths for l_p for each $p \in S_2$. Now, $S_1, \ldots, S_{h+1}$ is a complete pseudo binary tree for T. Thus, there exists a complete pseudo binary tree of T in the output of Algorithm OHPP with at most $h+1$ levels. Hence the lemma. □

We now use this complete pseudo binary tree to get our lower bound on deterministic online algorithms for path coloring on a tree T. As mentioned before, we essentially plug this complete pseudo binary tree into the lower bound argument of Bartal and Leonardi [1].

Theorem 3. *Let T be a tree. Then any deterministic online path coloring algorithm has a competitive ratio of $\Omega(\frac{h(T)}{\log 4h(T)})$.*

Description of the Adversarial Path Coloring Request Sequence: Before we present a proof of this theorem, we describe the sequence of requests presented by an adversary to a deterministic online path coloring algorithm. The lower bound is against a deterministic online algorithm and is established by an adversary by using the complete pseudo binary tree $S = \{S_1, S_2 \ldots, S_h\}$ contained in the optimal HPP $\{H_1, H_2 \ldots, H_h\}$ output by Algorithm OHPP. This complete pseudo binary tree has $h(T)$ levels, with S_h containing the root pseudo-node, and S_1 containing all the leaf pseudo-nodes. Each pseudo-node here corresponds to a path in S. We consider a complete binary tree T' of depth $h(T)-1$ where there is a bijective correspondence between the pseudo-nodes and the vertices of T', that respects the parent-child relationships between the paths in the family S.

For example, the root of T' corresponds to the path in S_h, and the leaves of T' corresponds to the paths in S_1. A path coloring request in T' is converted to a path coloring request in T by *expanding* the pseudo-nodes into the corresponding paths in the family S. In the following let ρ denote the best competitive ratio possible by any deterministic online algorithm.
We now present the request sequence generated by Bartal and Leonardi [1] on T'. The description is identical to the description in [1] except in the Lemma 3 where we reason about the path coloring requests in T obtained from the requests in T' as described below. The sequence of requests for coloring paths is generated in stages. Maintain the following invariant at the end of any stage $i \geq 0$: There exists a set C_i of i colors, a level $\overline{l_i}$, $l_i \leq \overline{l_i} \leq h(T) - 1 - i$, such that there are at least $r_i = \frac{2^{h(T)-1}}{(8\rho h(T))^i}$ pairs of paths with the following properties:

1. Each pair is formed by the two paths from two leaves in T' to their least common ancestor (LCA) at level $\overline{l_i}$.
2. Each vertex of level $\overline{l_i}$ in T', is the LCA of at most one pair of paths.
3. For any path in the r_i pairs of paths and for any color $c \in C_i$, there is one edge in the path included in a path coloring request with color c.
4. In T', any edge of a path is included in at most one request.

At stage 0, $\overline{l_i} = l_0 = h(T) - 1$ and $C_0 = \phi$. We associate a set of $r_0 = 2^{h(T)-1}$ pairs of empty paths, two with each leaf, with both endpoints equal to leaf itself. No path coloring requests are presented. Hence all 4 properties trivially hold. At stage $i + 1$, r_i new path coloring requests are presented, one for each pair of paths. Let u_1, u_2 be the two leaves that are endpoints of the two paths of a pair, and let $LCA(u_1, u_2)$ be the LCA at level $\overline{l_i}$ of these two leaves. Let v be the direct ancestor of $LCA(u_1, u_2)$. For each pair of paths we present a path coloring request having as endpoints one of the two leaves, say u_1, and v. The online algorithm must color the set of path coloring requests presented at stage $i + 1$. Clearly, due to the 4 invariants being respected at the beginning of stage $i+1$, any color in C_i cannot be used for these path coloring requests. These path coloring requests on T' are converted to path coloring requests in the complete pseudo binary tree. We will show in Lemma 3 that optimal number of colors at any stage of the sequence is at most 2 in the complete pseudo binary tree. Hence, in order for the online algorithm to be ρ competitive, it must use less than 2ρ colors for this set of path coloring requests. Therefore, by the pigeonhole principle there must be a set of path coloring requests R_i of cardinality at least $\frac{r_i}{2\rho}$ assigned with the same color. Let us call this color c_{i+1} and let $C_{i+1} = C_i \cup \{c_{i+1}\}$. In the following, we concentrate on this set of path coloring requests. As shown in Bartal and Leonardi [1] we first identify a set of paths in R_i that satisfy condition 3 and 4. Recall that each path coloring request in R_i is from a leaf u_1 to a node v, is constructed from some level i pair of paths from the leaves u_1 and u_2 to $LCA(u_1, u_2)$, which is a child of v. Moreover, for any pair only one path coloring request in R_i is constructed. Therefore, for any path coloring request in R_i, conditions 3 and 4 are satisfied at stage $i + 1$ for the path connecting the leaf u_2 to v or any ancestor of v. In fact, at most one

path coloring request includes any edge from u_2 to $LCA(u_1, u_2)$ and any color $c \in C_i$ is associated with a path coloring request that crosses only one edge in the path, using the invariant for level i. Moreover, the edge from $LCA(u_1, u_2)$ to v is associated with only one path coloring request with color c_{i+1}. We thus have a set of $\frac{r_i}{2\rho}$ paths satisfying conditions 3 and 4 of the invariant. We call this set of paths P_{i+1}.

The level l_{i+1} is now selected such that $2^{l_{i+1}+1} \leq \frac{r_i}{2\rho}$. We derive from P_{i+1} a new set P'_{i+1} as follows: we consider each path in P_{i+1} according to its ancestor in level l_{i+1}. If a vertex in level l_{i+1} is an ancestor of an odd number of paths we exclude one of these paths. Since the number of vertices of level l_{i+1} is at most $\frac{1}{2}\frac{r_i}{2\rho}$, the cardinality of P'_{i+1} is at least $\frac{1}{2}\frac{r_i}{2\rho}$.

We now scan paths in P'_{i+1} from left to right, following the order of the leaves that are endpoints of those paths. We associate each pair of successive leaves with their LCA, that is a vertex of level between l_{i+1} and $h(T) - i - 1$. Again, as shown in [1] we know that each vertex in a binary tree is the LCA of at most one pair of successive leaves.

Finally, let $\overline{l_{i+1}}$ be a level between $h(T) - i - 1$ and l_{i+1}, achieving the maximum cardinality set of pairs of successive paths that have LCA at that level. We define the set of pairs of paths for the stage $i + 1$ to be the set of pairs of successive paths that have LCA at level $\overline{l_{i+1}}$. Since the number of levels is $h(T)$, it follows that the number of pairs at stage $i + 1$ is atleast $\frac{1}{4}\frac{r_i}{2\rho h(T)} = \frac{2^{h(T)-1}}{(8\rho h(T))^{i+1}} = r_{i+1}$. From the above construction, it follows that both conditions 1 and 2 hold for this set of pairs. Therefore, the four invariants are satisfied at the beginning of stage $i + 1$.

We now come to the crucial and only modification to the argument of Bartal and Leonardi [1].

Lemma 3. *The optimal solution in the complete pseudo binary tree for the path coloring sequence described above uses at most 2 colors.*

Proof. The proof is obtained by the fact that in T' each edge is included in at most 2 path coloring requests, and that all such path coloring requests are directed from a leaf to an ancestor. Further, at each vertex in the tree, at most one edge to a child is present in the path coloring requests. Therefore, when these requests are considered as requests in the complete pseudo binary tree, each edge is present in at most two path coloring requests. Therefore, the paths can be colored offline with 2 colors. □

Proof of Theorem 3: Let ρ be the competitive ratio of the best deterministic online algorithm for coloring paths in tree T. Clearly, $\rho \leq \frac{h(T)(3\omega-2)}{\omega}$. Therefore, $\rho \leq 2h(T)$. The online algorithm uses at least i colors after i stages of the construction above. Hence by Lemma 3, the competitive ratio $\rho \geq \frac{i}{2}$. The lower bound on ρ is thus obtained by computing the maximum number of stages in the sequence. To carry out the sequence, we require that $l_i = h(T) - 1 - i\log(8\rho h(T)) \geq 1$. Since $\rho \geq \frac{i}{2}$, we get $\rho \geq \frac{h(T)-2}{2\log(8\rho h(T))}$. Since we know that $\rho \leq 2h(T)$, it follows that $\rho \geq \frac{h(T)-2}{2\log(16h(T)^2)}$. Therefore, ρ is $\Omega(\frac{h(T)}{\log 4h(T)})$. Hence the theorem. □

Acknowledgments. We would like to thank Krithika Ramaswamy, Dhannya S.M., and Stefano Leonardi for helpful suggestions.

References

1. Bartal, Y., Leonardi, S.: On-line routing in all-optical networks. Theor. Comput. Sci. **221**(1–2), 19–39 (1999)
2. Erlebach, T., Jansen, K., Elvezia, C.: The complexity of path coloring and call scheduling. Theoret. Comput. Sci. **255**, 2001 (2000)
3. Gavril, F.: Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. SIAM J. Comput. **1**(2), 180–187 (1972)
4. Gavril, F.: The intersection graphs of subtrees in trees are exactly the chordal graphs. J. Comb. Theor. Ser. B **16**(1), 47–56 (1974)
5. Gavril, F.: A recognition algorithm for the intersection graphs of paths in trees. Discrete Math. **23**(3), 211–227 (1978)
6. Golumbic, M.C., Jamison, R.E.: Edge and vertex intersection of paths in a tree. Discrete Math. **55**(2), 151–159 (1985)
7. Golumbic, M.C., Jamison, R.E.: The edge intersection graphs of paths in a tree. J. Comb. Theo. Ser. B **38**(1), 8–22 (1985)
8. Kaklamanis, C., Persiano, P.: Efficient wavelength routing on directed fiber trees. In: Diaz, J., Serna, M. (eds.) ESA 1996. LNCS, vol. 1136, pp. 460–470. Springer, Heidelberg (1996). doi:10.1007/3-540-61680-2_75
9. Kierstead, H.A., Trotter, W.T.: An extremal problem in recursive combinatorics. Congressus Numerantium **33**(143–153), 98 (1981)
10. Kumar, V., Schwabe, E.J.: Improved access to optical bandwidth in trees. In Proceedings of SODA 1997, pp. 437–444 (1997)
11. Mihail, M., Kaklamanis, C., Rao, S.: Efficient access to optical bandwidth. In: Proceedings of the 36th Annual Symposium on Foundations of Computer Science, FOCS 1995, p. 548. IEEE Computer Society, Washington, DC (1995)
12. Narayanaswamy, N.S., Subhash, R.: Babu. A note on first-fit coloring of interval graphs. Order **25**(1), 49–53 (2008)
13. Pemmaraju, S.V., Raman, R., Varadarajan, K.R.: Buffer minimization using max-coloring. In: Ian Munro, J. (ed.) SODA, pp. 562–571. SIAM (2004)
14. Raghavan, P., Upfal, E.: Efficient routing in all-optical networks. In: Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing, STOC 1994, pp. 134–143. ACM, New York (1994)
15. Takai, H., Kanatani, T., Matsubayashi, A.: Path coloring on binary caterpillars. IEICE Trans. **89–D**(6), 1906–1913 (2006)
16. Tarjan, R.E.: Decomposition by clique separators. Discrete Math. **55**(2), 221–232 (1985)
17. West, D.B.: Introduction to Graph Theory, 2nd edn. Prentice Hall, Upper Saddle River, NJ (2000)