Approximation Algorithms for Orienting Mixed Graphs[☆]

Michael Elberfeld[a,1], Danny Segev[b,1], Colin R. Davidson[c], Dana Silverbush[d], Roded Sharan[d]

[a]*Institute of Theoretical Computer Science, University of Lübeck, 23538 Lübeck, Germany*
[b]*Department of Statistics, University of Haifa, Haifa 31905, Israel*
[c]*Faculty of Mathematics, University of Waterloo, Waterloo, Canada, N2L 3G1*
[d]*Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel*

Abstract

Graph orientation is a fundamental problem in graph theory that has recently arisen in the study of signaling-regulatory pathways in protein networks. Given a graph and a list of source-target vertex pairs, one wishes to assign directions to the edges so as to maximize the number of pairs that admit a directed source-to-target path. When the input graph is undirected, a sub-logarithmic approximation is known for this problem. However, the approximability of the biologically-relevant variant, in which the input graph has both directed and undirected edges, was left open. Here we give the first approximation algorithms to this problem. Our algorithms provide a sub-linear guarantee in the general case, and logarithmic guarantees for structured instances.

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

1. Introduction

Protein-protein interactions (PPIs) form the skeleton of signal transduction in the cell. While many of these interactions carry directed signaling information, current PPI measurement technologies, such as yeast two hybrid [16] and co-immunoprecipitation [20], cannot reveal the direction in which the signal flows. The problem of inferring this hidden directionality information is fundamental to our understanding of how these biological networks function. Previous work on this problem has relied on information from perturbation experiments [30], in which a gene is perturbed (cause) and as a result other genes change their expression levels (effects), to guide the orientation inference. Specifically, it is assumed that for an effect to take place, there must be a directed path in the network from the causal gene to the affected gene. The arising combinatorial problem, formally defined in Section 2, is to orient the edges of the network such that a maximum number of cause-effect pairs admit a directed path from the causal to the affected gene. When studying a PPI network in isolation, the input network is undirected. However, the more biologically relevant variant also considers protein-DNA interactions as these are necessary to explain the expression changes. Moreover, the directionality of some PPIs, like kinase-substrate interactions, is known in advance. Thus, in general, the input network is a mixed graph containing both directed and undirected edges.

Previous work. Even though the specific optimization problem we study in this paper draws its recent interest from applications in network biology, it is rooted at practical applications studied for decades now. In 1939, Robbins [27], who was motivated by applications in street network design, showed that an undirected graph has a strongly connected orientation if and only if it has no bridge edges. The corresponding decision problem can be solved in linear time [29]. The characterization of Robbins was extended to mixed graphs by Boesch et al. [5], and linear-time algorithms for

[☆]This is the accepted author manuscript of the article `http://dx.doi.org/10.1016/j.tcs.2012.03.044`. An extended abstract of this paper appeared as [15].

Email addresses: `elberfeld@tcs.uni-luebeck.de` (Michael Elberfeld), `segevd@stat.haifa.ac.il` (Danny Segev), `colinrdavidson@gmail.com` (Colin R. Davidson), `danasilv@post.tau.ac.il` (Dana Silverbush), `roded@post.tau.ac.il` (Roded Sharan)

[1]These authors contributed equally to this work.

deciding whether a mixed graph admits a strongly connected orientation were presented by Chung et al. [11]. Hakimi et al. [21] presented a polynomial algorithm for the problem of orienting an undirected graph so as to maximize the number of source-target pairs (out of all possible ordered vertex pairs) that admit a directed source-to-target path. A recent work by Dorn et al. [13] studies the parameterized complexity of orienting graphs. We refer to the textbook of Bang-Jensen and Gutin [3] for a comprehensive discussion of various graph orientation problems.

More recently, graph orientations have found new real-life motivation, due to applications in network biology. Medvedovsky et al. [25], who introduced the problem we study here, focused on restricted instances where the input graph is undirected, providing a logarithmic approximation algorithm for the problem. The approximation guarantee was later improved to $\Omega(\log\log n/\log n)$ by Gamzu et al. [19], where n denotes the number of vertices in the input graph. See [14] for a comprehensive discussion of the approximability of the orientation problem for undirected graphs. Gamzu et al. also showed that the orientation problem on mixed graphs can be approximated to within a poly-logarithmic ratio of $\Omega(1/\log^{\ell} n)$ where ℓ is the maximum number of alternations between undirected and directed edges on a source-to-target path; in general, this parameter can be arbitrarily large. Silverbush et al. [28] developed an integer-programming-based algorithm to optimally orient mixed networks, but the approximability of the problem (for non-constant ℓ) was left open.

Results and techniques. In this work, we study the approximability of the orientation problem on mixed graphs. Our main contributions can be summarized as follows:

- We observe that the problem in question is NP-hard to approximate to within a factor of $7/8$, and explain how to reduce it (in an approximation-preserving way) to the somewhat simpler setting of mixed *acyclic* graphs.
- By making use of various structural properties, we provide an $\Omega(1/\log n)$ approximation algorithm for instances that have a constant feedback vertex number, and an $\Omega(1/\log^2 n)$ approximation for instances with constant tree width.
- For general instances, we show how to compute an orientation that satisfies an $\Omega(1/(\max\{n,|P|\}^{1/\sqrt{2}}\log n))$ fraction of all input pairs, where P denotes the collection of source-target pairs. In addition, we propose a variant of this algorithm that attains an $\Omega(1/\sqrt{\Delta|P|\log n})$ approximation, where Δ stands for the maximum length of a shortest source-target path.

Our algorithms are inspired by the so-called *junction-vertex* technique (see, for instance, [6, 8, 7]). Very informally, in this particular setting, we transform *junction instances*, where paths connecting source-target pairs go through a small part of the input graph, to the easier-to-handle orientation problem for undirected graphs. For graphs with a small feedback vertex number, the original instance is broken down into junction instances, for each part that is covered by the feedback set, and for the remaining graph, without the feedback set. In the case of having small tree width, the graph is split (using a tree decomposition) into separate parts for which the junction instance case can be applied. Finally, our approximation algorithm for the general case makes use of the above-mentioned technique within a greedy framework, that initially satisfies source-target pairs through short paths, as long as these paths do not intersect too many other paths; this idea is similar in spirit to how some *edge-disjoint-paths* algorithms work [24, 9]. When such paths can no longer be found, we argue that the remaining graph necessarily contains a junction instance that involves many input pairs. These methods are explained in greater detail throughout the technical part of this paper.

Outline. The paper is organized as follows: In the next section we formally define the orientation problem, discuss its complexity and describe a generic reduction to acyclic mixed graphs. In Section 3 we present approximation algorithms for instances that are trees, have a small feedback vertex number, or a small tree width. Section 4 presents the sub-linear approximation algorithms for the general case.

2. Preliminaries

Notation and terminology. We focus on simple graphs with no loops or parallel edges. A *mixed graph* is a triple $G = (V, E_{\mathrm{U}}, E_{\mathrm{D}})$ that consists of a vertex set V, a set of *undirected edges* $E_{\mathrm{U}} \subseteq \{e \subseteq V : |e| = 2\}$, and a set of *directed* *edges* $E_{\mathrm{D}} \subseteq V \times V$. We assume that every pair of vertices is either connected by a single edge of a specific type (directed or undirected) or not connected at all. We also write $V(G)$, $E_{\mathrm{U}}(G)$, and $E_{\mathrm{D}}(G)$ to refer to the sets V, E_{U}, and E_{D}, respectively. When G is clear from the context, we will denote $n = |V|$.

Let G_1 and G_2 be two mixed graphs. The graph G_1 is a *subgraph* of G_2 when the relations $V(G_1) \subseteq V(G_2)$, $E_{\mathrm{U}}(G_1) \subseteq E_{\mathrm{U}}(G_2)$, and $E_{\mathrm{D}}(G_1) \subseteq E_{\mathrm{D}}(G_2)$ hold. A *path* of length ℓ in a mixed graph G is a sequence $\langle v_1, \ldots, v_{\ell+1} \rangle$ of distinct vertices such that for every $1 \le i \le \ell$, we have $\{v_i, v_{i+1}\} \in E_U(G)$ or $(v_i, v_{i+1}) \in E_D(G)$. A path $\langle v_1, \ldots, v_{\ell+1} \rangle$ *crosses* a vertex $v \in V(G)$ when $v = v_i$ for some $1 \le i \le \ell + 1$. A path $\langle v_1, \ldots, v_{\ell+1} \rangle$ is a *cycle* when $v_1 = v_{\ell+1}$. Given a pair of vertices (s,t) from G, we say that t *is reachable from* s when there exists a path in G that goes from s to t. We say that G *satisfies* a pair (s,t) when there exists a path from s to t in G that uses only directed edges. A mixed graph with no cycles is called a *mixed acyclic graph* (MAG).

Let G be a mixed graph. An *orientation* $\vec{G}$ of G is a directed graph on the same vertex set, i.e. $V(G) = V(\vec{G})$, whose edge set contains all the directed edges of G, i.e. $E_{\mathrm{D}}(G) \subseteq E_{\mathrm{D}}(\vec{G})$, and a single oriented version of every undirected edge, i.e. for every $\{v,w\} \in E_{\mathrm{U}}(G)$ either $(v,w) \in E_{\mathrm{D}}(\vec{G})$ or $(w,v) \in E_{\mathrm{D}}(\vec{G})$ holds. When only a subset of the undirected edges have been oriented, we obtain a *partial orientation*.

Problem statement. The MAXIMUM-MIXED-GRAPH-ORIENTATION problem is defined as follows (see Figure 1 for an example).
Instance: A mixed graph G, and a collection of source-target vertex pairs $P \subseteq V(G) \times V(G)$.
Solution: An orientation $\vec{G}$ of G and the collection $\vec{P} \subseteq P$ of all pairs that are satisfied by $\vec{G}$.
Objective: Maximize the number of satisfied pairs, $|\vec{P}|$.

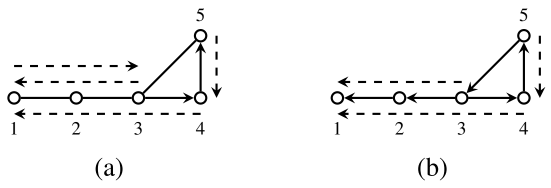

Figure 1: (a) An example instance (G,P) to MAXIMUM-MIXED-GRAPH-ORIENTATION. The input pairs $P = \{(1,3),(3,1),(4,1),(5,4)\}$ are denoted by dashed arrows that are directed from the source vertex to the target vertex. (b) An optimal orientation for this instance, satisfying all pairs other than $(1,3)$.

Hardness result. Arkin and Hassin [1] showed that it is NP-complete to decide whether, for a given mixed graph G and a collection of source-target pairs P, the graph G can be oriented to satisfy all pairs in P. Their reduction from the 3-SAT problem guarantees that for every $k \in \mathbb{N}$ there exists an assignment with k satisfied clauses if and only if there exists an orientation with k satisfied pairs. Thus, the inapproximability of MAX-3-SAT [22] directly transfers to MAXIMUM-MIXED-GRAPH-ORIENTATION, implying that it is NP-hard to approximate it to within a factor of $7/8$. We note that this bound is slightly stricter than the $12/13$-bound known for the special case of undirected graphs [14].

Reduction to mixed acyclic graphs. Given an instance (G,P), we can orient the undirected edges of any cycle in a consistent direction (i.e., either clockwise or counter-clockwise) without affecting the maximum number of source-target pairs that are satisfied by an optimal orientation. This observation gives rise to a polynomial-time reduction from mixed graphs to MAGs: First, we iteratively orient mixed cycles in the input graph. Then, we contract strongly connected components into single vertices, and connect two components by an undirected (directed) edge when some vertex in the first component is connected by such an edge to some vertex in the second component. Note that there cannot be more than one edge type as otherwise the two strongly connected components would have been merged earlier on. The pairs from P are adjusted accordingly from vertices of G to component vertices. Figure 2 shows an example of this reduction; a formal correctness proof is given in [28].

By the reduction above, we may focus our attention on treating MAGs. In addition, we may assume that each of the input pairs can be satisfied by some orientation; otherwise, it can be eliminated without affecting the optimum solution. Thus, throughout the paper, all instances considered are assumed to satisfy these two properties.

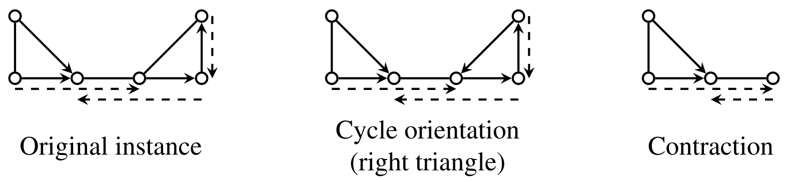

Figure 2: An example of the reduction to mixed acyclic graphs.

Given a MAG G, the components of the undirected graph $(V(G), E_{\mathrm{U}}(G))$ are called the *undirected components*; they must be trees that are connected by directed edges from $E_{\mathrm{D}}(G)$ without producing cycles. The *graph of undirected components of G* is the directed acyclic graph G_{UCC} with $V(G_{\mathrm{UCC}}) = \{T_i : T_i$ is an undirected component of $G\}$, and there is a directed edge from a node T_i to a node T_j when there is an edge from some vertex in T_i to some vertex in T_j.

3. Logarithmic approximations for tree-like instances

In this section we provide logarithmic approximations that apply to instances where the graph is "similar" to a tree, as formally defined in the sequel. In the remainder of this section, we make use of the following result about orienting undirected trees.

Lemma 3.1 (Medvedovsky et al. [25]). *There is a polynomial-time algorithm that, for instances (G,P) where G is an undirected tree, computes an orientation satisfying at least $|P|/(4\lceil \log n \rceil)$ pairs.*

It is worth mentioning that we could have alternatively used a slightly improved algorithm due to Gamzu et al. [19], who showed how to compute (in undirected trees) an orientation in which the number of satisfied pairs is within factor $\Omega(\log\log n/\log n)$ of optimal. However, their approximation guarantee is stated with respect to OPT, the maximum number of pairs that can be satisfied by any orientation, instead of with respect to $|P|$. Obviously, a bound of the latter type allowed us to prove that a certain fraction of all given pairs can always be satisfied, and for this reason we prefer not to save an additional $O(\log\log n)$ factor.

3.1. Orienting mixed trees

The above lemma guarantees that a logarithmic fraction of the input pairs can always be satisfied, and since it is constructive, we immediately derive an $\Omega(1/\log n)$ approximation algorithm for undirected trees. The following claims are of similar nature: We prove the existence of orientations satisfying a certain fraction of all input pairs, and this leads to approximation algorithms with the corresponding ratio. We start with orientations for mixed trees.

Lemma 3.2. *There is a polynomial-time algorithm that, for instances (G,P) where G is a mixed tree, computes an orientation satisfying at least $|P|/(4\lceil \log n \rceil)$ of the pairs.*

PROOF. We describe a reduction from mixed trees to undirected trees. The reduction applies the following edge contraction step as long as there are directed edges in the given mixed tree: Let (G,P) be an instances to the graph orientation problem and let $(v,w) \in U_{\mathrm{D}}(G)$. The *edge contraction step*: (1) deletes v from $V(G)$; (2) replaces each $\{v,u\} \in U_{\mathrm{U}}(G)$ by $\{w,u\}$, each $(v,u) \in U_{\mathrm{D}}(G)$ by (w,u), and each $(u,v) \in U_{\mathrm{D}}(G)$ by (u,w); and (3) replaces each $(v,t) \in P$ by (w,t), and each $(s,v) \in P$ by (s,w). The edge contraction step maintains that there are source-to-target paths for all pairs from P and orientations for the reduced tree can be transformed into orientations of the same solution sizes for the initial tree by reversing the contraction step.

After contracting all directed edges, the resulting graph is an undirected tree with source-to-target paths for every pair in P. By Lemma 3.1, there exists a polynomial-time algorithm that finds an orientation of this tree satisfying at least $|P|/(4\lceil \log n \rceil)$ pairs. Carrying over the edge orientations to the input mixed tree produces an orientation that satisfies the same number of pairs. □

3.2. Crossings through a junction component

Let (G,P) be a given instance and let $T_1, T_2, \ldots$ be the undirected components of G. A subgraph $S \subseteq G$ is said to be a *skeleton* of G if it can be obtained by deleting all but one directed edge between any pair of trees T_i and T_j. Clearly, G may have multiple skeletons, and we use $S(G)$ to denote the set of all skeletons. Now consider some skeleton $S \in S(G)$. It is not difficult to verify that S contains source-to-target paths for each pair in P, and that every orientation of S satisfying certain pairs directly translates into an orientation for G satisfying at least the same pairs. Figure 3 shows an example of a graph G and a skeleton for it. Note that for any MAG G and skeleton $S \in S(G)$, we have $G_{\text{UCC}} = S_{\text{UCC}}$.

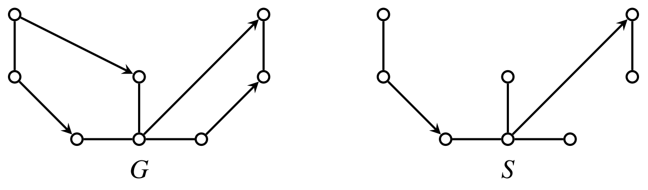

Figure 3: An example MAG G and a skeleton S.

An instance (G,P) is called a *junction instance* if there exists an undirected component T, such that each pair in P admits a source-to-target path that crosses a vertex from T. In Section 4 we will use Lemma 3.3 for instances where all pairs have source-to-target paths crossing a single distinguished vertex r.

Lemma 3.3. *There is a polynomial-time algorithm that, for junction instances (G,P), computes an orientation satisfying at least $|P|/(4\lceil \log n \rceil)$ of the pairs.*

PROOF. Let (G,P) be a junction instance and let T be an undirected component of G, such that each pair in P admits a source-to-target path that crosses a vertex from T. We begin by picking an arbitrary skeleton $S \in S(G)$. Since S is a MAG and, therefore, S_{UCC} is a directed acyclic graph, for every undirected component $T' \neq T$ of S_{UCC}, exactly one of the following options holds: (1) T' is reachable from T in S_{UCC}; (2) T is reachable from T' in S_{UCC}; or (3) there is no path between T and T' in either direction. Consequently, we can consider two subtrees that are rooted at T: The first subtree spans the vertices of S_{UCC} that are reachable from T, and the second subtree spans the vertices of S_{UCC} that can reach T. We merge both subtrees at T and call the resulting directed tree T_{UCC}. To compute an orientation for G, we consider the subtree of S that is constructed by taking all undirected components from T_{UCC}, and connect two vertices in different components by a directed edge if this edge is already present in S and the components are connected in T_{UCC}. This subtree of S contains a source-to-target path for each pair in P. Therefore, by Lemma 3.2, we can construct (in polynomial time) an orientation satisfying at least $|P|/(4\lceil \log n \rceil)$ pairs in S and, thus, also in the original graph G. □

3.3. Orientations for small feedback vertex sets or tree width

We end this section by providing logarithmic approximations to the orientation problem on tree-like instances. Precisely, we consider two graph parameters: *feedback vertex number* and *tree width*, showing that whenever either one of these is bounded by a constant, it is possible to compute an orientation that satisfies a poly-logarithmic fraction of the input pairs.

Theorem 3.4. *There is a polynomial-time algorithm that, given an instance (G,P), computes an orientation satisfying at least $|P|/(4(2k+1)\lceil \log n \rceil)$ pairs, where k is the minimum number of vertices whose deletion turns the underlying undirected graph of G_{UCC} into a tree.*

PROOF. We begin by detecting a small-sized feedback vertex set $F = \{T_1, \ldots, T_\ell\}$, consisting of ℓ vertices whose removal turns the underlying undirected graph of G_{UCC} into a tree. Even though finding a minimum cardinality vertex set of this type is NP-hard [23], this problem can be approximated to within a factor of 2 in undirected graphs [2], implying that we can assume $\ell \leq 2k$. We now partition P into two subsets, the collection of pairs P^+ for which we can find source-to-target paths in G that cross undirected components from F, and the collection $P^- = P \setminus P^+$. We further partition P^+ into ℓ subsets $P_1^+,\dots,P_\ell^+$, where a pair $(s,t)\in P^+$ lies in P_i^+ if i is the minimal index for which there exists a source-to-target path for this pair that crosses the undirected component T_i. With these definitions in place, note that by deleting the undirected components F from G, we can use Lemma 3.2 to efficiently compute an orientation of G satisfying at least $|P^-|/(4\lceil\log n\rceil)$ pairs; after deleting F the skeleton of the resulting graph is a tree and all pairs in P^- remain connected since they are only connected through paths that do not visit vertices from F. On the other hand, for each collection P_i^+ we can satisfy at least $|P_i^+|/(4\lceil\log n\rceil)$ pairs by applying Lemma 3.3. Picking the option that generates the highest number of satisfied pairs results in an orientation satisfying at least $|P|/(4(2k+1)\lceil\log n\rceil)$ of the pairs in P. □

We note that the above approximation result can be improved by a factor of 2 if the feedback vertex set has bounded size. For such instances we can invoke an exact fixed parameter algorithm [10] to find an optimal feedback vertex set, rather than using the 2-approximation algorithm. This is formally stated in the following corollary:

Corollary 3.5. *For every $k\in\mathbb{N}$, there is a polynomial-time algorithm that, given an instance (G,P) where the underlying undirected graph of G_{UCC} can be turned into a tree by deleting k vertices, computes an orientation satisfying at least $|P|/(4(k+1)\lceil\log n\rceil)$ pairs.*

Theorem 3.6. *For every $k\in\mathbb{N}$, there is a polynomial-time algorithm that, given an instance (G,P) where G_{UCC} has tree width k, computes an orientation satisfying at least $|P|/(4(k+1)\lceil\log n\rceil^2)$ pairs.*

PROOF. We first compute a tree decomposition of width k for the undirected underlying graph of G_{UCC}. The tree decomposition $(\mathcal{T},\{B_t\}_{t\in V(\mathcal{T})})$ consists of a tree $\mathcal{T}$ whose nodes are labeled with subsets B_t of vertices of G_{UCC}, called *bags*, such that: (1) the incident vertices of every edge are both contained in some bag; and (2) for every original vertex, the nodes of the bags that contain it make up a connected subtree. Its width is defined as the maximum number of vertices in a bag minus 1. For a comprehensive discussion on tree decompositions and their polynomial-time computability in the case of bounded tree width, we refer to the book of Flum and Grohe [17].

Based on the tree decomposition $(\mathcal{T},\{B_t\}_{t\in V(\mathcal{T})})$, we partition P into subsets $P_1,P_2,\dots,P_L$ with $L\le\lceil\log n\rceil$ such that for every subset we can efficiently find an orientation that satisfies a fraction of at least $1/(4(k+1)\lceil\log n\rceil)$ of its pairs. By picking the largest subset of pairs and its corresponding orientation, we obtain an orientation satisfying at least $|P|/(4(k+1)\lceil\log n\rceil^2)$ pairs.

For the purpose of constructing P_1, consider a *centroid* node t of $\mathcal{T}$ whose removal breaks this tree into subtrees of cardinality at most $|V(\mathcal{T})|/2$; noting that any tree necessarily contains a centroid (see, for instance, [18]). Let P_1 be the pairs in P with source-to-target paths that cross vertices from undirected components of $B_t=\{T_1,\dots,T_l\}$, where $l\le k+1$. We further partition P_1 into l collections $P_1^1,\dots,P_1^l$ such that a pair $(s,t)\in P_1$ lies in P_i^1 if there exists an *s*-*t* path that crosses vertices from T_i but no *s*-*t* paths that cross vertices from components T_j with $j<i$. By Lemma 3.3, we can compute an orientation that satisfies at least $|P_1^i|/(4\lceil\log n\rceil)$ of the pairs in P_1^i, for every $1\le i\le l$. By taking the largest collection, we can satisfy at least $|P_1|/(4(k+1)\lceil\log n\rceil)$ pairs in P_1.

To construct P_2, we proceed with the pair collection $P\setminus P_1$ that contains exactly the pairs from P with no source-to-target paths that cross vertices from the components of P_1. We delete the node t from $\mathcal{T}$, as well as the components in B_t from G. This results in a graph that contains source-to-target paths for all pairs from $P\setminus P_1$ and a forest of tree decompositions for the graph. For each tree decomposition we compute a centroid bag and, in the same way as above, the collection P_2 of pairs in $P\setminus P_1$ with source-to-target paths that cross components from these centroid bags. Using the same arguments as above, we can compute an orientation that satisfies at least $|P_2|/(4(k+1)\lceil\log n\rceil)$ of the pairs in P_2. We proceed recursively in the same way to construct $P_3,P_4,\dots,P_L$ as long as each tree decomposition (and the corresponding subgraph of G) is not empty. Since the maximal size of a subtree decreases by a factor of at least 2 in each level of the recursion, this process terminates within $\lceil\log n\rceil$ steps. □

4. Sub-linear approximations for general instances

In what follows, we focus our attention on approximating the orientation problem in its utmost generality, that is, without making simplifying structural assumptions on the underlying (mixed-acyclic) graph G or on the collection of input pairs P. The main result of this section is stated in the following theorem:

Theorem 4.1. *There is a polynomial-time algorithm that approximates* MAXIMUM-MIXED-GRAPH-ORIENTATION *to within a factor of* $\Omega(1/(M^{1/\sqrt{2}}\log n))$*, where* $M = \max\{n, |P|\}$.

In addition, we provide an improved approximation guarantee for input instances with bounded-distance pairs. This result is described in Section 4.3.

4.1. Algorithm

For each pair $(s_i, t_i) \in P$, let p_i be a shortest path from s_i to t_i in G, and let $\mathcal{P}$ be the set of all shortest paths, i.e., $\mathcal{P} = \{p_i : (s_i, t_i) \in P\}$. Our algorithm is based on a greedy framework where paths in $\mathcal{P}$ are oriented (from source to target) one after the other, trying not to interfere with future orientations of too many other paths by picking the shortest path in each step.

Greedy iteration. At any point in time, we will be holding a partial orientation G_ℓ of G and a subset $\mathcal{P}_\ell \subseteq \mathcal{P}$ of shortest paths, where these sets are indexed according to the step number that has just been completed. In other words, at the conclusion of step ℓ we have G_ℓ and $\mathcal{P}_\ell$, where initially $G_0 = G$ and $\mathcal{P}_0 = \mathcal{P}$. Now, as long as none of the termination conditions described below is met, we proceed as follows:

1. Let $\hat{p} = \langle s, \ldots, t \rangle$ be a shortest path in $\mathcal{P}_\ell$.
2. Orient $\hat{p}$ in the direction from s to t to obtain $G_{\ell+1}$.
3. Discard from $\mathcal{P}_\ell$ the path $\hat{p}$ as well as any path that has a non-empty edge intersection with $\hat{p}$. This way, we obtain $\mathcal{P}_{\ell+1}$.

Termination. There are two conditions that will cause the greedy iterations to terminate. For now, we state both conditions in terms of two parameters $\alpha \geq 0$ and $\beta \geq 0$, whose values will be optimized later on.

- Termination condition 1: $|\mathcal{P}_\ell| \leq n^\alpha$. In this case, we will orient an arbitrary path from $\mathcal{P}_\ell$, and update the current orientation to G_ℓ, as in the preceding greedy iterations. We then complete the orientation by arbitrarily orienting all yet-unoriented edges.

- Termination condition 2: There exists a vertex r such that at least $|\mathcal{P}_\ell|^\beta$ paths in $\mathcal{P}_\ell$ go through r. We construct a junction instance with input graph G_ℓ, junction vertex r, and pairs $\{(s_i, t_i) : p_i \in \mathcal{P}_\ell \text{ goes through } r\}$. We then apply the algorithm described in Section 3.2 for this special case, and return its output as our final orientation.

4.2. Analysis

To establish a lower bound on the number of satisfied pairs, we break the analysis into two cases, depending on the condition that caused the greedy iterations to terminate. In the remainder of this section, we assume that L greedy iterations have been completed prior to satisfying one of the termination conditions.

- Connections due to condition 1: In this case we satisfy a single pair out of $\{(s_i, t_i) : p_i \in \mathcal{P}_L\}$, noting that $|\mathcal{P}_L| \leq n^\alpha$.

- Connections due to condition 2: Following Lemma 3.3, the number of pairs satisfied out of $\{(s_i, t_i) : p_i \in \mathcal{P}_L\}$ is $\Omega(1/\log n) \cdot |\mathcal{P}_L|^\beta$.

We proceed by arguing that an $\Omega(1/n^{1-\alpha(1-2\beta)})$ fraction of the pairs in $\{(s_i, t_i) : p_i \notin \mathcal{P}_L\}$ are already satisfied by the partial orientation G_L. To this end, note that in each iteration $1 \leq \ell \leq L$ we satisfy a single pair by orienting the shortest path $\hat{p} \in \mathcal{P}_{\ell-1}$, and eliminating several others to obtain $\mathcal{P}_\ell$. To prove the claim above, it is sufficient to show that the number of eliminated paths satisfies $|\mathcal{P}_{\ell-1} \setminus \mathcal{P}_\ell| \leq n^{1-\alpha(1-2\beta)}$. Denote by $E(p)$ the set of edges of a path p, so that $|E(p)|$ is its length. We begin by observing that, since condition 2 has not been met in iteration ℓ, each edge can have at most $|\mathcal{P}_{\ell-1}|^{\beta}$ paths from $\mathcal{P}_{\ell-1}$ going through it, implying that $|\mathcal{P}_{\ell-1} \setminus \mathcal{P}_{\ell}| \leq |E(\hat{p})| \cdot |\mathcal{P}_{\ell-1}|^{\beta}$. Since $|E(\hat{p})|$ is upper bounded by the average length of the paths in $\mathcal{P}_{\ell-1}$, we have

$$\begin{aligned} |E(\hat{p})| &\leq \frac{1}{|\mathcal{P}_{\ell-1}|} \sum_{p_i \in \mathcal{P}_{\ell-1}} |E(p_i)| \leq \frac{1}{|\mathcal{P}_{\ell-1}|} \sum_{p_i \in \mathcal{P}_{\ell-1}} |V(p_i)| \\ &= \frac{1}{|\mathcal{P}_{\ell-1}|} \sum_{v \in V} |\{p_i \in \mathcal{P}_{\ell-1} : v \in V(p_i)\}| \\ &\leq \frac{1}{|\mathcal{P}_{\ell-1}|} \cdot n \cdot |\mathcal{P}_{\ell-1}|^{\beta} = \frac{n}{|\mathcal{P}_{\ell-1}|^{1-\beta}} \ , \end{aligned}$$

where the third inequality holds since condition 2 has not been met. Hence,

$$|\mathcal{P}_{\ell-1} \setminus \mathcal{P}_{\ell}| \leq \frac{n}{|\mathcal{P}_{\ell-1}|^{1-2\beta}} \leq \frac{n}{n^{\alpha(1-2\beta)}} = n^{1-\alpha(1-2\beta)} \ ,$$

where the second inequality follows from $|\mathcal{P}_{\ell-1}| > n^{\alpha}$, as condition 1 has not been met.

Based on the above discussion, it follows that the number of satisfied pairs when we terminate the algorithm due to condition 1 is

$$\begin{aligned} \Omega\left(\frac{1}{n^{1-\alpha(1-2\beta)}}\right)(|P| - |\mathcal{P}_L|) + 1 &= \Omega\left(\frac{1}{n^{1-\alpha(1-2\beta)}}\right)(|P| - n^{\alpha}) + \frac{1}{n^{\alpha}} n^{\alpha} \\ &= \Omega\left(\frac{1}{\max\{n^{1-\alpha(1-2\beta)}, n^{\alpha}\}}\right)|P| \\ &= \Omega\left(\frac{1}{n^{\max\{1-\alpha(1-2\beta),\alpha\}}}\right)|P| \ . \end{aligned}$$

Similarly, the number of satisfied pairs when the algorithm is terminated due to condition 2 is

$$\begin{aligned} &\Omega\left(\frac{1}{n^{1-\alpha(1-2\beta)}}\right)(|P| - |\mathcal{P}_L|) + \Omega\left(\frac{1}{\log n}\right)|\mathcal{P}_L|^{\beta} \\ &\quad = \Omega\left(\frac{1}{n^{1-\alpha(1-2\beta)}}\right)(|P| - |\mathcal{P}_L|) + \Omega\left(\frac{1}{|\mathcal{P}_L|^{1-\beta}\log n}\right)|\mathcal{P}_L| \\ &\quad = \Omega\left(\frac{1}{\max\{n^{1-\alpha(1-2\beta)}, |P|^{1-\beta}\log n\}}\right)|P| \\ &\quad = \Omega\left(\frac{1}{M^{\max\{1-\alpha(1-2\beta),1-\beta\}}}\right)\frac{1}{\log n}|P| \ . \end{aligned}$$

To obtain the best-possible performance guarantee, we pick values for α and β so as to minimize $\max\{\alpha, 1-\beta, 1-\alpha(1-2\beta)\}$. As explained below, the last term is optimized for $\alpha^* = \sqrt{1/2}$ and $\beta^* = \sqrt{1/2}/(2\sqrt{1/2}+1) = 1-\sqrt{1/2}$, in which case its value is $\sqrt{1/2} \approx 0.707$.

Optimizing α and β. Suppose we know the value of α^*. In this case, β^* should be picked so as to minimize $\max\{1-\beta, 1-\alpha^*(1-2\beta)\}$. Since $1-\beta$ is a decreasing linear function of β and $1-\alpha^*(1-2\beta)$ is an increasing linear function, this minimum is attained when $1-\beta = 1-\alpha^*(1-2\beta)$, that is, $\beta^* = \alpha^*/(2\alpha^*+1)$. For this value, we have $\max\{1-\beta^*, 1-\alpha^*(1-2\beta^*)\} = (\alpha^*+1)/(2\alpha^*+1)$. It remains to find a value of α that minimizes $\max\{\alpha, (\alpha+1)/(2\alpha+1)\}$. Using similar arguments, it is not difficult to verify that the right value to pick is $\alpha^* = \sqrt{1/2}$.

4.3. An improved approximation for bounded-distance pairs

In practice, the diameter of biological networks is sub-logarithmic due to their scale-free property [4, 12, 26]. For example, in the yeast physical network described by Silverbush et al. [28], the maximum source-target distance is 14. This motivates examining the approximation guarantee in terms of the maximum length of a shortest source-target path in the reduced mixed acyclic graph, which we denote by $\Delta = \Delta(G,P)$. In the following we present an $\Omega(1/\sqrt{\Delta|P|\log n})$ approximation to the orientation problem.

Our algorithm remains essentially unchanged, except for its termination conditions. Unlike the more general procedure, we ignore condition 1, and terminate the greedy iterations as soon as condition 2 is met, i.e., when there exists a vertex $r \in V$ such that at least $|\mathcal{P}_\ell|^\beta$ paths in $\mathcal{P}_\ell$ go through r. In this case, we construct a junction instance as before, with input graph G_ℓ, junction vertex r, and pairs $\{(s_i,t_i) : p_i \in \mathcal{P}_\ell \text{ goes through } r\}$. Our logarithmic approximation for this particular setting is then applied.

Similarly to the analysis in Section 4.2, we can prove the next two claims:

- Connections due to termination condition 2: The number of pairs satisfied out of $\{(s_i,t_i) : p_i \in \mathcal{P}_L\}$ is $\Omega(1/\log n) \cdot |\mathcal{P}_L|^\beta$.
- Connections due to greedy iterations: A fraction of $\Omega(1/(\Delta|P|^\beta))$ of the pairs in $\{(s_i,t_i) : p_i \notin \mathcal{P}_L\}$ are already satisfied by the partial orientation $\mathcal{G}_L$. This follows by observing that the number of paths that are eliminated from $\mathcal{P}_{\ell-1}$ in iteration ℓ is at most $\Delta|\mathcal{P}_{\ell-1}|^\beta \leq \Delta|P|^\beta$.

Consequently, the number of satisfied pairs upon termination is:

$$\begin{aligned}
&\Omega\left(\frac{1}{\Delta|P|^\beta}\right)(|P| - |\mathcal{P}_L|) + \Omega\left(\frac{1}{\log n}\right)|\mathcal{P}_L|^\beta \\
&\qquad = \Omega\left(\frac{1}{\Delta|P|^\beta}\right)(|P| - |\mathcal{P}_L|) + \Omega\left(\frac{1}{|\mathcal{P}_L|^{1-\beta}\log n}\right)|\mathcal{P}_L| \\
&\qquad = \Omega\left(\frac{1}{\max\{\Delta|P|^\beta, |P|^{1-\beta}\log n\}}\right)|P| \,.
\end{aligned}$$

By choosing $\beta = \frac{1}{2}(1 + \log_{|P|}(\frac{\log n}{\Delta}))$, we obtain an approximation ratio of $\Omega(1/\sqrt{\Delta|P|\log n})$.

In this section we used the standard definition of path lengths: the length of a path is its number of edges. The above analysis works in a similar way if we measure the length of a path by its number of undirected edges or even by the number of undirected components the path visits. This yields the same asymptotic worst-case bounds with respect to the size of the input, but highlights the increasing performance of the algorithm for structured inputs where these path length measures are small.

5. Conclusions

In this paper we presented approximation algorithms for the MAXIMUM-MIXED-GRAPH-ORIENTATION problem, which has recently arisen in the study of biological networks. We first showed that tree-like instances admit orientations (that can be computed in polynomial time) satisfying a poly-logarithmic fraction of the input pairs. Then we extended these algorithms to develop the first approximation algorithm for the problem whose ratio depends only on the size of the input instance, where no structural properties are assumed. The algorithm has a sub-linear approximation ratio, which can be improved when the input pairs are connected by short paths. The known upper and lower bounds for the approximation ratio of MAXIMUM-MIXED-GRAPH-ORIENTATION are far from being tight, both in the undirected and mixed cases. Closing this gap, both in the undirected and mixed cases, remains an open problem.

Acknowledgments

M.E. was supported by a research grant from the Dr. Alexander und Rita Besser-Stiftung. C.R.D. would like to thank Gerry Schwartz, Heather Reisman, and the University of Waterloo-Haifa International Experience Program for funding his visit to the University of Haifa, during which part of this work was done. R.S. was supported by a research grant from the Israel Science Foundation (grant no. 241/11).

References

[1] E. M. Arkin and R. Hassin. A note on orientations of mixed graphs. *Discrete Applied Mathematics*, 116(3):271–278, 2002. doi:10.1016/S0166-218X(01)00228-1.

[2] V. Bafna, P. Berman, and T. Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM Journal on Discrete Mathematics*, 12(3):289–297, 1999. doi:10.1137/S0895480196305124.

[3] J. Bang-Jensen and G. Gutin. *Digraphs: Theory, Algorithms and Applications*. Springer, 2nd edition, 2008.

[4] A.-L. Barabási and Z. N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5(2):101–113, February 2004. doi:10.1038/nrg1272.

[5] F. Boesch and R. Tindell. Robbins's theorem for mixed multigraphs. *The American Mathematical Monthly*, 87(9):716–719, 1980. URL: http://www.jstor.org/stable/2321858.

[6] M. Charikar, C. Chekuri, T.-Y. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li. Approximation algorithms for directed Steiner problems. *Journal of Algorithms*, 33(1):73–91, 1999.

[7] C. Chekuri, G. Even, A. Gupta, and D. Segev. Set connectivity problems in undirected graphs and the directed Steiner network problem. *ACM Transactions on Algorithms*, 7(2):18, 2011.

[8] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for non-uniform buy-at-bulk network design. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 677–686, 2006.

[9] C. Chekuri and S. Khanna. Edge-disjoint paths revisited. *ACM Transactions on Algorithms*, 3(4), 2007.

[10] J. Chen, F. V. Fomin, Y. Liu, S. Lu, and Y. Villanger. Improved algorithms for feedback vertex set problems. *Journal of Computer and System Sciences*, 74(7):1188–1198, 2008. doi:10.1016/j.jcss.2008.05.002.

[11] F. R. K. Chung, M. R. Garey, and R. E. Tarjan. Strongly connected orientations of mixed multigraphs. *Networks*, 15(4):477–484, 1985. doi:10.1002/net.3230150409.

[12] R. Cohen, S. Havlin, and D. ben-Avraham. Structural properties of scale-free networks. In *Handbook of Graphs and Networks: From the Genome to the Internet*, pages 85–110. Wiley-VCH, 2002.

[13] B. Dorn, F. Hüffner, D. Krüger, R. Niedermeier, and J. Uhlmann. Exploiting bounded signal flow for graph orientation based on cause-effect pairs. *Algorithms for Molecular Biology*, 6(1):21, 2011. doi:10.1186/1748-7188-6-21.

[14] M. Elberfeld, V. Bafna, I. Gamzu, A. Medvedovsky, D. Segev, D. Silverbush, U. Zwick, and R. Sharan. On the approximability of reachability-preserving network orientations. *Internet Mathematics*, 7(4):209–232, 2011. doi:10.1080/15427951.2011.604554.

[15] M. Elberfeld, D. Segev, C. R. Davidson, D. Silverbush, and R. Sharan. Approximation algorithms for orienting mixed graphs. In *Proceedings of the 22nd Annual Symposium on Combinatorial Pattern Matching (CPM 2011)*, volume 6661 of *Lecture Notes in Computer Science*, pages 416–428. Springer, 2011. doi:10.1007/978-3-642-21458-5_35.

[16] S. Fields. High-throughput two-hybrid analysis. The promise and the peril. *The FEBS Journal*, 272(21):5391–5399, 2005. doi:10.1111/j.1742-4658.2005.04973.x.

[17] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. doi:10.1007/3-540-29953-X.

[18] 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)*, volume 85 of *Lecture Notes in Computer Science*, pages 221–233. Springer, 1980. doi:10.1007/3-540-10003-2_73.

[19] 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)*, volume 6293 of *Lecture Notes in Computer Science*, pages 215–225. Springer, 2010. doi:10.1007/978-3-642-15294-8_18.

[20] 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. doi:10.1038/415141a.

[21] S. L. Hakimi, E. F. Schmeichel, and N. E. Young. Orienting graphs to optimize reachability. *Information Processing Letters*, 63(5):229–235, 1997. doi:10.1016/S0020-0190(97)00129-4.

[22] J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

[23] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[24] J. Kleinberg. *Approximation Algorithms for Disjoint Paths Problems*. Ph.D. Thesis, Department of Electrical Engineering and Computer Science, MIT, 1996.

[25] 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)*, volume 5251 of *Lecture Notes in Computer Science*, pages 222–232. Springer, 2008. doi:10.1007/978-3-540-87361-7_19.

[26] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003. doi:10.1137/S003614450342480.

[27] 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. URL: http://www.jstor.org/stable/2303897.

[28] D. Silverbush, M. Elberfeld, and R. Sharan. Optimally orienting physical networks. *Journal of Computational Biology*, 18(11):1437–1448, 2011. doi:10.1089/cmb.2011.0163.

[29] R. E. Tarjan. A note on finding the bridges of a graph. *Information Processing Letters*, 2(6):160–161, 1974. doi:10.1016/0020-0190(74)90003-9.

[30] C. Yeang, T. Ideker, and T. Jaakkola. Physical network models. *Journal of Computational Biology*, 11(2-3):243–262, 2004. doi:10.1089/1066527041410382.