



A simple algorithm and min–max formula for the inverse arborescence problem[☆]

András Frank^{*}, Gergely Hajdu

MTA-ELTE Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Pázmány P. s. 1/c, Budapest, H-1117, Hungary



ARTICLE INFO

Article history:

Received 5 October 2020

Received in revised form 10 February 2021

Accepted 12 February 2021

Available online 2 March 2021

Keywords:

Inverse combinatorial optimization

Arborescence

Min–max theorem

Simple algorithm

ABSTRACT

In 1998, Hu and Liu developed a strongly polynomial algorithm for solving the inverse arborescence problem that aims at minimally modifying a given cost-function on the edge-set of a digraph D so that an input spanning arborescence of D becomes a cheapest one. In this note, we develop a conceptually simpler algorithm along with a new min–max formula for the minimum modification of the cost-function. The approach is based on a link to a min–max theorem and a simple (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.

© 2021 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

1. Introduction

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. Let $D = (V, A)$ be a loopless digraph with n nodes and m edges, and let $r_0 \in V$ be a specified node called the root-node of D . By an r_0 -**arborescence** F of D , we mean a subgraph of D which is an arborescence of root r_0 containing all the nodes of D (that is, F is a spanning arborescence of D with root r_0). Without loss of generality, we shall assume that no edge of D enters r_0 . In 1965, Chu and Liu [5] developed a simple strongly polynomial algorithm for computing a cheapest (= minimum cost) r_0 -arborescence of D with respect to a given cost-function on A . In 1967, Edmonds [7] described an algorithm and a min–max theorem for the closely related problem concerning maximum weight branchings.

In the **inverse arborescence problem**, we are given a spanning arborescence F_0 of D with root r_0 (that is, an r_0 -arborescence) and a cost-function $w_0 : A \rightarrow \mathbf{R}_+$. The goal is to modify w_0 so that F_0 becomes a cheapest r_0 -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 [16] 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 which is based on a new min–max formula (Theorem 5.2) for the minimum deviation μ^* of the revised cost-function w for which the input r_0 -arborescence is a cheapest one of D . The approach is based on a link to a paper by the

[☆] The research was partially supported by the National Research, Development and Innovation Fund of Hungary (FK-18) – No. NKFI-128673.

^{*} Corresponding author.

E-mail addresses: frank@cs.elte.hu (A. Frank), hgergely91@gmail.com (G. Hajdu).

first author [10] from 1979 that includes a particularly simple algorithm for solving a natural extension of the cheapest r_0 -arborescence problem. Not only the algorithm is simple but so is the proof of its correctness.

Cai and Li [3,4] 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 [18]) can be applied. Since r_0 -arborescences form the common bases of two special matroids, the solution of Cai and Li can be specialized to r_0 -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 [9] did not make superfluous the simpler and more efficient direct algorithm of Chu and Liu [5] concerning cheapest r_0 -arborescences. It should, however, be noted that the minimum 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 $c : A \rightarrow \mathbf{R}_+$ is a given cost-function.

There is a rich history of inverse combinatorial optimization problems. One of the earliest papers is due to Burton and Toint [2] and appeared in 1992. This is about inverse shortest paths problems and was strongly motivated by practical applications (coming from geophysical sciences). Since then a great number of papers appeared which are wonderfully overviewed in two excellent survey papers due to Heuberger [15] and to Demange and Monnot [6].

In a recent work, Frank and Murota [13] developed a general min–max formula for a wide class of inverse combinatorial optimization problems. For example, their framework covers the generalization of the present problem for the case when upper bounds are imposed on the change of w_0 , and/or the weighted ℓ_1 - or ℓ_2 -norm or some other norms are used. It should, however, be emphasized that the proofs in [13] are not algorithmic and it remains a major challenge for further investigations to develop polynomial algorithms.

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** 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 **covers** $\mathcal{F}$ if L enters each member of $\mathcal{F}$. For two elements s and t , a set Z is called a **$t\bar{s}$ -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), or that D includes an r_0 -arborescence. An easy and well-known property is that an inclusionwise minimal root-connected subgraph of D is an r_0 -arborescence. More generally, D is **rooted k -edge-connected** if $\varrho_D(Z) \geq k$ holds for every non-empty subset $Z \subseteq V - r_0$. It is also a well-known property that in an inclusionwise minimal rooted k -edge-connected digraph the in-degree of every node $v \in V - r_0$ is k .

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

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** [10] if $\varrho_D(Z) > 0$ for each $Z \in \mathcal{F}$. All other notions and notation can be found in [12].

2. Arborescences and kernel systems

2.1. Cheapest r_0 -arborescences

Let $D = (V, A)$ be a root-connected digraph with a root-node r_0 and let $c : A \rightarrow \mathbf{R}_+$ 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 $y : \mathcal{F} \rightarrow \mathbf{R}$ defined on a set-system $\mathcal{F} \subseteq 2^V$ is **c -feasible** if $y \geq 0$ and

$$\sum [y(Z) : Z \in \mathcal{F}, Z \text{ is entered by } a] \leq c(a) \text{ for every edge } a \in A. \quad (2.1)$$

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 r_0 -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 [14] proved the following min–max formula.

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

$$\max \left\{ \sum [y(Z) : Z \subseteq V - r_0] : y \text{ } c\text{-feasible} \right\}. \quad (2.2)$$

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 [14] 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 an r_0 -arborescence of D for which the following optimality criteria hold:*

F_0 consists of tight edges,

$y^(Z) > 0$ implies $\varrho_{F_0}(Z) = 1$.*

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

The book of Schrijver [17] 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 [10] extended the problem of cheapest r_0 -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 covers $\mathcal{F}$). A special case is when $\mathcal{F}$ consists 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 r_0 -arborescences.

Another special case is when $\mathcal{F}$ consists of all the $t\bar{s}$ -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 r_0 -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 $y : \mathcal{F} \rightarrow \mathbf{R}$ for which

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

is as large as possible. In [10], the algorithm of Fulkerson [14] was extended in a natural way to general kernel systems. In addition, [10] 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 ([10]). *Let $\mathcal{F}$ be an intersecting set-system on the node-set of digraph $D = (V, A)$ for which A covers $\mathcal{F}$, and let $c : A \rightarrow \mathbf{R}_+$ be a cost-function. Then*

$$\min\{\tilde{c}(L) : L \subseteq A, L \text{ covers } \mathcal{F}\} = \max\{\tilde{y}(\mathcal{F}) : y \text{ } c\text{-feasible}\}. \quad (2.3)$$

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**:

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

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

then y is an optimal dual solution and L is a c -cheapest cover 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 cover $L^* \subseteq A$ 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 r_0 -arborescences of D , it follows in this case that the cheapest covers of $\mathcal{F}$ are the cheapest r_0 -arborescences, and hence the general Theorem 2.3 reduces to Theorem 2.1.

We note that [10] actually includes a min–max theorem (concerning a cheapest cover 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 [10] is not constructive anyway, unlike the simple algorithmic proof of Theorem 2.3.

We also remark that the min–max theorem and the two-phase greedy algorithm in [10] was generalized in [11] for the problem of finding a cheapest cover of an intersecting bi-set system. See also the book of Frank [12], page 395.

3. Algorithm for kernel systems

In [10], 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 [14] for the dual of the cheapest r_0 -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 cover L of $\mathcal{F}$. This algorithm will be needed to compute the optimal dual structure appearing in the min–max formula in Theorem 5.2 concerning the minimum deviation of the desired 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 [10]. Given a cost-function $c' : A \rightarrow \mathbf{R}_+$, we call a subset $Z \subset V$ **c' -positive** if the c' -cost of every edge entering Z is positive. Let $\mathcal{F}$ be an intersecting family of subsets of V for which $\varrho_A(Z) \geq 1$ holds for every $Z \in \mathcal{F}$.

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 cover 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 sequence of non-negative real-valued cost-functions on A and a sequence $Z_1, Z_2, Z_3, \dots$ 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 $i = 1$, let Z_1 be 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.

$$c_2(f) := \begin{cases} c_1(f) & \text{if } f \text{ does not enter } Z_1 \\ c_1(f) - y^*(Z_1) & \text{if } f \text{ enters } Z_1. \end{cases} \quad (3.1)$$

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 an 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.

$$c_{i+1}(f) := \begin{cases} c_i(f) & \text{if } f \text{ does not enter } Z_i \\ c_i(f) - y^*(Z_i) & \text{if } f \text{ enters } Z_i. \end{cases} \quad (3.2)$$

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 (K): *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 $\mathcal{F}$.*

In Section 4, we shall describe how Subroutine (K) can efficiently be realized for a special intersecting family $\mathcal{F}_0$ that will be defined in (4.1) and used for the algorithmic solution of the inverse arborescence problem.

It was pointed out in [10] 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 [10] for computing a cheapest cover 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. (Indeed, a simple induction on i shows that, at the moment when $y^*(Z_i)$ becomes positive and c_{i+1} is defined, $c_{i+1}(f) = c(f) - \sum [y^*(Z) : Z \text{ is entered by } f]$ holds, and hence if $c_i(f) > 0 = c_{i+1}(f)$, then f is tight.)

In order to construct a cheapest cover 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 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 cover of $\mathcal{F}$ obtained by the termination of Phase 2. The following lemma from [10] 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 ([10]). *The dual solution y^* provided by Phase 1 and the cover 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 an r_0 -arborescence of $D = (V, A)$. The key idea of our approach to the inverse arborescence 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,

$$\mathcal{F}_0 := \{Z \subseteq V - r_0 : \varrho_{F_0}(Z) = 1\}. \quad (4.1)$$

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 **(K)** 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 (4.1). To this end, we show how Subroutine **(K)** required for the general algorithm can be implemented in the special case of $\mathcal{F} := \mathcal{F}_0$. Recall that Subroutine **(K)** 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 determine such a member which is inclusionwise minimal.

To construct such a subroutine, we describe Subroutine **(K)_f** that decides for any given element $f = uv$ of the input r_0 -arborescence F_0 whether there is a c' -positive member of $\mathcal{F}_0^f := \{Z \in \mathcal{F}_0 : f \text{ enters } Z\}$, and if there is one, the subroutine computes the (unique) smallest member of $\mathcal{F}_0^f$.

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,

$$c'_f(e) := \begin{cases} c'(e) & \text{if } e = f, \\ 0 & \text{if } e \in F_0 - f, \\ c'(e) & \text{if } e \in A - F_0. \end{cases} \quad (4.2)$$

Let $A_f := \{a \in A : c'_f(a) = 0\}$ and let $D_f = (V, A_f)$. Subroutine **(K)_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 $\mathcal{F}_0^f$ has no c' -positive member. 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 $\mathcal{F}_0^f$.*

Proof. If v is reachable from r_0 , then all the nodes of D_f are reachable from r_0 since the c'_f -cost of each edge 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. Since the c' -cost and the c'_f -cost of edges entering a member Z of $\mathcal{F}_0^f$ coincide, $\mathcal{F}_0^f$ has no c' -positive member.

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 Subroutine **(K)_f** separately to each element f of F_0 , the requested Subroutine **(K)** 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 **(K)_f** to every member f of F_0 . This disadvantage is overcome by the following more compact approach.

4.2. More efficient algorithm

The algorithm in Section 4.1 provided a realization of Subroutine **(K)** by developing Subroutine **(K)_f** and applying it separately to each element f of the input r_0 -arborescence F_0 . Here we present a more efficient realization of Subroutine **(K)** by considering the elements of F_0 in a specific order $f_1, \dots, f_{n-1}$ having Property **(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 **(O)**. We remark that Property **(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 stages where Stage j is concerned with edge f_j . During one stage, 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 Stage j no new set receives a positive dual variable and it is also possible that more than one such set receives positive dual variable.

At the beginning of Stage 1, $c' := c$. Consider now Stage j concerning edge f_j and let c' denote the current cost-function. With the help of Subroutine $(\mathbf{K})_{f_j}$ 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, Stage j terminates. In this case, if $j = n - 1$, then the whole algorithm terminates, while if $j < n - 1$, then we turn to Stage $j + 1$ concerning edge f_{j+1} .

Suppose now that the c' -positive set in question does exist and consider the smallest such set Z' computed by Subroutine $(\mathbf{K})_{f_j}$. 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.

$$c'(f) := \begin{cases} c'(f) & \text{if } f \text{ does not enter } Z' \\ c'(f) - y^*(Z') & \text{if } f \text{ enters } Z'. \end{cases} \quad (4.3)$$

(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 (4.3), then Stage j terminates. In this case, if $j = n - 1$, then the whole algorithm terminates while if $j < n - 1$, then we turn to Stage $j + 1$ concerning edge f_{j+1} . If $c'(f_j) > 0$, continue the run of Stage j with the same f_j and with cost-function c' revised in (4.3).

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 of $\mathcal{F}_0$.*

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, Stage h preceded Stage j . At the termination of Stage 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 $|F_0| = n - 1$ stages. At the beginning of each stage, 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 stages are used to determine the positive dual-variables. Instead of estimating the number of these steps separately stage-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 $O(mn)$. ■

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) r_0 -arborescence F_0 to be a cheapest one by revising a given non-negative cost-function w_0 in a minimal way. That is, the goal is to find a new cost-function $w : A \rightarrow \mathbf{R}$ for which F_0 is a cheapest r_0 -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 r_0 -arborescence forms a polyhedron. This implies that the minimum of deviation does indeed exist and 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 an r_0 -arborescence of D' 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 $w : A \rightarrow \mathbf{R}$ w_0 -adequate or just **adequate** if F_0 is a cheapest r_0 -arborescence with respect to w and

$$\begin{cases} w(f) \leq w_0(f) & \text{for each } f \in F_0 \\ w(e) \geq w_0(e) & \text{for each } e \in A - F_0. \end{cases} \quad (5.1)$$

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**. A major new result of the present work is the following min-max formula, which serves as the structural basis of the new algorithm for the inverse arborescence problem.

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 an r_0 -arborescence of D . Let $\mathcal{F}_0 := \{Z \subseteq V - r_0 : \varrho_{F_0}(Z) = 1\}$. Then*

$$\begin{aligned} \mu^* &:= \min\{|w - w_0| : w \text{ a cost-function for which } F_0 \text{ is a cheapest } r_0 - \text{arborescence}\} \\ &= \max\{\tilde{w}_0(F_0) - \tilde{w}_0(L) : L \subseteq A, L \text{ covers } \mathcal{F}_0\}. \end{aligned} \quad (5.2)$$

Moreover, there exists a strongly adequate primal optimal solution $w = w^*$ to (5.2) 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 r_0 -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

$$|w - w_0| \geq \tilde{w}_0(F_0) - \tilde{w}_0(L) \quad (5.3)$$

holds for every cover $L \subseteq A$ of $\mathcal{F}_0$ and for every adequate cost-function w . As w_0 is non-negative, it suffices to prove (5.3) only when L is an inclusionwise minimal cover of $\mathcal{F}_0$.

Let w be an adequate cost-function and L an inclusionwise minimal cover 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 $\tilde{w}(L') = \tilde{w}(L)$. It also follows from these definitions that F_0 is a cheapest r_0 -arborescence (with respect to the extended cost-function w on A^+) in the digraph (V, A^+) , and hence F_0 is a cheapest r_0 -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 [8] on disjoint r_0 -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 r_0 -arborescence in D' with respect to w , we have

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

from which

$$\tilde{w}(L) \geq \tilde{w}(F_0) \quad (5.4)$$

follows. Since w is adequate, for $\bar{F}_0 := A - F_0$, we have:

$$\begin{aligned}\tilde{w}(\bar{F}_0) - \tilde{w}_0(\bar{F}_0) &= \\ &= [\tilde{w}(L) - \tilde{w}_0(L)] + [\tilde{w}(\bar{F}_0 - L) - \tilde{w}_0(\bar{F}_0 - L)] - [\tilde{w}(F_0 \cap L) - \tilde{w}_0(F_0 \cap L)] \\ &\geq \tilde{w}(L) - \tilde{w}_0(L) + 0 + 0.\end{aligned}\tag{5.5}$$

This and (5.4) imply:

$$\begin{aligned}|w - w_0| &= \\ &= [\tilde{w}_0(F_0) - \tilde{w}(F_0)] + [\tilde{w}(\bar{F}_0) - \tilde{w}_0(\bar{F}_0)] \\ &\geq [\tilde{w}_0(F_0) - \tilde{w}(F_0)] + [\tilde{w}(L) - \tilde{w}_0(L)] \\ &= [\tilde{w}_0(F_0) - \tilde{w}_0(L)] + [\tilde{w}(L) - \tilde{w}_0(F_0)] \\ &\geq \tilde{w}_0(F_0) - \tilde{w}_0(L),\end{aligned}\tag{5.6}$$

as required for (5.3), and hence $\max \leq \min$ follows.

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 cover L^* of $\mathcal{F}_0$ for which (5.3) holds with equality. By (5.6), we have equality in (5.3) precisely if both inequalities in (5.6) hold with equality. Furthermore, we have equality in (5.6) precisely if both (5.4) and (5.5) hold with equality, which are equivalent to the following optimality criteria.

$$(OC.1) \quad \tilde{w}^*(L^*) = \tilde{w}^*(F_0),$$

$$(OC.2) \quad \tilde{w}^*(\bar{F}_0 - L^*) = \tilde{w}_0^*(\bar{F}_0 - L^*),$$

$$(OC.3) \quad \tilde{w}^*(F_0 \cap L^*) = \tilde{w}_0^*(F_0 \cap L^*).$$

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 cover of $\mathcal{F}_0$, and let y^* be the optimal dual solution in (2.3). By the theorem, $\tilde{y}^*(\mathcal{F}_0) = \tilde{w}_0(L^*)$. Define w^* as follows.

$$w^*(a) := \begin{cases} w_0(a) & \text{if } a \in A - F_0 \\ \sum [y^*(Z) : a \text{ enters } Z] & \text{if } a \in F_0. \end{cases}\tag{5.7}$$

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 (2.1)) 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 r_0 -arborescence of D with respect to w^* . Therefore w^* is strongly w_0 -adequate and, by Theorem 2.3, we have $\tilde{w}^*(F_0) = \tilde{y}^*(\mathcal{F}_0) = \tilde{w}_0(L^*) = \tilde{w}^*(L^*)$, that is, (OC.1) holds.

The strong adequacy of w^* immediately implies (OC.2). For $a \in F_0$ we have $w^*(a) = \sum [y^*(Z) : Z \in \mathcal{F}_0, a \text{ enters } Z]$, while for $a \in L^*$ we have $w_0(a) = \sum [y^*(Z) : Z \in \mathcal{F}_0, a \text{ enters } Z]$. Therefore $w^*(a) = w_0(a)$ holds for $a \in F_0 \cap L^*$, and hence (OC.3) follows.

Finally, observe that if w_0 is integer-valued, then the optimal dual solution y^* can also be chosen integer-valued by Theorem 2.3, and hence w^* defined in (5.7) is also integer-valued. ■

Remark 5.3. One may feel that using Edmonds' deep theorem in the proof of the 'trivial' inequality $\max \leq \min$ is not really appropriate. Indeed, this can be avoided if we rely on the polyhedral description of the polytope of spanning r_0 -arborescences and use a slightly more technical argument. Namely, the convex hull of r_0 -arborescences of D is given by: $\{x : x \geq 0, \varrho_x(Z) \geq 1 \text{ for every non-empty subset } Z \subseteq V - r_0, \text{ and } \varrho_x(v) = 1 \text{ for every node } v \in V - r_0\}$. (This is an immediate consequence, for example, of Theorem 2.1.) 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 r_0 -arborescences of D , we obtain that z can be expressed as the convex combination of r_0 -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 $\tilde{w}(F_0) + \tilde{w}(L') = \tilde{w}(A') = 2wz = 2 \sum [\lambda_i \tilde{w}(F_i) : i = 1, \dots, q] \geq 2 \sum [\lambda_i \tilde{w}(F_0) : i = 1, \dots, q] = 2\tilde{w}(F_0)$, from which $\tilde{w}(F_0) \leq \tilde{w}(L') = \tilde{w}_0(L)$, 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 r_0 -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 cover of kernel system $\mathcal{F}_0 := \{Z \subseteq V - r_0 : \varrho_{F_0} = 1\}$. Then the cost-function w^* defined by (5.7) 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 (2.3). We proved that w^* , as defined in (5.7) by y^* , is an optimal (strongly adequate) primal solution in (5.2) to the inverse arborescence problem while L^* is an optimal dual solution in (5.2).

Remark 5.5. The statement in Theorem 5.2 that the desired 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 [16]. 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.

Remark 5.6. It was one of our major goals to develop a polynomial algorithm for the inverse arborescence problem which is definitely simpler than the existing algorithm of Hu and Liu [16]. This was the reason why we emphasized throughout the description above that the requested “background” algorithm concerning kernel systems is a conceptually simple two-phase greedy algorithm. On the other hand, the present approach may possibly be adapted to other inverse problems for which no polynomial algorithm were developed so far. In such a case, it is not an expectation that the analogous background algorithm should be particularly simple, the only requirement is that it must be polynomial.

Acknowledgements

The authors are grateful to the anonymous reviewers for their careful work and useful advices.

References

- [1] F. Bock, An algorithm to construct a minimum directed spanning tree in a directed network, in: B. Avi-Itzhak (Ed.), *Developments of Operations Research*, Vol. 1, Tel Aviv, 1969, in: *Proceedings of the Third Annual Israel Conference on Operations Research*, Gordon and Breach, New York, 1971, pp. 29–44.
- [2] D. Burton, Ph.L. Toint, On an instance of the inverse shortest paths problem, *Math. Program.* 53 (1992) 45–61.
- [3] M.-C. Cai, Inverse problems of matroid intersections, *J. Combin. Optim.* 3 (4) (1999) 465–474.
- [4] M.-C. Cai, Y.-J. Li, Inverse matroid intersection problem, *Math. Methods Oper. Res.* 45 (2) (1997) 235–243.
- [5] Y.-J. Chu, T.-H. Liu, On the shortest arborescence of a directed graph, *Sci. Sinica (Peking)* 14 (1965) 1397–1400.
- [6] M. Demange, J. Monnot, An introduction to inverse combinatorial problems, in: V.Th. Paschos (Ed.), *Paradigms of Combinatorial Optimization: Problems and New Approaches*, second ed., ISTE Ltd and John Wiley and Sons, 2014, pp. 547–586, Chapter 17.
- [7] J. Edmonds, Optimum branchings, *J. Res. Nat. Bur. Stand. B* 71 (1967) 233–240.
- [8] J. Edmonds, Edge-disjoint branchings, in: B. Rustin (Ed.), *Combinatorial Algorithms*, Acad. Press, New York, 1973, pp. 91–96.
- [9] J. Edmonds, Matroid intersection, *Ann. Discrete Math.* 4 (1979) 39–49.
- [10] A. Frank, Kernel systems of directed graphs, *Acta Sci. Math. (Szeged)* 41 (1–2) (1979) 63–76.
- [11] A. Frank, Increasing the rooted connectivity of a digraph by one, in: A. Frank (Ed.), *Connectivity Augmentation of Networks: Structures and Algorithms*, Mathematical Programming, Ser. B, Vol. 84, No. 3, 1999, pp. 565–576.
- [12] A. Frank, *Connections in Combinatorial Optimization*, in: *Oxford Lecture Series in Mathematics and its Applications*, vol. 38, Oxford University Press, ISBN: 978-0-19-920527-1, 2011.
- [13] A. Frank, K. Murota, A discrete convex min–max formula for box-TDI polyhedra, *Math. Oper. Res.* (2021) in press.
- [14] D.R. Fulkerson, Packing rooted directed cuts in a weighted directed graph, *Math. Program.* 6 (1974) 1–13.
- [15] C. Heuberger, Inverse combinatorial optimization: A survey on problems, methods, and results, *J. Combin. Optim.* 8 (2004) 329–361.
- [16] Z. Hu, Z. Liu, A strongly polynomial algorithm for the inverse shortest arborescence problem, *Discrete Appl. Math.* 82 (1998) 135–154.
- [17] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, in: *Algorithms and Combinatorics*, vol. 24, Springer, 2003.
- [18] É. Tardos, A strongly polynomial minimum cost circulation algorithm, *Combinatorica* 5 (1985) 247–255.