Steiner Forest Orientation Problems

Marek Cygan* Guy Kortsarz† Zeev Nutov‡

November 20, 2011

Abstract

We consider connectivity problems with orientation constraints. Given a directed graph D and a collection of ordered node pairs P let $P[D] = \{(u, v) \in P : D \text{ contains a } uv\text{-paths}\}$. In the Steiner Forest Orientation problem we are given an undirected graph $G = (V, E)$ with edge-costs and a set $P \subseteq V \times V$ of ordered node pairs. The goal is to find a minimum-cost subgraph H of G and an orientation D of H such that $P[D] = P$. We give for this problem a 4-approximation algorithm.

In the Maximum Pairs Orientation problem we are given a graph G and a multi-collection of ordered node pairs P on V. The goal is to find an orientation D of G such that $|P[D]|$ is maximum. Generalizing the result of [1] for $|P| = 2$, we will show that for a mixed graph G (that may have both directed an undirected edges), one can decide in $n^{O(|P|)}$ time whether G has an orientation D with $P[D] = P$ (for undirected graphs this problem admits a polynomial time algorithm for any P, but it is NP-complete on mixed graphs). For undirected graphs, we will show that one can decide whether G admits an orientation D with $|P[D]| \geq k$ in $O(n + m) + 2^{O(k \cdot \log \log k)}$ time; hence this decision problem is fixed parameter tractable, which answers an open question from [2]. We also show that Maximum Pairs Orientation admits ratio $O(\log |P| / \log \log |P|)$, which is usually better than the ratio $O(\log n / \log \log n)$ of [6].

Finally, we will also show that the following node-connectivity problem can be solved in polynomial time: given a graph $G = (V, E)$ with edge-costs, $s, t \in V$, and an integer ℓ, find a min-cost subgraph H of G with an orientation D such that D contains ℓ internally-disjoint st-paths, and ℓ internally-disjoint ts-paths.

1 Introduction

1.1 Problems considered and our results

We consider connectivity problems with orientation constraints. Unless stated otherwise, graphs are assumed to be undirected (and may not be simple), but we also consider directed graphs, and even *mixed graphs*, which may have both directed an undirected edges. Given a mixed graph H, an

*Institute of Informatics, University of Warsaw, Poland, `cygan@mimuw.edu.pl`. Partially supported by National Science Centre grant no. N206 567140, Foundation for Polish Science and ONR Young Investigator award when at the University at Maryland.

†Rutgers University, Camden, `guyk@camden.rutgers.edu`. Partially supported by NSF support grant award number 0829959.

‡The Open University of Israel, `nutov@openu.ac.il`.

orientation of H is a directed graph D obtained from H by assigning to each undirected edge one of the two possible directions. For a mixed graph H on node set V and a multi-collection of ordered node pairs (that is convenient to consider as a set of directed edges) P on V let $P[H]$ denote the subset of the pairs (or edges) in P for which H contains a uv-path. We say that H satisfies P if $P[H] = P$, and that H is P-*orientable* if H admit an orientation D that satisfies P. We note that for undirected graphs it is easy to check in polynomial time whether H is P-orientable, c.f. [8] and Section 3 in this paper.

Our first problem is the classic Steiner Forest problem with orientation constraints.

Steiner Forest Orientation
Instance: A graph $G = (V, E)$ with edge-costs and a set $P \subseteq V \times V$ of ordered node pairs.
Objective: Find a minimum-cost subgraph H of G with an orientation D that satisfies P.

Theorem 1.1 Steiner Forest Orientation *admits a* 4*-approximation algorithms.*

Our next bunch of results deals with maximization problems of finding an orientation that satisfies the maximum amount of the demand.

Maximum Pairs Orientation
Instance: A graph G and a multi-collection of ordered node pairs (i.e., a set of directed edges) P on V.
Objective: Find an orientation D of G such that $|P[D]|$ is maximum.

Let k Pairs Orientation be the decision problem of determining whether Maximum Pairs Orientation has a solution of value at least k. Let P-Orientation be the decision problem of determining whether G is P-orientable (this is the k Pairs Orientation with $k = |P|$). As was mentioned, for undirected graphs P-Orientation can be easily decided in polynomial time. Arkin and Hassin [1] proved that that on mixed graphs, P-Orientation is NP-complete, but it is polynomial time solvable for $|P| = 2$. We widely generalize the result of [1] as follows.

Theorem 1.2 *Given a mixed graph H and $P \subseteq V \times V$ one can decide in $n^{O(|P|)}$ time whether H is P-orientable; namely,* P-Orientation *with a mixed graph H can be decided in $n^{O(|P|)}$ time. In particular, the problem can be decided in polynomial time for any instance with constant $|P|$.*

In several papers, c.f. [11], it is stated that any instance of Maximum Pairs Orientation admits a solution D such that $|P[D]| \geq |P|/(4 \log n)$. Furthermore Gamzu et al. [6] show that Maximum Pairs Orientation admits an $O(\log n/ \log \log n)$-approximation algorithm. In [2] it is shown that k Pairs Orientation is fixed parameter tractable[1] when parameterized by the maximum number of pairs that can be connected via one node. They posed an open question if the problem is fixed parameter tractable when parameterized by k. Our next result answers this open question and improves the bounds of [11, 6].

Theorem 1.3 *Any instance of* Maximum Pairs Orientation *admits a solution D, that can be computed in polynomial time, such that $|P[D]| \geq |P|/(4 \log_2(3|P|))$. Furthermore*

[1]In the parameterized complexity setting, an instance of a decision problem comes with an integer parameter k. A problem is said to be *fixed parameter tractable* (w.r.t. k) if there exists an algorithm that decides any instance $(|I|, k)$ in time $f(k)\text{poly}(|I|)$ for some (usually exponential) computable function f.

(i) k Pairs Orientation *can be decided in* $O(n+m)+2^{O(k\cdot\log\log k)}$ *time; thus it is fixed parameter tractable when parameterized by* k.

(ii) Maximum Pairs Orientation *admits an* $O(\log|P|/\log\log|P|)$*-approximation algorithm.*

Note that there are values of $|P|$ that are much smaller than n for which exhaustive search cannot be performed in polynomial time, say $|P| = 2^{\sqrt{\log n}}$. For this value of $|P|$, we can get ratio $O(\sqrt{\log n}/\log\log n)$, which is better than the ratio $O(\log n/\log\log n)$ of Gamzu et al. [6].

One may also consider "high-connectivity" orientation problems, to satisfy prescribed connectivity demands. Many papers considered edge-connectivity orientation problems, c.f. [10]. Almost nothing is known about node-connectivity orientation problems. We consider the following simple but nontrivial variant.

ℓ Disjoint Paths Orientation
Instance: A graph $G=(V,E)$ with edge-costs, $s,t\in V$, and an integer ℓ.
Objective: Find a min-cost subgraph H of G with an orientation D such that D contains ℓ internally-disjoint st-paths, and ℓ internally-disjoint ts-paths.

Checking whether ℓ Disjoint Paths Orientation admits a feasible solution can be done in polynomial time using the characterization of feasible solutions of Egawa, Kaneko, and Matsumoto [3] (see Theorem 5.1 in Section 5); we use this characterization to prove the following.

Theorem 1.4 ℓ Disjoint Paths Orientation *can be solved in polynomial time.*

Theorems 1.1, 1.2, 1.3, and 1.4, are proved in sections 2, 3, 4, and 5, respectively.

1.2 Previous and related work

Let $\lambda_H(u,v)$ denote the (u,v)-connectivity in a graph H, namely, the maximum number of pairwise edge-disjoint uv-paths in H. Similarly, let $\kappa_H(u,v)$ denote the maximum number of pairwise internally node-disjoint uv-paths in H. Given edge-connectivity demand function $r=\{r(u,v):(u,v)\in V\times V\}$, we say that H satisfies r if $\lambda_H(u,v)\geq r(u,v)$ for all $(u,v)\in V\times V$; similarly, for node connectivity demands, we say that H satisfies r if $\kappa_H(u,v)\geq r(u,v)$ for all $(u,v)\in V\times V$.

Survivable Network Orientation
Instance: A graph $G=(V,E)$ with edge-costs and edge/node-connectivity demand function $r=\{r(u,v):(u,v)\in V\times V\}$.
Objective: Find a minimum-cost subgraph H of G with orientation D that satisfies r.

So far we assumed that the orienting costs are symmetric; this means that orienting an undirected edge connecting u and v in each one of the two directions is the same, namely, that $c(u,v)=c(v,u)$. This assumption is reasonable in practical problems, but in a theoretic more general setting, we might have non-symmetric costs $c_{uv}\neq c_{vu}$. Note that the version with non-symmetric costs includes the min-cost version of the corresponding directed connectivity problem, and also the case when the input graph G is a mixed graph. For example, Steiner Forest Orientation includes the Directed Steiner Forest problem, which is Label-Cover hard to approximate; this is why we considered the symmetric costs version.

Khanna, Naor, and Shepherd [10] considered several orientation problems with non-symmetric costs. They showed that when D is required to be k-edge-outconnected from a given roots s (namely, D contains k-edge-disjoint paths from s to every other node), then the problem admits a polynomial time algorithm. In fact they considered a more general problem of finding an orientation that covers an intersecting supermodular function; further generalization of this result due to Frank, T. Király, and Z. Király was presented in [5]. For the case when D should be strongly connected, [10] obtained a 4-approximation algorithm; note that our Steiner Forest Orientation has more general demands, but we consider symmetric edge-costs. For the case when D is required to be k-edge-connected, $k \geq 2$, [10] obtained a pseudo-approximation algorithm that computes a $(k-1)$-edge-connected subgraph of cost at most $2k$ times the cost of an optimal k-connected subgraph.

We refer the reader to [4] for a survey on characterization of graphs that admit orientations satisfying prescribed connectivity demands, and here mention only the following central theorem, that can be used to obtain a pseudo-approximation for edge-connectivity orientation problems.

Theorem 1.5 (Well-Balanced Orientation Theorem, Nash-Williams [12])
Any undirected graph $H = (V, E_H)$ has an orientation D for which $\lambda_D(u,v) \geq \left\lfloor \frac{1}{2}\lambda_H(u,v) \right\rfloor$ for all $(u,v) \in V \times V$.

We note that given H, an orientation as in Theorem 1.5 can be computed in polynomial time. It is easy to see that if H has an orientation D that satisfies r then H satisfies the demand function q defined by $q(u,v) = r(u,v) + r(v,u)$. Theorem 1.5 implies that edge-connectivity Survivable Network Orientation admits a polynomial time algorithm that computes a subgraph H of G and an orientation D of H such that $c(H) \leq 2\mathsf{opt}$ and $\lambda_D(u,v) \geq \left\lfloor \frac{1}{2}(r(u,v) + r(v,u)) \right\rfloor$ for all $(u,v) \in V \times V$. This is achieved by applying Jain's [9] algorithm to compute a 2-approximate solution H for the corresponding undirected edge-connectivity Survivable Network instance with demands $q(u,v) = r(u,v) + r(v,u)$, and then computing an orientation D of H as in Theorem 1.5. The above algorithm also applies for non-symmetric edge-costs, invoking an additional cost factor of $\max_{uv \in E} c(v,u)/c(u,v)$. Summarizing, we have the following observation, which we failed to find in the literature.

Corollary 1.6 *Edge-connectivity* Survivable Network Orientation *(with non-symmetric costs) admits a polynomial time algorithm that computes a subgraph H of G and an orientation D of H such that $c(H) \leq 2\mathsf{opt} \cdot \max_{uv \in E} c(v,u)/c(u,v)$ and $\lambda_D(u,v) \geq \left\lfloor \frac{1}{2}(r(u,v) + r(v,u)) \right\rfloor$ for all $(u,v) \in V \times V$. In particular, the problem admits a 2-approximation algorithm if both the costs and the demands are symmetric.*

2 Proof of Theorem 1.1

For a mixed graph or an edge set H on a node set V and $X, Y \subseteq V$ let $\delta_H(X,Y)$ denote the set of all (directed and undirected) edges in H from X to Y and let $d_H(X,Y) = |\delta_H(X,Y)|$ denote their number; for brevity, $\delta_H(X) = \delta_H(X, \bar{X})$ and $d_H(X) = d_H(X, \bar{X})$, where $\bar{X} = V \setminus X$.

Given an integral set function f on subsets of V we say that H covers f if $d_H(X) \geq f(X)$ for all

$X \subseteq V$. Define a set-function f_r by $f_r(\emptyset) = f_r(V) = 0$ and for every $\emptyset \neq X \subset V$

$$f_r(X) = \max\{r(u,v) : u \in X, v \in \bar{X}\} + \max\{r(v,u) : u \in X, v \in \bar{X}\} \ . \tag{1}$$

Note that the set-function f_r is symmetric, namely, that $f_r(X) = f_r(\bar{X})$ for all $X \subseteq V$.

Lemma 2.1 *If H has an orientation D that satisfies an edge-connectivity demand function r then H covers f_r.*

Proof: Let $X \subseteq V$. By Menger's Theorem, any orientation D of H that satisfies r has at least $\max\{r(u,v) : u \in X, v \in \bar{X}\}$ edges from X to $\bar{X}$, and at least $\max\{r(v,u) : u \in X, v \in \bar{X}\}$ edges from $\bar{X}$ to X. The statement follows. $\square$

Recall that in the Steiner Forest Orientation problem we have $r(u,v) = 1$ if $(u,v) \in P$ and $r(u,v) = 0$ otherwise. We will show that if $r_{\max} = \max_{u,v \in V} r(u,v) = 1$ then the inverse to Lemma 2.1 is also true, namely, if H covers f_r then H has an orientation that satisfies r; for this case, we also give a 4-approximation algorithm for the problem of computing a minimum-cost subgraph that covers f_r. We do not know if these results can be generalized for $r_{\max} \geq 2$.

Lemma 2.2 *For $r_{\max} = 1$, if H covers f_r then H has an orientation that satisfies r.*

Proof: Observe that if $(u,v) \in P$ (namely, if $r(u,v) = 1$) then u, v belong to the same connected component of H. Hence it sufficient to consider the case when H is connected. Let D be an orientation of H obtained as follows. Orient every 2-edge-connected component of H to be strongly connected (recall that a directed graph is strongly connected if there is a directed path from any its node to the other); this is possible by Theorem 1.5. Now we orient the bridges of H. Consider a bridge e of H. The removal of e partitions V into two connected components $X, \bar{X}$. Note that $\delta_P(X, \bar{X}) = \emptyset$ or $\delta_P(\bar{X}, X) = \emptyset$, since $f_r(X) \leq d_H(X) = 1$. If $\delta_P(X, \bar{X}) \neq \emptyset$, we orient e from X to $\bar{X}$; if $\delta_P(\bar{X}, X) \neq \emptyset$, we orient e from $\bar{X}$ to X; and if $\delta_P(X, \bar{X}), \delta_P(\bar{X}, X) = \emptyset$, we orient e arbitrarily. It is easy to see that the obtained orientation D of H satisfies P. $\square$

We say that an edge-set or a graph H covers a set-family $\mathcal{F}$ if $d_H(X) \geq 1$ for all $X \in \mathcal{F}$. A set-family $\mathcal{F}$ is said to be uncrossable if for any $X, Y \in \mathcal{F}$ the following holds: $X \cap Y, X \cup Y \in \mathcal{F}$ or $X \setminus Y, Y \setminus X \in \mathcal{F}$. The problem of finding a minimum-cost set of undirected edges that covers an uncrossable set-family $\mathcal{F}$ admits a primal-dual 2-approximation algorithm, provided the inclusion-minimal members of $\mathcal{F}$ can be computed in polynomial time [7]. It is known that the undirected Steiner Forest problem is a particular case of the problem of finding a min-cost cover of an uncrossable family, and thus admits a 2-approximation algorithm.

Lemma 2.3 *Let $H = (V, J \cup P)$ be a mixed graph, where edges in J are undirected and edges in P are directed, such that for every $uv \in P$ both u, v belong to the same connected component of the graph (V, J). Then the set-family $\mathcal{F} = \{S \subseteq V : d_J(S) = 1 \wedge d_P(S), d_P(\bar{S}) \geq 1\}$ is uncrossable, and its inclusion minimal members can be computed in polynomial time.*

Proof: Let $\mathcal{C}$ be the set of connected components of the graph (V, J). Let $C \in \mathcal{C}$. Any bridge e of C partitions C into two parts $C'(e), C''(e)$ such that e is the unique edge in J connecting them. Note that the condition $d_J(S) = 1$ is equivalent to the following condition (C1), while if the condition (C1) holds then the condition $d_P(S), d_P(\bar{S}) \geq 1$ is equivalent to the following condition (C2) (since no edge in P connects two distinct connected components of (V, J)).

(C1) There exists $C_S \in \mathcal{C}$ and a bridge e_S of C_S, such that S is a union of one of the sets $X' = C'_S(e_S), X'' = C''_S(e_S)$ and sets in $\mathcal{C} \setminus \{C_S\}$.

(C2) $d_P(X', X''), d_P(X'', X') \geq 1$.

Hence we have the following characterization of the sets in $\mathcal{F}$: $S \in \mathcal{F}$ *if, and only if, conditions (C1),(C2) hold for* S. This implies that every inclusion-minimal members of $\mathcal{F}$ is $C'(e)$ or $C''(e)$, for some bridge e of $C \in \mathcal{C}$. In particular, the inclusion-minimal members of $\mathcal{F}$ can be computed in polynomial time.

Now let $X, Y \in \mathcal{F}$ (so conditions (C1),(C2) hold for each one of X, Y), let $C_X, C_Y \in \mathcal{C}$ be the corresponding connected components and e_X, e_Y the corresponding bridges (possibly $C_X = C_Y$, in which case we also may have $e_X = e_Y$), and let X', X'' and Y', Y'' be the corresponding partitions of C_X and C_Y, respectively. Since e_X, e_Y are bridges, at least one of the sets $X' \cap Y', X' \cap Y'', X'' \cap Y', X'' \cap Y''$ must be empty, say $X' \cap Y' = \emptyset$. Note that the set-family $\mathcal{F}$ is symmetric, hence to prove that $X \cap Y, X \cup Y \in \mathcal{F}$ or $X \setminus Y, Y \setminus X \in \mathcal{F}$, it is sufficient to prove that $A \setminus B, B \setminus A \in \mathcal{F}$ for any pair A, B such that $A \in \{X, \bar{X}\}, B \in \{Y, \bar{Y}\}$. E.g., if $A = X$ and $B = \bar{Y}$, then $A \setminus B = X \cap Y$ and $B \setminus A = V \setminus (X \cup Y)$, hence $A \setminus B, B \setminus A \in \mathcal{F}$ together with the symmetry of $\mathcal{F}$ implies $X \cap Y, X \cup Y \in \mathcal{F}$. Similarly, if $A = \bar{X}$ and $B = \bar{Y}$, then $A \setminus B = Y \setminus X$ and $B \setminus A = X \setminus Y$, hence $A \setminus B, B \setminus A \in \mathcal{F}$ implies $Y \setminus X, X \setminus Y \in \mathcal{F}$. Thus w.l.o.g. we may assume that $X' \subseteq X$ and $Y' \subseteq Y$, see Figure 1, and we show that $X \setminus Y, Y \setminus X \in \mathcal{F}$. Note that $X \cap Y$ is a (possibly empty) union of some sets in $\mathcal{C} \setminus \{C_X, C_Y\}$. Thus $X \setminus Y$ is a union of X' and some sets in $\mathcal{C} \setminus \{C_X, C_Y\}$. This implies that conditions (C1),(C2) hold for $X \setminus Y$, hence $X \setminus Y \in \mathcal{F}$; the proof that $Y \setminus X \in \mathcal{F}$ is similar. This concludes the proof of the lemma. □

Lemma 2.4 *Given a* Steiner Forest Orientation *instance, the problem of computing a minimum-cost subgraph H of G that covers f_r admits a 4-approximation algorithm.*

Proof: The algorithm has two phases. In the first phase we solve the corresponding undirected Steiner Forest instance with the same demand function r. The Steiner Forest problem admits a 2-approximation algorithms, hence $c(J) \leq 2\mathsf{opt}$. Let J be a subgraph of G computed by such a 2-approximation algorithm. Note that $f_r(S) - d_J(S) \leq 1$ for all $X \subseteq V$. Hence to obtain a cover of f_r it is sufficient to cover the family $\mathcal{F} = \{S \subseteq V : f_r(S) - d_J(S) = 1\}$ of the deficient sets w.r.t. J. The key point is that the family $\mathcal{F}$ is uncrossable, and that the inclusion-minimal members of $\mathcal{F}$ can be computed in polynomial time. In the second phase we compute a 2-approximate cover of this $\mathcal{F}$ using the algorithm of [7]. Consequently, the problem of covering f_r is reduced to solving two problems of covering an uncrossable set-family.

To show that $\mathcal{F}$ is uncrossable we us Lemma 2.3. Note that for any $uv \in P$ both u, v belong to the same connected component of (V, J), and that $f_r(S) - d_J(S) = 1$ if, and only if, $d_J(S) = 1$ and $d_P(S), d_P(\bar{S}) \geq 1$, hence $\mathcal{F} = \{S \subseteq V : d_J(S) = 1 \wedge d_P(S), d_P(\bar{S}) \geq 1\}$. Consequently, by Lemma 2.3, the family $\mathcal{F}$ is uncrossable and its inclusion-minimal members can be computed in polynomial time. This concludes the proof of the lemma. □

The proof of Theorem 1.1 is complete.

3 Proof of Theorem 1.2

The following (essentially known) statement is straightforward.

Lemma 3.1 *Let G be a mixed graph, let P be a set of directed edges on V, and let C be a subgraph of G that admits a strongly connected orientation. Let G', P' be obtained from G, P by contracting C into a single node. Then G is P-orientable if, and only if, G' is P'-orientable. In particular, this is so if C is a cycle.*

Corollary 3.2 P-orientation *(with an undirected graph G) can be decided in polynomial time.*

Proof: By repeatedly contracting a cycle of G, we obtain an equivalent instance, by Lemma 3.1. Hence we may assume that G is a tree. Then for every $(u, v) \in P$ there is a unique uv-path in G, which imposes an orientation on all the edges of this path. Hence if suffices to check that no two pairs in P impose different orientations of the same edge of the tree. $\square$

Our algorithm for mixed graphs is based on a similar idea. We say that a mixed graph is an *ori-cycle* if it admits an orientation that is a directed simple cycle. We need the following statement.

Lemma 3.3 *Let $G = (V, E \cup A)$ be a mixed graph and let G' be obtained from G by contracting every connected component of the undirected graph (V, E) into a single node. If there is a directed cycle (possibly a self-loop) C' in G' then there is an ori-cycle C in G, and such C can be found in polynomial time.*

Proof: If C' is also a directed cycle in G, then we take $C = C'$. Otherwise, we replace every node v_X of C' that corresponds to a contracted connected component X of (V, E) by a path, as follows. Let a_1 be the arc entering v_X in C' and let a_2 be the arc leaving v_X in C'. Let v_1 be the head of a_1 and similarly let v_2 be a the tail of a_2. Since X is a connected component in (V, E), there is a v_1v_2-path in (V, E), and we replace X by this path. The result is the required ori-cycle C (possibly a self-loop) in G. It is easy to see that such C can be obtained from C' in polynomial time. $\square$

By Lemmas 3.3 and 3.1 we may assume that the directed graph G' obtained from G by contracting every connected components of (V, E), is a directed acyclic multigraph (with no self-loops). Let $p = |P|$. Let $f : V \to V(G')$ be the function which for each node v of G assigns a node $f(v)$ in G' that represents the connected component of (V, E) that contains v (in other words the function f shows a correspondence between nodes before and after contractions).

The first step of our algorithm is to guess the first and the last edge for each pair i of the p pairs in P, by trying all $n^{O(p)}$ possibilities. If for pair i an undirected edge is selected as the first or the last one on the corresponding path, then we orient it accordingly and move it from E to A. Thus by functions $\text{last}, \text{first} : \{1, \ldots, p\} \to A$ we denote the guessed first and last arc for each of the p paths.

Now we present a branching algorithm with exponential time complexity which we later turn to $n^{O(p)}$ time by applying memorization. Let π be a topological ordering of G'. By $\text{cur} : \{1, \ldots, p\} \to A$ we denote the most recently chosen arc from A for each of the i-paths (initially $\text{cur}(i) = \text{first}(i)$).

In what follows we consider subsequent nodes v_C of G' with respect to π and branch on possible orientations of the connected component C of G. We use this orientation to update the function cur for all the arguments i such that $\mathrm{cur}(i)$ is an arc entering a node mapped to v_C.

Let $v_C \in V(G')$ be the first node w.r.t. to π which was not yet considered by the branching algorithm. Let $I \subseteq \{1, \ldots, p\}$ be the set of indices i such that $\mathrm{cur}(i) = (u, v) \in A$ for $f(v) = v_C$, and $\mathrm{cur}(i) \neq \mathrm{last}(i)$. If $I = \emptyset$ than we skip v_C and proceed to the next node in π. Otherwise for each $i \in I$ we branch on choosing an arc $(u, v) \in A$ such that $f(u) = v_C$, that is we select an arc that the i-path will use just after leaving the connected component of G corresponding to the node v_C (note that there are at most $|A|^{|I|} = n^{O(p)}$ branches). Before updating the arcs $\mathrm{cur}(i)$ for each $i \in I$ in a branch, we check whether the connected component C of (V, E) consisting of nodes $f^{-1}(v_C)$ is accordingly orientable by using Lemma 3.2 (see Fig. 2). Finally after considering all the nodes in π we check whether for each $i \in \{1, \ldots, p\}$ we have $\mathrm{cur}(i) = \mathrm{last}(i)$. If this is the case our algorithm returns YES and otherwise it returns NO in this branch.

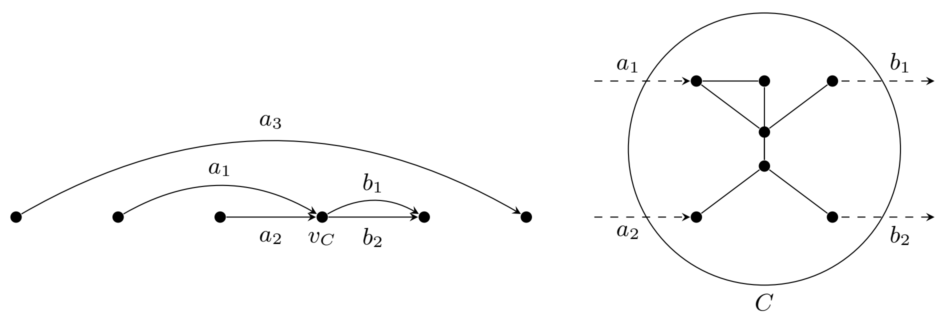

Figure 2: Our algorithm considers what orientation the connected component C (of (V, E)) will have. Currently we have $\mathrm{cur}(1) = a_1$,$\mathrm{cur}(2) = a_2$ and $\mathrm{cur}(3) = a_3$, hence $I = \{1, 2\}$. If in a branch we set new values $\mathrm{cur}(1) = b_1$ and $\mathrm{cur}(1) = b_2$ then by Lemma 3.2 we can verify that it is possible to orient C in such a way there is a path from the end-point of a_1 to the start-point of b_1 and from the end-point of a_2 to the start-point of b_2. However the branch when new values $\mathrm{cur}(1) = b_2$ and $\mathrm{cur}(2) = b_1$ will be terminated since it is not possible to orient C accordingly.

The correctness of our algorithm follows from the invariant that each node v_C is considered at most once since and all the updated values $\mathrm{cur}(i)$ are changed to arcs that are to the right with respect to π.

To improve the currently exponential time complexity we observe that we can apply the standard technique of memorization since when considering a node v_C it is not important what orientations previous nodes in π have, because all the relevant information is stored in the cur function. Hence in a state of memorization we store the index of the currently considered node in π and current values of the function cur, which leads to $n^{O(p)}$ states and $n^{O(p)}$ total time complexity.

4 Proof of Theorems 1.3

Lemma 4.1 *There exists a linear time algorithm that given an instance of* Maximum Pairs Orientation *or of* k Pairs Orientation, *transforms it into an equivalent instance such that the input graph is a tree with at most* $3p - 1$ *nodes.*

Proof: As is observed in [8], and also follows from Lemma 3.1, we can assume that the input graph is a tree; such tree can be constructed in linear time by contracting 2-edge-connected components. If a leaf of the tree does not belong to any pair in P, it can be discarded. If a leaf v belongs to exactly one pair in P, then without loss of generality we may contract the edge incident to v. If a node v has degree 2 in the tree and does not belong to any pair, we contract one of the edges incident to v; this is since in any inclusion minimal solution, one of the two edges enters v if, and only if, the other leaves v. It is not hard to verify that the described reduction can be performed in linear time.

We claim that after these reductions are implemented, the tree G' obtained has at most $3p - 1$ nodes. Let ℓ be the number of leaves and t the number of nodes of degree 2 in G'. As each node of degree less than 3 in G' is an s_i or t_i, $\ell + t \leq 2p$. Since each leaf belongs to at least two pairs in P, $\ell \leq p$. The number of nodes of degree at least 3 is at most $\ell - 1$ and so $|V(G')| \leq 2\ell + t - 1 \leq 3p - 1$. $\square$

After applying Lemma 4.1, the number of nodes n of the returned tree is at most $3p-1$. Therefore, by [11], one can find in polynomial time a solution D, such that $|P[D]| \geq p/(4\log_2 n) \geq p/(4\log_2(3p))$. Therefore, if for a given k Pairs Orientation instance we have $k \leq p/(4\log_2(3p))$, then clearly it is a YES instance. However if $k > p/(4\log_2(3p))$, then $p = \Theta(k \log k)$. In order to solve the k Pairs Orientation instance we consider all possible $\binom{p}{k}$ subsets P' of exactly k pairs from P, and check if the graph is P'-orientable. Observe that

$$\binom{p}{k} \leq \frac{p^k}{k!} \leq \frac{p^k}{(k/e)^k} \leq \frac{p^k}{(p/(4e\log_2(3p)))^k} = (4e\log_2(3p))^k = 2^{O(k\log\log k)}$$

where the second inequality follows from the Stirling's formula. Therefore the running time is $O(m + n) + 2^{O(k\log\log k)}$, which proves (i).

Combining Lemma 4.1 with the $O(\log n/\log\log n)$-approximation algorithm of Gamzu et al. [6] proves (ii), and consequently the proof of Theorem 1.3 is complete.

5 Proof of Theorem 1.4

We need the following characterization due to [3] of feasible solutions to ℓ Disjoint Paths Orientation.

Theorem 5.1 ([3]) *Let* $H = (V, E_H)$ *be an undirected graph and let* $s, t \in V$. *Then* H *has an orientation* D *such that* $\kappa_D(s,t), \kappa_D(t,s) \geq \ell$ *if, and only if,*

$$\lambda_{H\setminus C}(s,t) \geq 2(\ell - |C|) \quad \text{for every } C \subseteq V \setminus \{s,t\} \text{ with } |C| < \ell \ . \tag{2}$$

Furthermore, if H *satisfies (2), then an orientation* D *of* H *with* $\kappa_D(s,t), \kappa_D(t,s) \geq \ell$ *can be computed in polynomial time.*

Now, let us use the following version of Menger's Theorem for node and edge capacitated graphs; this version can be deduced from the original Menger's theorem by elementary constructions.

Lemma 5.2 *Let s, t be two nodes in a directed/undirected graph $H = (V, E)$ with edge and node capacities $\{u(a) : a \in E \cup (V \setminus \{s, t\}\}$. Then the maximum number of st-paths such that every $a \in E \cup (V \setminus \{s, t\})$ appears in at most $u(a)$ of them equals to* $\min\{u(A) : A \subseteq E \cup (V \setminus \{s, t\}), \lambda_{H \setminus A}(s, t) = 0\}$.
□

From Lemma 5.2 we deduce the following.

Corollary 5.3 *An undirected graph $H = (V, E)$ satisfies (2) if, and only if,*

$$H \text{ contains } 2\ell \text{ edge-disjoint st-paths such that every } v \in V \setminus \{s, t\} \text{ belongs to at most 2 of them .} \quad (3)$$

Proof: Assign capacity $u(e) = 1$ to every $e \in E$ and capacity $u(v) = 2$ to every $v \in V \setminus \{s, t\}$. By Lemma 5.2, (3) is equivalent to the condition

$$\min\{u(C \cup F) : F \subseteq E, C \subseteq (V \setminus \{s, t\}), \lambda_{H \setminus (C \cup F)}(s, t) = 0\} \geq 2\ell \ .$$

Since $u(C) = 2$ for all $C \subseteq V \setminus \{s, t\}$ and $u(F) = |F|$ for all $F \subseteq E$, the latter condition is equivalent to the condition

$$\min\{|F| : F \subseteq E, \lambda_{(H \setminus C) \setminus F}(s, t) = 0\} \geq 2\ell - 2|C| \quad \text{for every } C \subseteq V \setminus \{s, t\} \text{ with } |C| < \ell \ .$$

The above condition is equivalent to (2), since for every $C \subseteq V \setminus \{s, t\}$ we have $\min\{|F| : F \subseteq E, \lambda_{(H \setminus C) \setminus F}(s, t) = 0\} = \lambda_{H \setminus C}(s, t)$, by applying Menger's Theorem on the graph $H \setminus C$. The statement follows. □

Now consider the following problem.

Node-Capacitated Min-Cost k-Flow
Instance: A graph $G = (V, E)$ with edge-costs, $s, t \in V$, node-capacities $\{b_v : v \in V \setminus \{s, t\}\}$, and an integer k.
Objective: Find a set Π of k edge-disjoint paths such that every $v \in V \setminus \{s, t\}$ belongs to at most b_v paths in Π.

From Corollary 5.3, we see that ℓ Disjoint Paths Orientation is a particular case of Node-Capacitated Min-Cost k-Flow when H is undirected, $k = 2\ell$, and all node capacities are 2.

Node-Capacitated Min-Cost k-Flow can be solved in polynomial time, for both directed and undirected graphs, by reducing the problem to the standard Edge-Capacitated Min-Cost k-Flow problem. For directed graphs this can be done by a standard reduction of converting node-capacities to edge-capacities: replace every node $v \in V \setminus \{s, t\}$ by the two nodes v^+, v^-, connected by the edge v^+v^- having the same capacity as v, and redirect the heads of the edges entering v to v^+ and the tails of the edges leaving v to v^-. The undirected case is easily reduced to the directed one, by solving the problem on the bidirection graph of G, obtained from G by replacing every undirected edge e connecting u, v by a pair of antiparallel directed edges uv, vu of the same cost as e.

The proof of Theorem 1.4 is complete.

References

[1] E. Arkin and R. Hassin. A note on orientations of mixed graphs. *Discrete Applied Mathematics*, 116(3):271–278, 2002.

[2] 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 First international ICST conference on Theory and practice of algorithms in (computer) systems*, pages 104–115, 2011.

[3] Y. Egawa, A. Kaneko, and M. Matsumoto. A mixed version of Menger's theorem. *Combinatorica*, 11:71–74, 1991.

[4] A. Frank and T. Király. Combined connectivity augmentation and orientation problems. *Discrete Applied Mathematics*, 131(2):401–419, 2003.

[5] A. Frank, T. Király, and Z. Király. On the orientation of graphs and hypergraphs. *Discrete Applied Mathematics*, 131:385400, 2003.

[6] I. Gamzu, D. Segev, and R. Sharan. Improved orientations of physical networks. In *WABI*, pages 215–225, 2010.

[7] M. Goemans, A. Goldberg, S. Plotkin, D. Shmoys, E. Tardos, and D. Williamson. Improved approximation algorithms for network design problems. In *SODA*, pages 223–232, 1994.

[8] R. Hassin and N. Megiddo. On orientations and shortest paths. *Linear Algebra and Its Applications*, pages 589–602, 1989.

[9] K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[10] S. Khanna, J. Naor, and B. Shepherd. Directed network design with orientation constraints. In *SODA*, pages 663–671, 2000.

[11] 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 *WABI*, pages 222–232, 2008.

[12] Nash-Williams. On orientations, connectivity and odd vertex pairings in finite graphs. *Canad. J. Math.*, 12:555–567, 1960.