Approximation algorithms and complexity results for path problems in trees of rings

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

T. Erlebach

Approximation Algorithms and Complexity Results for Path Problems in Trees of Rings

TIK-Report
Nr. 109, August 2001

Institut für Technische Informatik und Kommunikationsnetze
Computer Engineering and Networks Laboratory

T. Erlebach
Approximation Algorithms and Complexity Results for Path Problems in Trees of Rings
August 2001
Version 1
TIK-Report Nr. 109

Computer Engineering and Networks Laboratory,
Swiss Federal Institute of Technology (ETH) Zurich

Institut für Technische Informatik und Kommunikationsnetze,
Eidgenössische Technische Hochschule Zürich

Gloriastrasse 35, ETH Zentrum, CH-8092 Zürich, Switzerland

Abstract

A tree of rings is a network that is obtained by interconnecting rings in a tree structure such that any two rings share at most one node. A connection request (call) in a tree of rings is given by its two endpoints and, in the case of prespecified paths, a path connecting these two endpoints. We study undirected trees of rings as well as bidirected trees of rings. In both cases, we show that the path packing problem (assigning paths to calls so as to minimize the maximum load) can be solved in polynomial time, that the path coloring problem with prespecified paths can be approximated within a constant factor, and that the maximum (weight) edge-disjoint paths problem is APX-hard and can be approximated within a constant factor (no matter whether the paths are prespecified or can be determined by the algorithms). We also consider fault-tolerance in trees of rings: If a set of calls has been established along edge-disjoint paths and if an arbitrary link fails in every ring of the tree of rings, we show that at least one third of the calls can be recovered if rerouting is allowed. Furthermore, computing the optimal number of calls that can be recovered is shown to be polynomial in undirected trees of rings and APX-hard in bidirected trees of rings.

Contents

1 Introduction

A *ring* is a graph that consists of a single cycle of length at least three. The *trees of rings* are the class of graphs that can be defined inductively as follows:

1. A single ring is a tree of rings.
2. If T is a tree of rings, then the graph obtained by adding a node-disjoint ring R to T and then identifying one node of R with one node of T is also a tree of rings.
3. No other graphs are trees of rings.

Equivalently, a tree of rings is a connected graph T whose edges can be partitioned into rings such that any two rings have at most one node in common and such that, for all pairs (u, v) of nodes in T, all simple paths from u to v touch precisely the same rings. (We say that a path touches a ring if it contains at least one edge of that ring. Furthermore, a path touches a node if it starts at that node, ends at that node, or passes through that node. Two paths *intersect* if they share an edge.) A third equivalent definition is that a tree of rings is a connected graph in which all biconnected components are rings.

Trees of rings are an interesting topology for the construction of communication networks. On the one hand, they are not expensive to build and require few additional links as compared to a tree topology. On the other hand, a tree of rings remains connected even if an arbitrary link fails in each ring, thus achieving much better fault tolerance than a tree network. Trees of rings are also a natural topology for all-optical networks: fiber rings have been employed for a long time (e.g., SONET rings), and it is natural to interconnect different rings in a tree structure.

In modern communication networks, establishing a connection between two nodes often requires allocating resources on all links along a path between the two nodes. For example, this is the case in ATM networks (where bandwidth is reserved along the path from sender to receiver) or in all-optical networks with wavelength-division multiplexing (where a wavelength is reserved along the path from sender to receiver). Resource allocation and call admission control in such networks lead to optimization problems involving edge-disjoint paths. (See [15] for more about the background of such problems.) In this paper, we investigate the complexity and approximability of these problems in networks with tree-of-rings topology.

1.1 Preliminaries and Problem Definitions

Each of the following problems can be studied for undirected paths in undirected graphs and for directed paths in bidirected graphs. (A bidirected graph is the graph obtained from an undirected graph by replacing each undirected edge by two directed edges with opposite directions. We use the term "link" to refer to an undirected edge in an undirected graph or to a pair of directed edges with opposite directions in a bidirected graph.) In the undirected case, a connection request (call) is given by its two endpoints, and the call is established along an undirected path between these two endpoints. In the bidirected case, a call specifies sender and receiver, and it is established along a directed path from sender to receiver.

Given a set of paths in a graph, the *load* $L(e)$ of a (directed or undirected) edge e is the number of paths containing that edge. The *maximum load* is the maximum of the values $L(e)$, taken over

all edges e of the graph. In this paper, the maximum load is denoted by L. The set of paths that creates this maximum load will always be clear from the context.

Now we define the relevant optimization problems.

Path Coloring (PC): Given a set of calls, assign paths and colors to the calls such that calls receive different colors if their paths intersect. Minimize the number of colors. (Application: Minimize the number of wavelengths in an all-optical WDM network.)

PC with Prespecified Paths (PCwPP): Same as path coloring, but the paths are specified as part of the input.

Path Packing (PP): Given a set of calls, assign paths to the calls such that the maximum load is minimized. (Application: Minimize the required link capacity if all calls request the same bandwidth.)

Maximum Edge-Disjoint Paths (MEDP): Given a set of calls, select a subset of the calls and assign edge-disjoint paths to the calls in that subset. Maximize the cardinality of the subset. (Application: Maximize the number of established calls.)

Maximum Weight Edge-Disjoint Paths (MWEDP): Given a set of calls that are assigned positive weights, select a subset of the calls and assign edge-disjoint paths to the calls in that subset. Maximize the total weight of the subset.

MEDP with Prespecified Paths (MEDPwPP): Same as MEDP, but the paths are specified as part of the input.

MWEDP with Prespecified Paths (MWEDPwPP): Same as MWEDP, but the paths are specified as part of the input.

Optimal Recovery (OR): Given a set C of calls that are assigned edge-disjoint paths in a graph $T = (V, E)$, and a set $F \subseteq E$ of faulty links, select a subset $C' \subseteq C$ and assign edge-disjoint paths in $T \setminus F$ to the calls in C'. Maximize the cardinality of C'. (Application: Maximize the number of calls that can remain active in spite of the link failures.)

Optimal Weighted Recovery (OWR): Same as OR, but the calls are assigned positive weights and the goal is to maximize the total weight of the calls in C'.

For maximization problems, an algorithm is a ρ-approximation algorithm if it runs in polynomial time and always computes a solution whose objective value is at least a $\frac{1}{\rho}$-fraction of the optimum. For minimization problems, an algorithm is a ρ-approximation algorithm if it runs in polynomial time and always computes a solution whose objective value is at most ρ times the optimum. APX is the class of all optimization problems in NPO (i.e., problems whose instances and solutions can be recognized in polynomial time, which have solutions of polynomial size, and for which the objective value of a given solution can be computed in polynomial time) that admit constant-factor approximation algorithms. A problem can be proved APX-hard by giving an approximation-preserving reduction (usually an L-reduction [18] or an AP-reduction [1]) from a known APX-hard problem. A problem is APX-complete if it is APX-hard and contained in APX. We refer to [1] for further background information concerning complexity classes for optimization problems, approximation-preserving reductions, and a discussion of the relationship of APX-hardness to the earlier concept of MAX SNP-hardness.

MEDP and OR as well as their variants can also be studied for the case that the network is an all-optical network with W wavelengths. In the case of MEDP, for example, this means that the goal is to compute a subset of the given calls and an assignment of paths and at most W different colors to the calls in that subset such that calls with the same color are routed along edge-disjoint paths. Approximation algorithms for MEDP and its variants in the case of W wavelengths can be obtained by calling an algorithm for one wavelength W times. The resulting approximation ratio is only slightly worse than for one wavelength, i.e., at most $1/(1-e^{-1/\rho}) \leq \rho+1$ if the approximation ratio for one wavelength is ρ [20, 2].

1.2 Previous Work on Rings and Trees

Most of the problems defined in the previous section have been studied intensively for rings and for trees (and, of course, for many other topologies). For undirected and bidirected ring networks, it is known that PC is $\mathcal{NP}$-hard no matter whether the paths are prespecified [12] or not [9, 21], and that PP can be solved optimally in polynomial time [21]. MEDP, MWEDP, MEDPwPP, and MWEDPwPP are polynomial for undirected and bidirected rings.

In undirected and bidirected tree networks, PC is also $\mathcal{NP}$-hard [9, 16]. A given set of paths with maximum load L can be efficiently colored using at most $3L/2$ colors in the undirected case [19] and at most $\min\{2L-1, \lceil 5L/3 \rceil\}$ colors in the bidirected case [11]. MEDP and MWEDP are polynomial in undirected trees of arbitrary degree and in bidirected trees of constant degree, but MAX SNP-hard and APX-hard in bidirected trees of arbitrary degree [10]. For every positive constant ε, there is a $(\frac{5}{3}+\varepsilon)$-approximation algorithm for MEDP and MWEDP in bidirected trees of arbitrary degree [10, 8].

1.3 Previous Work on Trees of Rings

It is known that an algorithm for PC in trees that uses at most αL colors can be used to obtain a 2α-approximation algorithm for PC in trees of rings, both in the undirected [19] and in the bidirected case [17]: It is sufficient to remove an arbitrary link from each ring in the tree of rings (the "cut-one-link" heuristic) and to use the tree algorithm in the resulting tree; the maximum load of the obtained paths is at most twice the load of the paths in the optimal solution, which in turn is a lower bound on the optimal number of colors. In this way, a 3-approximation algorithm is obtained in the undirected case and a $\frac{10}{3}$-approximation algorithm in the bidirected case. For undirected trees of rings, a 2-approximation algorithm for PCwPP was given in [6] for the special case in which each node is contained in at most two rings (i.e., in trees of rings with maximum vertex degree equal to four).

The all-to-all instance (the set of calls containing one call for every ordered pair of nodes) in bidirected trees of rings was studied in [4]. It was shown that a routing that minimizes the maximum load L can be computed in polynomial time, and that the resulting paths can be colored optimally with L colors.

1.4 Summary of Results

In this paper, we consider the path problems defined in Section 1.1 for networks with tree-of-rings topology. We show that PP is polynomial for trees of rings (Section 2) and give a practical

approximation algorithm with constant approximation ratio for PCwPP (Section 3). In Section 4, we show that at least one third of the active calls in a tree of rings can be recovered after an arbitrary link fails in every ring of the tree of rings, provided that rerouting is allowed. OR and OWR are proved to be polynomial for undirected trees of rings, while OR is shown to be APX-hard for bidirected trees of rings even if $T \setminus F$ is a tree. In Section 5, we consider MEDP, MWEDP, MEDPwPP, and MWEDPwPP. We prove that all these problems are APX-hard for trees of rings and give constant-factor approximation algorithms.

2 Path Packing

Path packing in trees of rings can be reduced to path packing in rings. Let c be a call and let p be a path connecting the endpoints of c. For each ring r in the given tree of rings that is touched by p, the node $in(r, c)$ at which p enters the ring (or begins) and the node $out(r, c)$ at which p leaves the ring (or terminates) are uniquely determined by the endpoints of c. Therefore, PP can be tackled by considering each ring separately. For a ring r, simply view each call c that touches r as a call from $in(r, c)$ to $out(r, c)$ and compute a routing that minimizes the maximum load using the known polynomial algorithms for PP in ring networks. The resulting routings can be combined to obtain an optimal routing for the tree of rings.

Theorem 1 *PP can be solved optimally in polynomial time for undirected and bidirected trees of rings.*

Note that Theorem 1 implies that one can decide in polynomial time whether *all* given calls can be established along edge-disjoint paths. Hence, this decision version of the edge-disjoint paths problem is polynomial for trees of rings.

3 Path Coloring with Prespecified Paths

As PCwPP is $\mathcal{NP}$-hard in rings, PCwPP is also $\mathcal{NP}$-hard in trees of rings. Given a set P of paths in a tree of rings $T = (V, E)$, we propose the following greedy approximation algorithm. It is a generalization of the simple greedy algorithm that uses at most $2L - 1$ colors for PC in trees.

algorithm GreedyColoring(T, P):

1. Initially, all paths are uncolored.
2. Choose an arbitrary node $s \in V$ and perform a depth-first search (DFS) of T starting at s. When the DFS reaches a node v, consider all uncolored paths $p \in P$ that touch v, in arbitrary order, and assign to each of them the smallest available color (i.e., the color with smallest index such that no path intersecting p has already been assigned that color).

In order to derive an upper bound on the number of colors used by GreedyColoring, we consider an arbitrary path p at the time it is assigned its color and show that it can intersect only a bounded number of paths that have already been assigned a color prior to p.

See Fig. 1. Assume that the dark nodes have been visited by the DFS already, and that the DFS now reaches v for the first time. Let r denote the ring containing v and a node adjacent to

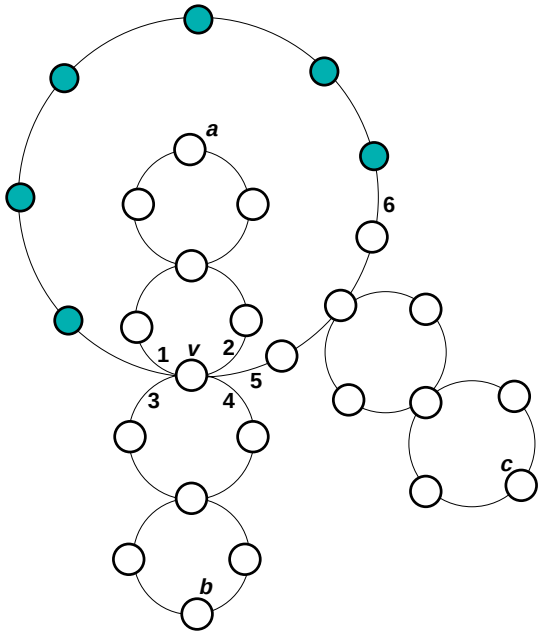

Figure 1: Illustration of the argument used to prove Theorem 2.

v that has been visited already (there is only one such ring). The uncolored paths touching v do not touch any dark node. They can be classified into two basic types. The first type does not use any edge in r; for the arguments given below, take a path connecting nodes a and b in the figure as a representative path for the first type. The second type uses at least one edge in r; take a path connecting b and c as a representative path.

First, consider the case of undirected paths. If p belongs to the first type, all previously colored paths that intersect p must also touch v and touch a ring containing v that is touched by p. But there are at most four edges incident to v that belong to rings touched by p (in the figure, these are the edges labeled 1, 2, 3, 4), and each conflicting path must use one of these edges. As the maximum load is L, there are less than $4L$ colored paths intersecting p. If p belongs to the second type, the number of colored paths that can intersect p is also less than $4L$: if we denote the edges as shown in Fig. 1, there can be at most $L-1$ colored paths using edge 5, at most L colored paths using edge 6, and at most $2L-1$ colored paths using edges 3 and 4. Thus, GreedyColoring uses at most $4L$ colors in the undirected case.

The same argument allows us to bound the number of conflicting paths by $8L-1$ in the bidirected case. We lose a factor of two because now there can be $2L$ paths using the link between two adjacent nodes, L in each direction.

Theorem 2 *For PCwPP in trees of rings, GreedyColoring is a polynomial-time algorithm that uses at most $4L$ colors in the undirected case and at most $8L$ colors in the bidirected case.*

Note that a PC algorithm can be derived from a PCwPP algorithm as follows. First, compute a routing that minimizes the maximum load L (Theorem 1); then, use a PCwPP algorithm to assign colors to the resulting paths. If the PCwPP algorithm uses at most αL colors, an α-approximation algorithm for PC is obtained in this way. Using Theorem 2, we obtain a PC algorithm with approximation ratio at most 4 in the undirected case and at most 8 in the bidirected case. These

ratios are worse than the ratios achieved by the "cut-one-link" heuristic (see Section 1.3), but it is not clear whether our upper bounds are tight, and it is conceivable that our new algorithms perform better in practice.

4 Recovery after Link Failures

In this section we study the problems OR and OWR in trees of rings. For bidirected trees of rings, we assume that a failure always affects both edges in a pair of directed edges with opposite directions between two adjacent nodes.

Let $T = (V, E)$ be a tree of rings and let $F \subset E$ be the set of faulty links. The worst case among all failure patterns that leave the remaining network connected is the case that $T \setminus F$ is a tree.

Assume that a set C of calls has been active at the time of the failure, i.e., the calls in C are assigned edge-disjoint paths in T. In the worst case, every call in C used a link in F, and all calls are affected by the link failures. Therefore, it is important to allow *rerouting*, i.e., changing the paths assigned to the calls.

If $T \setminus F$ is a tree, there is only one possible routing in $T \setminus F$ for each call $c \in C$. Let P be the set of paths obtained from C by routing each call in $T \setminus F$. The maximum load L created by the paths in P is at most two. (For any edge $e \in E \setminus F$, the paths in P that contain e either contained e already before the link failures, or they contained the faulty link in the ring to which e belongs.) In this case, solving the OR (OWR) problem reduces to solving the MEDP (MWEDP) problem for the set of paths P in the tree $T \setminus F$. As MWEDP is polynomial in undirected trees, this shows that OR and OWR are polynomial for undirected trees of rings (under the assumption that $T \setminus F$ is a tree). For bidirected trees of arbitrary degree, it was not known previously whether MEDP is already $\mathcal{NP}$-hard if the input is a set of paths with maximum load two. However, we have obtained the following theorem.

Theorem 3 *MEDP in bidirected trees is* APX*-hard even if the given set of paths has maximum load* $L = 2$.

The proof of Theorem 3 is given in Appendix A.

For any set P of paths in a bidirected tree T_1 with maximum load $L = 2$, one can construct a set C of calls that can be routed along edge-disjoint paths in a bidirected tree of rings T_2 and a set F of faulty links in T_2 such that the conflict graph of the paths obtained by routing the calls in C in $T_2 \setminus F$ is identical to the conflict graph of P in T_1. Therefore, Theorem 3 implies that OR and OWR are APX-hard for bidirected trees of rings T even if $T \setminus F$ is a tree.

The $(\frac{5}{3}+\varepsilon)$-approximation for MWEDP in bidirected trees from [8] gives a $(\frac{5}{3}+\varepsilon)$-approximation for OR and OWR in bidirected trees of rings provided that $T \setminus F$ is a tree. Furthermore, we can extend this approach to arbitrary sets F of faulty links (i.e., $T \setminus F$ could be disconnected or still contain some complete rings). In general, $T \setminus F$ consists of several connected components. All calls in C whose endpoints are in different components cannot be recovered. Consider one particular connected component H of $T \setminus F$. Intuitively, only the "tree part" of H is relevant for OR and OWR. Each ring in H that does not contain a faulty link can be shrunk into a single node. The resulting network is a tree T'. Then the MWEDP algorithm for trees is applied to T'. The resulting

set of edge-disjoint paths in T' gives a set of edge-disjoint paths in H by using the original routing in the non-faulty rings. Calls that touch only non-faulty rings are always recovered. Thus, we obtain the following theorem.

Theorem 4 *For undirected trees of rings, OR and OWR can be solved optimally in polynomial time. For bidirected trees of rings, OR and OWR are* APX*-hard even if $T \setminus F$ is a tree, and there is a $(\frac{5}{3}+\varepsilon)$-approximation for arbitrary sets F of faulty links.*

A set P of paths in a tree with $L = 2$ can always be colored (efficiently and optimally) with at most 3 colors, both in the undirected case and in the bidirected case. The paths assigned the same color (a color class) are edge-disjoint. Therefore, there is a set $S \subseteq P$ of edge-disjoint paths such that $|S| \geq |P|/3$. In the weighted case we can also infer that there is a set $S' \subseteq P$ of edge-disjoint paths whose total weight is at least one third of the total weight of P. The set S resp. S' can be computed efficiently by coloring P and taking the best of the color classes. If $T \setminus F$ still contains some rings, we can again shrink the rings into single nodes before applying the path coloring algorithm.

Theorem 5 *Assume that a set C of calls has been established along edge-disjoint paths in an undirected or bidirected tree of rings $T = (V, E)$. For any set F of faulty links such that F contains at most one link from every ring in T, it is possible to recover (efficiently) at least $|C|/3$ calls (or calls whose total weight is at least one third of the total weight of C), provided that rerouting is allowed.*

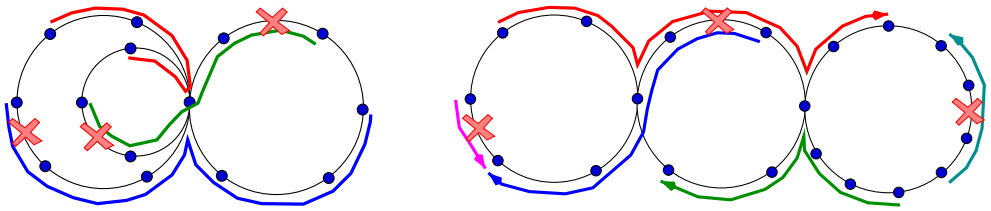

Figure 2: Examples of bad link failures.

For undirected trees of rings, the bound given in Theorem 5 is tight, i.e., there exists a set of edge-disjoint paths in a tree of rings and a set of link failures such that only one third of the paths can be recovered. See the left-hand side of Fig. 2 for an example. For bidirected trees of rings, we can construct an example where at most $2|C|/5$ calls can be recovered after failure, even if an optimal rerouting is used (right-hand side of Fig. 2). So we know that the fraction of calls that can be recovered after such link failures (where $T \setminus F$ is a tree) in the worst case is at least $\frac{1}{3}$ and at most $\frac{2}{5}$ for bidirected trees of rings.

Now we consider the effect of link failures in an all-optical network with wavelength-division multiplexing (WDM). If W wavelengths (colors) are available, a set C of calls can be established simultaneously if the calls are assigned paths and colors such that intersecting paths receive different colors and such that at most W different colors are used altogether. Corollary 1 follows from Theorem 5 by considering the calls with each wavelength separately.

Corollary 1 *Assume that a set C of calls has been established in an undirected or bidirected tree of rings $T = (V, E)$ using W wavelengths. For any set F of faulty links such that F contains at most one link from every ring in T, it is possible to recover at least $|C|/3$ calls (or calls whose total weight is at least one third of the total weight of C) without changing the wavelength of a call, provided that rerouting is allowed.*

5 Maximum (Weight) Edge-Disjoint Paths

While MEDP and MWEDP are polynomial for rings and for undirected trees, we can show that MEDP and its variants are all $\mathcal{NP}$-hard in undirected and bidirected trees of rings. In fact, we can prove that these problems are MAX SNP-hard and APX-hard, implying that there is a constant $r > 1$ such that approximating any of these problems with approximation ratio at most r is $\mathcal{NP}$-hard.

Theorem 6 *MEDP, MWEDP, MEDPwPP, and MWEDPwPP are all* APX*-hard for trees of rings, both in the undirected case and in the bidirected case.*

The proof of Theorem 6 is given in Appendix B. We remark that these problems can be solved optimally in polynomial time for trees of rings whose degree is bounded by a constant (i.e., where each node can be only in a bounded number of different rings), because the dynamic programming approach from [7] that works for bidirected trees of bounded degree can be generalized to graphs of bounded treewidth and bounded degree. Note that trees of rings have treewidth 2.

In view of the hardness results for trees of rings with arbitrary degree, we are interested in obtaining efficient approximation algorithms.

5.1 Arbitrary Paths

Consider the "cut-one-link" heuristic for MEDP and MWEDP: remove an arbitrary link from each ring and then use an algorithm for MEDP or MWEDP in trees. Theorem 5 implies that the optimal solution in the resulting tree has an objective value that is at least one third of the optimal solution in the full tree of rings.

Theorem 7 *Using the "cut-one-link" heuristic and an α-approximation algorithm for MEDP (for MWEDP) in trees gives a 3α-approximation algorithm for MEDP (for MWEDP) in trees of rings, both in the undirected case and in the bidirected case.*

Noting that MWEDP is polynomial (i.e., admits a 1-approximation algorithm) in undirected trees and that there is a $(\frac{5}{3} + \varepsilon)$-approximation for MWEDP in bidirected trees, we obtain the following corollary.

Corollary 2 *There is a 3-approximation algorithm for MWEDP in undirected trees of rings and a $(5 + \varepsilon)$-approximation algorithm for MWEDP in bidirected trees of rings, where ε is an arbitrary positive constant.*

5.2 Prespecified Paths

Next, we consider MEDPwPP and MWEDPwPP in trees of rings. Given a set P of paths in a tree of rings $T = (V, E)$, we propose the following greedy algorithm for MEDPwPP.

algorithm GreedySelection(T, P):

1. Maintain a set $S \subseteq P$ of selected paths. Initially, $S = \emptyset$.
2. Choose an arbitrary node $s \in V$ and perform a DFS of T starting at s. When the DFS is about to retreat from a node v (i.e., when all neighbors of v have been visited and the DFS goes back to the parent of v in the DFS tree), consider all paths $p \in P$ that touch v and that do not touch any node from which the DFS has not yet retreated, in arbitrary order. If p is edge-disjoint from all paths in S, insert p into S, otherwise discard p.
3. Output S.

Intuitively, the algorithm processes the DFS-tree of T in a bottom-up fashion and greedily selects paths that touch only nodes in subtrees that have already been processed. For the analysis of the approximation ratio of GreedySelection, please refer again to Fig. 1. Now the dark nodes represent nodes from which the DFS has not yet retreated. If the DFS is about to retreat from v, the paths that are considered can be classified into the same two types as in Section 3. If GreedySelection selects a path p that is not in the optimal solution, we can bound the number of later paths (paths that are considered after p by GreedySelection) that are in the optimal solution and that are intersected by p using the same arguments as in Section 3. In this way we obtain that there can be at most 4 such paths in the undirected case and at most 8 such paths in the bidirected case. (For example, in the undirected case, all later paths that are blocked by a path p from a to b in Fig. 1 must use one of the four edges labeled 1 to 4.) Therefore, each path selected by GreedySelection blocks at most 4 resp. 8 later paths that could be selected in an optimal solution instead. This reasoning can easily be turned into a formal proof that establishes the following theorem.

Theorem 8 *GreedySelection has approximation ratio at most* 4 *for MEDPwPP in undirected trees of rings and approximation ratio at most* 8 *for MEDPwPP in bidirected trees of rings.*

GreedySelection can be converted into an approximation algorithm for MWEDPwPP by adopting a two-phase approach similar to the one used by Berman et al. for an interval selection problem [5]. (This can also be seen as an application of the local ratio technique, see [3].) The algorithm maintains a stack S of paths. When a path is pushed on the stack, it is assigned a *value* that may be different from its weight. In the end, the paths are popped from the stack and selected in a greedy way.

algorithm TwoPhaseSelection(T, P):

1. Maintain a stack S of paths with values. Initially, S is the empty stack.
2. Phase One: Choose an arbitrary node $s \in V$ and perform a DFS of T starting at s. When the DFS is about to retreat from a node v, consider all paths $p \in P$ that touch v and that do not touch any node from which the DFS has not yet retreated, in arbitrary order. Compute $total(p, S)$, the sum of the values of the paths in S that intersect p. If the weight $w(p)$ of p is strictly larger than $total(p, S)$, then push p onto the stack and assign it the value $w(p) - total(p, S)$.

3. Phase Two: After the DFS is complete, pop the paths from the stack and select each path if it does not intersect any previously selected path.

4. Output the set of selected paths.

We can prove the same upper bound on the approximation ratio of TwoPhaseSelection for the problem MWEDPwPP as for GreedySelection for MEDPwPP.

Theorem 9 *TwoPhaseSelection is a* 4*-approximation algorithm for MWEDPwPP in undirected trees of rings and an* 8*-approximation algorithm for MWEDPwPP in bidirected trees of rings.*

Proof. Let $\overline{S}$ denote the contents of the stack at the end of Phase One, and let t denote the sum of the values of the paths in $\overline{S}$. Fix an arbitrary optimal solution P^*. The proof consists of two parts.

First, one can show that the total weight of the paths in P^* is at most $4t$ (at most $8t$) for undirected (bidirected) trees of rings. To see this, let $val(p, \overline{S})$ denote the sum of the values of paths in $\overline{S}$ that intersect p and that were processed in Phase One not after p. (If $p \in \overline{S}$, then the value of p also contributes to $val(p, \overline{S})$.) By the definition of the algorithm, $val(p, \overline{S}) \geq w(p)$ for all paths p. Now the key observation is that the value of every path $q \in \overline{S}$ is counted at most four (eight) times in the sum $\sum_{p \in P^*} val(p, \overline{S}) \geq w(P^*)$. This can again be seen by referring to Fig. 1 and considering the path q to be either of the first type (from a to b) or of the second type (from b to c). In the undirected case, there can be at most four paths in P^* that are processed after q in Phase One and that intersect q. In the bidirected case, there can be at most eight such paths. Therefore, $w(P^*) \leq 4t$ in the undirected case (resp. $w(P^*) \leq 8t$ in the bidirected case).

Second, the solution output by the algorithm has weight at least t: when a path p is selected in Phase Two, the weight of the solution increases by $w(p)$, while the sum of the values of the paths that are still on the stack and that are not intersected by p decreases by at most $w(p)$. □

References

[1] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation. Combinatorial Optimization Problems and their Approximability Properties*. Springer, Berlin, 1999.

[2] B. Awerbuch, Y. Azar, A. Fiat, S. Leonardi, and A. Rosén. On-line competitive algorithms for call admission in optical networks. *Algorithmica*, 31(1):29–43, 2001.

[3] A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. S. Naor, and B. Schieber. A unified approach to approximating resource allocation and scheduling. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing STOC'00*, pages 735–744, 2000.

[4] B. Beauquier, S. Pérennes, and D. Tóth. All-to-all routing and coloring in weighted trees of rings. In *Proceedings of the 11th Annual ACM Symposium on Parallel Algorithms and Architectures SPAA'99*, pages 185–190, 1999.

[5] P. Berman and B. DasGupta. Multi-phase algorithms for throughput maximization for real-time scheduling. *Journal of Combinatorial Optimization*, 4(3):307–323, 2000.

[6] X. Deng, G. Li, W. Zang, and Y. Zhou. A 2-approximation algorithm for path coloring on trees of rings. In *Proceedings of the 11th Annual International Symposium on Algorithms and Computation ISAAC 2000*, LNCS 1969, pages 144–155, 2000.

[7] T. Erlebach. *Scheduling Connections in Fast Networks*. PhD thesis, Technische Universität München, 1999. http://www.tik.ee.ethz.ch/∼erlebach/dissertation.ps.gz.

[8] T. Erlebach and K. Jansen. Conversion of coloring algorithms into maximum weight independent set algorithms. In *ICALP Workshops 2000*, Proceedings in Informatics 8, pages 135–145. Carleton Scientific, 2000.

[9] T. Erlebach and K. Jansen. The complexity of path coloring and call scheduling. *Theoretical Computer Science*, 255(1–2):33–50, 2001.

[10] T. Erlebach and K. Jansen. The maximum edge-disjoint paths problem in bidirected trees. *SIAM J. Disc. Math.*, 14(3):326–355, 2001.

[11] T. Erlebach, K. Jansen, C. Kaklamanis, M. Mihail, and P. Persiano. Optimal wavelength routing on directed fiber trees. *Theoretical Computer Science*, 221:119–137, 1999. Special issue of ICALP'97.

[12] M. R. Garey, D. S. Johnson, G. L. Miller, and C. H. Papadimitriou. The complexity of coloring circular arcs and chords. *SIAM J. Algebraic Discrete Methods*, 1(2):216–227, 1980.

[13] N. Garg, V. V. Vazirani, and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1):3–20, 1997.

[14] V. Kann. Maximum bounded 3-dimensional matching is MAX SNP-complete. *Information Processing Letters*, 37:27–35, 1991.

[15] J. Kleinberg. *Approximation algorithms for disjoint paths problems*. PhD thesis, MIT, 1996.

[16] S. R. Kumar, R. Panigrahy, A. Russel, and R. Sundaram. A note on optical routing on trees. *Inf. Process. Lett.*, 62:295–300, 1997.

[17] M. Mihail, C. Kaklamanis, and S. Rao. Efficient access to optical bandwidth. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science FOCS'95*, pages 548–557, 1995.

[18] C. Papadimitriou and M. Yannakakis. Optimization, approximization and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

[19] P. Raghavan and E. Upfal. Efficient routing in all-optical networks. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing STOC'94*, pages 134–143, 1994.

[20] P.-J. Wan and L. Liu. Maximal throughput in wavelength-routed optical networks. In *Multichannel Optical Networks: Theory and Practice*, volume 46 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 15–26. AMS, 1998.

[21] G. Wilfong and P. Winkler. Ring routing and wavelength translation. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms SODA'98*, pages 333–341, 1998.

A MEDP for Paths in Trees with Load Two

In this appendix, we give a proof of Theorem 3. For convenience, we repeat the theorem here.

Theorem 3 *MEDP in bidirected trees is* APX*-hard even if the given set of paths has maximum load* $L = 2$*.*

In order to prove Theorem 3, we use an approximation-preserving reduction from the maximum bounded 3-dimensional matching problem where each element occurs in at most two triples. First, we present a hardness result for the latter problem. Then, we give the approximation-preserving reduction.

Let us define the 3-dimensional matching problem formally.

MAX-3DM–*B* (Maximum bounded 3-dimensional matching). An instance is given by a set $M \subseteq X \times Y \times Z$ of ordered triples, where X, Y and Z are disjoint sets. The number of occurrences in M of an element in X, Y or Z is bounded by the constant B. A feasible solution is a matching, i.e., a subset $M' \subseteq M$ such that no two triples in M' agree in any coordinate. The goal is to maximize the cardinality of the matching.

Kann proved that MAX-3DM–B is MAX SNP-hard for any $B \geq 3$ by giving an approximation-preserving reduction from MAX-3SAT–B (Maximum bounded 3-satisfiability) to MAX-3DM–3 [14]. However, we are interested in the case $B = 2$. To our knowledge, no hardness results for MAX-3DM–2 have been known previously. Nevertheless, we can derive such a result easily by applying the reduction in [14] to MAX-2SAT–B instead of MAX-3SAT–B.

The reduction in [14] takes an instance of MAX-3SAT–B and constructs an instance of MAX-3DM–3. In the resulting instance, most of the elements of X, Y or Z occur in at most two triples. The only elements that appear in three triples are the two elements created for each clause containing three literals. If we reduce from MAX-2SAT–B instead of MAX-3SAT–B, each clause contains at most two literals, so each element of X, Y or Z will occur in at most two triples, and the resulting instance of MAX-3DM is actually an instance of MAX-3DM–2. Furthermore, it is known that MAX-2SAT–B is APX-complete for any $B \geq 3$ [1, p. 280]. So we obtain an approximation-preserving reduction from the APX-complete problem MAX-2SAT–B to MAX-3DM–2. Since MAX-3DM–2 is easily seen to be contained in APX, we obtain the following result.

Lemma 1 *MAX-3DM–*2 *is* APX*-complete.*

Now we show how to reduce MAX-3DM-B, for any $B \geq 2$, to MEDP in bidirected trees. The reduction is similar to the reduction used by Garg, Vazirani and Yannakakis to prove MAX SNP-hardness of integral multicommodity flow in undirected trees with edge capacities one and two [13].

Lemma 2 *For any* $B \geq 2$*, there is a polynomial-time reduction from MAX-3DM-B to MEDP in bidirected trees such that the set of paths in the constructed instance of MEDP has maximum load at most* B*.*

Proof. Let an instance I of MAX-3DM-B be given by three disjoint sets X, Y, Z and a set of triples $M \subseteq X \times Y \times Z$.

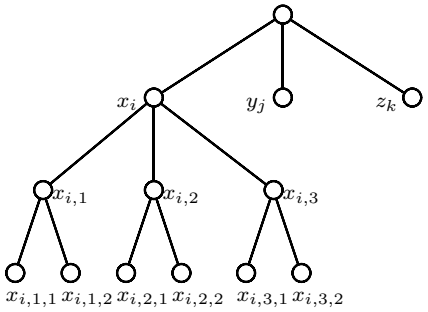

Figure 3: Tree constructed from the MAX-3DM-B instance.

We create a bidirected tree T of depth three as follows. The root of T has $|X| + |Y| + |Z|$ children: one for each $x_i \in X$, one for each $y_j \in Y$, and one for each $z_k \in Z$. Each node $x_i \in X$ has p_i children $x_{i,1},\ldots,x_{i,p_i}$, where $p_i \leq B$ is the number of occurrences of x_i in M. Each $x_{i,j}$ has two children $x_{i,j,1}$ and $x_{i,j,2}$. See Fig. 3 for a sketch of this construction; note that only a small subset of the nodes of the tree is actually shown there, and that pairs of oppositely directed edges are depicted as undirected edges for the sake of simplicity.

Now, we create a set P of paths in T. Number the occurrences of x_i in the triples of M from 1 to p_i arbitrarily. For every triple in M we add three paths to P. Let triple (x_i, y_j, z_k) contain the l-th occurrence of x_i in M. Then we add a path from $x_{i,l,1}$ to y_j, a path from z_k to $x_{i,l,2}$, and a path from $x_{i,l,1}$ to $x_{i,l,2}$. The set P of paths has maximum load at most B, since each of the x_i, y_j and z_k occurs in at most B triples in M.

We claim that M contains d disjoint triples if and only if P contains a subset P' of at least $|M| + d$ edge-disjoint paths, and that d disjoint triples in M can be computed efficiently from a given subset $P' \subseteq P$ of at least $|M| + d$ edge-disjoint paths.

Assume that M contains d disjoint triples $(x_{i_1}, y_{j_1}, z_{k_1}),\ldots,(x_{i_d}, y_{j_d}, z_{k_d})$. Let triple $(x_{i_t}, y_{j_t}, z_{k_t})$ contain the l_t-th occurrence of x_{i_t}. Then the following $|M| + d$ paths form a set P' of edge-disjoint paths: for each t, $1 \leq t \leq d$, choose the path from $x_{i_t,l_t,1}$ to y_{j_t}, the path from z_{k_t} to $x_{i_t,l_t,2}$, and $p_{i_t} - 1$ paths from $x_{i_t,l,1}$ to $x_{i_t,l,2}$ for $l \in \{1,\ldots,p_{i_t}\} \setminus \{l_t\}$; for each x_i that does not occur in the d disjoint triples, choose the p_i paths from $x_{i,l,1}$ to $x_{i,l,2}$ for $1 \leq l \leq p_i$.

Conversely, assume that there is a subset P' of P containing $|M| + d$ edge-disjoint paths. Note that P' can contain at most one path entering the subtree rooted at x_i from above and at most one path leaving the subtree rooted at x_i. The only possibility for P' to contain more than p_i paths using edges of the subtree rooted at x_i is to contain one path from $x_{i,l_i,1}$ to some y_j, one path from some z_k to $x_{i,l_i,2}$, and $p_i - 1$ paths from $x_{i,l,1}$ to $x_{i,l,2}$ for $l \in \{1,2,\ldots,p_i\} \setminus \{l_i\}$. In that case, P' contains $p_i + 1$ paths using edges of the subtree rooted at x_i. The only way for P' to contain at least $|M| + d$ paths is that P' contains exactly $p_i + 1$ paths using edges of the subtree rooted at x_i for at least d values of i. Then a set of d disjoint triples is obtained by taking, for each of the d values of i, the triple containing the l_i-th occurrence of x_i. □

For any set M of triples such that each element of $X \cup Y \cup Z$ occurs in at most B triples, there is always a matching that contains at least $|M|/(3B-2)$ triples, because each triple is in conflict with at most $3B-3$ other triples. Using this fact, it is not difficult to show that the reduction presented in the proof of Lemma 2 is approximation preserving for any constant B, in the sense

that it is an L-reduction [18] and an AP-reduction [1]. For the details of the calculation, see [13] or [10].

Since MAX-3DM-B is APX-hard for $B \geq 2$ by Lemma 1, this shows that MEDP in bidirected trees is APX-hard even if the maximum load of the given paths is bounded by 2. This establishes Theorem 3.

B Hardness of MEDP and MEDPwPP in Trees of Rings

In this appendix, we prove Theorem 6. For convenience, we repeat the theorem here.

Theorem 6 *MEDP, MWEDP, MEDPwPP, and MWEDPwPP are all* APX*-hard for trees of rings, both in the undirected case and in the bidirected case.*

It is enough to prove the theorem for the unweighted versions of the problems, i.e., for MEDP and MEDPwPP. The proofs for the individual cases are given in the following subsections. Some of our arguments use the reduction from MAX-3DM-B to MEDP given in Lemma 2 in Appendix A.

Note that any given tree can be turned into a tree of rings by adding a new vertex v_e for every edge e of the tree and making v_e adjacent to the two endpoints of e. We call the resulting tree of rings the *tor-extension* of the given tree.

B.1 MEDPwPP in Bidirected Trees of Rings

MEDPwPP for bidirected trees of rings is obviously APX-hard, because MEDP is APX-hard for bidirected trees and any set of paths in a tree can be viewed as a set of paths in the tor-extension of the tree.

B.2 MEDPwPP in Undirected Trees of Rings

We claim that we can transform any instance of MEDP in bidirected trees (which is APX-hard) into an instance of MEDPwPP in undirected trees of rings such that the paths have the same conflict structure. Start with the given bidirected tree. Replace each pair of directed edges (u, v) and (v, u) of the bidirected tree by an undirected ring $\{u, x\}$, $\{x, v\}$, $\{v, y\}$, $\{y, u\}$, where x and y are two new nodes. On each path using the directed edge (u, v) (the directed edge (v, u)), replace that edge by the two undirected edges $\{u, x\}$ and $\{x, v\}$ (by the two undirected edges $\{v, y\}$ and $\{y, u\}$). In this way we obtain a set of undirected paths in an undirected tree of rings such that two paths intersect if and only if the corresponding paths in the original bidirected tree intersect. Therefore, MEDPwPP in undirected trees of rings is at least as hard to approximate as MEDP in bidirected trees. This shows APX-hardness of MEDPwPP in undirected trees of rings.

B.3 MEDP in Bidirected Trees of Rings

We adapt the reduction used in the proof of Lemma 2. From a given instance of MAX-3DM-B we obtain a set of calls as in the proof of Lemma 2, but now these calls are not in a tree but in a tree of rings. The tree of rings is obtained form the tree constructed in the proof of Lemma 2 as follows: the links between the root and its children are replaced by rings of length 3, the nodes

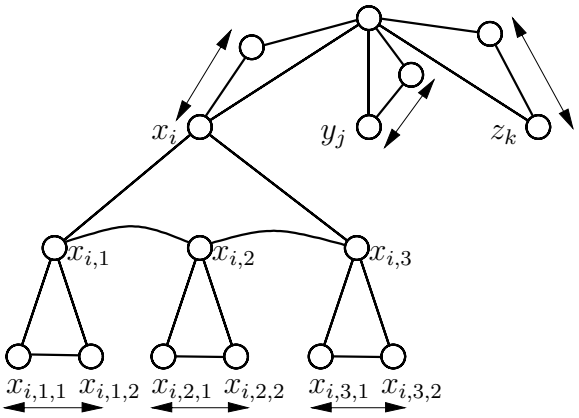

Figure 4: Tree of rings constructed from a MAX-3DM-B instance.

x_i and $x_{i,1}, x_{i,2}, \ldots, x_{i,p_i}$ form a ring, and an edge is added between $x_{i,l,1}$ and $x_{i,l,2}$ for $1 \leq l \leq p_i$. See Fig. 4. Finally, we add $2(|X| + |Y| + |Z| + |M|)$ calls between certain pairs of adjacent nodes as follows. For two adjacent nodes u and v, we add a call from u to v and a call from v to u. We add such calls on one link of every ring containing the root and on all links connecting some $x_{i,l,1}$ with $x_{i,l,2}$. These additional calls are called *short calls*. They are indicated by the short lines with arrows at both ends in Fig. 4.

The purpose of adding the short calls is made clear by the following claim.

Claim 1 *Let an instance of MEDP in any graph be given. Let S be a set of edge-disjoint paths for a subset of the calls in the instance. Then S can be transformed into a new solution S' with $|S'| \geq |S|$ such that S' contains, for every directed edge (u, v) of the graph for which one call from u to v is part of the instance, at least one path from u to v consisting only of the edge (u, v).*

The claim is easy to prove: If the solution S does not satisfy the condition, we can either add a call or reroute some calls in order to bring the solution closer to the desired form.

Claim 1 implies that we can assume without loss of generality that any solution to the constructed instance of MEDP in a bidirected tree of rings accepts all short calls and routes them along the direct edge. Then the situation for the remaining calls is just the same as it was in the tree case: a solution containing $|M| + d$ of the remaining calls can be converted into a matching of cardinality at least d for the original instance of MAX-3DM-B, and vice versa. Taking into account the short calls, we have that the resulting instance of MEDP has a solution with $2(|X| + |Y| + |Z|) + 3|M| + d$ edge-disjoint paths if and only if there exists a matching with d triples. The reduction is still approximation preserving (since $|X| + |Y| + |Z| \leq 3|M|$), and we get that MEDP is APX-hard for bidirected trees of rings.

B.4 MEDP in Undirected Trees of Rings

In order to prove APX-hardness for MEDP in undirected trees of rings, we can essentially use the same reduction as for MEDP in bidirected trees of rings. The only differences are: we view the tree of rings (shown in Fig. 4) as an undirected graph and all constructed calls as undirected calls,

we do not add short calls in the rings containing any of the x_i, and we replace each remaining pair of short calls from u to v and from v to u by a single undirected call between u and v. The resulting instance of MEDP in undirected trees of rings has a solution with $|Y| + |Z| + 2|M| + d$ edge-disjoint paths if and only if there exists a matching with d triples. Again, it is easy to show that the reduction is approximation-preserving.