

# On the Approximability of Reachability Preserving Network Orientations\*

Michael Elberfeld

Institute of Theoretical Computer Science  
University of Lübeck  
23538 Lübeck, Germany  
elberfeld@tcs.uni-luebeck.de

Iftah Gamzu

Microsoft R&D Center  
Herzliya 46725, Israel  
iftah.gamzu@cs.tau.ac.il

Danny Segev

Department of Statistics  
University of Haifa  
Haifa 31905, Israel  
segevd@stat.haifa.ac.il

Uri Zwick

Blavatnik School of Computer Science  
Tel Aviv University  
Tel Aviv 69978, Israel  
zwick@post.tau.ac.il

Vineet Bafna

Department of Computer Science  
University of California San Diego  
San Diego, USA  
vbafna@cs.ucsd.edu

Alexander Medvedovsky

Blavatnik School of Computer Science  
Tel Aviv University  
Tel Aviv 69978, Israel  
medv@post.tau.ac.il

Dana Silverbush

Blavatnik School of Computer Science  
Tel Aviv University  
Tel Aviv 69978, Israel  
danasilv@post.tau.ac.il

Roded Sharan<sup>†</sup>

Blavatnik School of Computer Science  
Tel Aviv University  
Tel Aviv 69978, Israel  
roded@post.tau.ac.il

## Abstract

We introduce a graph orientation problem arising in the study of biological networks. Given an undirected graph and a list of ordered source-target vertex pairs, the goal is to orient the graph such that a maximum number of pairs admit a directed source-to-target path. We study the complexity and approximability of this problem. We show that the problem is NP-hard even on star graphs and hard to approximate to within some constant factor. On the positive side, we provide an  $\Omega(\log \log n / \log n)$ -factor approximation algorithm for the problem on  $n$ -vertex graphs. We further show that for any instance of the problem there exists an orientation of the input graph that satisfies a logarithmic fraction of all pairs and that this bound is tight up to a constant factor. Our techniques also lead to constant factor approximation algorithms for some restricted variants of the problem.

**Key words:** network orientation, graph orientation, approximation algorithm, biological network, protein-protein interaction

## 1 Introduction

A major role of a protein-protein interaction (PPI) network is to transmit signals within the cell in response to genetic and environmental cues. Technologies for measuring PPIs such as yeast two-hybrid [5] and co-immunoprecipitation [9] are unable to provide information on the direction in which

---

\*This is a preprint of the article <http://dx.doi.org/10.1080/15427951.2011.604554>. Part of this work has been presented at the Workshop on Algorithms in Bioinformatics in the years 2008 [17] and 2010 [8].

<sup>†</sup>To whom correspondence should be addressed. Tel: +972-36407139. Fax: +972-36409357.

the signal flows. Such information can be inferred from indirect, causal information on cellular events. One such source of information are perturbation experiments in which a gene is perturbed (cause) and as a result other genes change their expression levels (effects).

In graph theoretic terms, one is given an undirected graph and a list of source-target pairs that represent the experimentally observed cause-effect pairs. The goal is to predict an orientation of the graph, i.e. a directed graph on the same vertex set that contains a single directed version of every undirected edge, so that for a maximum number of pairs the target is reachable from the source. We study the complexity of approximating the resulting `MAXIMUM-GRAPH-ORIENTATION` problem.

The algorithmic research on graph orientations that preserve reachability was initially focused on producing strongly connected orientations. It started with the 1939 paper of Robbins [19] who was motivated by applications in street network design and showed that an undirected graph has a strongly connected orientation if and only if it has no bridge edge. Hakimi et al. [11] presented a polynomial algorithm for the problem of orienting an undirected graph  $G = (V, E)$  to preserve reachability for a maximum number of all vertex pairs (i.e., from the set  $V \times V$ ). The problem we study is a generalization of the latter as it considers any subset of pairs. We refer to the textbook of Bang-Jensen and Gutin [2] for a comprehensive discussion of various graph orientation problems.

Yeang et al. [23] were the first to use perturbation experiments to annotate protein networks. They proposed a probabilistic model and an accompanying inference approach to predict edge directions and signs of activation and repression from cause-effect data. Ourfali et al. [18] provided an integer linear program formulation for the problem of inferring edge signs that maximize the expected number of explained cause-effect pairs. Gitter et al. [10] used `SATISFIABILITY`-based approximations to tackle the orientation problem. The main caveat of all these approaches is that they depend on enumerating all possible paths between a pair of genes and, hence, they are limited to paths of very short length (3 for the first two works and 5 for the latter). Finally, a work of Dorn et al. [3] studies the complexity of solving the orientation problem with respect to structural parameters that measure how often vertices (or edges) are used by source-to-target paths.

In this paper we show that the `MAXIMUM-GRAPH-ORIENTATION` problem can be reduced to the same problem on instances where the input graph is a tree. We focus on the latter problem, called `MAXIMUM-TREE-ORIENTATION`. We show that it is NP-hard and hard to approximate to within a factor of  $12/13$ . On the positive side, we show that for an  $n$ -vertex tree the problem can be approximated to within a factor of  $\Omega(\log \log n / \log n)$ . We also study combinatorial properties of graph orientations, showing that for every undirected graph and collection of source-target vertex pairs, there exists an orientation that satisfies a logarithmic fraction of the pairs and that this bound is tight up to a constant factor. We also present algorithms with constant approximation ratios for restricted instances of `MAXIMUM-TREE-ORIENTATION`. In particular, we provide a constant approximation algorithm for instances where the distance between source and target vertices is bounded by a constant.

The paper is organized as follows: In Section 2 we define the graph orientation problem that we consider and provide hardness results. In Section 3 we present combinatorial bounds on the number of pairs that can be satisfied in different orientation instances. In Section 4 we provide a constant-factor approximation algorithm for the restriction of `MAXIMUM-TREE-ORIENTATION` where the end vertices of pairs are connected by short paths. In Section 5 we present a sublogarithmic approximation algorithm for the general case.

## 2 Preliminaries

Let  $G = (V, E)$  be an undirected graph on  $n$  vertices. We also use  $V(G)$  and  $E(G)$  to refer to its sets  $V$  of vertices and  $E$  of edges, respectively. The graph  $G$  is a *tree* if it is connected and has no cycles. An *orientation* of  $G$  is an assignment of directions to the edges of  $G$  such that each edge is assigned a single direction.

Given a *source* vertex  $s \in V(G)$  and a *target* vertex  $t \in V(G)$ , we say that  $t$  is *reachable from*  $s$  if there exists a path in  $G$  from  $s$  to  $t$ . In this case we also say that  $G$  *satisfies* the *source-target vertex pair*  $(s,t)$ . The problem we study is formally defined as follows:

**Problem 2.1** (MAXIMUM-GRAPH-ORIENTATION).

**Input:** An undirected graph  $G = (V, E)$  and a multiset of source-target vertex pairs  $P$  from  $G$ .

**Output:** An orientation  $G'$  of  $G$  that satisfies a maximum number of pairs from  $P$ .

A tuple  $(G, P)$  of an undirected graph  $G$  and a multiset of source-target vertex pairs  $P$  from  $G$  is called an *orientation instance*;  $|P|$  denotes the number of pairs in the multiset  $P$ .

**Lemma 2.2.** *There exists a linear time algorithm that given an orientation instance  $(G, P_G)$ , computes an orientation instance  $(T, P_T)$  with tree  $T$ , such that for every  $k \in \mathbb{N}$  there exists an orientation  $G'$  of  $G$  that satisfies  $k$  pairs from  $P_G$  if and only if there exists an orientation  $T'$  of  $T$  that satisfies  $k$  pairs from  $P_T$ .*

*Proof.* The algorithm computes the tree  $T$  of  $G$ 's two-edge connected components. Vertex pairs  $P_G$  in  $G$  are transformed into vertex pairs  $P_T$  in  $T$  by the following rule: For every pair  $(s, t) \in P_G$  we construct a pair  $(C, C')$  where  $C$  and  $C'$  are the two-edge connected components of  $G$  that contain  $s$  and  $t$ , respectively. Computing the tree of two-edge connected components can be done in linear time as shown by Tarjan [21, 22]; the transformation of the pairs can also be done in linear time.

An orientation  $G'$  of  $G$  translates into an orientation  $T'$  of  $T$  by taking the oriented versions of the bridge edges between two-edge connected components. For every vertex pair  $(s, t) \in P_G$  that is satisfied in  $G'$ , the corresponding pair of components  $(C, C')$  is also satisfied in  $T'$ . Conversely, consider an orientation  $T'$  of  $T$ . We will use the orientations from the edges of  $T$  for the bridge edges between the two-edge connected components in  $G$  and a strongly connected orientation for each two-edge connected component. Such orientations exist by the Theorem of Robbins [19] and satisfy every pair whose source and target vertices lie in the same two-edge connected component.  $\square$

By Lemma 2.2, it is sufficient to solve MAXIMUM-GRAPH-ORIENTATION for orientation instances  $(T, P)$  with  $T$  being a tree. This results in the formal MAXIMUM-TREE-ORIENTATION problem. In the following, we show NP-hardness for an even more restricted problem variant where the input graph is a star.

**Theorem 2.3.** MAXIMUM-TREE-ORIENTATION is NP-hard to approximate to within a factor of  $12/13$ .

*Proof.* We reduce from the problem MAXIMUM-DIRECTED-CUT [14] where we are given a directed graph  $G$  and wish to find a subset  $A \subseteq V(G)$  such that a maximum number of edges cross from  $A$  to  $V(G) \setminus A$  in  $G$ .



For the reduction, we map a directed graph  $G$  to an orientation instance  $(S, P)$  with  $S$  being a star graph in the following way: We set  $V(S) = V(G) \cup \{v_c\}$  where  $v_c$  is a vertex not in  $V(S)$  that will be the vertex in the center of  $S$ . For every vertex  $v \in V(G)$ , we insert an undirected edge between  $v$  and  $v_c$  into the edge set  $E(S)$ . We define  $P$  to be the set of directed edges of  $G$ . We claim that for every  $k \in \mathbb{N}$  there exists a set  $A \subseteq V(G)$ , such that at least  $k$  edges from  $E(G)$  go from  $A$  to  $V(G) \setminus A$  if and only if there exists an orientation  $S'$  of  $S$  that satisfies at least  $k$  pairs from  $P$ .

For the ‘‘only if’’-direction consider a set  $A \subseteq V(G)$  with  $k$  crossing edges. For all  $v \in A$ , orient the edge between  $v$  and  $v_c$  from  $v$  toward  $v_c$ . All other edges are oriented away from  $v_c$ . Every pair with a corresponding edge that goes from  $A$  to  $V(G) \setminus A$  is satisfied by this orientation.

For the ‘‘if’’-direction let  $S'$  be an orientation of  $S$  that satisfies  $k$  pairs from  $P$ . Let  $A$  be the set of vertices from  $V(S) \setminus \{v_c\}$  whose incident edges are oriented toward  $v_c$ . Pairs that are satisfied in  $S'$

have their source vertices in the set  $A$  and their target vertices in the set  $V(S) \setminus A$ . To each of these pairs corresponds an edge that goes from  $A$  to  $V(G) \setminus A$  in  $G$ .

Since MAXIMUM-DIRECTED-CUT is NP-hard to approximate to within a factor of  $12/13$  [12] and the reduction is approximation-preserving, the claim follows.  $\square$

The MAXIMUM-TREE-ORIENTATION problem is also NP-hard when restricted to complete binary trees [17] or caterpillar trees of degree at most 3 like  [16]. In contrast, on path graphs (graphs like ) it can be solved in polynomial time by a dynamic programming algorithm [17, 3].

### 3 Combinatorial Logarithmic Bounds

In this section we prove bounds on the number of pairs that can be satisfied in various orientation instances. We first introduce the concept of *covers* and show that orientation instances with restricted covers admit orientations satisfying a constant fraction of the pairs. Then we turn to general orientation instances and show that they always admit an orientation satisfying a logarithmic fraction of their pairs and that this is tight up to a constant factor.

#### 3.1 Constant Factor Bounds

In the following we will consider tree instances and, instead of orienting edges individually, partition the input tree into subtrees and orient all edges of a subtree at the same time in a consistent direction.

A *cover for  $T$*  is a tuple  $(\mathcal{T}, \mathcal{W})$  that consists of: (1) a class  $\mathcal{T} = \{T_1, \dots, T_l\}$  of induced subtrees of  $T$  where every  $T_i$  is connected and  $E(T_1), \dots, E(T_l)$  is a partition of  $E$ ; and (2) a collection of vertices  $\mathcal{W} = \{w_1, \dots, w_l\}$  with  $w_i \in V(T_i)$  for every  $i \in \{1, \dots, l\}$ . The *size* of the cover is  $|\mathcal{T}| = |\mathcal{W}|$ . For every subtree  $T_i$ , we consider two orientations: the orientation  $T_i^{\text{receiver}}$ , where all edges are oriented toward the vertex  $w_i$  (i.e., every edge is oriented toward its incident vertex nearest to  $w_i$ ), and the orientation  $T_i^{\text{sender}}$ , where all edges are oriented in the opposite direction. We consider the collection of orientations of  $T$  that are produced by choosing, for every subtree  $T_i$ , one of  $T_i^{\text{receiver}}$  and  $T_i^{\text{sender}}$  and combining them to an orientation for  $T$ . For a cover of size  $l$  this collection contains  $2^l$  orientations. For an orientation instance  $(T, P)$ , we define a *pair-cover* as a cover  $(\mathcal{T}, \mathcal{W})$  for  $T$ , such that for every pair  $(s, t) \in P$  there exists an orientation in the cover that satisfies  $(s, t)$ . The *crossing number* of a pair-cover is the maximum number of subtrees whose edges are used by a pair on the path from its source to its target vertex.

**Lemma 3.1.** *Let  $(T, P)$  be an orientation instance admitting a pair-cover  $(\mathcal{T}, \mathcal{W})$  with crossing number  $c$ . Then there exists an orientation  $T'$  of  $T$  that satisfies at least  $|P|/2^c$  pairs from  $P$ .*

*Proof.* We consider a uniform probability distribution on the collection of orientations of  $T$  that can be constructed with respect to  $(\mathcal{T}, \mathcal{W})$ . Note that it is equivalent to say that we choose, for every subtree  $T_i \in \mathcal{T}$ , with probabilities  $1/2$  the orientation  $T_i^{\text{receiver}}$  or the orientation  $T_i^{\text{sender}}$ . For every pair  $(s, t)$ , we consider a random variable  $X_{(s,t)}$  that is evaluated to 1 if the particular orientation satisfies  $(s, t)$  and 0, otherwise. The satisfaction of a pair depends only on the orientations of the at most  $c$  subtrees whose edges are used by its source-to-target path. Thus every pair is satisfied with probability at least  $1/2^c$  (in particular,  $E[X_{(s,t)}] \geq 1/2^c$ ). Let us denote by  $X$  the random variable that equals the total number of pairs that are satisfied in an orientation. By linearity of expectation,  $E[X] = \sum_{(s,t) \in P} E[X_{(s,t)}] \geq |P|/2^c$ . Thus, there must exist an orientation as desired.  $\square$

Since the conditional expectation with respect to partial orientations of the tree can be computed in polynomial time by deleting all non-satisfied pairs from  $P$ , contracting the oriented edges, and computing the expectation value for the remaining instance, we can apply the method of conditional

expectations [1] to construct a deterministic polynomial-time algorithm that produces the orientations from Lemma 3.1.

In some restricted cases, the input trees admit the polynomial-time construction of pair-covers with small crossing numbers and, hence, orientations satisfying a constant fraction of all input pairs. This leads to the following bounds for star and caterpillar graphs:

**Lemma 3.2.** *The following two properties hold:*

1. *Let  $(S, P)$  be an orientation instance with a star graph  $S$ . Then there exists a polynomial-time computable orientation  $S'$  of  $S$  that satisfies at least  $|P|/4$  pairs from  $P$ .*
2. *Let  $(C, P)$  be an orientation instance with a caterpillar graph  $C$ . Then there exists a polynomial-time computable orientation  $C'$  of  $C$  that satisfies at least  $|P|/8$  pairs from  $P$ .*

*Proof.* To prove the lemma we show that orientation instances with star graphs and caterpillar graphs admit polynomial-time computable pair-covers with crossing number 2 and 3, respectively. Examples of the constructed pair-covers are given in Figure 1.

We start with orientation instances  $(S, P)$  where  $S$  is a star with center node  $v_c$ . We define the pair-cover  $(\mathcal{T}, \mathcal{W})$  to be the collection of all edges from  $E(S)$  together with  $w_i = v_c$  for all  $i \in \{1, \dots, |E(S)|\}$ . It can be computed in polynomial time for a given star graph and, since it has crossing number at most 2, applying Lemma 3.1 and the subsequent derandomization proves the claim.

Recall that a caterpillar graph is made up by a *backbone* path and edges attached to this path. For caterpillars, we construct a pair-cover  $(\mathcal{T}, \mathcal{W})$  by taking the backbone path and the edges attached to it into the collection  $\mathcal{T}$ . Note that all graphs in  $\mathcal{T}$  are paths. For each of them we use any end vertex as  $w_i$ . This cover is polynomial-time computable and has crossing number at most 3. Similar to the star graph case, the claim follows.  $\square$

Figure 1 shows examples of the pair-covers constructed in the proof of Lemma 3.2.

We note that `MAXIMUM-TREE-ORIENTATION` on star graphs reduces to `MAXIMUM-DIRECTED-CUT` by the following two-step reduction: Without loss of generality, we start with an orientation instance  $(S, P)$  with a center vertex  $v_c$  where each pair has distinct end vertices. First, consider pairs  $(s, t)$  where one of  $s$  and  $t$  equals  $v_c$ . Delete  $(s, t)$  from  $P$ , insert a new vertex  $v_{(s,t)}$  into  $V(S)$  that is connected to  $v_c$ , and insert a copy of  $(s, t)$  into  $P$  where the vertex equal to  $v_c$  is replaced by  $v_{(s,t)}$ . This does not change the size of an optimal orientation and no pair contains the center vertex. Next, consider the new instance  $(S', P')$  from the first step, and construct the graph  $(V(S'), P')$ . Similar to the proof of Theorem 2.3, every optimal orientation for  $(S', P')$  can be turned into an optimal cut for  $(V(S'), P')$ , and vice versa. It follows that the orientation problem on stars admits a 0.874-approximation by using the corresponding approximation algorithm for `MAXIMUM-DIRECTED-CUT` [15].

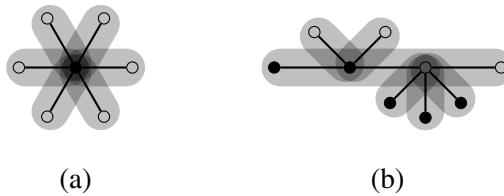


Figure 1: (a) A star tree and (b) a caterpillar tree along with their pair-covers as constructed in the proof of Lemma 3.2. The cover subtrees from  $\mathcal{T}$  are depicted with gray background bars, the vertices from  $\mathcal{W}$  are filled in black.

### 3.2 Logarithmic Factor Bounds

**Lemma 3.3.** *Let  $(T, P)$  be an orientation instance with a tree  $T$ . There exists an orientation  $T'$  of  $T$  that satisfies at least  $|P|/(4\lceil\log n\rceil)$  pairs from  $P$ .*

*Proof.* We partition the multiset  $P$  into  $\lceil\log n\rceil$  multiset  $P_1, \dots, P_{\lceil\log n\rceil}$ , such that every orientation instance  $(T, P_i)$  admits an orientation satisfying at least  $|P_i|/4$  of the pairs in  $P_i$ . Taking the instance with the largest set  $P_i$  and an orientation that satisfies at least  $1/4$  of its pairs, results in an orientation that satisfies  $1/(4\lceil\log n\rceil)$  of all pairs.

Let  $v \in V(T)$  be a vertex whose removal breaks the tree into components of size at most  $\lceil|V(T)|/2\rceil$ ; such a *centroid vertex* always exists in trees and can be found in polynomial time [6]. Let  $P_1$  be the pairs from  $P$  with source-to-target paths crossing the vertex  $v$  and consider the orientation instance  $(T, P_1)$ . We construct a pair-cover  $(\mathcal{T}_1, \mathcal{W}_1)$  for  $(T, P_1)$  where  $\mathcal{T}$  consists of all subtrees rooted at  $v$  and vertices in  $\mathcal{W}$  equal  $v$ . This pair-cover has crossing number 2. Thus, by Lemma 3.1, there exists an orientation satisfying a fraction of  $1/4$  of the pairs  $P_1$ . Next, let  $F$  be the forest that arises by deleting  $v$  from  $T$ . Note that the path of every pair from  $P \setminus P_1$  lies completely inside one of the trees from this forest. We consider centroid vertices for all trees of  $F$  and let  $P_2$  be the multiset of pairs from  $P \setminus P_1$  that cross any of these vertices. We use the same construction of pair-covers as above for every tree of the forest, and merge the covers to obtain a pair-cover with crossing number 2 for the orientation instance  $(T, P_2)$ . It witnesses that there exists an orientation  $T'$  of  $T$  satisfying  $1/4$  of the pairs from  $P_2$ . We proceed recursively by breaking  $F$  into subforests using centroid vertices. This produces  $\lceil\log n\rceil$  orientation instances  $(T, P_1), \dots, (T, P_{\lceil\log n\rceil})$ , each admitting an orientation that satisfies a fraction of  $1/4$  of its pairs using the same arguments.  $\square$

The following lemma shows that the logarithmic bound from Lemma 3.3 is tight up to a constant factor. We refer to Appendix A for its proof.

**Lemma 3.4.** *For every  $r \in \mathbb{N}$ , there exists an orientation instance  $(T_r, P_r)$  with  $|V(T_r)| = 2^{r+1} - 1$  and  $|P_r| = 2r4^{r-1}$ , such that every orientation of  $T_r$  satisfies at most  $(4^r - 1)/3$  pairs. This is a fraction of less than  $2/(3r)$  of all pairs, which is logarithmic in the size of  $T_r$ .*

## 4 Constant Factor Approximation for Pairs of Bounded Distance

In this section we consider tree orientation instances  $(T, P)$  where the distance between the source and target vertices of each pair is at most a constant  $d \in \mathbb{N}$ . We prove that such instances always admit orientations satisfying a fraction of  $1/(4d)$  of the input pairs (Lemma 4.1) and provide a polynomial-time algorithm with approximation ratio  $1/(2d)$  (Lemma 4.3).

**Lemma 4.1.** *Let  $d \in \mathbb{N}$ . Let  $(T, P)$  be an orientation instance with a tree  $T$ , such that for every pair  $(s, t) \in P$  the path from  $s$  to  $t$  in  $T$  has length at most  $d$ . Then there exists an orientation  $T'$  of  $T$  that satisfies at least  $|P|/(4d)$  pairs from  $P$ .*

*Proof.* Choose any vertex  $r \in V(T)$  and consider, for every pair  $(s, t) \in P$ , the unique vertex  $v_{(s,t)}$  from the  $s$ - $t$  path that has shortest distance to  $r$ . We partition the pairs  $P$  into  $d$  multisets  $P_0, \dots, P_{d-1}$  as follows: A pair  $(s, t) \in P$  lies in  $P_i$  if and only if  $i \equiv d_{(s,t)} \pmod{d}$  where  $d_{(s,t)}$  is the distance between  $r$  and  $v_{(s,t)}$ . An example instance and its partition are depicted in Figure 2.

We prove that for every orientation instance  $(T, P_i)$  there exists an orientation satisfying at least  $|P_i|/4$  pairs. Taking such an orientation for the largest multiset  $P_i$  results in an orientation that satisfies at least  $|P|/(4d)$  pairs. We use Lemma 3.1 and the following pair-cover with crossing number 2: For an instance  $(T, P_i)$ , we produce  $\mathcal{T}$  by splitting  $T$  at every vertex  $v$  whose distance from  $r$  modulo  $d$  is  $i$ . For every subtree  $T_j$  from  $\mathcal{T}$  its vertex  $w_j$  equals the vertex from  $V(T_j)$  with lowest distance to

$r$  in  $T$ . Since the distance between the source and target vertices of each pair is at most  $d$ , the path of every pair lies in at most two subtrees from  $\mathcal{T}$ . Moreover, every pair in  $P_i$  can be satisfied by an orientation with respect to this pair-cover: either it lies completely inside a subtree  $T_j$  and one of its end vertices is  $w_j$ , or it lies in two subtrees  $T_j$  and  $T_k$  with  $w_j = w_k$ .  $\square$

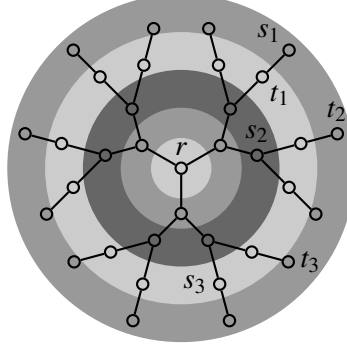


Figure 2: An example orientation instance  $(T, P)$  with  $P = \{(s_1, t_1), (s_2, t_2), (s_3, t_3)\}$  is shown. The construction from the proof of Lemma 4.1 for the depicted root node  $r$  and source-to-target distance 3 partitions  $P$  into  $P_0 = \{(s_1, t_1)\}$ ,  $P_1 = \emptyset$ , and  $P_2 = \{(s_2, t_2), (s_3, t_3)\}$ . The multiset for a pair  $(s, t)$  is chosen with respect to the vertex  $v_{(s,t)}$  and its distance from  $r$ . The distances of the vertices from the root modulo 3 are depicted using different shades of grey.

Let  $(T, P)$  be an orientation instance with tree  $T$ . The *conflict graph* of  $(T, P)$ , denoted by  $C_{(T,P)}$ , is the undirected graph with vertex set  $P$  and there is an edge between two pairs  $p_1 = (s_1, t_1) \in P$  and  $p_2 = (s_2, t_2) \in P$  if and only if  $p_1$  and  $p_2$  can not be satisfied at the same time in any orientation of  $T$  – equivalently the path from  $s_1$  to  $t_1$  and the path from  $s_2$  to  $t_2$  use some edge in different directions. We say that  $p_1$  and  $p_2$  are *conflicting* if there is an edge between them in the conflict graph and *non-conflicting*, otherwise. Sets of non-conflicting pairs can be satisfied simultaneously, which implies the following fact:

**Fact 4.2.** *Let  $(T, P)$  be an orientation instance with a tree  $T$ . For every  $k \in \mathbb{N}$  there exists an orientation of  $T$  that satisfies at least  $k$  pairs from  $P$  if and only if there exists an independent set of size at least  $k$  in  $C_{(T,P)}$ .*

**Theorem 4.3.** *Let  $d \in \mathbb{N}$ . There exists a polynomial-time algorithm that, given an orientation instance  $(T, P)$  where for every pair the length of the path between the source and the target vertex is at most  $d$ , approximates its optimum solution to within a factor of  $1/(2d)$ .*

*Proof.* Let  $\text{OPT}$  be the size of an optimal solution to MAXIMUM-TREE-ORIENTATION for the given orientation instance  $(T, P)$ . The algorithm consists of two steps: First it computes a subset  $P'$  of the pairs  $P$  with  $2 \cdot \text{OPT}/d \leq |P'|$ . Then it computes an orientation  $T'$  for  $T$  satisfying  $|P'|/4$  pairs. As a result we will satisfy at least  $\text{OPT}/(2d)$  of the pairs.

For the first step of the algorithm consider a vertex  $r \in V(T)$  and the partition of the pairs  $P$  into multisets  $P_0, \dots, P_{d-1}$  from the proof of Lemma 4.1. Let  $C'_{(T,P)}$  be the graph that arises from the conflict graph  $C_{(T,P)}$  of  $(T, P)$  by deleting all edges between pairs  $(s_i, t_i)$  and  $(s_j, t_j)$  with  $v_{(s_i, t_i)} = v_{(s_j, t_j)}$ . Two pairs  $(s_i, t_i)$  and  $(s_j, t_j)$  from the same set  $P_i$  have the same vertex  $v_{(s_i, t_i)} = v_{(s_j, t_j)}$  or their paths do not overlap. Thus, the partition  $P_0, \dots, P_{d-1}$  is a valid coloring with  $d$  colors of  $C'_{(s,t)}$ . For graphs with colorings of  $d$  colors we can use an algorithm of Hochbaum [13] that computes an independent set whose size approximates the size of a maximum independent set to within a factor of  $2/d$  in time  $O(nm \log n)$ , where  $n = |P|$  and  $m \leq |P|^2$  are the number of vertices and edges in the considered graph. Let  $P'$  be such a set for  $C'_{(s,t)}$ . Since  $C'_{(s,t)}$  is a subgraph of  $C_{(s,t)}$ , we have  $2 \cdot \text{OPT}/d \leq |P'|$ .

For the second step consider the partition  $P'_0, \dots, P'_{d-1}$  with  $P'_i = P_i \cap P'$  of  $P'$ . Similar to the proof of Lemma 4.1, we compute an orientation that satisfies  $1/4$  of the pairs for every orientation instance  $(T, P'_i)$ , but this time we are not limited to use only the orientation of a single instance. Since  $P'$  is an independent set there are no conflicts between pairs from different sets  $P'_i$  and  $P'_j$ . Thus, we are able to merge the orientations for the instances  $(T, P'_i)$  into an orientation that satisfies a fraction of  $1/4$  of all pairs from  $P'$ . Altogether this results in a set of satisfied pairs of size at least  $\text{OPT}/(2d)$ .  $\square$

## 5 Sublogarithmic Factor Approximation

In this section we devise a deterministic algorithm that achieves a sublogarithmic approximation guarantee of  $\Omega(\log \log n / \log n)$  for `MAXIMUM-TREE-ORIENTATION`. Since general instances reduce to tree instances in an approximation preserving manner, this leads to the same approximation ratio for the general problem `MAXIMUM-GRAPH-ORIENTATION`.

The algorithm first partitions the input pairs  $P$  into  $\Omega(\log \log n / \log n)$  multisets  $P_i$ . For each orientation instance  $(T, P_i)$  it computes an orientation that satisfies a constant fraction of the optimal number of satisfiable pair. Consequently, the above-mentioned approximation ratio follows by picking, out of the set of all the computed orientations, the one that satisfies a maximum number of pairs. Below we describe the partition and orientation steps in detail.

### 5.1 Pair Partitioning

The process by which we partition the pairs is a modification of the centroid decomposition used in the proof of Lemma 3.3. Specifically, we will use a decomposition that splits a tree into  $k = \lceil \log n \rceil$  subtrees of almost the same size, formalized by the concept of an almost-balanced decomposition: Let  $T = (V, E)$  be a tree. An *almost-balanced  $k$ -decomposition* of  $T$  is a partition of  $T$  into  $k$  edge-disjoint subtrees  $T_1, \dots, T_k$  such that each subtree contains between  $|E|/(3k)$  and  $3|E|/k$  edges and the number of vertices shared by at least two subtrees is at most  $k$ . Gamzu and Segev [7] showed that for every tree  $T$  and integer  $k \leq |E(T)|$  there exists an almost-balanced  $k$ -decomposition for  $T$ , and that such a decomposition can be computed in polynomial-time.

The partition of the pairs corresponds to a recursive decomposition of the input tree  $T$ . Let  $\mathcal{T}_1 = \{T_1, \dots, T_k\}$  be an almost-balanced  $k$ -decomposition of  $T$ . We say that a decomposition *separates* a pair  $(s_i, t_i)$  when its end vertices reside in different subtrees of the decomposition (see Figure 3 for an example). The first multiset of pairs,  $P_1$ , consists of all pairs separated by  $\mathcal{T}_1$ . To partition the remaining set of pairs,  $P \setminus P_1$ , we recursively apply the previously-described procedure with respect to the collection of subtrees in  $\mathcal{T}_1$ . Specifically, in the second level of the recursion, an almost-balanced  $k$ -decomposition is computed in each of the subtrees  $T_1, \dots, T_k$ , to obtain a set  $\mathcal{T}_2$ , comprising of  $k^2$  subtrees. The second multiset of pairs,  $P_2$ , consists of all pairs from  $P \setminus P_1$  that are separated by  $\mathcal{T}_2$ . The remaining multisets  $P_3, P_4, \dots$  are defined in a similar manner. The recursive process ends as soon as we arrive at a subtree with strictly less than  $k$  edges. In this case we use the decomposition that breaks a tree into its individual edges.

For  $k = \lceil \log n \rceil$ , the overall number of levels in the recursion, or equivalently, the number of pair multisets is  $O(\log_k n) = O(\log n / \log \log n)$ .

### 5.2 A Constant Factor Approximation for a Single Part

Notice that a multiset of pairs, say  $P_\ell$ , generally consists of several subsets of pairs, each created when different subtrees in  $\mathcal{T}_{\ell-1}$  are partitioned by the decomposition  $\mathcal{T}_\ell$ . More specifically, assuming that the subtrees in  $\mathcal{T}_{\ell-1}$  are  $T_1, T_2, \dots$ , the class  $P_\ell$  can be written as the disjoint union of  $P_\ell^1, P_\ell^2, \dots$ , where  $P_\ell^j$  is the set of pairs that are first separated when  $T_j$  is partitioned. Recall that the path of any



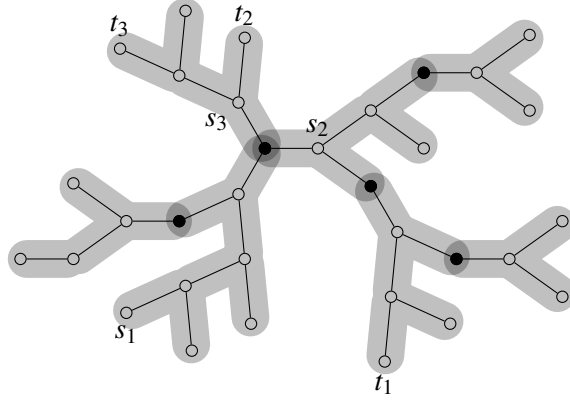


Figure 3: The example shows a tree  $T = (V, E)$  with 32 edges and an almost-balanced 7-decomposition for it. The 7 subtrees of the decomposition are highlighted with gray backgrounds, the 5 border vertices (see below) are filled in black. The number of edges in the subtrees varies between 3 and 7; this satisfies the lower bound  $|E|/3k = 32/21 \leq 3$  and upper bound  $3|E|/k = 96/7 \geq 7$  for the number of edges that are allowed in the trees of almost-balanced 7-decompositions. The pairs  $(s_1, t_1)$  and  $(s_2, t_2)$  are separated by the decomposition, while the pair  $(s_3, t_3)$  is not separated.

pair separated by some subtree decomposition must be contained in that subtree (otherwise, this pair would have been separated in previous recursion steps). This observation implies that it is sufficient to compute an orientation for a single subtree decomposition and its induced set of separated pairs. Given a polynomial-time algorithm that computes such an orientation, one can apply it to each of the subtree decompositions in the same recursion level. The resulting orientations of edge-disjoint subtrees define an orientation for the whole input tree, satisfying at least as many pairs as the overall number of pairs satisfied in all individual subtrees.

In what follows, we focus our attention on a single decomposition, and devise a randomized algorithm that computes an orientation which satisfies, in expectation, a constant fraction of the optimal number of satisfiable pairs for this decomposition. Formally, an instance of the problem in question consists of a tree  $T = (V, E)$ , and a partition  $\mathcal{T} = \{T_1, \dots, T_k\}$  of this tree into  $k$  edge-disjoint subtrees, where  $k \leq \lceil \log n \rceil$ , and the number of vertices shared by at least two subtrees is less than  $k$ . In addition, we are given a multiset  $P$  of pairs that are separated by the the decomposition  $\mathcal{T}$ .

We need the following notation (exemplified in Figure 4). Let  $\text{opt}$  denote the number of satisfied pairs in some fixed optimal orientation of  $T$ . Let  $V_B \subseteq V$  be the set of *border vertices* of  $\mathcal{T}$ , that is, the set of vertices that are shared by at least two subtrees in  $\mathcal{T}$ . Moreover, let  $S \subseteq T$  be the *skeleton* of  $\mathcal{T}$ , namely, the minimal subtree spanned by all border vertices. Note that this subtree consists of the union of paths connecting any two vertices in  $V_B$ . Finally, let  $V_J \subseteq V$  the set of *junction vertices*, defined as non-border skeleton vertices with degree at least 3 (counting only skeleton edges).

We are now ready to present the orientation algorithm. Our algorithm consists of two phases: *segment guessing*, where the optimal direction state of disjoint subpaths of the skeleton is attained, followed by *randomized assignment*, in which individual edges are assigned a direction.

**Segment guessing.** Let us name the vertex set  $V_B \cup V_J$  the *core* of the skeleton  $S$ . One can verify that  $|V_B \cup V_J| < 2k$  as  $|V_J| < |V_B| < k$ . We now partition the skeleton into a collection  $\Sigma(S)$  of edge-disjoint paths, which are referred to as *segments*. Each such segment is a subpath of  $S$  whose endpoints are core vertices, but its interior traverses only non-core vertices. Clearly,  $|\Sigma(S)| = |V_B \cup V_J| - 1 < 2k$ . We now argue that one could obtain in polynomial time the direction state that the optimal orientation induces on each segment  $\sigma \in \Sigma(S)$ , *simultaneously* for all segments. To this end, notice that any

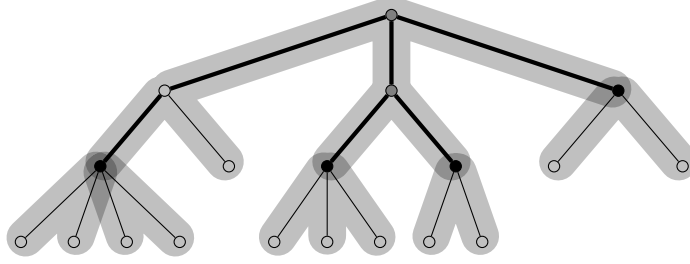


Figure 4: An almost-balanced 5-decomposition. Black vertices are border vertices, gray vertices are junction vertices, and the gray areas make up the skeleton of the decomposition.

skeleton segment  $\sigma = \langle v_1, v_2, \dots, v_\ell \rangle$  may be in one of three possible direction states:

- *Right direction*: all edges are consistently directed from  $v_1$  toward  $v_\ell$ , which means  $v_1 \rightarrow v_2, v_2 \rightarrow v_3, \dots, v_{\ell-1} \rightarrow v_\ell$ .
- *Left direction*: all edges are consistently directed from  $v_\ell$  toward  $v_1$ , namely,  $v_1 \leftarrow v_2, v_2 \leftarrow v_3, \dots, v_{\ell-1} \leftarrow v_\ell$ .
- *Mixed direction*: the direction of segment edges is non-consistent.

These definitions imply that the total number of segment direction states to be examined is of polynomial size since  $3^{|\Sigma(S)|} < 3^{2k} \leq 3^{2\lceil \log n \rceil} = O(n^{2 \cdot \log 3})$ . As a consequence, we may assume without loss of generality that the set of direction states induced by the optimal orientation on all the segments of  $\Sigma(S)$  is known in advance. This assumption can be enforced by enumerating over all  $O(n^{2 \cdot \log 3})$  possible segment direction states.

**Randomized assignment.** The goal of this phase is to orient the graph while making sure that the edge directions respect the outcome of the segment guessing phase. For this purpose, we begin by considering skeleton segments that have a consistent direction, namely, segments in either right or left direction states, and assign all the edges in these segments their implied direction. The assignment procedure proceeds with two randomized assignment steps: (1) Each segment in a mixed direction state is assigned, independently and uniformly at random, a right or left direction. All segment edges are oriented according to the chosen direction. (2) Each of the decomposition subtrees  $T_1, \dots, T_k$  is assigned, independently and uniformly at random, the role of a *sender* or a *receiver*. All the non-skeleton edges of each sender subtree are oriented toward the skeleton (in its simplest form, when the subtree contains a single border vertex, all edges are oriented toward that vertex). In contrast, all the non-skeleton edges of each receiver subtree are oriented away from the skeleton. We refer the reader to an example in Figure 5(a).

We turn to prove that the expected number of satisfied pairs is within a constant factor of optimal, as formally stated in the following claim.

**Claim 5.1.** *The resulting orientation satisfies at least  $\text{opt}/16$  pairs in expectation.*

*Proof.* Recall that we have previously assumed the endpoints of each pair to reside in different subtrees of the decomposition  $\mathcal{T}$ . In particular, this implies that each pair path must traverse at least one border (core) vertex. For this reason, as shown in Figure 5(b), we can divide each pair path, with endpoints  $s_i$  and  $t_i$ , into five (some possibly empty) parts: (1) A subpath between  $s_i$  and its closest skeleton vertex  $v_{s_i}$ . (2) A subpath, along a partial skeleton segment, between  $v_{s_i}$  and its closest core vertex  $r_{s_i}$ . (3) A subpath between  $t_i$  and its closest skeleton vertex  $v_{t_i}$ . (4) A subpath, along a partial skeleton

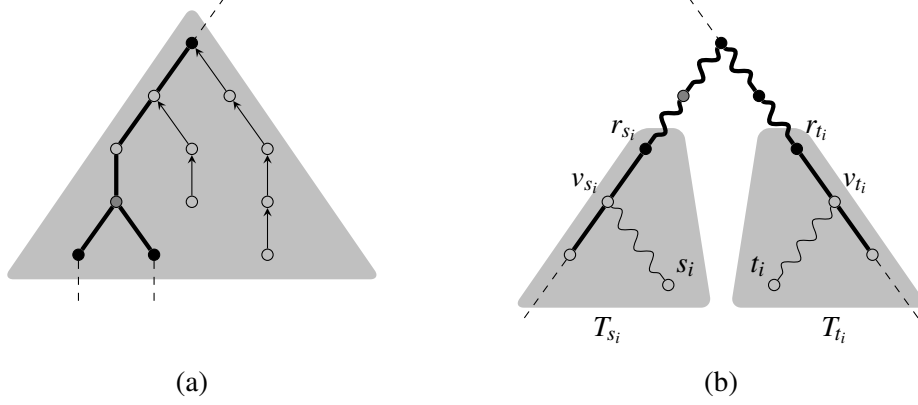


Figure 5: (a) An orientation of a sender subtree, where the thick edges are part of the skeleton. (b) A partition of a pair path into five parts.

segment, between  $v_{t_i}$  and its closest core vertex  $r_{t_i}$ . (5) A subpath between  $r_{s_i}$  and  $r_{t_i}$ , along a sequence of complete skeleton segments.

With these definitions in mind, let us focus on some pair  $(s_i, t_i)$  that is satisfied in the optimal orientation. We now argue that, with probability at least  $1/16$ , this pair is satisfied in the random orientation constructed by the algorithm. Consequently, by linearity of expectation, the overall expected number of satisfied pairs is  $\text{OPT}/16$ . The key observation we make to establish this argument is that all the segments along the subpath between  $r_{s_i}$  and  $r_{t_i}$  must have a consistent direction in the optimal orientation; otherwise, this pair would not have been satisfied. Accordingly, we may assume that our algorithm assigned the same direction to all the edges in these segments. Now, notice that the pair under consideration is satisfied if the following four probabilistic events occur: (1) the edges in the subpath between  $s_i$  and  $v_{s_i}$  are oriented toward  $v_{s_i}$ ; (2) the edges in the subpath between  $v_{s_i}$  and  $r_{s_i}$  are oriented toward  $r_{s_i}$ ; (3) the edges in the subpath between  $v_{t_i}$  and  $r_{t_i}$  are oriented toward  $v_{t_i}$ ; and (4) the edges in the subpath between  $t_i$  and  $v_{t_i}$  are oriented toward  $t_i$ . One can validate that these four events are independent, and that each one of them occurs with probability of at least  $1/2$ . For example, the edges in the subpath between  $s_i$  and  $v_{s_i}$  are oriented toward  $v_{s_i}$  if the underlying subtree  $T_{s_i}$  is selected as a sender. As a result, the probability that pair  $i$  is satisfied in the random orientation is at least  $1/16$ .  $\square$

**Derandomization.** The extent to which we utilize randomization is limited; its purpose is to make the presentation of our algorithm simpler. Each segment in a mixed direction state is randomly assigned one of two possible directions, resulting in at most  $2^{|\Sigma(S)|} < 2^{2k} \leq 2^{2\lceil \log n \rceil} = O(n^2)$  possibilities. Each decomposition subtree is randomly assigned one of two possible roles, resulting in at most  $2^k \leq 2^{\lceil \log n \rceil} = O(n)$  possibilities. To obtain a deterministic polynomial-time algorithm, we can construct the whole space of possible assignments.

In summary, we obtain the following theorem:

**Theorem 5.2.** *There exists a polynomial-time algorithm that approximates MAXIMUM-TREE-ORIENTATION to within a factor of  $\Omega(\log \log n / \log n)$  on  $n$ -vertex trees.*

Theorem 5.2 and Lemma 2.2 imply the same approximation bound for the problem on general graphs:

**Corollary 5.3.** *There exists a polynomial-time algorithm that approximates MAXIMUM-GRAPH-ORIENTATION to within a factor of  $\Omega(\log \log n / \log n)$  on  $n$ -vertex graphs.*

## 6 Conclusions

In this paper we studied the complexity and approximability of the maximum graph orientation problem. We showed that this problem is NP-hard to approximate to within a factor of  $12/13$ . On the positive side, we provided an  $\Omega(\log \log n / \log n)$ -approximation algorithm for the problem. In addition, we provided insights into the combinatorial structure of the problem, showing that every orientation instance admits an orientation that satisfies a fraction of  $1/(4\lceil \log n \rceil)$  of its pairs, and that this bound is tight up to a constant factor. We also designed constant factor approximation algorithms for restricted variants of the problem where the instance can be decomposed by restricted covers, including star graphs, caterpillars and graphs in which the distances between the source and target vertices of every pair are bounded.

There are several directions for future research: One direction is to close the gap between our approximation and hardness-of-approximation results. Another direction is to develop algorithms that work on instances in which some of the graph's edges are pre-directed. This problem is motivated by biological scenarios in which the directions of some of the edges are known, such as for protein-DNA interaction and kinase-substrate interactions. We have recently developed a polynomial integer linear programming formulation for the problem and showed that the resulting orientations are much more in line with current biological knowledge compared to orientations that ignore the pre-set directions [20]. We also proved a sub-linear approximation ratio for this problem [4]. It is open whether this ratio can be improved. A third research direction is to understand the structure of real world instances and exploit it for algorithms solving the NP-complete orientation problem fast in practice. A recent work in this direction by Dorn et al. [3] determines the values of many structural parameters for different PPI networks and studies the complexity of the orientation problem when parameterized by them.

## Acknowledgments

M.E. was supported by a research grant from the Dr. Alexander und Rita Besser-Stiftung. R.S. was supported by a research grant from the Israel Science Foundation (grant no. 385/06). U.Z. was supported by BSF grant no. 2006261. We thank Rani Hod for his help with the proof of Lemma 3.4.

## References

- [1] N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley-Interscience, New York, 2nd edition, 2000.
- [2] J. Bang-Jensen and G. Gutin. *Digraphs: Theory, Algorithms and Applications*. Springer, 2nd edition, 2008.
- [3] B. Dorn, F. Hüffner, D. Krüger, R. Niedermeier, and J. Uhlmann. Exploiting bounded signal flow for graph orientation based on cause-effect pairs. In *Proceedings of the 1st International ICST Conference on Theory and Practice of Algorithms in (Computer) Systems (TAPAS 2011)*, pp. 104–115. Springer, Lecture Notes in Computer Science 6595, 2011.  
Online at [http://dx.doi.org/10.1007/978-3-642-19754-3\\_12](http://dx.doi.org/10.1007/978-3-642-19754-3_12)
- [4] M. Elberfeld, D. Segev, C. R. Davidson, D. Silverbush, and R. Sharan. Approximation algorithms for orienting mixed graphs. To appear in *Proceedings of the 22nd Annual Symposium on Combinatorial Pattern Matching (CPM 2011)*. Springer, Lecture Notes in Computer Science 6661, 2011.

- [5] S. Fields. High-throughput two-hybrid analysis. The promise and the peril. *The FEBS Journal* 272(21):5391–5399, 2005.  
Online at <http://www.ncbi.nlm.nih.gov/pubmed/16262681>
- [6] G. N. Frederickson and D. B. Johnson. Generating and searching sets induced by networks. In *Proceedings 7th International Colloquium on Automata, Languages and Programming (ICALP 1980)*, pp. 221-233. Springer, Lecture Notes in Computer Science 85, 1980.  
Online at [http://dx.doi.org/10.1007/3-540-10003-2\\_73](http://dx.doi.org/10.1007/3-540-10003-2_73)
- [7] I. Gamzu and D. Segev. A sublogarithmic approximation for highway and tollbooth pricing. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP 2010)*, pp. 582–593. Springer, Lecture Notes in Computer Science 6198, 2010.  
Online at [http://dx.doi.org/10.1007/978-3-642-14165-2\\_49](http://dx.doi.org/10.1007/978-3-642-14165-2_49)
- [8] I. Gamzu, D. Segev, and R. Sharan. Improved orientations of physical networks. In *Proceedings of the 10th International Workshop on Algorithms in Bioinformatics (WABI 2010)*, pp. 215-225. Springer, Lecture Notes in Computer Science 6293, 2010.  
Online at [http://dx.doi.org/10.1007/978-3-642-15294-8\\_18](http://dx.doi.org/10.1007/978-3-642-15294-8_18)
- [9] A. Gavin, M. Bösche, R. Krause, P. Grandi, M. Marzioch, A. Bauer, J. Schultz, J. M. Rick, A. Michon, C. Cruciat, M. Remor, C. Höfert, M. Schelder, M. Brajenovic, H. Ruffner, A. Merino, K. Klein, M. Hudak, D. Dickson, T. Rudi, V. Gnau, A. Bauch, S. Bastuck, B. Huhse, C. Leutwein, M. Heurtier, R. R. Copley, A. Edelmann, E. Querfurth, V. Rybin, G. Drewes, M. Raida, T. Bouwmeester, P. Bork, B. Seraphin, B. Kuster, G. Neubauer, and G. Superti-Furga. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature* 415(6868):141–147, Jan. 2002.  
Online at <http://dx.doi.org/10.1038/415141a>
- [10] A. Gitter, J. Klein-Seetharaman, A. Gupta, and Z. Bar-Joseph. Discovering pathways by orienting edges in protein interaction networks. *Nucleic Acids Research* 39(4):e22, 2011.  
Online at <http://dx.doi.org/10.1093/nar/gkq1207>
- [11] S. L. Hakimi, E. F. Schmeichel, and N. E. Young. Orienting graphs to optimize reachability. *Information Processing Letters* 63(5):229–235, 1997.  
Online at [http://dx.doi.org/10.1016/S0020-0190\(97\)00129-4](http://dx.doi.org/10.1016/S0020-0190(97)00129-4)
- [12] J. Håstad. Some optimal inapproximability results. *Journal of the ACM* 48(4):798–859, 2001.  
Online at <http://dx.doi.org/10.1145/502090.502098>
- [13] D. S. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Applied Mathematics* 6(3):243–254, 1983.  
Online at [doi:http://dx.doi.org/10.1016/0166-218X\(83\)90080-X](http://dx.doi.org/10.1016/0166-218X(83)90080-X)
- [14] V. Kann, J. Lagergren, and A. Panconesi. Approximability of maximum splitting of  $k$ -sets and some other APX-complete problems. *Information Processing Letters* 58(3):105–110, 1996.  
Online at [http://dx.doi.org/10.1016/0020-0190\(96\)00046-4](http://dx.doi.org/10.1016/0020-0190(96)00046-4)
- [15] M. Lewin, D. Livnat, and U. Zwick. Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems. In *Proceedings of the 9th International Conference on Integer Programming and Combinatorial Optimization (IPCO 2002)*, pp. 67–82. Springer, Lecture Notes in Computer Science 2337, 2006.  
Online at [http://dx.doi.org/10.1007/3-540-47867-1\\_6](http://dx.doi.org/10.1007/3-540-47867-1_6)
- [16] A. Medvedovsky. An algorithm for orienting graphs based on cause-effect pairs and its applications to orienting protein networks. Master’s thesis, Tel-Aviv University, Israel, 2009.  
Online at <http://www.cs.tau.ac.il/~roded/Sasha-thesis.pdf>

- [17] A. Medvedovsky, V. Bafna, U. Zwick, and R. Sharan. An algorithm for orienting graphs based on cause-effect pairs and its applications to orienting protein networks. In *Proceedings of the 8th International Workshop on Algorithms in Bioinformatics (WABI 2008)*, pp. 222–232. Springer, Lecture Notes in Computer Science 5251, 2008.  
Online at [http://dx.doi.org/10.1007/978-3-540-87361-7\\_19](http://dx.doi.org/10.1007/978-3-540-87361-7_19)
- [18] O. Ourfali, T. Shlomi, T. Ideker, E. Ruppın, and R. Sharan. SPINE: a framework for signaling-regulatory pathway inference from cause-effect experiments. *Bioinformatics* 23(13):i359–i366, 2007.  
Online at <http://dx.doi.org/10.1093/bioinformatics/btm170>
- [19] H. E. Robbins. A theorem on graphs, with an application to a problem of traffic control. *The American Mathematical Monthly* 46(5):281–283, 1939.  
Online at <http://www.jstor.org/stable/2303897>
- [20] D. Silverbush, M. Elberfeld, and R. Sharan. Optimally orienting physical networks. In *Proceedings of the 15th Annual International Conference on Research in Computational Molecular Biology (RECOMB 2011)*, pp. 424–436. Springer, Lecture Notes in Computer Science 6577, 2011.  
Online at [http://dx.doi.org/10.1007/978-3-642-20036-6\\_39](http://dx.doi.org/10.1007/978-3-642-20036-6_39)
- [21] R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing* 1(2):146–160, 1972.  
Online at <http://dx.doi.org/10.1137/0201010>
- [22] R. E. Tarjan. A note on finding the bridges of a graph. *Information Processing Letters* 2(6):160–161, 1974.  
Online at [http://dx.doi.org/10.1016/0020-0190\(74\)90003-9](http://dx.doi.org/10.1016/0020-0190(74)90003-9)
- [23] C. Yeang, T. Ideker, and T. Jaakkola. Physical network models. *Journal of Computational Biology* 11(2-3):243–262, 2004.  
Online at <http://dx.doi.org/10.1089/1066527041410382>

## A Appendix: Proof of Lemma 3.4

*Proof.* Let  $r \in \mathbb{N}$ . The orientation instance  $(T_r, P_r)$  is made up by an undirected rooted complete binary tree with depth  $r$  and the following multiset of pairs: For every ordered pair  $(v, w)$  of distinct leaves in  $T_r$  we insert  $2^{(2r-d(v,w))/2}$  copies of the pair  $(v, w)$  into  $P_r$ , where  $d(v, w)$  is the distance between  $v$  and  $w$ . Figure 6 shows an example of this orientation instance.

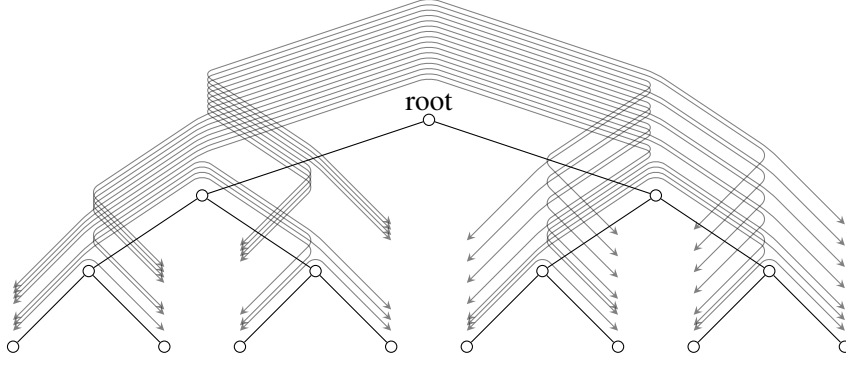
Let  $\text{OPT}(r)$  denote the maximum number of satisfied pairs from  $P_r$  in any orientation of  $T_r$ . Let  $\text{OPT}(r, k)$  denote the maximum number of satisfied pairs from  $P_r$  in orientations where the root can be reached from exactly  $k$  leaves. Due to the definition of the pairs, this is equivalent to saying that there exist exactly  $k$  leaves that are reachable from the root – if we flip all edge orientations, then a pair  $(v, w)$  is satisfied if, and only if, the pair  $(w, v)$  was satisfied before. Thus, flipping all edge orientations does not change the number of satisfied pairs. By definition we have  $\text{OPT}(r) = \max_{0 \leq k \leq 2^r} \text{OPT}(r, k)$ .

The main technical arguments of this proof are encapsulated by the following claim. For  $k \geq 1$ , define  $g(k) = 2^{\lceil \log k \rceil}$  – the largest power of two that is at most  $k$ . We claim that for every  $r \geq 1$  and  $k \geq 0$ , we have

$$\text{OPT}(r, k) = \begin{cases} \frac{4^r + 2g(k)^2}{3} - g(k)k & \text{if } k \geq 1 \\ \frac{4^r - 4}{3} & \text{if } k = 0 \end{cases}.$$

Once the claim is proven, we know that  $\text{OPT}(r, k)$  is maximized for  $k = 1$  since  $\text{OPT}(r, k)$  is monotone decreasing in  $k$ , starting from  $k \geq 1$ . Thus  $\text{OPT}(r) = \text{OPT}(r, 1) = (4^r - 1)/3$ . An example of an optimal orientation where exactly one leaf has a path to the root is shown in Figure 6(b).

(a) Orientation instance  $(T_3, P_3)$ :



(b) An optimal orientation  $T'_3$  of  $T_3$  and the satisfied pairs from  $P_3$ :

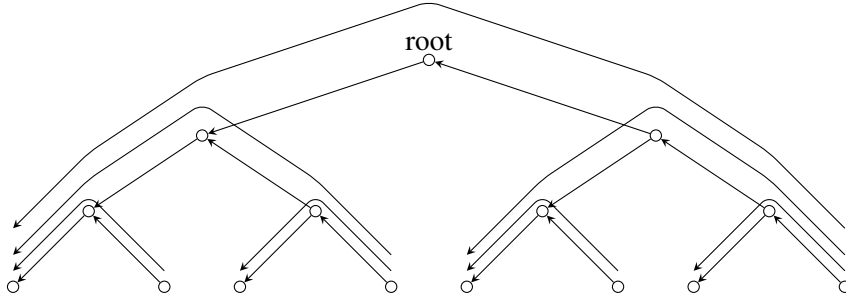


Figure 6: (a) The orientation instance constructed in the proof of Lemma 3.4 is shown for  $r = 3$ . The tree  $T_3$  is drawn by using solid edges; it has  $2^4 - 1$  vertices. The pairs from  $P_3$  are grouped to pairs having the same end vertices (though, not necessarily the same source and target vertices); every group is depicted by a grey bidirectional path that connects the two end vertices. The number of ordered combinations of two leaves whose pairs cross the root is  $2 \cdot 4^2$ ;  $4^2$  combinations with first vertex on the left and second vertex on the right side. In  $P_3$  there is exactly one copy for each of these pairs and, thus, the number of pairs in  $P_3$  that cross the root is  $2 \cdot 4^2$ . All leaf combinations whose pairs do not cross the root, but a vertex one level beneath, appear twice in  $P_3$ . In total, the corresponding number of pairs is  $2 \cdot 4^2$ . The same holds if we go one level further beneath where all leaf combinations appear 4 times as pairs in  $P_3$ . In total, the number of pairs in  $P_3$  is  $3 \cdot 2 \cdot 4^2$ . (b) An optimal orientation  $T'_3$  of  $T_r$  that satisfies  $1 + 4 + 16 = (4^3 - 1)/3$  pairs. The pattern of edge orientations in this example (for every vertex, one child edge is oriented upwards, and the other child edge is oriented downwards) can be generalized to construct optimal orientations for every orientation instance  $(T_r, P_r)$ .

We prove the claim by induction over  $r$ . For  $r = 1$ , we have  $T_1 = \overset{\text{root}}{\circ} \begin{array}{l} \swarrow \\ \searrow \end{array}$  and by checking all 4 orientations for this tree we can see that the claim holds for all  $k$ . We assume that the claim holds for some  $r \geq 1$  and all  $k$ , and prove that it also holds for  $r + 1$  and all  $k$ .

Before we proceed with the proof, we will make a short break for a technical observation that is enabled by the induction hypothesis and used later on. We define  $\text{DIFFOPT}(r, k) = \text{OPT}(r, k) - \text{OPT}(r, k + 1)$ , which is the change of the number of satisfied pairs in optimal orientations if we require that one more leaf has a path to the root. So far  $g(k)$  is only defined for  $k \geq 1$ . We extend its definition to  $g(0) = -1$  and show  $\text{DIFFOPT}(r, k) = g(k)$  for all  $k \geq 0$ . For  $k = 0$ , we have  $\text{DIFFOPT}(r, 0) = \text{OPT}(r, 0) -$

$\text{OPT}(r, 1) = (4^r - 4)/3 - (4^r - 1)/3 = -1$ . For  $k \geq 1$ , we consider the equalities


$$\begin{aligned} \text{DIFFOPT}(r, k) &= \text{OPT}(r, k) - \text{OPT}(r, k+1) \\ &= \frac{4^r + 2g(k)^2}{3} - g(k)k - \left( \frac{4^r + 2g(k+1)^2}{3} - g(k+1)(k+1) \right) \\ &= \frac{2}{3}(g(k)^2 - g(k+1)^2) - g(k)k + g(k+1)(k+1) . \end{aligned}$$

If  $g(k) = g(k+1)$ , which happens if  $k+1$  is not a power of 2, we can derive  $\text{DIFFOPT}(r, k) = g(k)$ . If  $g(k) \neq g(k+1)$ , we know  $g(k+1) = k+1 = 2g(k)$  and can extend the equations from above to

$$\begin{aligned} \text{DIFFOPT}(r, k) &= \frac{2}{3}(g(k)^2 - 4g(k)^2) - g(k)k + 4g(k)^2 \\ &= g(k)(2g(k) - k) = g(k)(k+1 - k) = g(k) . \end{aligned}$$

We proceed to prove the claim for  $(T_{r+1}, P_{r+1})$  and first consider the case  $k = 0$ . The orientations that are possible in this case do not satisfy any of the pairs that cross the root (because there is no path from a leaf to the root.) As a result, we can restrict to optimally orient the two orientation instances that correspond to the two subtrees beneath the root; they equal  $T_r$ , but with each pair from (the multiset)  $P_r$  occurring twice. Thus, by using the induction hypothesis and the fact that  $\text{OPT}(r, k)$  is maximized for  $k = 1$ , we can derive  $\text{OPT}(r+1, 0) = 2 \cdot 2 \cdot \max_{0 \leq k \leq 2^r} \text{OPT}(r, k) = 4 \cdot (4^r - 1)/3 = (4^{r+1} - 4)/3$ .

Next, we consider  $k \geq 1$  and optimal orientations among the orientations where exactly  $k$  leaves have a path to the root. We distinguish two cases: In the first case, both edges incident to the root are directed toward the root, and in the second case one edge incident to the root is oriented away from the root and the other toward the root.

We start with the first case and consider orientations of the form . Since no pair that crosses the root is satisfied and a total of  $k$  leaves have a path to the root, we obtain the equation  $\text{OPT}(r+1, k) = \max_{0 \leq l \leq k} 2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k-l)$  where  $l$  and  $k-l$  are the number of leaves with paths to the root from the left and right subtrees, respectively. Which  $l$  maximizes the expression  $2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k-l)$ ? We consider the difference between the expression for two consecutive values of  $l$  and derive the following equalities by using the induction hypothesis and the technical observation from above:

$$\begin{aligned} &2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k-l) - (2 \cdot \text{OPT}(r, l+1) + 2 \cdot \text{OPT}(r, k-l-1)) \\ &= 2 \cdot \text{DIFFOPT}(r, l) - 2 \cdot \text{DIFFOPT}(r, k-l-1) = 2(g(l) - g(k-l-1)) . \end{aligned}$$


The expression  $2(g(l) - g(k-l-1))$  is at most 0 for  $l = 0$  and at least 0 for  $l = k-1$ . Moreover, it is monotonically increasing for constant  $k$  and increasing  $l$ . Thus the value of  $l$  where  $2(g(l) - g(k-l-1))$  is the first time at least 0 gives us an  $l$  that maximizes  $2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k-l)$ . If  $k$  is a power of 2, this happens for  $l = k/2 = g(l)$  and implies

$$\text{OPT}(r+1, k) = 2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k-l) = 4 \cdot \text{OPT}(r, \frac{k}{2}) = \frac{4^{r+1} + 2g(k)^2}{3} - g(k)k .$$

If  $k$  is not a power of 2, which means  $k = 2^t + d$  for some  $t$  and  $d$  with  $0 < d < 2^t$ , it happens at  $l = 2^{t-1}$ , whenever  $d < 2^{t-1}$ , and  $d$ , whenever  $d \geq 2^{t-1}$ . Both times we have  $g(l) = g(k-l-1) = g(k)/2$ . This implies

$$\begin{aligned} \text{OPT}(r+1, k) &= 2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k-l) \\ &= 2 \cdot \text{OPT}(r, l) - 2 \cdot \text{DIFFOPT}(r, k-l-1) + 2 \cdot \text{OPT}(r, k-l-1) \\ &= 2 \left( \frac{4^r + 2g(l)^2}{3} - g(l)l \right) - 2g(k-l-1) + 2 \left( \frac{4^r + 2g(k-l-1)^2}{3} - g(k-l-1)(k-l-1) \right) \\ &= \frac{4^{r+1}}{3} + \frac{8}{3} \frac{g(k)^2}{4} - g(k)(l+k-l-1+1) = \frac{4^{r+1} + 2g(k)^2}{3} - g(k)k . \end{aligned}$$



Now we consider the second case with orientations like . In this case, we can write the optimum as  $\text{OPT}(r+1, k) = \max_{0 \leq l \leq 2^r} 2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k) + lk$  where  $l$  denotes that number of leaves that are reachable from the root in the left subtree. Similar to the above case, we write down the difference between two subsequent expressions, this time deriving the equation

$$\begin{aligned} & 2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k) + lk - (2 \cdot \text{OPT}(r, l+1) + 2 \cdot \text{OPT}(r, k) + (l+1)k) \\ &= 2 \cdot \text{DIFFOPT}(r, l) - k = 2g(l) - k . \end{aligned}$$

The expression  $2g(l) - k$  increases with growing values of  $l$  and has its first positive value at  $l = g(k)$ . This can be used to derive

$$\begin{aligned} \text{OPT}(r+1, k) &= 2 \cdot \text{OPT}(r, l) + 2 \cdot \text{OPT}(r, k) + lk \\ &= 2 \left( \frac{2^r + 2g(k)^2}{3} - g(k)k \right) + 2 \left( \frac{2^r + 2g(k)^2}{3} - g(k)^2 \right) \\ &= \frac{4^{r+1} + 2g(k)^2}{3} - g(k)k . \end{aligned}$$

Taking all steps of the induction proof together proves the claim.

Note that the proved statement also holds if the orientation instance pairs are kept in sets instead of multisets. In this case we cannot have multiples copies of the same pair, but the proof can be fixed by replacing every leaf with a cycle whose size equals the number of pairs that use the leaf as an end vertex. Then the pairs are redirected to have unique end vertices. Since for optimal orientations we can restrict to the case that cycles are oriented in a consistent direction, the proof generalizes to this case.  $\square$