Egerváry Research Group
on Combinatorial Optimization

Technical reportS

TR-2020-15. Published by the Egerváry Research Group, Pázmány P. sétány 1/C, H–1117, Budapest, Hungary. Web site: www.cs.elte.hu/egres. ISSN 1587–4451.

Simple algorithm and min-max formula for the inverse arborescence problem

András Frank and Gergely Hajdu

September 2020

Simple algorithm and min-max formula for the inverse arborescence problem

András Frank and Gergely Hajdu ⋆

Abstract

In 1998, Hu and Liu developed a strongly polynomial algorithm for solving the inverse arborescence problem that aims at modifying minimally a given cost-function on the edge-set of a digraph so that an input arborescence becomes a cheapest one. In this note, we develop a conceptually simpler algorithm along with a min-max theorem for the minimum modification of the cost-function. The approach is based on a link to a min-max theorem and a two-phase greedy algorithm by the first author from 1979 concerning the primal optimization problem of finding a cheapest subgraph of a digraph that covers an intersecting family along with the corresponding dual optimization problem, as well.

1 Introduction

Let $D = (V, A)$ be a loopless digraph with n nodes and m edges. Let r_0 be a root-node of D. An **arborescence** is a directed tree in which the in-degree of all but one node is 1. The exceptional node is called the **root**, its in-degree is 0. In 1965, Chu and Liu [4] developed a simple strongly polynomial algorithm for computing a spanning arborescence of D of minimum cost with respect to a given cost-function on A.

In the **inverse arborescence problem**, we are given a spanning arborescence F_0 of D with root r_0 and a cost-function $w_0 : A \to \mathbf{R}_+$. The goal is to modify w_0 so that F_0 becomes a cheapest arborescence with respect to the revised cost-function w, and the deviation of w from w_0 is as small as possible. The **deviation** $|w - w_0|$ of w (from w_0) is defined by $\sum(|w(a) - w_0(a)| : a \in A)$, and we use throughout the paper this L_1-norm to measure the optimality of w.

In 1998, Hu and Liu [11] described a strongly polynomial algorithm for this inverse problem. Both their algorithm and the proof of its correctness were rather complex. The goal of the present work is to develop a conceptually simpler algorithm and a min-max formula for the minimum deviation μ^* of the revised cost-function w for

⋆MTA-ELTE Egerváry Research Group, Department of Operations Research, Eötvös University, Pázmány P. s. 1/c, Budapest, Hungary, H-1117. e-mail: `frank@cs.elte.hu and hgergely91@gmail.com` . Project no. NKFI-128673 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the FK18 funding scheme.

which the input arborescence is a cheapest one of D. The approach is based on a link to a paper by the first author [7] from 1979 that includes a two-phase greedy algorithm for solving a natural extension of the cheapest arborescence problem. Not only the algorthm is simple but so is the proof of its correctness.

Cai and Li [2, 3] showed how the inverse matroid intersection problem can be reduced to a minimum cost circulation problem and therefore a purely combinatorial strongly polynomial algorithm for circulation (e.g. the first one due to Tardos [13]) can be applied. Since arborescences form the common bases of two special matroids, the solution of Cai and Li may be specialized to arborescences. Our main goal is to present an approach giving rise to a simpler and more efficient algorithm, and to a min-max formula, as well. The situation is analogous to the one where the general weighted matroid intersection algorithm of Edmonds' [6] did not make superfluous the simpler and more efficient direct algorithm of Chu and Liu [4] concerning cheapest arborescences. It should, however, be noted that the min-cost flow approach of Cai and Li provides a solution to the significantly more general problem when there is an upper-bound constraint $g(a)$ on $|w(a) - w_0(a)|$ for every edge a, and the objective is to minimize $\sum[c(a)|w(a) - w_0(a)| : a \in A]$ where $c : A \to \mathbf{R}_+$ is a given cost-function.

1.1 Terminology and notation

For a directed edge (or arc) $a = uv$, v is called the **head** of a while u is its **tail**. We say that uv **enters** (**leaves**) subset Z of nodes if $v \in Z$ and $u \notin Z$ ($v \notin Z$ and $u \in Z$). In a digraph $D = (V, A)$, the number of edges entering Z is denoted by $\varrho_D(Z) = \varrho_A(Z)$ while the number of edges leaving Z is denoted by $\delta_D(Z) = \delta_A(Z)$. A subset L of edges is said to **enter** (or **cover**) Z if L contains an edge entering Z, that is, if $\varrho_L(Z) \geq 1$. For a family $\mathcal{F}$ of subsets, we say that L **enters** (or **covers**) $\mathcal{F}$ if L enters each member of $\mathcal{F}$. For two elements s and t, a set Z is called a $t\overline{s}$**-set** if $t \in Z$ and $s \notin Z$.

A digraph D is called **root-connected** with respect to a root-node r_0 if $\varrho_D(Z) \geq 1$ holds for every non-empty subset $Z \subseteq V - r_0$. Clearly, root-connectivity is equivalent to requiring that every node of D is reachable from r_0 (along a dipath). An easy and well-known property is that an inclusionwise minimal root-connected subgraph of D is an arborescence. In what follows, an r_0-arborescence or a spanning arborescence always means a spanning arborescence of root r_0. More generally, D is **rooted** k-**edge-connected** if $\varrho(Z) \geq k$ holds for every non-empty subset $Z \subseteq V - r_0$.

A function $x : S \to \mathbf{R}$ on S can be extended to a set-function $\widetilde{x}$ by $\widetilde{x}(Z) := \sum[x(s) : s \in Z]$ $(Z \subseteq S)$. Analogously, for a set-function y on S and for a family $\mathcal{F}$ of subsets of S, we use the notation $\widetilde{y}(\mathcal{F}) := \sum[y(Z) : Z \in \mathcal{F}]$.

Two sets X and Y are called **intersecting** if $X \cap Y \neq \emptyset$. If, in addition, $X - Y$ and $Y - X$ are non-empty, then X and Y are **properly intersecting**. A family $\mathcal{F}$ of sets is **laminar** if it has no two properly intersecting members. $\mathcal{F}$ is **intersecting** if both $X \cap Y$ and $X \cup Y$ belong to $\mathcal{F}$ whenever X and Y are intersecting members of $\mathcal{F}$. Given a digraph $D = (V, A)$, we say that an intersecting family $\mathcal{F}$ of distinct subsets of V is a **kernel system** [7] if $\varrho_D(Z) > 0$ for each $Z \in \mathcal{F}$. All other notions and notation can be found in [9].

2 Arborescences and kernel systems

2.1 Cheapest arborescences

Let $D = (V, A)$ be a root-connected digraph with a root-node r_0 and let $c : A \to \mathbf{R}_+$ be a non-negative cost-function on the edge-set. The primal problem consists of determining a cheapest r_0-arborescence. We say that a function $y : \mathcal{F} \to \mathbf{R}$ defined on a set-system $\mathcal{F} \subseteq 2^V$ is c-**feasible** if $y \geq 0$ and if

$$\sum[y(Z) : Z \in \mathcal{F},\ Z \ \text{ is entered by } \ a] \leq c(a) \ \text{ for every edge } a \in A. \tag{1}$$

When $\mathcal{F} := \{X : \emptyset \neq X \subseteq V - r_0\}$, a c-feasible function y will be referred to as a **dual solution** to the cheapest arborescence problem. We call an edge a of D c-**tight** (or just **tight**) (with respect to y) if $\sum[y(Z) : Z \in \mathcal{F},\ Z \ \text{ is entered by } \ a] = c(a)$. Bock [1] and Fulkerson [10] proved the following min-max formula.

THEOREM 2.1 (Bock, Fulkerson)**.** *Let c be a non-negative cost-function on the edge-set of root-connected digraph $D = (V, A)$. The minimum cost of a spanning arborescence of root r_0 is equal to*

$$\max\{\sum[y(Z) : Z \subseteq V - r_0] : y \ c\text{-feasible}\}. \tag{2}$$

There is an optimal dual solution y for which $\{Z : y(Z) > 0\}$ is laminar. If c is integer-valued, the optimal y can also be chosen integer-valued. ■

Note that Fulkerson [10] developed a simple greedy algorithm for computing the optimal dual vector y occurring in the theorem. The theorem immediately implies the following optimality criteria.

Corollary 2.2. *Let y^* be a c-feasible function on the family of non-empty subsets of $V - r_0$ and let F_0 be a spanning r_0-arborescence for which the following opitmality criteria hold.*

(A) *F_0 consists of tight edges, and*
(B) *$y^*(Z) > 0$ implies $\varrho_{F_0}(Z) = 1$.*

Then F_0 is a spanning arborescence of minimum c-cost for which $\widetilde{c}(F_0) = \sum[y^*(Z) : Z \subseteq V - r_0]$. ■

The book of Schrijver [12] provides a rich background and bibliography of algorithms and min-max results concerning arborescences and related objects like branchings.

2.2 Known min-max theorem for kernel systems

In 1979, Frank [7] extended the problem of cheapest arborescences to kernel systems when one is interested in finding a cheapest subset L of edges that enters every member of an intersecting family $\mathcal{F}$ of subsets (for short, L enters or covers $\mathcal{F}$). A special case is when $\mathcal{F}$ consist of all the non-empty subsets of $V - r_0$. In this case, the inclusionwise minimal subsets of edges covering $\mathcal{F}$ are exactly the spanning r_0-arborescences.

Another special case is when $\mathcal{F}$ consists of all the $t\overline{s}$-sets: here the inclusionwise minimal edge-sets covering $\mathcal{F}$ are precisely the st-dipaths. We shall need a third special case where $\mathcal{F}$ consists of those subsets Z of $V - r_0$ which are entered exactly once by a specified arborescence F_0 (that is, $\varrho_{F_0}(Z) = 1$).

The primal problem consists of finding a cheapest subset of edges covering $\mathcal{F}$. The dual problem consists of finding a c-feasible function $y : \mathcal{F} \to \mathbf{R}$ for which

$$\widetilde{y}(\mathcal{F}) = \sum [y(Z) : Z \in \mathcal{F}]$$

is as large as possible. In [7], the algorithm of Fulkerson [10] was extended in a natural way to general kernel systems. In addition, [7] described a second phase of the algorithm which computes in a greedy way the cheapest subset L of edges covering $\mathcal{F}$.

This two-phase greedy algorithm proved the following extension of Theorem 2.1.

THEOREM 2.3 ([7])**.** *Let $\mathcal{F}$ be an intersecting set-system on the node-set of digraph $D = (V, A)$ for which A enters $\mathcal{F}$, and let $c : A \to \mathbf{R}_+$ be a cost-function. Then*

$$\min\{\widetilde{c}(L) : L \subseteq A,\ L \text{ covers } \mathcal{F}\} = \max\{\widetilde{y}(\mathcal{F}) :\ y \text{ c-feasible}\}. \tag{3}$$

If c is integer-valued, the optimal dual solution y can also be chosen integer-valued. Moreover, there is an optimal y for which the set-system $\{Z : y(Z) > 0\}$ is laminar.

■

The trivial direction max $\leq$ min of the theorem implies for a dual solution y and for a subset L of edges covering $\mathcal{F}$ that if they meet the following **optimality criteria**:

(A) every edge in L is tight (with respect to y),

(B) $y(Z) > 0$ implies $\varrho_L(Z) = 1$,

then y is an optimal dual solution and L is a c-cheapest covering of $\mathcal{F}$. The non-trivial direction max $\geq$ min of the theorem is equivalent to stating that there exist a dual solution y^* and a covering $L^* \subseteq A$ of $\mathcal{F}$ meeting the optimality criteria.

In the special case when $\mathcal{F} := \{Z : \emptyset \neq Z \subseteq V - r_0\}$, the subsets of edges covering $\mathcal{F}$ are exactly the edge-sets of root-connected subgraphs of D. Since minimal members of root-connected subgraphs are the spanning r_0-arborescences of D, it follows in this case that the cheapest coverings of $\mathcal{F}$ are the cheapest arborescences, and hence the general Theorem 2.3 reduces to Theorem 2.1.

We note that [7] actually includes a min-max theorem (concerning a cheapest covering of an intersecting supermodular function) which is significantly more general than Theorem 2.3. Here we do not need this general result, and only remark that its proof in [7] is not constructive anyway, unlike the simple algorithmic proof of Theorem 2.3.

We also remark that the min-max theorem and that two-phase greedy algorithm in [7] was generalized in [8] for the problem of finding a cheapest covering of an intersecting bi-set system. See also the book of Frank [9], Page 395.

3 Algorithm for kernel systems

In [7], Theorem 2.3 was proved with the help of a two-phase greedy algorithm. The first phase for computing an optimal c-feasible dual solution y^* is a straight extension of Fulkerson's algorithm [10] for the dual of the cheapest arborescence problem. Since we need this first phase for our suggested solution to the inverse arborescence problem, we describe it with full detail. We shall outline the second phase, as well, which computes, also in a greedy way, a cheapest covering L of $\mathcal{F}$. This algorithm will be needed to compute the optimal dual object occurring in the min-max formula in Theorem 5.2 concerning the minimum deviation of the wanted cost-function in the inverse arborescence problem.

3.1 Known algorithm for general kernel systems

We start by outlining the two-phase greedy algorithm described in [7]. Given cost-function $c' : A \to \mathbf{R}_+$, we call a subset $Z \subset V$ **c'-positive** if the c'-cost of every edge entering Z is positive.

First Phase If $\mathcal{F}$ has no c-positive member, then $y^* \equiv 0$ is an optimal dual solution (and the set of edges with zero c-cost is a cheapest covering of $\mathcal{F}$), in which case the algorithm terminates. Therefore, we assume that $\mathcal{F}$ admits a c-positive member.

The algorithm determines one by one the members $c_1 := c, c_2, c_3, \dots$ of a series of non-negative cost-functions and a series $Z_1, Z_2, Z_3, \dots$ of members of $\mathcal{F}$ along with positive dual variables $y^*(Z_i)$ assigned to these sets Z_i, which are integer-valued when c is integer-valued.

In Step $i = 1$, Z_1 is an inclusionwise minimal c_1-positive member of $\mathcal{F}$. Let

$$y^*(Z_1) := \min\{c_1(f) : f \in A, f \text{ enters } Z_1\}$$

and define c_2, as follows.

$$c_2(f) := \begin{cases} c_1(f) & \text{if } \; f \text{ does not enter } Z_1 \\ c_1(f) - y^*(Z_1) & \text{if } \; f \text{ enters } Z_1. \end{cases} \tag{4}$$

For the general case $i \geq 2$, suppose that c_i has already been computed. If every member of $\mathcal{F}$ is entered by a 0 c_i-cost edge, then the first phase terminates. If this is not the case, then let Z_i be a inclusionwise minimal c_i-positive member of $\mathcal{F}$. Let

$$y^*(Z_i) := \min\{c_i(f) : f \in A, f \text{ enters } Z_i\}$$

and define c_{i+1}, as follows.

$$c_{i+1}(f) := \begin{cases} c_i(f) & \text{if } \; f \text{ does not enter } Z_i \\ c_i(f) - y^*(Z_i) & \text{if } \; f \text{ enters } Z_i. \end{cases} \tag{5}$$

The algorithm is greedy in the sense that once a positive dual variable $y^*(Z_i)$ is determined, it is not changed anymore in later steps. The algorithm needs the following subroutine.

Subroutine (A): *determines for a given non-negative cost-function c' whether $\mathcal{F}$ has a c'-positive member or not, and if it does, then the subroutine computes an inclusionwise minimal c'-positive member of $\mathcal{F}$.*

It was pointed out in [7] that the following property of the dual solution y^* is an easily provable consequence of the greedy algorithm above.

Claim 3.1. *The optimal dual solution y^* provided by the algorithm has the property that the set-system $\mathcal{F}^* := \{Z : y^*(Z) > 0\}$ is laminar.* ■

The second phase of the algorithm in [7] for computing a cheapest covering of $\mathcal{F}$ is as follows.

Second Phase Let c' denote the cost-function obtained by the end of Phase 1, and let $A_0 := \{a \in A : c'(a) = 0\}$. By the termination rule of Phase 1, A_0 covers $\mathcal{F}$. Note that the edges in A_0 are c-tight with respect to the dual solution y^* obtained by the first phase.

In order to construct a cheapest covering L of $\mathcal{F}$, we pick up edges from A_0 one by one, as follows. At the beginning, L is empty. In the general step, we check if the current L covers $\mathcal{F}$ or not. If it does, Phase 2 (and the whole algorithm) terminates. If L does not yet cover $\mathcal{F}$, we select a maximal member Z of $\mathcal{F}$ not covered (= entered) by L, and select an edge a from A_0 entering Z whose cost became zero at earliest during Phase 1 (or, in other words, a became tight at earliest). Let L^* denote the covering of $\mathcal{F}$ obtained by the termination of Phase 2. The following lemma from [7] immediately implies Theorem 2.3. We do not include its proof here only remark that it is an easy consequence of Claim 3.1 and the earliest-choice rule (used for selecting the subsequent element of L).

Lemma 3.2 ([7]). *The dual solution y^* provided by Phase 1 and the covering L^* of $\mathcal{F}$ obtained in Phase 2 meet the optimality criteria.* ■

4 Specific algorithm for kernel system $\mathcal{F}_0$

Let F_0 be a spanning arborescence of $D = (V, A)$ with root-node r_0. The key idea of our approach to the inverse aborescence problem is that we apply the min-max formula in Theorem 2.3 and the algorithm in Section 3.1 concerning general kernel systems to a specific kernel system $\mathcal{F}_0$ assigned to F_0. Namely, let $\mathcal{F}_0$ denote the system of subsets $Z \subseteq V - r_0$ for which $\varrho_{F_0}(Z) = 1$, that is,

$$\mathcal{F}_0 := \{Z \subseteq V - r_0 : \ \varrho_{F_0}(Z) = 1\}. \tag{6}$$

The following is a standard observation.

Claim 4.1. *The set-system $\mathcal{F}_0$ is intersecting.*

Proof. Let X and Y be two intersecting members of $\mathcal{F}_0$. Then $\varrho_{F_0}(X \cap Y) \geq 1$ and $\varrho_{F_0}(X \cup Y) \geq 1$ imply

$$1 + 1 = \varrho_{F_0}(X) + \varrho_{F_0}(Y) \geq \varrho_{F_0}(X \cap Y) + \varrho_{F_0}(X \cup Y) \geq 1 + 1$$

and hence $\varrho_{F_0}(X \cap Y) = 1$ and $\varrho_{F_0}(X \cup Y) = 1$ follow, that is, both $X \cap Y$ and $X \cup Y$ are in $\mathcal{F}_0$. ∎

By Claim 4.1, Theorem 2.3 can indeed be applied to $\mathcal{F}_0$. Our goal is to show how the requested Subroutine **(A)** for the general algorithm can be implemented for this concrete kernel system $\mathcal{F}_0$.

4.1 Simple approach

We compute the optimal y^* by applying the first phase of the algorithm described in Section 2 for general kernel systems to the specific kernel system $\mathcal{F}_0$ defined by (6). To this end, we show how Subroutine **(A)** required for the general algorithm can be implemented in the special case of $\mathcal{F} := \mathcal{F}_0$. Recall that Subroutine **(A)** in Section 3.1 decides for an input cost-function c' whether there exists a c'-positive member of $\mathcal{F}$ (that is one for which every entering edge has positive c'-cost), and if there is one, the subroutine must compute such a member which is inclusionwise minimal.

To construct such a subroutine, we describe Subroutine $\mathbf{(A)}_f$ that decides for any given element $f = uv$ of the input arborescence F_0 whether there is a c'-positive member of $\mathcal{F}_0$ for which f is the only edge in F_0 entering it, and if there is one, the subroutine computes the (unique) smallest member of $\mathcal{F}_0$ with this property.

Let c'_f denote the cost-function arising from c' in such a way that the c'-cost of each element of $F_0 - f$ is reduced to 0, that is,

$$c'_f(e) := \begin{cases} c'(e) & \text{if} \quad e = f, \\ 0 & \text{if} \quad e \in F_0 - f, \\ c'(e) & \text{if} \quad e \in A - F_0. \end{cases} \tag{7}$$

Let $A_f := \{a \in A : c'_f(a) = 0\}$ and let $D_f = (V, A_f)$. Subroutine $\mathbf{(A)}_f$ computes the set S_f of those nodes from which v is reachable in D_f. (This can be done, for example, by a BFS in linear time). If r_0 is in S_f, then the subroutine terminates with the conclusion that $\mathcal{F}_0$ has no c'-positive member for which f is the unique element of F_0 entering it. If r_0 is not in S_f, then the subroutine outputs S_f as the requested minimal member of $\mathcal{F}_0$. The correctness of the subroutine follows from the following claim.

Claim 4.2. *If the head v of edge $f \in F_0$ is reachable from r_0 (in D_f), then $\mathcal{F}_0$ admits no c'-positive member entered by f. If v is not reachable from r_0, then S_f is the (unique) minimal c'-positive member of $\mathcal{F}_0$ entered by f.*

Proof. If v is reachable from r_0, then all the nodes of D_f is reachable from r_0 since the c'_f-cost of each edges in $F_0 - f$ is 0. But in this case, for each subset $Z \subseteq V - r_0$, there is an entering edge with 0 c'_f-cost. If v is not reachable from r_0, then it is not reachable from u either and hence f enters S_f. Since the c'_f-cost of every other element of F_0 is 0 and S_f is c'_f-positive, f is the unique edge in F_0 that enters S_f and hence $S_f \in \mathcal{F}_0$. Moreover, there is a dipath of 0 c'_f-cost from every node in S_f to v, and hence S_f cannot have a proper c'_f-positive subset containing v. ∎

By applying separately Subroutine $(\mathbf{A})_f$ to each element f of F_0, the requested subroutine $(\mathbf{A})$ for the special kernel system $\mathcal{F}_0$ is indeed available.

The algorithm above is strongly polynomial and conceptually simple. One may, however, feel that it is not particularly efficient since the computation of the subsequent positive dual variables $y^*(Z)$ needs the application of Subroutine $(\mathbf{A})_f$ to every member f of F_0. This disadvantage is overcome by the following more compact approach.

4.2 More efficient algorithm

Consider the elements of the input arborescence F_0 in a special order $f_1, \dots, f_{n-1}$ having Property $(\mathbf{O})$: edge $f \in F_0$ precedes edge $e \in F_0$ in this ordering if F_0 admits a dipath from the head of $e \in F_0$ to the tail of f. Note that such an ordering can easily be computed in linear time: consider a building up of F_0 that starts from r_0 and adds new edges one by one in such a way that each newly added edge leaves the already constructed sub-arborescence. (This building up procedure of F_0 is also realizable in linear time.) It follows immediately that by reversing the building up ordering, we obtain $f_1, \dots, f_{n-1}$ with the Property $(\mathbf{O})$. We remark that Property $(\mathbf{O})$ will be used only in the proof of Lemma 4.3 concerning the correctness of the algorithm.

The algorithm considers the elements of F_0 one-by-one in the given ordering. It consists of $n-1$ subsequent segments where Segment j is concerned with edge f_j. During one segment, we compute a (possibly empty) sequence of subsets (forming an increasing chain) for which f_j is the single edge in F_0 entering these sets and for which their dual variable will be positive. According to the rule of Phase 1 of the general algorithm, when such a dual variable is defined, we reduce the current cost-function, denoted by c'. It should be emphasized that it may be the case that in Segment j no new set gets a positive dual variable and it is also possible that more than one such set gets positive dual variable.

Consider now Segment j concerning edge f_j and let c' denote the current cost-function. At the beginning of Segment 1, $c' := c$. With the help of Subroutine $(\mathbf{A})_{f_j}$ described above, decide if there is a c'-positive set for which f_j is the single member of F_0 entering this set. If no such a set exists, Segment j terminates. In this case, if $j = n-1$, then the whole algorithm terminates, while if $j < n-1$, then we turn to Segment $j+1$ concerning edge f_{j+1}.

Suppose now that the c'-positive set in question does exist and consider the smallest such set Z' cumputed by Subroutine $(\mathbf{A})_{f_j}$. By copying the (first phase of the) general algorithm for kernel systems, let

$$y^*(Z') := \min\{c'(f) : f \in A, f \text{ enters } Z'\}$$

and revise c' as follows.

$$c'(f) := \begin{cases} c'(f) & \text{if} \quad f \text{ does not enter } Z' \\ c'(f) - y^*(Z') & \text{if} \quad f \text{ enters } Z'. \end{cases} \tag{8}$$

(Note that the present notation harmonizes with the one used in the description of the general algorithm, with the only difference that here we use Z' in place of Z_i used in the description of the general algorithm, and, also, we use here c' in place of c'_i or c'_{i+1}.)

If $c'(f_j) = 0$ holds for c' defined in (8), then Segment j terminates. In this case, if $j = n - 1$, then the whole algorithm terminates while if $j < n - 1$, then we turn to Segment $j + 1$ concerning edge f_{j+1}. If $c'(f_j) > 0$, continue the run of Segment j with the same f_j and with cost-function c' revised in (8).

Our final goal is to verify the correctness of the algorithm, that is, to prove that the procedure outputs an optimal dual solution. The algorithm in Section 3.1 for computing an optimal dual solution to a general kernel system was generic in the sense that the currently chosen set Z_i was required to be an inclusionwise minimal c_i-positive member of $\mathcal{F}$ but, within these requirements, it did not matter which of these sets was actually selected to be Z_i.

In order to prove the correctness of the present algorithm concerning $\mathcal{F}_0$, we show that the choice of Z' may be interpreted as a specific choice of Z_i occurring in the general algorithm. This is exactly the content of the next lemma and hence the lemma, along with the correctness of the general algorithm for kernel systems, imply the correctness of the present algorithm.

Lemma 4.3. *At the moment when the algorithm finds a minimal c'-positive set Z' for which f_j is the unique element of F_0 entering Z', the set Z' is an inclusionwise minimal c'-positive member $\mathcal{F}_0$.*

Proof. Suppose, indirectly, that at the moment of finding Z', $\mathcal{F}_0$ has a c'-positive member Z'' for which $Z'' \subset Z'$. Let f_h denote the unique edge in F_0 entering Z''. The minimal choice of Z' shows that $h \neq j$, implying that f_h must lie completely in Z'. Therefore, the unique dipath in F_0 from the root r_0 to f_h must go through f_j and hence Property **(O)** implies that $h < j$, that is, Segment h preceded Segment j. At the termination of Segment h, there was an edge $a \in A$ entering Z'' whose current cost at that moment was 0. But then $c'(a) = 0$, in a contradiction with the indirect assumption that Z'' was w-positive at the moment of defining Z'. ∎

THEOREM 4.4. *The complexity of the algorithm above is $O(mn)$ where m and n denote the number of edges and nodes of D, respectively.*

Proof. The algorithm consists of $|F_0| = n - 1$ segments. At the beginning of each segment, we compute the set S_{f_j} of nodes from which the head of f_j is reachable in D_{f_j}. By a BFS, this can be done in $O(m)$ time and hence these sets can be computed in $O(mn)$ time.

The other steps of the segments seek for computing the positive dual-variables. Instead of estimating the number of these steps separately segment-wise, we provide an upper bound $O(n)$ for the total number of sets getting positive dual variables. Indeed, the second half of Theorem 2.3 implies that the family of sets with positive dual variables is laminar and hence its cardinality is at most $2n$. Summing up, the total number of steps is indeed $O(nm)$. ∎

5 Solution to the inverse arborescence problem

Let us turn to the inverse arborescence problem in which we want to make a given (so-called input) arborescence F_0 to be a cheapest one by revising a given cost-function w_0 in a minimal way. That is, tha goal is to find a new cost-function $w : A \to \mathbf{R}$ for which F_0 is a cheapest arborescence and the deviation $|w - w_0| := \sum(|w(a) - w_0(a)| : a \in A)$ of w from w_0 is as small as possible. This minimum will be denoted by μ^*.

An essential and natural observation is that the set of cost-functions w for which F_0 is a cheapest arborescence forms a polyhedron. This implies that the minimum of deviation does indeed exist and the the set of deviation-minimizer cost-functions (that is, the ones with deviation μ^*) is also a polyhedron.

5.1 Min-max theorem and algorithm

As a preparation, we need the following easy observation.

Proposition 5.1. *Let $D' = (V, F_0 \cup L')$ be a digraph in which F_0 is a spanning arborescence of root r_0 which is disjoint from L'. Let $\mathcal{F}_0 := \{Z \subseteq V - r_0 : \varrho_{F_0}(Z) = 1\}$. Then D' is rooted 2-edge-connected if and only if L' covers $\mathcal{F}_0$. Moreover, if L' is an inclusionwise minimal set of edges covering $\mathcal{F}_0$, then $\varrho_{L'}(v) = 1$ (or equivalently $\varrho_{D'}(v) = 2$) for each node $v \in V - r_0$.*

Proof. The first statement is an immediate consequence of the definition of rooted 2-edge-connectivity. To see the second one, observe that if Z_1 and Z_2 are two intersecting subsets of $V - r_0$ for which $\varrho_{D'}(Z_i) = 2$, then $2 + 2 = \varrho_{D'}(Z_1) + \varrho_{D'}(Z_2) \geq \varrho_{D'}(Z_1 \cap Z_2) + \varrho_{D'}(Z_1 \cup Z_2) \geq 2 + 2$ from which $\varrho_{D'}(Z_1 \cap Z_2) = 2$ follows. This implies that every node $v \in V - r_0$ is contained in a unique smallest subset Z_v for which $\varrho_{D'}(Z_v) = 2$. This and the minimality of L' imply that each edge $f \in L'$ entering v enters Z_v. But then $2 = \varrho_{D'}(Z_v) = \varrho_{F_0}(Z_v) + \varrho_{L'}(v) \geq 1 + \varrho_{L'}(v)$, from which $\varrho_{L'}(v) \leq 1$. On the other hand, $\{v\} \in \mathcal{F}_0$ implies that $\varrho_{L'}(v) \geq 1$, and hence $\varrho_{L'}(v) = 1$ follows. ■

We call a cost-function $w : A \to \mathbf{R}$ **w_0-adequate** or just **adequate** if F_0 is a cheapest arborescence with respect to w and

$$w(f) \leq w_0(f) \text{ for each } f \in F_0 \text{ and } w(e) \geq w_0(e) \text{ for each } e \in A - F_0. \tag{9}$$

If, in addition, $w \geq 0$ and $w(e) = w_0(e)$ for each $e \in A - F_0$, then we say that w is **strongly adequate**.

THEOREM 5.2. *Let $w_0 \geq 0$ be a cost-function on the edge-set of digraph $D = (V, A)$ and let F_0 be a spanning r_0-arborescence of D. Let $\mathcal{F}_0 := \{Z \subseteq V - r_0 : \varrho_{F_0}(Z) = 1\}$. Then*

$$\mu^* := \min\{|w - w_0| : w \textit{ a cost-function for which } F_0 \textit{ is a cheapest arborescence}\} = \tag{10}$$

$$\max\{\widetilde{w}_0(F_0) - \widetilde{w}_0(L) : L \subseteq A,\ L \textit{ covers } \mathcal{F}_0\}. \tag{11}$$

Moreover, there exists a strongly adequate optimal solution $w = w^*$ to (10) *which, in addition, is integer-valued when w_0 is integer-valued.*

Proof. If w is a cost-function for which F_0 is a cheapest arborescence and its deviation $|w - w_0|$ is minimum, then w is obviously adequate. Therefore, in order to prove max $\leq$ min, it suffices to show that

$$|w - w_0| \geq \widetilde{w}_0(F_0) - \widetilde{w}_0(L) \tag{12}$$

holds for every covering $L \subseteq A$ of $\mathcal{F}_0$ and for every adequate cost-function w. As w_0 is non-negative, it suffices to prove (12) only when L is an inclusionwise minimal covering of $\mathcal{F}_0$.

Let w be an adequate cost-function and L an inclusionwise minimal covering of $\mathcal{F}_0$. For each $e \in L \cap F_0$, let e' be a new edge parallel to e, and let $w(e') := w(e)$. Let N' be the set of new edges, $L' := (L - F_0) \cup N'$, and $A^+ := A \cup N'$. Obviously, $|L'| = |L|$ and $\widetilde{w}(L') = \widetilde{w}(L)$. It also follows from these definitions that F_0 is a cheapest arborescence (with respect to the extended cost-function w on A^+) in the digraph (V, A^+), and hence F_0 is a cheapest arborescence in $D' := (V, F_0 \cup L')$, as well.

By Proposition 5.1, we have $\varrho_{F_0}(v) + \varrho_{L'}(v) = 2$ for each $v \in V - r_0$. This and Edmonds' theorem [5] on disjoint arborescences imply that D' is the union of two disjoint spanning r_0-arborescences F_1 and F_2. Since F_0 is a cheapest arborescence in D' with respect to w, we have

$$\widetilde{w}(L) + \widetilde{w}(F_0) = \widetilde{w}(L') + \widetilde{w}(F_0) = \widetilde{w}(L' \cup F_0) = \widetilde{w}(F_1) + \widetilde{w}(F_2) \geq \widetilde{w}(F_0) + \widetilde{w}(F_0),$$

from which $\widetilde{w}(L) \geq \widetilde{w}(F_0)$ follows. This and (9) imply

$$\begin{aligned} |w - w_0| &\geq [\widetilde{w}_0(F_0) - \widetilde{w}(F_0)] + [\widetilde{w}(L) - \widetilde{w}_0(L)] \\ &= [\widetilde{w}_0(F_0) - \widetilde{w}_0(L)] + [\widetilde{w}(L) - \widetilde{w}(F_0)] \;\; \geq \;\; \widetilde{w}_0(F_0) - \widetilde{w}_0(L), \end{aligned} \tag{13}$$

as required for (12), and hence max $\leq$ min follows. (One may feel inadequate to use a non-trivial theorem in a proof of the 'trivial' inequality max $\leq$ min. The application of Edmonds' theorem, however, can be avoided with the help of a slightly more technical argument, see Remark 5.3.)

To see the reverse direction max $\geq$ min, it suffices to prove that there is an adequate cost-function w^* (integer-valued when w_0 is so) and a covering L^* of $\mathcal{F}_0$ for which (13) holds with equality. Actually, we shall show that w^* can be chosen strongly adequate, in which case the first inequality in (13) is met automatically by equality, while we have equality in the second inequality of (13) precisely if

$$\widetilde{w}^*(L^*) = \widetilde{w}^*(F_0). \tag{14}$$

Apply Theorem 2.3 to the special kernel system $\mathcal{F}_0$ in place of $\mathcal{F}$ and to the cost-function $c := w_0$. Let L^* denote the optimal primal solution, that is, L^* is a cheapest covering of $\mathcal{F}_0$, and let y^* be the optimal dual solution in (3). By the theorem, $\widetilde{y}^*(\mathcal{F}_0) = w_0(L^*)$. Define w^* as follows.

$$w^*(a) := \begin{cases} w_0(a) & \text{if } \ a \in A - F_0 \\ \sum[y^*(Z) : a \text{ enters } Z] & \text{if } \ a \in F_0. \end{cases} \tag{15}$$

This w^* is non-negative and $w^*(a) = w_0(a)$ for each $a \in A - F_0$. Furthermore, y^* is clearly w^*-feasible (for the definition of feasibility, see (1)) and F_0 consists of tight edges with respect to w^*. By applying Corollary 2.2 to w^* in place of c, we obtain that F_0 is a cheapest arborescence with respect to w^*. Therefore w^* is strongly w_0-adequate for which (by Theorem 2.3) $\widetilde{w}^*(F_0) = \widetilde{y}^*(\mathcal{F}_0) = \widetilde{w}^*(L^*)$ holds. When w_0 is integer-valued, the optimal dual solution y^* can also be chosen integer-valued by Theorem 2.3, and hence w^* defined in (15) is also integer-valued. ■

Remark 5.3. In the proof of inequality max $\leq$ min in Theorem 5.2, we relied on Edmonds' deep theorem. This, however, can be avoided by applying the easier polyhedral description of the convex hull of spanning r_0-arborescences: $\{x : x \geq 0, \varrho_x(Z) \geq 1$ for every non-empty subset $Z \subseteq V - r_0$, and $\varrho_x(v) = 1$ for every node $v \in V - r_0\}$. To see this alternative, recall that the digraph $D' := (V, A')$ (where $A' = F_0 \cup L'$) was shown to be rooted 2-edge-connected for which $\varrho_{D'}(v) = 2$ for each $v \in V - r_0$. Let $z := \chi(A')/2$ be the identically 1/2 vector on A'. By using the polyhedral description of the spanning r_0-arborescences, we obtain that z can be expressed as the convex combination of spanning arborescences: $z = \sum[\lambda_i \chi(F_i) : i = 1, \dots, q]$ where $\lambda_i > 0$ for each i and $\sum \lambda_i = 1$. Then we have $\widetilde{w}(F_0) + \widetilde{w}(L') = \widetilde{w}(A') = 2wz = 2\sum[\lambda_i \widetilde{w}(F_i) : i = 1, \dots, q] \geq 2\sum[\lambda_i \widetilde{w}(F_0) : i = 1, \dots, q] = 2\widetilde{w}(F_0)$, from which $\widetilde{w}(F_0) \leq \widetilde{w}(L') = \widetilde{w}_0(L)$, as required for max $\leq$ min.

This proof is technically a bit more complicated than the one using Edmonds' theorem. However, it may be used in other inverse optimization problems where the analogue of Edmonds' theorem (that is, the discrete Carathéodory property of arborescences) does not hold while the polyhedral description of the objects in question is available. •

An immediate consequence of the proof above is the following.

Corollary 5.4. *Let y^* be an optimal dual solution to the primal problem of finding a minimal w_0-cost covering of kernel system $\mathcal{F}_0 := \{Z \subseteq V - r_0 : \varrho_{F_0} = 1\}$. Then the cost-function w^* defined by* (15) *is an optimal solution to the inverse arborescence problem.* ■

Algorithm Corollary 5.4 implies that the algorithm developed in Section 4 for the special kernel system $\mathcal{F}_0 := \{Z \subseteq V - r_0 : \ \varrho_{F_0}(Z) = 1\}$, when applied to $c := w_0$, computes in $O(mn)$ time both the optimal primal solution L^* and the optimal dual solution y^* in (3). We proved that w^*, as defined in (15) by y^*, is an optimal (strongly adequate) primal solution in (10) to the inverse arborescence problem while L^* is an optimal dual solution in (11).

Remark 5.5. The statement in Theorem 5.2 that the wanted optimal cost-function w^* may be chosen in such a way $w^*(e) = w_0(e)$ holds for each edge $e \in A - F_0$ was proved already by Hu and Liu [11]. It is interesting to note that the corresponding property does not hold in the undirected counterpart of the inverse arborescence problem where the goal is to make an input spanning tree F_0 of an undirected graph G to be a cheapest tree. To see this, let G be a triangle with edges f, g, h whose costs are $1, 1, 0$, and let $F_0 = \{f, g\}$ be a spanning tree. If we are not allowed to change the costs outside F_0, then the cost of both f and g must be reduced to 0 to make F_0 a cheapest tree, and hence the total change (deviation) is 2. On the other hand, if we increase the cost of h by 1, then F_0 becomes a cheapest tree, that is, the deviation in this case is only 1. The same example shows that the property does not hold for matroids either where an input basis B_0 is to be made a cheapest basis.

References

[1] F. Bock, *An algorithm to construct a minimum directed spanning tree in a directed network*, in: Developments of Operations Research, Vol. 1 (Proceedings of the Third Annual Israel Conference on Operations Research, Tel Aviv, 1969, B. Avi-Itzhak, ed.) Gordon and Breach, New York, 1971, 29–24.

[2] M.-C. Cai and Y.-J. Li, *Inverse matroid intersection problem,* Mathematical Methods of Operations Research, Vol 45. No. 2, (1997), pp. 235–243.

[3] M.-C. Cai, *Inverse problems of matroid intersections*, J. Combinatorial Optimization, Vol 3. No. 4, (1999) pp. 465–474.

[4] Y.-J. Chu and T.-H. Liu, *On the shortest arborescence of a directed graph*, Scientia Sinica (Peking) Vol. 14 (1965) pp. 1397–1400.

[5] J. Edmonds, *Edge-disjoint branchings,* in: Combinatorial Algorithms (B. Rustin, ed.), Acad. Press, New York, (1973), 91–96.

[6] J. Edmonds, *Matroid intersection*, Annals of Discrete Math. 4, (1979) 39-49.

[7] A. Frank, *Kernel systems of directed graphs*, Acta Scientiarum Mathematicarum, (Szeged), 41, 1-2 (1979) 63–76.

[8] A. Frank, *Increasing the rooted connectivity of a digraph by one*, in: Connectivity Augmentation of Networks: Structures and Algorithms, Mathematical Programming, (ed. A. Frank) Ser. B, Vol. 84, No. 3 (1999), pp. 565-576.

[9] A. Frank, Connections in Combinatorial Optimization, Oxford University Press, 2011 (ISBN 978-0-19-920527-1). Oxford Lecture Series in Mathematics and its Applications, 38.

[10] D.R. Fulkerson, *Packing rooted directed cuts in a weighted directed graph*, Math. Programming 6 (1974) 1–13.

[11] Z. Hu and Z. Liu, *A strongly polynomial algorithm for the inverse shortest arborescence problem*, Discrete Applied Mathematics, 82 (1998) 135–154.

[12] A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency, Springer, 2003. Vol 24. of the series Algorithms and Combinatorics.

[13] É. Tardos, *A strongly polynomial minimum cost circulation algorithm*, Combinatorica, 5, (1985) pp. 247-255.

[14] Z. Zhang, S. Li, H.-J. Lai, and D.-Z. Du, *Algorithms for the partial inverse matroid problem in which weights can only be increased.* J. Global Optimization, 65 (2016) 801–811.