Approximation Algorithms for Path Coloring in Trees*

Ioannis Caragiannis[1], Christos Kaklamanis[1], and Giuseppe Persiano[2]

[1] Research Academic Computer Technology Institute and
Department of Computer Engineering and Informatics,
University of Patras, 26500 Rio, Greece
{caragian, kakl}@ceid.upatras.gr
[2] Dipartimento di Informatica ed Applicazioni,
Università di Salerno, 84081 Baronissi, Italy
giuper@dia.unisa.it

Abstract. The study of the path coloring problem is motivated by the allocation of optical bandwidth to communication requests in all-optical networks that utilize Wavelength Division Multiplexing (WDM). WDM technology establishes communication between pairs of network nodes by establishing transmitter-receiver paths and assigning wavelengths to each path so that no two paths going through the same fiber link use the same wavelength. Optical bandwidth is the number of distinct wavelengths. Since state-of-the-art technology allows for a limited number of wavelengths, the engineering problem to be solved is to establish communication minimizing the total number of wavelengths used. This is known as the *wavelength routing problem*. In the case where the underlying network is a tree, it is equivalent to the path coloring problem.

We survey recent advances on the path coloring problem in both undirected and bidirected trees. We present hardness results and lower bounds for the general problem covering also the special case of sets of symmetric paths (corresponding to the important case of symmetric communication). We give an overview of the main ideas of deterministic greedy algorithms and point out their limitations. For bidirected trees, we present recent results about the use of randomization for path coloring and outline approximation algorithms that find path colorings by exploiting fractional path colorings. Also, we discuss upper and lower bounds on the performance of on-line algorithms.

1 Introduction

1.1 Motivation

Optical fiber has been established as the standard transmission medium for backbone communication networks, since it can provide the required data rate, error rate and delay performance necessary for high speed networks of next

* This work was partially supported by the European Union under IST FET Project CRESCCO.

E. Bampis et al. (Eds.): Approximation and Online Algorithms, LNCS 3484, pp. 74–96, 2006.

generation [19,36]. Multiwavelength communication [19,36] is the most popular communication technology used on optical networks. Roughly speaking, it allows to send different streams of data on different wavelengths along an optical fiber. Multiwavelength communication is implemented through *Wavelength Division Multiplexing* (WDM).

In a WDM all-optical network, once the data stream has been transmitted in the form of light, it continues without conversion to electronic form until it reaches its destination. WDM takes all data streams travelling on an incoming link and routes each of them to the right outgoing link, provided that each data stream travels on the same wavelength on both links. For a packet transmission to occur, a transmitter at the source must be tuned to the same wavelength as the receiver at the destination for the duration of the packet transmission and no data stream collision may occur at any fiber.

A WDM all-optical network can thus be modelled as a graph, where nodes of the graph are the nodes of the network and edges are optical fibers connecting nodes. Communication requests are ordered pairs of nodes to be thought of as transmitter-receiver pairs. The problem of *wavelength routing* consists of finding, for each transmitter-receiver pair, a path connecting the transmitter with the receiver and assigning a wavelength to each path, so that no two paths going through the same edge (fiber link) use the same wavelength. Intuitively, we may think of the wavelengths as colors and the wavelength assignment to paths as coloring. So, the wavelength routing problem consists of two subproblems: a *routing* problem (for selecting the path for each transmitter-receiver pair) and a *wavelength assignment* or *path coloring* problem (for coloring the paths established after solving the routing subproblem).

The objective is to minimize the *optical bandwidth*, that is the number of different wavelengths (colors) that are used. Optical bandwidth is a scarce resource. State-of-the-art technology allows some hundreds of wavelengths per fiber in the laboratory, even less in manufacturing, and there is no anticipation for dramatic progress in the near future. An efficient allocation of the optical bandwidth (with respect to the number of wavelengths used) is an important engineering problem that can have a significant impact on the success of the technology.

Theoretical work on optical networks mainly focuses on the performance of wavelength routing algorithms on regular networks using oblivious (predefined) routing schemes for connecting each transmitter to the corresponding receiver. We point out the pioneering work of Pankaj [33] who considered shuffle exchange, De Bruijn, and hypercubic networks. Aggarwal *et al.* [1] consider oblivious routing schemes and wavelength assignment algorithms in several networks. Raghavan and Upfal in [35] consider mesh-like networks. Aumann and Rabani [3] improve the bounds of Raghavan and Upfal for mesh networks and also give tight results for hypercubic networks. Rabani in [34] gives almost optimal results for the wavelength routing problem on meshes and mesh-like networks improving earlier results implicit in [26].

These topologies reflect architectures of optical computers rather than wide-area networks. For fundamental practical reasons, the telecommunication

industry does not deploy massive regular architectures: backbone networks need to reflect irregularity of geography, non-uniform clustering of users and traffic, hierarchy of services, dynamic growth, etc. In this direction, Raghavan and Upfal [35], Aumann and Rabani [3], and Bermond *et al.* [7], among other results, focus on bounded-degree networks and give upper and lower bounds as functions of the network expansion.

However, wide-area multiwavelength technology is expected to grow around the evolution of current networking principles and existing fiber networks. These are mainly SONET (Synchronous Optical Networking Technology) rings and trees [31]. In this sense, even asymptotic results for expander graphs do not address the above telecommunication scenario. In this direction, Kumar [27] studies the wavelength routing problem in rings improving previous earlier results implicit in [38,24].

Most of the works mentioned above model the optical network as an undirected graph. Indeed, an optical fiber connecting two nodes can be used to carry traffic in both directions. However, additional devices which are placed on fibers like amplifiers work on only one direction and two "opposite directed" fibers are used in practice to support bidirectional communication between pairs of nodes. So, it is reasonable to model optical networks as bidirected graphs, i.e., graphs whose nodes are connected through pairs of opposite directed edges.

In this work, we consider tree topologies and survey recent methods and algorithms for efficiently coloring paths on undirected and bidirected trees. Notice that, in a tree, the path connecting two nodes is well-defined; this means that the wavelength routing in trees is equivalent to path coloring. We consider arbitrary sets of paths. Surveys on the path coloring problem for sets of paths which capture specific communication patterns like *broadcasting*, *gossiping*, and *permutation routing* can be found in [5,25].

1.2 Problem Definition

We now formulate the path coloring problem in detail by giving the necessary definitions. A path p on a tree T is a sequence $p = (u_1, \cdots, u_l)$ of nodes of the tree such that (u_i, u_{i+1}) is an edge of the tree for $i = 1, \cdots, l-1$. We say that a path touches a node if the node belongs to the sequence of nodes forming the path. Nodes u_1 and u_l are called the origin and the destination of the path, respectively. We also say that path p includes edges (u_i, u_{i+1}) for $i = 1, \cdots, l-1$. The tree can be undirected or bidirected. Respectively, the path can be undirected or directed; in the latter case, a path consists of directed edges of the tree.

Given a set of undirected (resp. directed) paths $P = \{p_1, \cdots, p_k\}$ on a undirected (resp. bidirected) tree, we define the load of an undirected (resp. directed) edge e as the number of paths of P that include e. The *load* of the set of paths P is defined as the maximum edge load over all edges of the tree. The decision version of the path coloring problem can be formalized as follows.

PATH COLORING IN TREES
INSTANCE: An undirected (resp. bidirected) tree T, a set of undirected (resp. directed) paths P on T and an integer B.
QUESTION: Is there a coloring $\chi : P \to \{1, \cdots, B\}$ such that, for any two paths p_1 and p_2 of P, if p_1 and p_2 share an undirected (resp. directed) edge of T, then $\chi(p_1) \neq \chi(p_2)$?

In the optimization version of the path coloring problem, we are given a tree T and a set of paths P on T, and the goal is to compute the minimum integer B such that the answer to the corresponding instance (T, P, B) of the decision problem is YES, i.e., the goal is to compute the minimum number of colors sufficient for coloring P and, usually, to provide the corresponding coloring. Obviously, if $B < L$, the answer to the instance (T, P, B) of the decision problem is certainly NO. Therefore, the load of a set of paths P is a lower bound on the minimum number of colors sufficient for coloring P.

Path coloring is similar to graph coloring. Given an undirected graph G, the graph coloring problem is to assign colors to the nodes of G so that any two adjacent nodes are colored with different colors. Note that path coloring of a set of paths P on a tree T is equivalent to graph coloring of the *conflict graph* of P, i.e., the graph which contains one node for each path in P and an edge between two nodes if the corresponding paths share an edge of the tree. In the text below, we also refer to the edge-coloring problem; here, we seek for assignment of colors to the edges of an undirected graph so that any two edges incident to the same node are assigned different colors.

1.3 Roadmap

The rest of this paper is structured as follows. In Section 2, we present hardness results and lower bounds for both the undirected and the directed version of the problem. We also address the special case of symmetric sets of paths (corresponding to an important communication pattern in today's optical networks) in bidirected trees. In Section 3, we describe deterministic greedy algorithms for path coloring in both undirected and bidirected trees and present the best known results for them. Especially for path coloring in bidirected trees, in Section 4 we discuss randomized greedy algorithms, while we present algorithms that exploit fractional path colorings to compute approximate path colorings in Section 5. In Section 6 we present algorithms and lower bounds for the on-line version of the problem. We conclude, in Section 7, with a list of open problems.

2 Hardness Results and Lower Bounds

As we mentioned in the previous section, the load of a set of paths is a lower bound on the number of necessary colors. We now present better lower bounds for both directed and undirected sets of paths. It is easy to construct a set of undirected paths on an undirected binary tree which requires at least $3L/2$ colors (see Figure 1a). The next result shows that the number of colors can be sensibly higher than the load in the case of directed path coloring as well.

Theorem 1 (Kumar and Schwabe [29]). *For any integer $l > 0$, there exists a set of directed paths of load $L = 4l$ on a binary bidirected tree T that requires at least $5L/4$ colors.*

For proving the theorem, a set of $5L/2$ directed paths is constructed on a binary bidirected tree with six nodes. The set of paths is such that no more than two paths can be colored with the same color. This implies that at least $5L/4$ colors are necessary. The construction is depicted in Figure 1b.

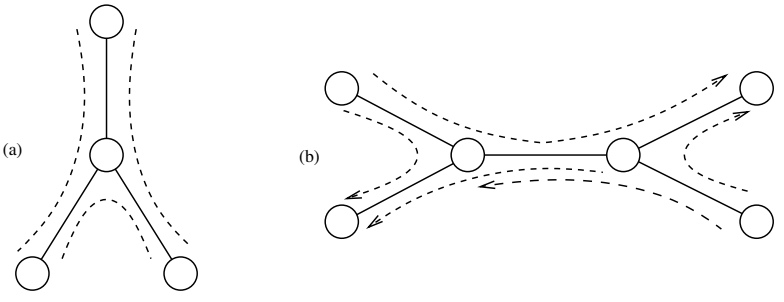

Fig. 1. (a) A set of undirected paths of load L that requires $3L/2$ colors. Each dashed line represents $L/2$ parallel undirected paths. (b) A set of directed paths of load L that requires $5L/4$ colors. Each arrow represents $L/2 = 2l$ parallel directed paths.

The next three theorems show that, in general, computing the optimal number of colors is NP-hard. Erlebach and Jansen [14] and, independently, Kumar *et al.* [28] have proved the following hardness results.

Theorem 2 (Erlebach and Jansen [14], Kumar *et al.* [28]). *Given an undirected star T and a set of paths P of load $L = 3$ on T, it is NP-complete to decide whether P can be colored with 3 colors.*

Theorem 3 (Erlebach and Jansen [14], Kumar *et al.* [28]). *Given a bidirected tree T of depth at least 2 and a set of paths P of load $L = 3$ on T, it is NP-complete to decide whether P can be colored with 3 colors.*

The proofs of the above theorems use reductions from the edge coloring problem in graphs of maximum degree 3 [21]. Actually, as it was first proved in [18], the path coloring problem in undirected stars is equivalent to edge coloring of multigraphs.

Note that the above statement holds in bidirected trees of arbitrary maximum degree. The following result applies to binary bidirected trees and sets of directed paths of arbitrary load. The proof uses a reduction from the circular arc coloring problem [16].

Theorem 4 (Erlebach and Jansen [14], Kumar *et al.* [28]). *Given a bidirected binary tree T and a set of paths P of load L on T, it is NP-complete to decide whether P can be colored with L colors.*

Thus, the corresponding optimization problems (minimizing the number of colors) are NP-hard, in general. On the positive side, a few special cases of the problem can be solved in polynomial time either by using dynamic programming or by exhaustively searching many different solutions.

Theorem 5 (Erlebach and Jansen [14], Kumar *et al.* [28]). *The path coloring problem can be solved optimally in undirected trees of bounded degree.*

Theorem 6 (Erlebach and Jansen [14]). *Given a set of (undirected or directed) paths P on a (undirected or bidirected) tree T with n nodes, such that at most $O\left(\frac{\log n}{\log \log n}\right)$ paths touch any node of T, an optimal coloring of P can be computed in polynomial time.*

As a corollary, we obtain that optimally coloring sets of paths of load $O\left(\frac{\log n}{\log \log n}\right)$ in bidirected trees of bounded degree can be done in polynomial time.

2.1 Coloring Symmetric Sets of Paths

We now consider the special case of sets of symmetric paths in bidirected trees. Two directed paths are called symmetric if the origin of the one is the destination of the other and vice versa. A set of directed paths is called symmetric if it can be partitioned into pairs of symmetric paths.

We can restrict the coloring of symmetric sets of paths to use the same color to color the two paths in each pair of symmetric paths. Then, we can solve the path coloring problem using any algorithm for the undirected version of the problem. In this way, up to $3L/2$ colors may be necessary (by the discussion at the beginning of Section 2) and sufficient (by using an algorithm presented in Section 3.1, see Theorem 9). The following two questions now arise.

- Can we improve this bound if we do not constrain symmetric paths to use the same color?
- Is the path coloring problem easier than in the general case if we restrict the input instances to sets of symmetric paths?

Although we give a positive answer for the first question in the case of binary bidirected trees in Section 4, we are not aware of complete answers for these questions. The following discussion shows some inherent similarities and differences between the general path coloring problem and the case where the input instance is restricted to sets of symmetric paths.

For these sets of paths, Caragiannis *et al.* [10] have proved some interesting statements (lower bounds). Both NP-completeness results (Theorems 3 and 4) hold in the case of sets of symmetric paths [10]. The proofs again use slightly modified reductions from the edge coloring and circular arc coloring problem. Notice that the lower bound of Figure 1 applies to non-symmetric sets of paths. A similar lower bound (but much more complicated than the one of Figure 1b) holds for sets of symmetric paths as well.

Theorem 7 (Caragiannis *et al.* [10]). *For any $\delta > 0$, there exists an integer $l > 0$, a binary bidirected tree T and a set of symmetric paths P with load $L = 4l$ on T, such that no algorithm can color P using less than $(5/4 - \delta)L$ colors.*

Does Theorem 7 indicate that the symmetric version of the problem is as "hard" as the general one? The following result gives evidence for a negative answer. Notice that for sets of symmetric paths, there exists an algorithm which may color paths in such a way that each pair of opposite directed edges sees at most L colors. This can be done in a trivial way if we consider two symmetric paths as an undirected one and use any algorithm for the undirected problem. However, this is not the case for the non-symmetric problem.

Theorem 8 (Caragiannis *et al.* [10]). *For any integer $l > 0$, there exists a bidirected tree T and a set of directed paths with load $L = 8l$ that cannot be colored in such a way that any pair of opposite directed edges of T sees less than $9L/8$ colors.*

3 Greedy Algorithms

Most of the known path coloring algorithms [12,14,17,22,29,35] belong to a special class of algorithms, the class of *greedy* algorithms. We devote this section to their study. Given a tree T and a set of paths P, we call greedy a path coloring algorithm that works as follows:

> Starting from an arbitrary node (the root of the tree), the algorithm computes a breadth-first (BFS) numbering of the nodes of the tree.
> The algorithm proceeds in phases, one per each node u of the tree. The nodes are considered following their BFS numbering. The phase associated with node u takes as input a proper coloring of all the paths that touch nodes with BFS number strictly smaller than u's; all other paths have not been colored yet.
> During the phase associated with node u, the partial proper coloring is extended to a proper one that assigns colors to paths that touch node u but have not been colored yet.
> During each phase, the algorithm does not recolor paths that have been colored in previous phases.

The various greedy algorithms differ among themselves with respect to the strategies followed to extend the partial proper coloring during a phase. In the following two sections, we describe the techniques used by greedy algorithms in undirected and bidirected trees. Then, we describe the main technique used for proving lower bounds for greedy path coloring algorithms in bidirected trees.

3.1 Greedy Path Coloring Algorithms in Undirected Trees

The algorithms in [14,35] reduce the coloring of a phase to an edge coloring problem on a multigraph. In the following we describe this reduction.

Consider a set of paths P of load L on an undirected tree T and the phase of a greedy algorithm associated with a node u of T. The phase receives as input the coloring of the paths of P in previous phases (i.e., in phases associated with nodes of T with numbers smaller than u's) and extends this coloring by coloring the paths touching node u which are still uncolored by solving an edge-coloring problem on a multigraph G_u. The multigraph G_u associated with node u is constructed as follows. We denote by v_0 the parent of u and let $v_1, ..., v_k$ be the children of u. For each node v_i of the nodes $v_0, v_1, ..., v_k$, the graph G_u has two nodes x_i and y_i. For each path originated from u and going to v_i, we add an edge between x_i and y_i in G_u. For each path between two neighbors v_i and v_j of u, we add an edge between nodes x_i and x_j in G_u. It can be easily seen that the graph G_u has maximum degree L. An example of the construction is depicted in Figure 2.

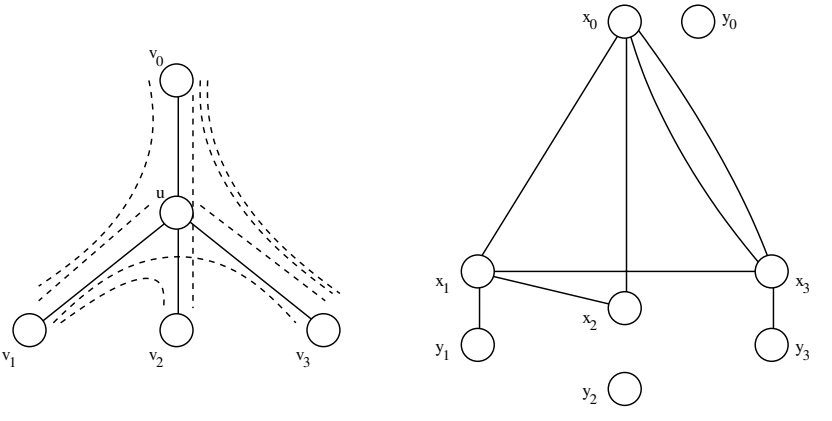

Fig. 2. Undirected paths touching node u and the corresponding multigraph

Intuitively, edges of G_u incident to a node x_i correspond to paths of P traversing the edge of T between nodes u and v_i. The edges of G_u incident to node x_0 correspond to paths of P traversing the edge of T between node u and its parent node v_0; these paths have been colored in phases associated with nodes having BFS numbers smaller than u's. We call the edges of G_u incident to node x_0 *color-forced* edges. The phase associated with node u colors the edges of G_u under the constraint that color-forced edges are assigned the colors assigned to the corresponding paths in previous phases. It can be easily seen that these constraints do not make the edge-coloring problem harder with respect to the number of colors required since, in any proper coloring of the edges of G_u (without constraints on the coloring of the edges incident to x_0), different colors must be assigned to the color-forced edges. These constraints just reduce the number of different proper colorings. The produced coloring of the edges of G_u trivially yields a coloring of the paths of P touching node u and nodes with BFS numbers smaller than u's. Also, note that the optimal number of colors required for coloring the edges of G_u is a lower bound on the number of colors required for coloring P.

For extending the coloring to all edges of G_u, colors used for coloring the edges of multigraphs associated with nodes with numbers smaller than u's which are not used in the edges incident to node x_0 of G_u can be used. New colors are used only when the colors already used do not suffice to complete the edge-coloring. So, the total number of colors used for coloring the paths of P is the maximum over all phases of the number of colors used for coloring the edges of the corresponding multigraph. Raghavan and Upfal [35] used an algorithm of Shannon [37] which colors the edges of a multigraph with maximum degree L using at most $3L/2$ colors to obtain the following result.

Theorem 9 (Raghavan and Upfal [35]). *There exists a polynomial-time algorithm which colors any set of paths of load L on an undirected tree using at most $3L/2$ colors.*

From the discussion on lower bounds in Section 2, we know that this bound is tight. Erlebach and Jansen [14] observed that using an algorithm which colors the edges of a multigraph G with at most $f(\chi(G))$ colors, where f is a non-decreasing function and $\chi(G)$ denotes the optimal number of colors for coloring the edges of G, we obtain a greedy path coloring algorithm for undirected trees which colors any set of paths P using $f(\chi(P))$ colors, where $\chi(P)$ denotes the optimal number of colors sufficient for coloring P. This can be explained as follows. Let u be the node for which the edge-coloring algorithm uses the maximum number of colors for coloring the edges of the corresponding multigraph G_u. Clearly, $\chi(P) \geq \chi(G_u)$, while the greedy algorithm uses at most $f(\chi(G_u)) \geq f(\chi(P))$ colors to color the paths of P. In [32], Nishizeki and Kashiwagi present an algorithm for coloring the edges of a multigraph G with at most $\lfloor 1.1\chi(G) + 0.8 \rfloor$ colors. Combined with the discussion above, this implies the following.

Theorem 10 (Erlebach and Jansen [14]). *There exists a polynomial-time algorithm which colors any set of paths P on an undirected tree using at most $\lfloor 1.1\chi(P) + 0.8 \rfloor$ colors, where $\chi(P)$ denotes the optimal number of colors sufficient for coloring P.*

Actually, Theorem 5 and Theorem 6 (for undirected trees) can be proved using greedy path coloring algorithms. In both cases, an edge-coloring algorithm is used in each phase to optimally color the edges of the corresponding multigraph in polynomial time. In the case of Theorem 5, the multigraph associated with each phase has a constant number of nodes; in this case, the edge-coloring problem can be solved in time polynomial in L. In the case of Theorem 6, the multigraph associated with each phase has at most $O\left(\frac{\log n}{\log \log n}\right)$ edges; it can be easily seen that an optimal edge coloring can be computed in time polynomial in n.

3.2 Greedy Path Coloring Algorithms in Bidirected Trees

The algorithms in [15,29] for path coloring in bidirected trees reduce the coloring of a phase to an edge-coloring problem on a bipartite multigraph. In the following we describe this reduction.

Consider a set of directed paths P of load L on a bidirected tree T and the phase of a greedy algorithm associated with a node u of T. The phase receives as input the coloring of the paths of P in previous phases (i.e., in phases associated with nodes of T with numbers smaller than u's) and extends this coloring by coloring the paths touching node u which are still uncolored by solving an edge-coloring problem on a bipartite multigraph H_u. The multigraph H_u associated with node u is constructed as follows. Again, we denote by v_0 the parent of u and let $v_1, \cdots, v_k$ be the children of u. For each node v_i, the bipartite multigraph has four nodes w_i, x_i, y_i, z_i and the left and right partitions are $\{w_i, x_i | i = 0, \cdots k\}$ and $\{z_i, y_i | i = 0, \cdots k\}$, respectively. For each path of the tree directed out of some node v_i into some node v_j, the bipartite multigraph H_u has an edge between w_i and z_j. For each path directed out of some node v_i and terminating at u, we have an edge between w_i and y_i. Finally, for each path directed out of u into some node v_i, we have an edge between z_i and x_i. It can be easily seen that the graph H_u has maximum degree L. An example of this construction is depicted in Figure 3.

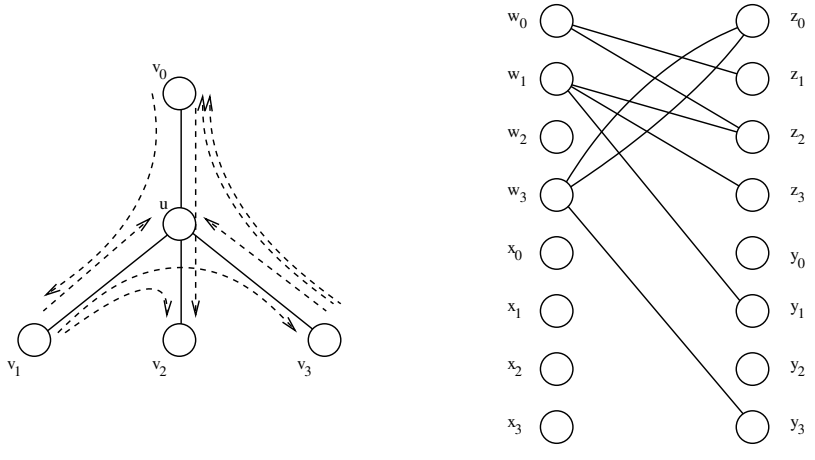

Fig. 3. Directed paths touching node u and the corresponding bipartite multigraph

Intuitively, edges of H_u incident to a node w_i correspond to paths of P traversing the directed edge of T from node v_i to node u, while edges of H_u incident to a node z_i correspond to paths of P traversing the directed edge of T from node u to node v_i. Observe that no edges extend across opposite nodes w_i and z_i as they would correspond to paths starting and ending on the same node. The edges of H_u incident to nodes w_0 and z_0 correspond to paths of P traversing the opposite directed edges between u and its parent node v_0; these paths have been colored in phases associated with nodes having BFS numbers smaller than u's. We call the edges of H_u incident to nodes w_0 and z_0 color-forced edges. The phase associated with node u colors the edges of H_u under the constraint that color-forced edges are assigned the colors assigned to the corresponding directed

paths in previous phases. The produced coloring of the edges of H_u trivially yields a coloring of the paths of P touching node u and nodes with BFS numbers smaller than u's. Similarly to the case of path coloring in undirected trees, the total number of colors used for coloring the paths of P can be made as small as the maximum over all phases of the number of colors used for coloring the edges of the corresponding bipartite multigraph.

Thus, the problem of coloring paths at each phase is reduced to the problem of coloring the edges of a bipartite multigraph of maximum degree L, under the constraint that some colors have already been assigned to the color-forced edges. We call this problem an *α-constrained bipartite edge coloring problem* on a bipartite graph with maximum degree L. The parameter α denotes that color-forced edges have been colored with at most αL colors ($1 \leq \alpha \leq 2$). Note that unlike the case of edge-coloring multigraphs produced by undirected sets of paths, the existence of constraints in the color-forced edges of the bipartite multigraph H_u associated with node u indeed make the problem harder with respect to the total number of colors required for coloring all the edges. Bipartite multigraphs with maximum degree L can always be colored with L colors (if no constraints on the color-forced edges exist). However, the bipartite multigraphs at the phases of a greedy algorithm for coloring directed paths on bidirected trees may require at least $5L/4$ colors due to the lower bound presented in Section 2 (Theorem 1).

In the following, we first describe a simple solution to the 2-constrained bipartite edge coloring. In particular we show how to color the edges of a bipartite multigraph H_u of maximum degree L in which the color-forced edges are constrained to be colored with at most $2L$ specific colors using at most $2L$ colors in total. Then, we briefly describe the main ideas that lead to an improved solution of the problem in [15].

We call *single* colors the colors that appear only in one color-forced edge and *double* colors the colors that appear in two color-forced edges (one incident to w_0 and one incident to z_0). We denote by S and D the number of single and double colors, respectively. Clearly, $2D + S \leq 2L$.

We decompose the bipartite graph into L matchings (this can be done in polynomial time since the graph is bipartite and has maximum degree L; see [6]). At least $S/2$ of these matchings have at least one single color in one of the two colored-forced edges. Thus, we can use this single color to color the uncolored edges of the matching. The uncolored edges of the matchings that have no color-forced edges colored with a single color are colored with extra colors (one extra color per matching). In total, we use at most

$$D + S + L - S/2 = L + D + S/2 \leq 2L$$

colors.

In each phase of the greedy algorithm when applied to a set of directed paths of load L on a bidirected tree, an instance of 2-constrained bipartite edge coloring on a bipartite multigraph of maximum degree L is solved. The number of colors used in each phase never grows larger than $2L$. In this way, we obtain a simple

greedy path coloring algorithm that uses at most $2L$ colors for any set of directed paths of load L on a bidirected tree.

Better solutions are obtained in [15,29]. Again consider the application of the greedy algorithm on a set of directed paths of load L on a bidirected tree and let u be a node with parent v_0 and children $v_1, ..., v_k$. For $i = 0, ..., k$, we call the pair of nodes w_i, z_i of the bipartite multigraph H_u a *line*. We say that the color c is used in the line w_i, z_i if the color c is used in some edge incident either to node w_i or to node z_i. Erlebach *et al.* [15] solve the problem proving the following theorem (a slightly inferior result was obtained in [29]).

Theorem 11 (Erlebach *et al.* [15]). *There exists a polynomial time algorithm which, for any $\alpha \in [1, 4/3]$, colors an instance of the α-constrained bipartite edge coloring problem on the bipartite multigraph H_u of maximum degree L using at most $\left(1 + \frac{\alpha}{2}\right) L$ total colors so that the number of colors used per each line of H_u is at most $4L/3$.*

Note that the edges of H_u incident to a line w_i, z_i correspond to directed paths traversing the opposite directed edges of the tree between u and a child v_i of u. This means that, if we use the edge-coloring algorithm of [15] in the phase associated with node u for solving an instance of the α-constrained bipartite edge-coloring on the bipartite multigraph H_u of maximum degree L with $\alpha \leq 4/3$, then, the edge-coloring problems that have to be solved in the phases associated with the child nodes of u are instances of the α-constrained bipartite edge coloring on bipartite multigraph of maximum degree L for $\alpha \leq 4/3$ as well. In the first phase of the greedy algorithm (i.e., the one associated with the root of the tree), there are no constraints on the edges of the corresponding bipartite multigraph, so its edges can be properly colored with at most L colors. Then, we can easily verify inductively that, at each of the next phases, the number of colors used in the paths colored in previous phases is never more than $4L/3$. So, in all phases of the greedy algorithm, an instance of the α-constrained bipartite edge coloring problem with $\alpha \leq 4/3$ has to be solved in a bipartite multigraph of maximum degree L and the total number of colors used is never more than $5L/3$. This is summarized in the following theorem.

Theorem 12 (Erlebach *et al.* [15]). *There exists a polynomial time greedy algorithm which colors any set of directed paths of load L on a bidirected tree using at most $5L/3$ colors.*

The interested reader may refer to the papers [15,29] for detailed description of the edge-coloring techniques. They either consider matchings in pairs and color them in sophisticated ways using detailed potential and averaging arguments for the analysis [29] or partition matchings into groups which can be colored and accounted for independently [15]. Note that one might think of bipartite edge coloring problems with different constraints. Tight bounds on the number of colors for more generalized constrained bipartite edge coloring problems can be found in [11].

The algorithms in [12,22] use much simpler methods to obtain the same upper bound in bidirected binary trees. The common characteristic of all the algorithms discussed so far is that they are deterministic.

3.3 A Lower Bound Technique

In this section, we present a lower bound technique for greedy path coloring algorithms in bidirected trees. We first briefly describe the technique; then we give the best known statement for deterministic greedy algorithms.

The technique is based on an adversary argument. An adversary algorithm $\mathcal{ADV}$ is exhibited that constructs a set of directed paths on a binary bidirected tree for which it can be proved that there exists a lower bound on the number of colors used by any greedy algorithm.

The adversary $\mathcal{ADV}$ constructs the set of directed paths P in an incremental way visiting the nodes of the tree according to a BFS order. At each node u, the $\mathcal{ADV}$ deals with the set of paths traversing one of the two parallel directed edges between u and its parent $p(u)$. For each *downward* path p (that is, for each path including the directed edge $(p(u), u)$), $\mathcal{ADV}$ has two options:

1. do nothing; i.e., make p stop at u;
2. propagate p to the left child $l(u)$ by appending edge $(u, l(u))$ to p.

Similarly, for each *upward* path p (that is, for each path including the directed edge $(u, p(u))$), $\mathcal{ADV}$ has two options:

1. do nothing; i.e. make p start from u;
2. make p originate from the right child $r(u)$ by pre-pending edge $(r(u), u)$ to p;

Moreover, the adversary algorithm $\mathcal{ADV}$ can introduce directed paths between the two children of u (see Figure 4). Initially, these paths will consist of only two edges (from a child to u and from u to the other child) and can be augmented when the adversary reaches the children of u.

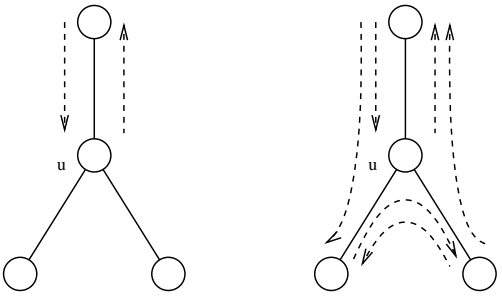

Fig. 4. The construction for the lower bound for greedy path coloring algorithms in bidirected trees

At each step, since the adversary $\mathcal{ADV}$ may know how a greedy algorithm performs the coloring, it can choose to augment paths in such a way that the number of common colors used in downward paths from $p(u)$ to $l(u)$ and upward paths from $r(u)$ to $p(u)$ is small. In this way, no matter what a greedy algorithm can do, both the total number of colors and the number of colors seen by the opposite directed edges between u and its children will increase.

By constructing such an adversary for deterministic greedy algorithms, Erlebach *et al.* [15] prove the following lower bound. The same lower bound technique can be used for greedy algorithms that use randomization (see Section 4).

Theorem 13 (Erlebach *et al.* [15]). *Let $\mathcal{A}$ be a deterministic greedy path coloring algorithm in bidirected trees. There exists an algorithm $\mathcal{ADV}$ which, on input $\delta > 0$ and integer $L > 0$, outputs a binary bidirected tree T and a set of directed paths P of load L on T, such that $\mathcal{A}$ colors P with at least $(5/3 - \delta)L$ colors.*

Thus, the greedy algorithm presented in [15] (see also Theorem 12) is the best possible within the class of deterministic greedy algorithms for path coloring in bidirected trees. In Section 4 we demonstrate how randomization can be used to beat the 5/3 barrier at least on binary bidirected trees.

4 Randomized Algorithms in Bidirected Trees

As we discussed in Section 3.1, the algorithm of Raghavan and Upfal [35] achieves a tight upper bound on the number of colors sufficient for path coloring in undirected trees while the use of efficient approximation algorithms for edge-coloring of multigraphs leads to very good approximations for the problem. This is not the case for the directed version of the problem. In this section, we discuss how randomization can be used in order to improve the upper bounds for path coloring in binary bidirected trees.

In an attempt to beat the 5/3 lower bound for deterministic greedy algorithms for path coloring in bidirected trees, Auletta *et al.* [2] define the class of randomized greedy path coloring algorithms. Randomized greedy algorithms have the same structure as deterministic ones; that is, starting from a node, they consider the nodes of the tree in a BFS manner. Their main difference is that a randomized greedy algorithm $\mathcal{A}$ uses a palette of colors and at each phase associated with a node, $\mathcal{A}$ picks a random proper coloring of the uncolored paths using colors of the palette according to some probability distribution.

The results presented in the following were originally obtained in [2]. The interested reader may see [2] for further details.

4.1 Lower Bounds

In this section, we present two lower bounds on the number of colors used by randomized greedy algorithms to color sets of paths of load L on bidirected trees. We first present a lower bound for large trees (i.e., trees of depth $\Omega(L)$) in

Theorem 14; then, in Theorem 15, we present a lower bound for trees of constant depth.

The first lower bound states that no randomized greedy algorithm can achieve a performance ratio better than 3/2 if the depth of the tree is large.

Theorem 14 (Auletta *et al.* [2]). *Let $\mathcal{A}$ be a (possibly randomized) greedy path coloring algorithm on binary bidirected trees. There exists a randomized algorithm $\mathcal{ADV}$ which, on input $\epsilon > 0$ and integer $L > 0$, outputs a binary bidirected tree T of depth $L + \epsilon \ln L + 2$ and a set of directed paths P of load L on T, such that the probability that $\mathcal{A}$ colors P with at least $3L/2$ colors is at least $1 - \exp(-L^{\epsilon})$.*

The following lower bound holds even for trees of constant depth.

Theorem 15 (Auletta *et al.* [2]). *Let $\mathcal{A}$ be a (possibly randomized) greedy path coloring algorithm on binary bidirected trees. There exists a randomized algorithm $\mathcal{ADV}$ which, on input $\delta > 0$ and integer $L > 0$, outputs a binary bidirected tree T of constant depth and a set of directed paths P with load L on T, such that the probability that $\mathcal{A}$ colors P with at least $(1.293 - \delta - o(1))L$ colors is at least $1 - O(L^{-2})$.*

For the proofs, constructions based on the lower bound technique presented in Section 3.3 are used.

Note that the adversary assumed in Theorems 14 and 15 has no knowledge of the probability distribution according to which the randomized greedy algorithm makes its random choices. Possibly, better lower bounds may be achieved by considering more powerful adversaries.

4.2 Upper Bounds

In this section, we give the main ideas of a randomized path coloring algorithm presented in [2]. The algorithm has a greedy structure but allows for limited recoloring at the phases associated with each node.

At each phase, the path coloring algorithm maintains the following two invariants:

I. The total number of colors is no greater than $7L/5$.
II. The number of colors seen by two opposite directed edges is exactly $6L/5$.

At a phase associated with a node u, a *coloring procedure* is executed which extends the coloring of paths that touch u and its parent node to the paths that touch u and are still uncolored. The coloring procedure is randomized. It selects the coloring of paths being uncolored according to a specific probability distribution on random colorings in which the probability that two paths crossing an edge between u and any child of u in opposite directions are assigned the same color is $\frac{4}{5L}$. In this way, the algorithm can complete the coloring at the phase associated with node u using at most $7L/5$ colors in total, keeping the number of colors seen by the opposite directed edges between u and its children to $6L/5$.

At each phase associated with a node u, the algorithm is enhanced by a *recoloring procedure* which recolors a small subset of paths in order to maintain some specific properties on the (probability distribution of the) coloring of paths touching u and its parent. This procedure is randomized as well.

The recoloring procedure at each phase of the algorithm works with very high probability. The coloring procedure at each phase always works correctly maintaining the two invariants. As a result, if the depth of the tree is not very large (i.e., $o(L^{1/3})$), the algorithm executes the phases associated with all nodes, with high probability.

After the execution of all phases, the set of paths being recolored by the executions of the recoloring procedure are colored using the simple deterministic greedy algorithm with at most $o(L)$ extra colors due to the fact that as far as the depth of the tree is not very large (i.e., $o(L^{1/3})$), the load of the set of paths being recolored is at most $o(L)$, with high probability.

In this way, the following result is proved. The interested reader may look at [2] for a detailed description of the algorithm and formal proofs.

Theorem 16 (Auletta *et al.* [2]). *Let $0 < \delta < 1/3$ be a constant. There exists a randomized path coloring algorithm that colors any set of directed paths of load L on a binary bidirected tree of depth at most $L^{\delta}/8$ using at most $7L/5 + o(L)$ colors, with probability at least $1 - \exp\left(-\Omega(L^{\delta})\right)$.*

Going back to the discussion of Section 2.1 on symmetric sets of paths, Theorem 16 immediately implies that, at least for symmetric sets of paths on binary bidirected trees (with some additional restrictions on their depth), allowing symmetric paths to use different colors leads to better colorings (i.e., colorings with less than $3L/2$ colors). A better algorithm for symmetric sets of paths on binary trees has been presented in [13]. It colors any symmetric set of paths of load L on a binary bidirected tree with the same restrictions with those in Theorem 16 using at most $1.367L + o(L)$ colors, with high probability. This algorithm follows the main ideas behind the randomized algorithm discussed above.

5 Fractional Path Coloring and Applications

So far, we have expressed the performance of the path coloring algorithms in terms of the load of the set of paths to be colored. The approximation guarantee is easily achieved since the load is a lower bound on the number of colors necessary for coloring a set of paths.

In this section, among other results, we present approximation algorithms for which we can express their performance in terms of both the load and the *fractional path chromatic number* of the set of paths to be colored. In this way, we obtain a better approximation guarantee at least for bidirected trees of bounded degree.

We first give some definitions. The graph coloring problem can be considered as finding a minimum cost integral covering of the nodes of a graph by independent sets of unit cost. Given a graph $G = (V, E)$, this means solving the following integer linear program:

$$\begin{array}{rll} \text{minimize} & \sum_{I \in \mathcal{I}} x(I) & \\ \text{subject to} & \sum_{I \in \mathcal{I}: v \in I} x(I) \geq 1 & \forall\ v \in V \\ & x(I) \in \{0,1\} & \forall\ I \in \mathcal{I} \end{array}$$

where $\mathcal{I}$ denotes the set of the independent sets of G.

This formulation has a natural relaxation into the following linear program:

$$\begin{array}{rll} \text{minimize} & \sum_{I \in \mathcal{I}} \bar{x}(I) & \\ \text{subject to} & \sum_{I \in \mathcal{I}: v \in I} \bar{x}(I) \geq 1 & \forall\ v \in V \\ & 0 \leq \bar{x}(I) \leq 1 & \forall\ I \in \mathcal{I} \end{array}$$

The corresponding combinatorial problem is called the *fractional coloring problem* (see [20]), and the value of an optimal solution is called the *fractional chromatic number*. Clearly, the fractional chromatic number of a graph is a lower bound on its chromatic number. If $\bar{x}$ is a function over the independent sets of the graph G satisfying the constraints of the linear program, we call it a *fractional coloring* of G.

We now extend some terms of graph coloring to path coloring. Given a set of directed paths P on a bidirected tree T, we define an *independent set of paths* as a set of pairwise edge-disjoint paths, i.e., a set of paths whose corresponding nodes form an independent set of the conflict graph. If G is the conflict graph of P on T, we will denote by $w_f(P)$ the fractional chromatic number of G and call it the *fractional path chromatic number* of P on T. Clearly, $w_f(P)$ is a lower bound on the optimal number of colors sufficient for coloring P. Also, a fractional coloring of the conflict graph G is called a *fractional path coloring* of P.

In general, the fractional chromatic number is as hard to approximate as the chromatic number. Indeed, the size of the above described linear program is exponential (proportional to the number of independent sets of G). However, in [8], it is shown how to compute an optimal fractional path coloring. In particular, the following statement is proved.

Theorem 17 (Caragiannis *et al.* [8]). *Fractional path coloring in bounded-degree trees can be computed in polynomial time.*

The main idea for the proof of Theorem 17 is to inductively construct a linear program whose solution is actually the optimal solution to the fractional path coloring problem. The linear program produced is proved to have polynomial size (i.e., polynomial number of variables) as far as the degree of the tree is bounded. In this way, fractional path coloring in trees is reduced to solving a polynomial size linear program.

5.1 Fractional Path Coloring on Binary Bidirected Trees

In this section we give an upper bound on the fractional path chromatic number of a set of directed paths on a binary bidirected tree. We express the result in terms of the load. This result was first discussed in [8].

In Section 4 we discussed the randomized algorithm presented in [2], which colors a set of directed paths of load L on a binary bidirected tree using at

most $7L/5 + o(L)$ colors. The algorithm works with high probability under some additional constraints (i.e., the depth of the tree is not very large). The hidden constants in the low order term are large and are due to the integrality constraints of the problem and the random choices of the algorithm.

In the case of fractional coloring we can design a deterministic algorithm having the same top-down structure as the path coloring algorithm in [2]. Given a set of directed paths P of load L on a bidirected tree, this fractional path coloring algorithm uses fractions of colors to color paths and at each step, it maintains the following invariant: the common fractions of colors assigned to any two paths traversing an edge of the tree in opposite directions is $\frac{4}{5L}$. Following simpler analysis than in [2], we can obtain that, given a set of directed paths P on a binary bidirected tree T, the algorithm computes a fractional path coloring of P with cost at most $7L/5$. Comparing this result to the randomized algorithm presented in Section 4 and particularly to Theorem 16, we observe that the low order term is now eliminated. This is due to the fact that, when we compute a fractional coloring, we have no integrality constraints and the algorithm is deterministic.

Theorem 18 (Caragiannis *et al.* [8]). *For any set of directed paths of load L on a binary bidirected tree, there exists a fractional path coloring of cost at most $7L/5$. Moreover, such a fractional coloring can be computed in polynomial time.*

A slightly better upper bound of $1.367L$ has been proved in [13] for the cost of the fractional coloring of symmetric sets of paths of load L on binary bidirected trees.

5.2 Applications to Path Coloring on Bounded-Degree Bidirected Trees

In this section we briefly outline how to exploit the (optimal) solution for fractional path coloring which can be obtained in polynomial time for bounded-degree bidirected trees to design a randomized algorithm with approximation ratio better than $5/3$.

Given a solution $\bar{x}$ of the fractional path coloring of the set of directed paths P on a bidirected tree T, the idea is to perform a randomized rounding to $\bar{x}$ and obtain an integral solution x. After rounding, x is not a feasible solution to the path coloring problem since some of the constraints of the form $\sum_{I \in \mathcal{I}: p \in I} x(I) \geq 1$ may be violated. However, this is a feasible solution to the path coloring problem on the set of paths $P' \subseteq P$, defined as the set of paths contained in independent sets I such that $x(I) = 1$. This means that we have properly colored some of the paths of P with $w_f(P)$ colors.

Following the analysis in [8] (similar arguments are used in [27] for the analysis of a path coloring algorithm in rings), we can show that if $L = \Omega(\log n)$, where n is the number of nodes in T, then after the rounding procedure the load of paths in $P \backslash P'$, i.e., the load of the paths not colored, is at most $\frac{L}{e} + o(L)$, with high probability. Now, we can apply the algorithm of [15] to color the paths in $P \backslash P'$ with $\frac{5L}{3e} + o(L)$ additional colors. In total, we use at most

$$w_f(P) + \frac{5L}{3e} + o(L)$$

colors. Since both the load L and the fractional path chromatic number of P are lower bounds on the optimal number of colors sufficient for coloring P, we obtain the following result.

Theorem 19 (Caragiannis *et al.* [8]). *There exists a randomized path coloring algorithm, which, given a set of directed paths of load $L = \Omega(\log n)$ on a bidirected tree of bounded degree with n nodes, computes an $1.613 + o(1)$-approximate path coloring for P, with high probability.*

In a recent work [9], a generalization of the randomized rounding technique used in [8,27] is proposed. It uses a parameter $q \in (0, 1]$, and, given a set of directed paths P of load $L = \Omega(\log n)$ on a bidirected tree with n nodes and a fractional coloring of P of cost $w_f(P)$, it computes a coloring of some of the paths in P with $q \cdot w_f(P) + o(L)$ colors so that the load of the subset of P consisting of the paths being uncolored is at most $e^{-q} \cdot w_f(P) + o(L)$, with high probability. For sets of paths on bidirected trees of bounded degree, applying this randomized rounding technique with parameter $q = \ln \frac{5}{3}$ and using the algorithm of [15] to color the uncolored paths we obtain a $1.511 + o(1)$-approximation algorithm. The restriction on the load is not very important since sets of paths of load $O\left(\frac{\log n}{\log \log n}\right)$ can be optimally colored in bidirected trees of bounded degree. Furthermore, the technique can be applied with $q = \ln \frac{7}{5}$ to sets of paths on binary bidirected trees with similar restrictions to those in Theorem 16 and, combined with the algorithm of [2], it yields a $1.336 + o(1)$-approximation path coloring algorithm on binary bidirected trees.

6 On-Line Algorithms

In the *on-line* version of the path coloring problem in trees, the input instance is a sequence of paths $P = \{p_1, ..., p_{|P|}\}$ arriving over time. An on-line path coloring algorithm must color the paths following the order of the sequence. For each path, the algorithm has to assign it a color so that no two paths including the same edge of the tree are assigned the same color. Once a path has been colored, it cannot be recolored.

The performance of an on-line algorithm is measured by comparing it to the performance (in our case, the number of colors) of the *off-line* algorithm, i.e., the algorithm that knows the sequence of paths in advance. We say that an on-line algorithm $\mathcal{A}_{on}$ for the path coloring problem is c-competitive (or has competitive ratio c) if, for every instance I, $\mathcal{A}_{on}$ uses at most $c \cdot \chi_{off}$ colors, where χ_{off} is the number of colors used by the optimal off-line algorithm $\mathcal{A}_{off}$. This definition applies to deterministic on-line algorithms. We say that a randomized on-line algorithm $\mathcal{A}_{on}$ is c-competitive if, for every instance I, the expectation of the number of colors used by $\mathcal{A}_{on}$ is at most $c \cdot \chi_{off}$. Usually, we analyze the performance of randomized on-line algorithms against oblivious adversaries,

i.e., we assume that input instances do not depend on the random choices of the algorithm (although they may depend on the probability distribution according to which the random choices of the algorithm are made).

The results presented in the following apply to both undirected and bidirected trees. A natural on-line algorithm is the First-Fit algorithm. Colors are denoted by positive integers. When a new path p arrives, it is assigned the minimum integer not assigned to paths sharing an edge with p.

The performance of the First-Fit algorithm has been studied in the context of on-line coloring of d-inductive graphs [23]. A graph is called d-inductive if its nodes can be numbered in such a way that each node has at most d adjacent nodes with higher numbers. Irani in [23] shows that First-Fit uses at most $O(d \cdot \log n)$ colors for coloring on-line a d-inductive graph with n nodes.

Bartal and Leonardi [4] observed that the conflict graph defined by the paths in an instance of the path coloring problem on a tree is $2(C-1)$-inductive, where C is the clique number of the conflict graph. Using the result of Irani [23], the set of paths can be colored on-line using at most $O(C \cdot \log |P|)$ colors. Note that any algorithm requires at least C colors to color an instance whose conflict graph has a clique of size C even if it knows the paths in advance. Hence, the competitive ratio of the First-Fit algorithm is $O(\log |P|)$. This is logarithmic in the number of nodes of the tree only if $|P|$ is polynomial in n.

Bartal and Leonardi [4] present the following algorithm which is a variation of the First-Fit algorithm. The nodes of the tree are classified into $O(\log n)$ levels so that for any two nodes of the same level i, the path connecting them contains at least one node of level $i-1$. When a path appears, it is first assigned the lowest of the levels of the nodes it touches. The algorithm assigns the path the minimum color not assigned to paths of the same level sharing an edge with p or to any other path of different level. Bartal and Leonardi show that the number of different colors used for on-line path coloring of a set of paths whose conflict graph has clique size C is at most $O(\log n) \cdot C$.

Theorem 20 (Bartal and Leonardi [4]). *There exists an $O(\log n)$-competitive deterministic on-line path coloring algorithm in (undirected or bidirected) trees with n nodes.*

The algorithm presented in [4] is almost optimal within the class of deterministic on-line path coloring algorithms. Indeed, Bartal and Leonardi [4] show the following statement.

Theorem 21 (Bartal and Leonardi [4]). *For any deterministic on-line path coloring algorithm $\mathcal{A}$ in (undirected or bidirected) trees, there exists a (resp. undirected or bidirected) tree T with n nodes and an input instance I on T which can be colored with a constant number of colors, such that $\mathcal{A}$ colors I with at least $\Omega(\log n / \log \log n)$ colors.*

An interesting issue is whether the use of randomization is helpful. Although no randomized algorithm with sublogarithmic competitive ratio is known, Leonardi and Vitaletti [30] have proved the following lower bound for randomized on-line path coloring algorithms.

Theorem 22 (Leonardi and Vitaletti [30]). *For any on-line path coloring algorithm $\mathcal{A}$ in (undirected or bidirected) trees, there exists an oblivious adversary that constructs a (resp. undirected or bidirected) tree T with n nodes and input instances $\mathcal{I}$ on T which can be colored with a constant number of colors, such that the expectation of the number of colors used by $\mathcal{A}$ to color any instance $I \in \mathcal{I}$ is at least $\Omega(\log \log n)$.*

7 Open Problems

Recent work on path coloring in trees has revealed many open problems; some of them are listed below.

- Improving known bounds for the path coloring problem in undirected trees is equivalent to improving known bounds for the edge-coloring problem on multigraphs. Although this appears to be very difficult, it is an intriguing question whether approximation schemes for the problem exist.
- For the path coloring problem in bidirected trees, an open problem is to close the gap between $5L/4$ and $5L/3$ for the number of colors sufficient for coloring sets of directed paths of load L on arbitrary bidirected trees (see Theorems 1 and 12). Closing the gap between $5L/4$ and $7L/5 + o(L)$ for binary trees also deserves some attention.
- Furthermore, although for deterministic greedy path coloring algorithms we know tight bounds on the number of colors, this is not true for randomized greedy algorithms in bidirected trees. Exploring the power of randomized greedy algorithms in more depth is interesting as well.
- Another intriguing open problem is to prove better bounds on the size of the gap between the path coloring and the fractional path coloring in bidirected trees by using different randomized rounding techniques. We believe that this approach is the most promising one for designing better approximation algorithms for the problem.
- Notice that the algorithms proposed are too far from optimality. Although an "absolute" inapproximability result of 4/3 is implied by the NP-completeness results in arbitrary trees, we are not aware of any asymptotic inapproximability results for the problem (i.e., inapproximability results for sets of paths of heavy load). This indicates another direction for future research.
- Also, closing the gap on the competitive ratio between the best known on-line path coloring algorithm in trees and the (randomized) lower bound is another interesting open problem.

References

1. A. Aggarwal, A. Bar-Noy, D. Coppersmith, R. Ramaswami, B. Schieber, and M. Sudan. Efficient Routing and Scheduling Algorithms for Optical Networks. *Journal of the ACM*, 43(6), 1996, pp. 973-1001.
2. V. Auletta, I. Caragiannis, C. Kaklamanis, and P. Persiano. Randomized Path Coloring on Binary Trees. *Theoretical Computer Science*, 289(1), pp. 355-399, 2002.

3. Y. Aumann and Y. Rabani. Improved Bounds for All Optical Routing. In *Proc. of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '95)*, 1995, pp. 567-576.
4. Y. Bartal and S. Leonardi. On-Line Routing in All-Optical Networks, *Theoretical Computer Science*, 221(1-2), 1999, pp. 19-39.
5. B. Beauquier, J.-C. Bermond, L. Gargano, P. Hell, S. Perennes, and U. Vaccaro. Graph Problems Arising from Wavelength-Routing in All-Optical Networks. *2nd Workshop on Optics and Computer Science (WOCS '97)*, 1997.
6. C. Berge. Graphs and Hypergraphs. *North-Holland*, 1973.
7. J.-C. Bermond, L. Gargano, S. Perennes, A.A. Rescigno, and U. Vaccaro. Efficient Collective Communication in Optical Networks. *Theoretical Compute Science*, 233 (1-2), 2000, pp. 165-189.
8. I. Caragiannis, A. Ferreira, C. Kaklamanis, S. Perennes, H. Rivano. Fractional Path Coloring with Applications to WDM Networks. In *Proc. of the 28th International Colloquium on Automata, Languages, and Programming (ICALP '01)*, LNCS 2076, Springer, 2001, pp. 732-743.
9. I. Caragiannis and C. Kaklamanis. Approximate Path Coloring with Applications to Wavelength Assignment in WDM Optical Networks. In *Proc. of the 21st International Symposium on Theoretical Aspects of Computer Science (STACS '04)*, LNCS 2996, Springer, pp. 258-269, 2004.
10. I. Caragiannis, C. Kaklamanis, and P. Persiano. Symmetric Communication in All-Optical Tree Networks. *Parallel Processing Letters*, 10(4), 2000, pp. 305-314.
11. I. Caragiannis, C. Kaklamanis, and P. Persiano. Edge Coloring of Bipartite Graphs with Constraints. *Theoretical Computer Science*, 270(1-2), pp. 361-399, 2002.
12. I. Caragiannis, C. Kaklamanis, and P. Persiano. Bounds on Optical Bandwidth Allocation in Directed Fiber Tree Topologies. *2nd Workshop on Optics and Computer Science*, 1997.
13. I. Caragiannis, C. Kaklamanis, P. Persiano, and A. Sidiropoulos. Fractional and Integral Coloring of Locally-Symmetric Sets of Paths on Binary Trees. In *Proc. of the 1st International Workshop on Approximation and Online Algorithms (WAOA '03)*, LNCS 2909, Springer, pp. 81-94, 2003.
14. T. Erlebach and K. Jansen. The Complexity of Path Coloring and Call Scheduling. *Theoretical Computer Science*, 255, pp. 33-50, 2001.
15. T. Erlebach, K. Jansen, C. Kaklamanis, M. Mihail, and P. Persiano. Optimal Wavelength Routing on Directed Fiber Trees. *Theoretical Computer Science* 221(1-2), 1999, pp. 119-137.
16. M.R. Garey, D.S. Johnson, G.L. Miller, and C.H. Papadimitriou. The Complexity of Coloring Circular Arcs and Chords. *SIAM Journal of Algebraic Discrete Methods*, 1(2), pp. 216-227, 1980.
17. L. Gargano, P. Hell, and S. Perennes. Colouring All Directed Paths in a Symmetric Tree with Applications to WDM Routing. In *Proc. of the 24th International Colloquium on Automata, Languages, and Programming (ICALP '97)*, LNCS 1256, Springer, pp. 505-515, 1997.
18. M.C. Golumbic and R.E. Jamison. The Edge Intersection Graphs of Paths in a Tree. *Journal of Combinatorial Theory Series B*, 38(1), pp. 8-22, 1985.
19. P. E. Green. Fiber-Optic Communication Networks. *Prentice-Hall*, 1992.
20. M. Grötschel, L.Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1, pp. 169-197, 1981.
21. I. Holyer. The NP-Completeness of Edge Coloring. *SIAM Journal of Computing*, 10(4), pp. 718-720, 1981.

22. K. Jansen. Approximation Results for Wavelength Routing in Directed Binary Trees. *2nd Workshop on Optics and Computer Science*, 1997.
23. S. Irani. Coloring Inductive Graphs On-line, *Algorithmica*, 11(1), pp. 53-72, 1994.
24. I. A. Karapetian. On the Coloring of Circular Arc Graphs. *Akad. Nauk Armeyan SSR Doklady*, 70(5), pp. 306-311, 1980 (in Russian).
25. R. Klasing. Methods and Problems of Wavelength-Routing in All-Optical Networks. In *Proc. of MFCS 98 Workshop on Communication*, pp. 1-9, 1998.
26. J. Kleinberg and E. Tardos. Disjoint Paths in Densely Embedded Graphs. In *Proc. of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS '95)*, pp. 52-61, 1995.
27. V. Kumar. An Approximation Algorithm for Circular Arc Coloring. *Algorithmica*, 30(3), pp. 406-417, 2001.
28. S.R. Kumar, R. Panigrahy, A. Russell, and R. Sundaram. A Note on Optical Routing in Trees, *Information Processing Letters*, 62, pp. 295-300, 1997.
29. E. Kumar and E. Schwabe. Improved Access to Optical Bandwidth in Trees. In *Proc. of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '97)*, pp. 437-444, 1997.
30. S. Leonardi and A. Vitaletti. Randomized Lower Bounds for On-line Path Coloring. In *Proc. of the 2nd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM '98)*, LNCS 1518, Springer, pp. 232-247, 1998.
31. D. Minoli. Telecommunications Technology Handbook, *Artech House*, 1991.
32. T. Nishizeki and K. Kashiwagi. On the 1.1 Edge-Coloring of Multigraphs. *SIAM Journal of Discrete Mathematics*, 3(3), pp. 391-410, 1990.
33. R. Pankaj. Architectures for Linear Lightwave Networks. PhD. Thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge MA, 1992.
34. Y. Rabani. Path Coloring on the Mesh. In *Proc. of the 37th IEEE Symposium on Foundations of Computer Science (FOCS '96)*, pp. 400-409, 1996.
35. P. Raghavan and E. Upfal. Efficient Routing in All-Optical Networks. In *Proc. of the 26th Annual ACM Symposium on the Theory of Computing (STOC '94)*, pp. 133-143, 1994.
36. R. Ramaswami and K. Sivarajan. Optical Networks: A Practical Perspective. *Morgan Kauffman Publishers*, 1998.
37. C.E. Shannon. A Theorem on Colouring Lines of a Network. *J. Math. Phys.*, 28, pp. 148-151, 1949.
38. A. Tucker. Coloring a Family of Circular Arcs. *SIAM Journal on Applied Mathematics*, 29(3), 493-502, 1975.