GLASNIK MATEMATIČKI
Vol. 55(75)(2020), 143 – 176

AN ALGEBRAIC FRAMEWORK FOR MULTI-OBJECTIVE AND ROBUST VARIANTS OF PATH PROBLEMS

Robert Manger
University of Zagreb, Croatia

Abstract. It is well known that various types of path problems in graphs can be treated together within a common algebraic framework. Thereby each type is characterized by a different "path algebra", i.e., a different instance of the same abstract algebraic structure. This paper demonstrates that the common algebraic framework, although originally intended for conventional problem variants, can be extended to cover multi-objective and robust variants. Thus the paper is mainly concerned with constructing and justifying new path algebras that correspond to such more complex problem varieties. A consequence of the obtained algebraic formulation is that multi-objective or robust problem instances can be solved by well-known general algorithms designed to work over an arbitrary path algebra. The solutions obtained in this way comprise all paths that are efficient in the Pareto sense. The efficient paths are by default described only implicitly, as vectors of objective-function values. Still, it is shown in the paper that, with slightly extended versions of the involved algebras, the same paths can also be identified explicitly. Also, for robust problem instances it is possible to select only one "robustly optimal" path according to a generalized min-max or min-max regret criterion.

1. Introduction

Path problems are a family of optimization and enumeration problems, which reduce to generation or comparison of paths in graphs. Some examples are: checking path existence, finding shortest or longest paths, finding most reliable paths or paths with maximum capacity, listing all paths, etc.

One strategy how to deal with path problems is to treat each particular type of problem separately and solve it by dedicated algorithms. Indeed,

2020 *Mathematics Subject Classification.* 90C35, 90C29, 05C38, 16Y60, 68R10.

Key words and phrases. Directed graphs, path problems, path algebras, multi-objective optimization, robust optimization, Pareto efficiency, min-max (regret).

we can develop specialized algorithms for shortest paths, similar but slightly different algorithms for checking path existence, etc [14]. Another strategy is to establish a general framework for the whole family of problems and to use general algorithms. Such approach can be realized by introducing a suitable algebraic structure.

There are many variants of the algebraic approach to path problems proposed in the literature [2, 3, 5, 9, 10, 12, 13, 16, 18, 24, 25, 26, 27]. In this paper we have chosen the variant from [5], which uses a structure whose instances are called *path algebras* (or alternatively *idempotent semirings*). The approach from [5] relies heavily on matrices and on analogies with linear algebra. Each type of path problem is formulated by using a different path algebra. Solving a concrete problem reduces to computing with matrices over the corresponding algebra. The same overall computing procedure can be used for any algebra, thus allowing construction of general (abstract) algorithms.

This paper is concerned with optimization path problems, such as finding shortest, longest, most-reliable paths or paths with maximum capacity. A conventional (single-objective) problem instance is specified by unique values (e.g., lengths, reliabilities, capacities, ...) given to graph arcs. The paper focuses on two complex problem variants, i.e., on multi-objective and on robust variants. A common property of both complex variants is that their problem instances are described by multiple values assigned to arcs.

In multi-objective optimization ([8, 19]), there are more objectives (criteria of optimality) that have to be fulfilled. For instance, in a graph whose arcs are given lengths and reliabilities, we could try to find paths that are at the same time as short and as reliable as possible. Of course, objectives may be in conflict, which means that usually there is no solution that can optimize all objective functions simultaneously. One way of dealing with conflicts is finding more solutions where each of them is in some aspect better than the others. Or a kind of aggregate criterion (e.g., a weighted sum of original objective functions) should be followed. Or priorities among objectives should be established.

In robust optimization ([1, 4, 15, 17]) there is only one objective, but the values of the associated parameters are uncertain. Such uncertainty is expressed through scenarios. For instance, in a robust maximum capacity problem each scenario is a list of possible arc capacities. According to [17] we will assume that the set of scenarios is finite and explicitly given. The usual procedure for solving a robust problem considers only solutions that are feasible for all scenarios. Then the "behavior" of any solution under any scenario is measured in some way. As the "robustly optimal" solution, the one is chosen whose worst behavior, measured over all scenarios, is the best possible. Depending on the chosen behavior measure, such procedure can lead either to the well known min-max (max-min) or to the min-max regret criterion of robustness ([1]).

The aim of this paper is to extend the algebraic framework from [5] in order to cover multi-objective and robust path problems. Or differently speaking, our aim is to show that the considered complex path problems can be regarded as members of the same family where conventional path problems already belong. For this purpose, the paper constructs new path algebras, which correspond to multi-objective or robust problems and are based on algebras corresponding to the respective conventional problems.

The aim of the paper is mostly motivated by "aesthetic" reasons. Indeed, it is nice to see that the already known algebraic framework is wider than originally assumed, thus including the considered complex optimization path problems together with simpler conventional problems. Apart from its aesthetic appeal, the idea of putting multi-objective and robust problems into the same framework brings additional consequences. Namely, according to this idea, multi-objective and robust problems could be solved by well-known and tested general (abstract) algorithms.

In this paper, a multi-objective path problem instance is solved by finding the full set of its *Pareto-efficient* solutions ([8, 19]). Each solution is characterized by a vector of its objective-function values according to different objectives. Efficiency means that the vectors in the set are incomparable when compared with the standard partial ordering of vectors. Or more precisely, for any two solutions in the set, there is an objective where the first one is better than the second one, and another objective where the second one is better than the first one.

In the paper, robust path problems are treated as a special case of multi-objective problems, and solved by using the same algebraic construction as for multi-objective problems. The consequence is that, in order to solve a robust problem instance, again the whole set of efficient solutions is found. Thereby each solution is again represented as an efficient vector. But now that vector comprises objective-function values under different scenarios.

Note that our way of solving a robust optimization problem differs from the usual practice where only one "robustly optimal" solution is chosen according to the previously mentioned min-max (max-min) or min-max regret criterion. Still, both ways of solving can easily be harmonized by defining suitable ranking functions for efficient vectors. In the paper we show that such rankings can be expressed in terms of path-algebra operations, so that the whole ranking procedure remains within the same algebraic framework. Moreover, we show that our rankings are in fact abstractions and generalizations of the usual robustness criteria.

Apart from this introduction, the rest of the paper is organized as follows. Section 2 contains all necessary preliminaries about the adopted algebraic approach to path problems. Section 3 presents our construction of a path algebra that can be used for solving multi-objective path problems. Thereby, solutions are represented only as sets of efficient vectors. Section 4 shows

how the algebra from Section 3 can be extended in order to identify not only vectors but also paths in graphs where those vectors are achieved. Section 5 is devoted to solving robust path problems and to ranking of efficient solutions. The final Section 6 gives conclusions.

2. Algebraic approach to path problems

We start with the definition of our algebraic structure. A *path algebra* is defined according to [5] as a set P equipped with two binary operations, $\vee$ and $\circ$, which have the following properties.

- The operation $\vee$ is idempotent, commutative and associative. Thus for all $x, y, z \in P$ it holds:
$$x \vee x = x,$$
$$x \vee y = y \vee x,$$
$$(x \vee y) \vee z = x \vee (y \vee z).$$
- The operation $\circ$ is associative, left-distributive and right-distributive over $\vee$. Thus for all $x, y, z \in P$:
$$(x \circ y) \circ z = x \circ (y \circ z),$$
$$x \circ (y \vee z) = (x \circ y) \vee (x \circ z),$$
$$(y \vee z) \circ x = (y \circ x) \vee (z \circ x).$$
- There exist a zero element $\phi \in P$ and a unit element $\epsilon \in P$. Thus for any $x \in P$:
$$\phi \vee x = x,$$
$$\phi \circ x = \phi = x \circ \phi,$$
$$\epsilon \circ x = x = x \circ \epsilon.$$

The same or similar structure is also known in the literature as *idempotent semiring*, see [2, 9, 10, 16, 24, 27]. The operation $\vee$ is called the *join operation*, and $\circ$ is called *multiplication*. For $x, y \in P$, the elements $x \vee y$ and $x \circ y$ are referred to as the *join* and the *product* of x and y, respectively.

Table 1. Extremal path algebras

notation	P	$x \vee y$	$x \circ y$	ϕ	ϵ	application
PS	$\mathbb{R} \cup \{\infty\}$	$\min\{x, y\}$	$x + y$	∞	0	shortest paths
PL	$\mathbb{R} \cup \{-\infty\}$	$\max\{x, y\}$	$x + y$	$-\infty$	0	longest paths
PR	$\{t \in \mathbb{R} \mid 0 \leq t \leq 1\}$	$\max\{x, y\}$	$x \cdot y$	0	1	most reliable paths
PC	$\{t \in \mathbb{R} \mid t \geq 0\} \cup \{\infty\}$	$\max\{x, y\}$	$\min\{x, y\}$	0	∞	max capacity paths

Some well-known path algebras from [5], mostly associated with optimization problems, are given in Table 1. In this text they are denoted with PS, PL, PR and PC. Each of them is based on real numbers (sometimes

extended with infinity symbols) and on ordinary extremal or arithmetic operations. Similar algebras can also be found in [12, 27].

TABLE 2. Linguistic path algebras

notation	P	$X \vee Y$	$X \circ Y$	Φ	E	application
PA	$\mathcal{P}(\Sigma^*)$	$X \cup Y$	$\{x*y \mid x \in X, y \in Y\}$	$\emptyset$	$\{\lambda\}$	listing all paths
PE	$\mathcal{B}(\Sigma^*)$	$\mathrm{bas}(X \cup Y)$	$\{x*y \mid x \in X, y \in Y\}$	$\emptyset$	$\{\lambda\}$	listing elem. paths
PO	$\mathcal{W}(\Sigma^*)$	$\mathrm{owl}(X \cup Y)$	$\{x*y \mid x \in X, y \in Y\}$	$\emptyset$	$\{\lambda\}$	listing one path

Additional path algebras associated with enumeration problems are given in Table 2. The first two of them, denoted with PA and PE, can be found in [5]. The third algebra PO is described in [22]. They are all based on linguistic concepts. Namely, Σ denotes here a finite alphabet, and Σ^* is the set of all words (finite sequences of letters) over Σ. Consequently, $\mathcal{P}(\Sigma^*)$ is the set of all languages (sets of words) over Σ. The operation $\vee$ is based on the set union $\cup$, and $\circ$ is based on word concatenation $*$. The symbol λ stands for the empty word, and $\emptyset$ is the empty set (empty language). The zero and the unit element are denoted with Φ and E, respectively.

The algebra PA deals with all languages without any restriction. The algebra PE is similar to PA, but it deals with basic languages. An abbreviation of a word w is defined as any word that can be obtained from w by removing at least one of its letters. For any language $L \subset \Sigma^*$, $\mathrm{bas}(L)$ is the basis of L, i.e., the language consisting of all words from L that do not have abbreviations in L. If $\mathrm{bas}(L) = L$, then L is a basic language. $\mathcal{B}(\Sigma^*)$ denotes the set of all basic languages over Σ. The algebra PO deals with words that can be compared according to a suitably defined total ordering of words - see [22]. Here $\mathcal{W}(\Sigma^*)$ denotes the set of all languages over Σ consisting of at most one word. The expression $\mathrm{owl}(L)$ denotes the smallest word from a given language L interpreted as a one-word language (specially $\mathrm{owl}(\emptyset) = \emptyset$).

An important concept in a path algebra P is its natural ordering relation, which is defined by the following rule:

$$x \preceq y \text{ if } x \vee y = y.$$

It is easy to prove that $\preceq$ is really an ordering (at least partial). For convenience, we can also introduce a strict version of the same relation:

$$x \prec y \text{ if } (x \preceq y \text{ and } x \neq y).$$

Also, it is easy to check that for the zero element ϕ and for any $x, y, z \in P$ it holds:

$$\begin{aligned}
&\phi \preceq x,\\
&x \preceq x \vee y \ \text{ (and indeed } y \preceq x \vee y),\\
&\text{if } x \preceq y \text{ then } x \vee z \preceq y \vee z,\\
&\text{if } x \preceq y \text{ then } (x \circ z \preceq y \circ z \text{ and } z \circ x \preceq z \circ y).
\end{aligned}$$

Now here are some additional remarks regarding notation. When evaluating algebraic expressions over a path algebra P, it is always assumed that $\circ$ takes precedence over $\vee$ unless otherwise regulated by parentheses. Similarly, both $\vee$ and $\circ$ are assumed to take precedence over $\preceq$. Sometimes we will work simultaneously with more than one path algebra. Still, in all algebras we will almost always use the same symbols $\vee$ and $\circ$ for their algebraic operations. Also, we will use the same symbol $\preceq$ for ordering.

Let us now consider $n \times n$ matrices over a path algebra P. We define joins and products of matrices by analogy with ordinary linear algebra. Thus for matrices $X = [x_{ij}]$ and $Y = [y_{ij}]$ over P, we put:

$$\begin{aligned}
X \vee Y &= [x_{ij} \vee y_{ij}]\,,\\
X \circ Y &= \left[\bigvee_{k=1}^{n} x_{ik} \circ y_{kj} \right].
\end{aligned}$$

It is easy to prove that the set of considered matrices equipped with the above operations is itself a path algebra. Its zero is the zero matrix Φ (filled with zeros from P), and its unit element is the unit matrix E (having unit elements from P on the diagonal and zeros elsewhere). For an $n \times n$ matrix X over a path algebra P we consider its powers:

$$X^0 = E, \ \ X^k = X^{k-1} \circ X \ \ (k = 1, 2, \ldots).$$

X is said to be *stable* if for some nonnegative integer q

$$\bigvee_{k=0}^{q} X^k = \bigvee_{k=0}^{q+1} X^k.$$

The smallest q with this property is called the *stability index* of X. The join

$$X^* = X^0 \vee X^1 \vee X^2 \vee \ldots \vee X^q$$

is called the *strong closure* of X, while

$$\widehat{X} = X^1 \vee X^2 \vee X^3 \vee \ldots \vee X^{q+1}$$

is the *weak closure* of X. Obviously, one type of closure can easily be transformed into the other. From now on, the word "closure" will refer to the weak closure.

In this paper we consider *directed* graphs, where nodes are denoted as integers and arcs as ordered pairs of nodes. We explore (directed) *paths* in graphs, i.e., nonempty sequences of adjacent arcs with appropriate orientation. A circular path is called a *cycle*. A path is *elementary* if it does not traverse any node more than once.

A graph G is said to be *labeled* with a path algebra P if each arc (i,j) of G is assigned a nonzero label $l(i,j) \in P$. The label $l(\mu)$ of a path μ in G is then computed as the product of its arc labels. G is said to be *absorptive* if for any elementary cycle γ in G and the unit element ϵ in P it holds:

$$l(\gamma) \preceq \epsilon.$$

An n-node labeled graph G is fully described by its $n \times n$ *adjacency matrix* $A = [a_{ij}]$, whose entries are defined as follows:

$$a_{ij} = \begin{cases} l(i,j) & \text{if the arc } (i,j) \text{ exists,} \\ \phi & \text{otherwise.} \end{cases}$$

The importance of adjacency matrices and absorptive graphs is stressed by the following theorem proved in [5].

THEOREM 2.1. *The adjacency matrix A of an absorptive n-node graph G is always stable with the stability index $q \leq n-1$. The (i,j)-th entry of the closure $\widehat{A}$ is then equal to the join of labels of all elementary paths from node i to node j (or ϕ if there are no such paths).*

Now we are finally ready to explain our algebraic approach to path problems. For a certain problem instance posed in a graph G, we choose a suitable path algebra P and assign appropriate arc labels $l(i,j) \in P$. Next we construct the adjacency matrix A of G. Finally, we compute the closure $\widehat{A}$ and read from it the solutions to the original instance. Feasibility and correctness of the whole procedure is usually guaranteed by Theorem 2.1.

Concrete examples of using the described algebraic approach can be found, e.g., in [5, 22, 23, 24, 25, 27], and later on in Sections 3-5 of this paper. Note that each type of problem requires a different algebra, although the overall problem structure remains the same. Thus it is possible to use general (abstract) algorithms, i.e., algorithms that evaluate the closure $\widehat{A}$ of a matrix A over an *arbitrary* path algebra P. Algorithms of that kind can be found, e.g., in [5, 6, 7, 20, 21, 24, 25, 26], and some of them can be regarded as counterparts of traditional methods for solving linear systems.

In this paper we will build new path algebras from existing ones. In some cases such constructions will rely on algebraic reduction. The needed theory is presented, e.g., in [3, 28]. Here is a short resume.

Let P be a path algebra whose join operation, multiplication, zero and unit element are denoted with $\vee$, $\circ$, ϕ and ϵ, respectively. A function of the

form $\rho : P \longrightarrow P$ is called a *reduction* if it satisfies the following properties:

$$\begin{aligned} \rho(\phi) &= \phi, \\ \rho(\epsilon) &= \epsilon, \\ \rho(\rho(x) \vee y) &= \rho(x \vee y) \quad \text{for all } x, y \in P, \\ \rho(\rho(x) \circ y) &= \rho(x \circ y) = \rho(x \circ \rho(y)) \quad \text{for all } x, y \in P. \end{aligned}$$

Given such ρ, the set P_ρ with the operations $\vee_\rho$ and $\circ_\rho$ is again a path algebra, where

$$\begin{aligned} P_\rho &= \{x \in P \,|\, \rho(x) = x\}, \\ x \vee_\rho y &= \rho(x \vee y), \\ x \circ_\rho y &= \rho(x \circ y). \end{aligned}$$

So far we have considered path algebras in general. But in some situations it will be necessary to restrict to algebras whose operations have additional properties. For instance, we can require that the join operation in an algebra P is a *choice operation*, i.e., for all $x, y \in P$:

$$x \vee y = x \text{ or } x \vee y = y.$$

Or we can assume that multiplication in P has the so-called *cancellation property*, i.e., for all $x, y, z \in P$:

$$\text{if } (x \circ z = y \circ z \text{ or } z \circ x = z \circ y) \text{ then } (x = y \text{ or } z = \phi).$$

Both properties are quite common, e.g., they can be found in most algebras from Table 1.

3. Solving multi-objective path problems

As explained in the previous section, an instance of a conventional path problem is specified by a graph G labeled with a suitable path algebra P. However, in an instance of a multi-objective path problem each objective will produce its own labeling with its specific algebra. Obviously, the labels assigned to the same arc by different objectives can be interpreted as a vector. Thus in order to work with multi-objective path problems we need a mechanism for processing vectors of path-algebra elements. Such mechanism is defined in the following paragraph and justified by Proposition 3.1.

Let $s \geq 2$ be an integer. For $i = 1, 2, \ldots, s$ let P_i be a path algebra with the zero element ϕ_i and the unit element ϵ_i. We consider vectors of length s (written as rows) whose i-th entry belongs to P_i. The set of all such vectors will be denoted with $\mathcal{V}(P_1, P_2, \ldots, P_s)$, i.e.,

$$\mathcal{V}(P_1, P_2, \ldots, P_s) = \{(x_1, x_2, \ldots, x_s) \,|\, x_i \in P_i,\ i = 1, 2, \ldots, s\}.$$

Let $\vec{x} = (x_1, x_2, \ldots, x_s)$ and $\vec{y} = (y_1, y_2, \ldots, y_s)$ be two vectors from the set $\mathcal{V}(P_1, P_2, \ldots, P_s)$. We define their join and product in the following way:

$$\vec{x} \vee \vec{y} = (x_1 \vee y_1, x_2 \vee y_2, \ldots, x_s \vee y_s),$$
$$\vec{x} \circ \vec{y} = (x_1 \circ y_1, x_2 \circ y_2, \ldots, x_s \circ y_s).$$

Assuming the above definitions and notation, we can state the following simple claim.

PROPOSITION 3.1. *The set $\mathcal{V}(P_1, P_2, \ldots, P_s)$ with its operations $\vee$ and $\circ$ constitutes a path algebra, whose zero and unit elements are $\vec{\phi} = (\phi_1, \phi_2, \ldots, \phi_s)$ and $\vec{\epsilon} = (\epsilon_1, \epsilon_2, \ldots, \epsilon_s)$, respectively.*

PROOF. It follows easily by straightforward verification of all path-algebra properties. □

The described path algebra $\mathcal{V}(P_1, P_2, \ldots, P_s)$ will be called a *vector* algebra, and $P_1, P_2, \ldots, P_s$ will be referred to as the corresponding *scalar* algebras. It is easy to show that the natural ordering $\preceq$ in $\mathcal{V}(P_1, P_2, \ldots, P_s)$ is compatible with $\preceq$ in P_i, $i = 1, 2, \ldots, s$, in the following sense:

$(x_1, x_2, \ldots, x_s) \preceq (y_1, y_2, \ldots, y_s)$ if and only if $x_i \preceq y_i$ for all $i = 1, 2, \ldots, s$.

Note that $\preceq$ in $\mathcal{V}(P_1, P_2, \ldots, P_s)$ is always a partial ordering even if $\preceq$ in each P_i is total.

As mentioned in Section 1, a multi-objective path problem instance will be solved by finding the whole set of its efficient solutions. Thus in order to work with multi-objective path problems we also need another mechanism for manipulating sets of efficient path-algebra elements. Such mechanism is described in the following paragraph and justified by Proposition 3.2. For our purposes it is enough to consider only *finite* sets.

Let P be a path algebra, with its natural ordering $\preceq$, whose zero and unit elements are again denoted with ϕ and ϵ, respectively. Let X be a set of elements from P. We say that $x \in X$ is *dominated* in X if there exists $y \in X$ such that $x \prec y$. If $x \in X$ is not dominated in X, we say that x is *efficient* in X. Next we define the operator eff( ). For a finite $X \subseteq P$, eff(X) is the set of all elements of X that are efficient in X. In other words, eff( ) deletes from X all dominated elements and leaves efficient elements. A finite set $X \subseteq P$ is said to be *efficient* if it does not contain dominated elements, i.e., if eff(X) $= X$. Finally, we consider the set $\mathcal{S}(P)$ consisting of all efficient subsets of P, i.e.,

$$\mathcal{S}(P) = \{X \mid X \subseteq P,\ X \text{ finite},\ \text{eff}(X) = X\}.$$

For $X, Y \in \mathcal{S}(P)$ we define their join and product:

$$X \vee Y = \text{eff}(X \cup Y),$$
$$X \circ Y = \text{eff}\left(\{x \circ y \mid x \in X,\ y \in Y\}\right).$$

Relying on the above definitions and notation, the following claim is valid.

PROPOSITION 3.2. *The set $\mathcal{S}(P)$ with its operations $\vee$ and $\circ$ is itself a path algebra, whose zero element is the empty set $\Phi = \emptyset$ and unit element is the single-element set $E = \{\epsilon\}$.*

PROOF. Let us first consider a "relaxed" version of $\mathcal{S}(P)$, which is defined in the same way as above but without the eff( ) operator. It is easy to prove that the relaxed $\mathcal{S}(P)$ is a path algebra. Namely, the relaxed $\mathcal{S}(P)$ is closed with respect to its relaxed $\vee$ and $\circ$. Also, all path-algebra properties are satisfied, as it can easily be shown by straightforward verification.

In order to prove that our "reduced" version of $\mathcal{S}(P)$ (with eff( ) present) is also a path algebra, it is enough to demonstrate that eff( ) is a reduction over the relaxed $\mathcal{S}(P)$. Consequently, four properties of a reduction listed in Section 2 must be verified.

The first two reduction properties are trivial. Let us check the third property. Consider any two sets X and Y from the relaxed $\mathcal{S}(P)$. Then the third property requires that

$$\text{eff}(\text{eff}(X) \cup Y) = \text{eff}(X \cup Y).$$

The above equality obviously holds. Indeed, the inner eff( ) on the left-hand side can be omitted. It is true that due to such omission it can happen that some inefficient $v \in X$ temporarily "survives" although it is dominated by an efficient $w \in X$. But then v will be deleted by the outer eff( ) since it will still be dominated by the same w in $X \cup Y$. Moreover, temporary survival of v cannot cause unnecessary elimination of other elements in $X \cup Y$. Namely, if v dominates some u in $X \cup Y$, then w (being $\succ v \succ u$) also dominates the same u, so that u will be deleted by the outer eff( ) regardless of v.

Next we check the left part of the fourth reduction property. The right part is checked analogously. Consider again any X and Y from the relaxed $\mathcal{S}(P)$. The left part of the fourth property requires that

$$\text{eff}\left(\{x \circ y \,|\, x \in \text{eff}(X),\, y \in Y\}\right) = \text{eff}\left(\{x \circ y \,|\, x \in X,\, y \in Y\}\right).$$

The above equality again holds. Skipping the inner eff( ) can be justified by similar arguments as in the corresponding part of the proof for Proposition 4.1 in Section 4. □

The described path algebra $\mathcal{S}(P)$ will be called a *set* algebra. In the same context P will be referred to as the corresponding *element* algebra.

As already explained in this section, solving multi-objective path problems means working with sets of efficient solutions. On the other hand, each of those solutions is characterized by a vector of values obtained according to different objectives. Consequently, the two mechanisms described above, for working with vectors and sets, respectively, should be combined. More precisely, we are interested in path algebras of the form $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$,

where P_i, $i = 1, 2, \ldots, s$, is a scalar path algebra, $\mathcal{V}(P_1, P_2, \ldots, P_s)$ consists of vectors whose i-th entry belongs to P_i, and $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$ consists of efficient sets of vectors from $\mathcal{V}(P_1, P_2, \ldots, P_s)$. Due to Propositions 3.1 and 3.2, the considered $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$ is indeed a path algebra, which will be called a *vector-set* algebra.

Now we are ready to explain in more detail our algebraic method for solving multi-objective path problems.

- Let the considered instance of a multi-objective problem be posed in a graph G with n nodes, and let it comprise s objectives. Suppose that the conventional (single-objective) problem associated with the i-th objective is characterized by a scalar path algebra P_i. Then the i-th objective produces a labeling of G with P_i.
- The labels assigned to the same arc according to different objectives are interpreted as a vector from $\mathcal{V}(P_1, P_2, \ldots, P_s)$. That vector is further interpreted as a single-element set. In this way, G is considered as labeled with the vector-set algebra $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$. Consequently, the given multi-objective problem instance is specified by the $n \times n$ adjacency matrix A of G over $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$.
- The problem instance is solved by computing the closure $\widehat{A}$. Computing is accomplished through path-algebra operations $\vee$ and $\circ$ within $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$. The overall algorithm simply follows the closure definition from Section 2, i.e., it evaluates joins of matrix powers. Or some more sophisticated abstract closure procedure from the literature is used, which is based, e.g., on Gaussian elimination or Gauss-Seidel iteration.
- The solutions of the considered problem instance are directly represented by $\widehat{A}$. Indeed, the (i, j)-th entry of $\widehat{A}$ is a set of efficient vectors - each of them corresponds to (at least) one path in G from node i to node j and comprises the labels of that path according to different objectives.

Within our method, a path corresponding to an efficient vector is also called *efficient*. For any two such paths connecting the same pair of nodes there is an objective where the first one is better than the second one, and vice versa - thus neither of them can be discarded as inferior. The method is further illustrated by Example 3.3.

EXAMPLE 3.3. We consider an instance of a two-objective path problem given in Figure 1. The first objective is to minimize path length, and the second objective is to maximize path reliability. Arc lengths and reliabilities are given by left-hand and right-hand arc labels, respectively. For any pair of nodes i and j we are looking for paths between i and j that are at the same time as short as possible and as reliable as possible. At this moment

we ignore arc identifiers (letters a, b, c, ...) - they will be used in the next example.

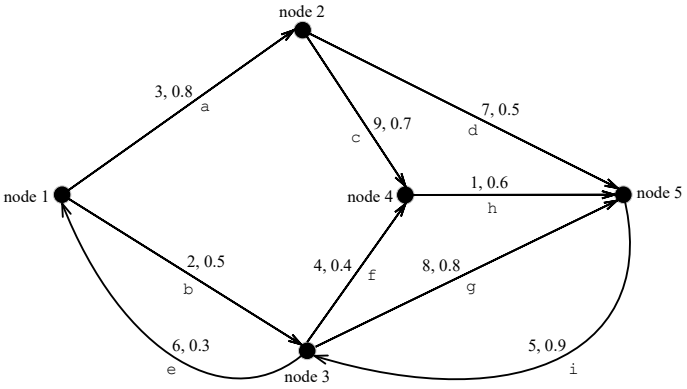

FIGURE 1. An instance of the two-objective shortest/most-reliable path problem

The considered multi-objective problem instance is specified by the following 5×5 adjacency matrix:

$$A = \begin{bmatrix} \emptyset & \{(3,0.8)\} & \{(2,0.5)\} & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \{(9,0.7)\} & \{(7,0.5)\} \\ \{(6,0.3)\} & \emptyset & \emptyset & \{(4,0.4)\} & \{(8,0.8)\} \\ \emptyset & \emptyset & \emptyset & \emptyset & \{(1,0.6)\} \\ \emptyset & \emptyset & \{(5,0.9)\} & \emptyset & \emptyset \end{bmatrix}.$$

We deal here with a matrix over the vector-set algebra $\mathcal{S}(\mathcal{V}(PS, PR))$, where PS and PR are the scalar algebras from Table 1 associated with the conventional shortest path and most-reliable path problem, respectively. The corresponding closure matrix computed within $\mathcal{S}(\mathcal{V}(PS, PR))$ looks as follows:

$$\widehat{A} = \begin{bmatrix} \left\{ \begin{matrix}(8,0.15)\end{matrix} \right\} & \left\{ \begin{matrix}(3,0.8)\end{matrix} \right\} & \left\{ \begin{matrix}(2,0.5)\end{matrix} \right\} & \left\{ \begin{matrix}(12,0.56),\\(6,0.2)\end{matrix} \right\} & \left\{ \begin{matrix}(10,0.4),\\(7,0.12)\end{matrix} \right\} \\ \left\{ \begin{matrix}(18,0.135)\end{matrix} \right\} & \left\{ \begin{matrix}(21,0.108)\end{matrix} \right\} & \left\{ \begin{matrix}(12,0.45)\end{matrix} \right\} & \left\{ \begin{matrix}(9,0.7)\end{matrix} \right\} & \left\{ \begin{matrix}(7,0.5)\end{matrix} \right\} \\ \left\{ \begin{matrix}(6,0.3)\end{matrix} \right\} & \left\{ \begin{matrix}(9,0.24)\end{matrix} \right\} & \left\{ \begin{matrix}(8,0.15),\\(13,0.72),\\(10,0.216)\end{matrix} \right\} & \left\{ \begin{matrix}(4,0.4)\end{matrix} \right\} & \left\{ \begin{matrix}(8,0.8),\\(5,0.24)\end{matrix} \right\} \\ \left\{ \begin{matrix}(12,0.162)\end{matrix} \right\} & \left\{ \begin{matrix}(15,0.1296)\end{matrix} \right\} & \left\{ \begin{matrix}(6,0.54)\end{matrix} \right\} & \left\{ \begin{matrix}(10,0.216)\end{matrix} \right\} & \left\{ \begin{matrix}(1,0.6)\end{matrix} \right\} \\ \left\{ \begin{matrix}(11,0.27)\end{matrix} \right\} & \left\{ \begin{matrix}(14,0.216)\end{matrix} \right\} & \left\{ \begin{matrix}(5,0.9)\end{matrix} \right\} & \left\{ \begin{matrix}(9,0.36)\end{matrix} \right\} & \left\{ \begin{matrix}(13,0.72),\\(10,0.216)\end{matrix} \right\} \end{bmatrix}$$

From $\widehat{A}$ we can read the solutions for our problem instance. Suppose that we are interested in paths from node 1 to node 5. Then we must observe the $(1,5)$-th entry in $\widehat{A}$. It consists of two vectors - each of them corresponds to (at least) one path between nodes 1 and 5 and comprises the length and reliability of that path.

By observing closely our small graph, it is easy to find out that the vector $(10, 0.4)$ corresponds either to the path through nodes 1, 2 and 5, or to the path through nodes 1, 3 and 5 (let us call them the first and the second path, respectively). On the other hand, the vector $(7, 0.12)$ corresponds to the path traversing nodes 1, 3, 4 and 5 (let it be called the third path). Since both vectors are efficient, it means that none of the three listed paths is inferior to another. Indeed, the first two of them are equivalent, i.e., they have the same lengths and reliabilities. The first or second path is more reliable than the third one, but the third one is shorter.

It is also easy to see that in our small graph there exists as well the fourth path between nodes 1 and 5, which traverses nodes 1, 2, 4 and 5, and which is characterized by the vector $(13, 0.336)$. However, that vector is discarded from the solution set since it is not efficient. More precisely, $(13, 0.336)$ is dominated by $(10, 0.4)$, which means that the fourth path is inferior to the first or second path since it is longer and less reliable.

Example 3.3 has shown that our algebraic method works well at least for one instance of one multi-objective path problem. But can we be sure that it will work in general? Or more precisely, can we be sure that it will always be feasible and correct? Feasibility means that the constructed adjacency matrix A over $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$ is really stable so that its closure $\widehat{A}$ can be computed in a finite number of operations. Correctness means that the entries of $\widehat{A}$ really contain the desired sets of efficient vectors.

In most cases, both feasibility and correctness can be guaranteed by Theorem 2.1 from Section 2. But in order to apply Theorem 2.1, we must be sure that the adjacency matrix A is absorptive in the sense of $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$. The next three propositions will establish necessary and sufficient conditions for such absorptiveness.

PROPOSITION 3.4. *Let a graph G be labeled simultaneously with path algebras $P_1, P_2, \ldots, P_s$ according to s objectives. Let the labels assigned to the same arc by different objectives be interpreted as a label from the vector algebra $\mathcal{V}(P_1, P_2, \ldots, P_s)$. Then G is absorptive in the sense of $\mathcal{V}(P_1, P_2, \ldots, P_s)$ if and only if it is absorptive in the sense of P_i for each $i = 1, 2, \ldots, s$.*

PROOF. As before, the unit element in P_i will be denoted by ϵ_i, and the unit element in $\mathcal{V}(P_1, P_2, \ldots, P_s)$ by $\vec{\epsilon}$. Let $l_i(\ )$ denote labeling with P_i according to the i-th objective, and $\vec{l}(\ )$ labeling with $\mathcal{V}(P_1, P_2, \ldots, P_s)$. We prove necessity. Suppose that G is absorptive in the sense of $\mathcal{V}(P_1, P_2, \ldots, P_s)$

and let γ be any elementary cycle in G. Then, according to the definition of absorptiveness:

$$\vec{l}(\gamma) \preceq \vec{\epsilon}.$$

Thanks to properties of $\mathcal{V}(P_1, P_2, \ldots, P_s)$ this is equivalent to

$$(l_1(\gamma), l_2(\gamma), \ldots, l_s(\gamma)) \preceq (\epsilon_1, \epsilon_2, \ldots, \epsilon_s),$$

and also to

$$l_i(\gamma) \preceq \epsilon_i, \quad i = 1, 2, \ldots, s.$$

Thus again according to the definition of absorptiveness, G is absorptive in the sense of P_i for any i. Sufficiency can be proved by reading the above necessity proof in opposite direction. □

PROPOSITION 3.5. *Let a graph G be labeled with a path algebra P. Let the label assigned to any arc be interpreted as a single-element set from the set algebra $\mathcal{S}(P)$. Then G is absorptive in the sense of $\mathcal{S}(P)$ if and only if it is absorptive in the sense of P.*

PROOF. As before, the unit element in P will be denoted by ϵ, and the unit element in $\mathcal{S}(P)$ by E. Let $l(\ )$ denote labeling with P and $L(\ )$ labeling with $\mathcal{S}(P)$. We prove necessity. Suppose that G is absorptive in the sense of $\mathcal{S}(P)$ and let γ be any elementary cycle in G. Then, according to the definition of absorptiveness:

$$L(\gamma) \preceq E.$$

Thanks to properties of $\mathcal{S}(P)$ this is equivalent to

$$\mathrm{eff}(\{l(\gamma), \epsilon\}) = \{\epsilon\}.$$

The above equality of sets is possible only if $l(\gamma)$ is dominated by ϵ or equal to ϵ, i.e., only if

$$l(\gamma) \preceq \epsilon.$$

Thus again according to the definition of absorptiveness, G is absorptive in the sense of P. Sufficiency is proved by reading the above necessity proof in opposite direction. □

PROPOSITION 3.6. *Let a graph G be labeled simultaneously with path algebras $P_1, P_2, \ldots, P_s$ according to s objectives. Let the labels assigned to the same arc by different objectives be interpreted as a vector from $\mathcal{V}(P_1, P_2, \ldots, P_s)$, which is further interpreted as a single-vector set from $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$. Then G is absorptive in the sense of $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$ if and only if it is absorptive in the sense of P_i for each $i = 1, 2, \ldots, s$.*

PROOF. It follows directly by applying first Proposition 3.5, then Proposition 3.4. □

Our algebraic method is primarily intended to solve multi-objective variants combined of well-known conventional optimization path problems, such as those listed in Table 1. Let us now analyze how the condition from Proposition 3.6 is interpreted within each of the extremal path algebras from Table 1. Indeed, in PR and PC the unit element ϵ is the greatest element according to the ordering $\preceq$, so that absorptiveness is achieved automatically. In PS (associated with shortest paths) absorptiveness means that any elementary cycle in a graph must have a nonnegative length - this is a reasonable assumption since otherwise the shortest path problem would not make sense. PL (associated with longest paths) is usually applied to acyclic graphs, so that absorptiveness holds trivially. So putting it all together, the condition from Proposition 3.6 is fulfilled for all algebras from Table 1, either automatically or trivially or with some additional but natural assumptions regarding the involved graph. Thus according to Proposition 3.6 and Theorem 2.1, our algebraic method is applicable to multi-objective combinations of (at least) the following conventional optimization problems: shortest paths, longest paths, most-reliable paths, paths of maximum capacity.

4. Identifying efficient paths

A potential drawback of the method from the previous section is that it represents efficient paths only implicitly, i.e., as vectors of objective-function values. Moreover, it is not clear whether a particular vector represents just one path or more paths that happen to produce the same values. In this section we will develop an extended version of the method, able to explicitly identify efficient paths together with vectors. For this purpose, we will need a more complex version of the set algebra $\mathcal{S}(P)$ from the previous section. The idea is to combine an extremal or vector path algebra (derived, e.g., from Table 1) with a linguistic path algebra (from Table 2). The first of the combined algebras should correspond to the considered optimization problem, while the second algebra should encode paths. The new version of $\mathcal{S}(P)$ is defined in the following three paragraphs and justified by Proposition 4.1.

Let P be a path algebra whose zero element is denoted with ϕ and unit element with ϵ. Let $\bar{P}$ be another path algebra with the zero element $\bar{\phi}$ and the unit element $\bar{\epsilon}$. The two algebras P and $\bar{P}$ will be called *active* and *passive* algebra, respectively. We consider the set of ordered pairs of the following form:

$$Q = \left\{(x, \bar{x}) \,|\, x \in P,\, x \neq \phi,\, \bar{x} \in \bar{P}\right\}.$$

For any pair $(x, \bar{x})$ from Q, x is called the *active* component and $\bar{x}$ is the *passive* component.

Next we define a new efficiency operator $\widetilde{\text{eff}}(\ )$ operating on finite sets of pairs from Q. For a finite $X \subseteq Q$, $\widetilde{\text{eff}}(X)$ is a subset of X obtained through the following two phases.

- *Phase 1.* All pairs from X with equal active components are merged into a single pair whose active component is the one found in the original pairs and passive component is the join of passive components from the original pairs. For instance, if X contains altogether three pairs with a given $x \in P$, i.e., $(x, \bar{x}_1)$, $(x, \bar{x}_2)$ and $(x, \bar{x}_3)$, then they are replaced with a single pair $(x, \bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$. Note that after phase 1 all pairs in X have distinct active components.
- *Phase 2.* All pairs from X that are dominated in X with respect to their active components are deleted from X. For instance, if X contains two pairs $(x, \bar{x})$ and $(y, \bar{y})$ such that $x \prec y$, then the pair $(x, \bar{x})$ is deleted. In other words, phase 2 leaves in X only those pairs that are efficient according to their active component.

A finite set $X \subseteq Q$ is said to be *efficient* if $\widetilde{\mathrm{eff}}(X) = X$. Our new version of the set algebra, denoted by $\widetilde{\mathcal{S}}(P, \bar{P})$ consists of all efficient subsets of Q, i.e.,

$$\widetilde{\mathcal{S}}(P, \bar{P}) = \left\{ X \,|\, X \subseteq Q,\ X \text{ finite},\ \widetilde{\mathrm{eff}}(X) = X \right\}.$$

For $X, Y \in \widetilde{\mathcal{S}}(P, \bar{P})$ we define their join and product:

$$X \vee Y = \widetilde{\mathrm{eff}}(X \cup Y),$$
$$X \circ Y = \widetilde{\mathrm{eff}}\left(\{(x \circ y, \bar{x} \circ \bar{y}) \,|\, (x, \bar{x}) \in X,\ (y, \bar{y}) \in Y\}\right).$$

By using the definitions and notation from the above three paragraphs, the following result can be stated.

PROPOSITION 4.1. *Suppose that multiplication in P has the cancellation property. Then the set $\widetilde{\mathcal{S}}(P, \bar{P})$ with its operations $\vee$ and $\circ$ constitutes a path algebra, whose zero element is the empty set $\Phi = \emptyset$ and unit element is the single-pair set $E = \{(\epsilon, \bar{\epsilon})\}$.*

PROOF. Let us consider a "relaxed" version of $\widetilde{\mathcal{S}}(P, \bar{P})$, which is defined in the same way as above but with the $\widetilde{\mathrm{eff}}(\ )$ operator missing. It is relatively easy to prove that the relaxed $\widetilde{\mathcal{S}}(P, \bar{P})$ is a path algebra. Within such proof, all path-algebra properties are verified by straightforward computation. In addition, it must also be shown that the relaxed $\widetilde{\mathcal{S}}(P, \bar{P})$ is closed with respect to its relaxed operations $\vee$ and $\circ$. More precisely, it is clear that both operations produce again finite sets of pairs. But it also has to be demonstrated that the active component in any newly produced pair cannot be zero, i.e., that the product of two nonzero elements from P is again nonzero. Indeed, this is a simple consequence of the assumed cancellation property. Namely, for $x, y \in P$, $x \neq \phi$, $y \neq \phi$, such that $x \circ y = \phi$ we could write $x \circ y = \phi \circ y$. Then due to the cancellation property (since $y \neq \phi$) it would follow that $x = \phi$, which would be a contradiction.

After proving that the relaxed $\widetilde{\mathcal{S}}(P,\bar{P})$ is a path algebra, it follows relatively easily that our "reduced" $\widetilde{\mathcal{S}}(P,\bar{P})$, (i.e., with $\widetilde{\mathrm{eff}}(\ )$ included) is also a path algebra. The only thing one must show is that $\widetilde{\mathrm{eff}}(\ )$ is a reduction over the relaxed $\widetilde{\mathcal{S}}(P,\bar{P})$. It means that four properties of a reduction from Section 2 must be verified.

The first two reduction properties are trivial. Let us concentrate on the left part of the fourth property. The right part is checked analogously. Consider any two sets of pairs X and Y from the relaxed $\widetilde{\mathcal{S}}(P,\bar{P})$. Then the left part requires that

$$\widetilde{\mathrm{eff}}\left(\{(x\circ y,\bar{x}\circ\bar{y})\,|\,(x,\bar{x})\in\widetilde{\mathrm{eff}}(X),\,(y,\bar{y})\in Y\}\right)$$
$$=\widetilde{\mathrm{eff}}\left(\{(x\circ y,\bar{x}\circ\bar{y})\,|\,(x,\bar{x})\in X,\,(y,\bar{y})\in Y\}\right).$$

We can see that that the above equality really holds. Indeed, omission of the inner $\widetilde{\mathrm{eff}}(\ )$ does not change the result. Here is the explanation.

- With such omission, some pairs from X with equal active components, say $(u,\bar{u}_1)$ and $(u,\bar{u}_2)$ will remain separated although the inner $\widetilde{\mathrm{eff}}(\ )$ should merge them into a single pair $(u,\bar{u}_1\vee\bar{u}_2)$. But after componentwise multiplication with the same $(y,\bar{y})\in Y$, those pairs, becoming $(u\circ y,\bar{u}_1\circ\bar{y})$ and $(u\circ y,\bar{u}_2\circ\bar{y})$, will still have the same active components so that they will eventually be merged by the outer $\widetilde{\mathrm{eff}}(\ )$ into $(u\circ y,\bar{u}_1\circ\bar{y}\vee\bar{u}_2\circ\bar{y})$. Thanks to the properties of $\bar{P}$, this is the same result as if the original pairs from X were merged by the inner $\widetilde{\mathrm{eff}}(\ )$ before multiplication with $(y,\bar{y})$.
- With such omission, a pair $(u,\bar{u})\in X$ dominated by some other pair $(v,\bar{v})\in X$ will temporarily "survive" although it should be deleted by the inner $\widetilde{\mathrm{eff}}(\ )$. Remember that domination means that $u\prec v$. Then the obsolete pair $(u,\bar{u})$ will produce new obsolete pairs of the form $(u\circ y,\bar{u}\circ\bar{y})$ for various $(y,\bar{y})\in Y$. But each of those new pairs will be dominated by the corresponding $(v\circ y,\bar{v}\circ\bar{y})$. Namely, from $u\prec v$ it follows that $u\circ y\prec v\circ y$. Thereby, the inequality is again strict thanks to the cancellation property of $\circ$ in P and the fact that $y\neq\phi$. Thus all obsolete pairs $(u\circ y,\bar{u}\circ\bar{y})$ will eventually be deleted by the outer $\widetilde{\mathrm{eff}}(\ )$.
- Temporary existence of the mentioned obsolete pairs $(u\circ y,\bar{u}\circ\bar{y})$ cannot produce side-effects among the remaining pairs. Indeed, if an obsolete pair $(u\circ y,\bar{u}\circ\bar{y})$ merges with some other pair, the obtained merged pair will be dominated by the same pair $(v\circ y,\bar{v}\circ\bar{y})$ that dominates $(u\circ y,\bar{u}\circ\bar{y})$. So the outer $\widetilde{\mathrm{eff}}(\ )$ will delete the merged pair thus removing the effects of obsolete merging. Similarly, if an obsolete pair $(u\circ y,\bar{u}\circ\bar{y})$ dominates some other pair $(u_1\circ y_1,\bar{u}_1\circ\bar{y}_1)$ and causes its deletion, then $(u_1\circ y_1,\bar{u}_1\circ\bar{y}_1)$

is also dominated by the pair $(v \circ y, \bar{v} \circ \bar{y})$ that dominates $(u \circ y, \bar{u} \circ \bar{y})$. So $(u_1 \circ y_1, \bar{u}_1 \circ \bar{y}_1)$ will be deleted by the outer $\widetilde{\text{eff}}(\ )$ regardless of $(u \circ y, \bar{u} \circ \bar{y})$.

Next we have to check the third reduction property. Consider again any two sets of pairs X and Y from the relaxed $\widetilde{\mathcal{S}}(P, \bar{P})$. The third property requires that

$$\widetilde{\text{eff}}(\widetilde{\text{eff}}(X) \cup Y) = \widetilde{\text{eff}}(X \cup Y).$$

The equality obviously holds since the inner $\widetilde{\text{eff}}(\ )$ can be skipped. Justification for such skipping relies on similar arguments as for the above analyzed fourth property. □

On one hand, the described path algebra $\widetilde{\mathcal{S}}(P, \bar{P})$ can be considered as a modification of the set algebra $\mathcal{S}(P)$ from the previous section. On the other hand, the same algebra can also be regarded as a generalization of the so-called composite path algebra from [23]. Consequently, $\widetilde{\mathcal{S}}(P, \bar{P})$ will be called a *composite set* algebra based on the active algebra P and the passive algebra $\bar{P}$. It should also be noted that our construction bears some resemblance to the structure developed in [11], which is based on the so-called lexicographic product and lexicographic ordering.

Similarly as in the previous section, in order to solve multi-objective path problems it is necessary to work with vectors over some scalar algebras. Or more precisely, our intention is to work with a composite set algebra whose active algebra consists of vectors. However, according to Proposition 3.1, correctness of such construction is guaranteed only if multiplication of those vectors satisfies the cancellation property. Unfortunately, the desired property does not hold in general. Indeed, if we consider the vector algebra $\mathcal{V}(P_1, P_2, \ldots, P_s)$ from the previous section, we can easily find nonzero vectors whose product is zero. But luckily enough, it is still possible to design a restricted version of $\mathcal{V}(P_1, P_2, \ldots, P_s)$, which is good enough for our purposes and has the required cancellation property. The design is described in the following paragraph and justified by Proposition 4.2.

Let $s \geq 2$ be an integer. For $i = 1, 2, \ldots, s$, let P_i be a path algebra whose zero and unit elements are denoted with ϕ_i and ϵ_i, respectively. We consider vectors of length s (written as rows) whose i-th entry is a nonzero element from P_i. The set of all such vectors together with the zero vector will be denoted with $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$, i.e.,

$$\begin{aligned}&\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)\\ &\quad = \{(x_1, x_2, \ldots, x_s) \,|\, x_i \in P_i,\ x_i \neq \phi_i,\ i = 1, 2, \ldots, s\} \cup \{(\phi_1, \phi_2, \ldots, \phi_s)\}.\end{aligned}$$

Let $\vec{x} = (x_1, x_2, \ldots, x_s)$ and $\vec{y} = (y_1, y_2, \ldots, y_s)$ be two vectors from the set $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$. We define their join and product in the following way:

$$\vec{x} \vee \vec{y} = (x_1 \vee y_1, x_2 \vee y_2, \ldots, x_s \vee y_s),$$
$$\vec{x} \circ \vec{y} = (x_1 \circ y_1, x_2 \circ y_2, \ldots, x_s \circ y_s).$$

Then, by assuming the above definitions and notation, the following result holds.

PROPOSITION 4.2. *Suppose that for all $i = 1, 2, \ldots, s$ the join operation in P_i is a choice operation and multiplication in P_i has the cancellation property. Then the set $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ with its operations $\vee$ and $\circ$ constitutes a path algebra, whose zero element is $\vec{\phi} = (\phi_1, \phi_2, \ldots, \phi_s)$ and unit element is $\vec{\epsilon} = (\epsilon_1, \epsilon_2, \ldots, \epsilon_s)$. Moreover, the operation $\circ$ in $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ has the cancellation property.*

PROOF. Note that $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ is a subset of $\mathcal{V}(P_1, P_2, \ldots, P_s)$ from the previous section. Also, the operations $\vee$ and $\circ$ in $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ are defined in the same way as in $\mathcal{V}(P_1, P_2, \ldots, P_s)$, and the proposed zero and unit elements are the same. Consequently, in order to prove that $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ is a path algebra (in fact a sub-algebra of $\mathcal{V}(P_1, P_2, \ldots, P_s)$), we only need to demonstrate that $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ is closed with respect to $\vee$ and $\circ$. Indeed, let $\vec{x} = (x_1, x_2, \ldots, x_s)$ and $\vec{y} = (y_1, y_2, \ldots, y_s)$ be two vectors from $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$. We must show that both $\vec{x} \vee \vec{y}$ and $\vec{x} \circ \vec{y}$ again belong to $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$. To exclude trivial cases we can assume that both $\vec{x}$ and $\vec{y}$ are nonzero, which means that all x_i, y_i, $i = 1, 2, \ldots, s$, are nonzero in P_i. But then, since $\vee$ in P_i is a choice operation, each entry of $\vec{x} \vee \vec{y}$, being a choice among two nonzero values, is again nonzero. Similarly, since $\circ$ in P_i has the cancellation property, each component of $\vec{x} \circ \vec{y}$, being a product of two nonzero vales, must be nonzero (as already demonstrated within the proof of Proposition 4.1). So both $\vec{x} \vee \vec{y}$ and $\vec{x} \circ \vec{y}$ consist of nonzero entries, thus belonging to $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$.

It remains to prove that $\circ$ in $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ has the cancellation property. We prove the first part of the property (cancellation from right). Indeed, let $\vec{x} = (x_1, x_2, \ldots, x_s)$, $\vec{y} = (y_1, y_2, \ldots, y_s)$ and $\vec{z} = (z_1, z_2, \ldots, z_s)$ be vectors from $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ such that $\vec{x} \circ \vec{z} = \vec{y} \circ \vec{z}$, or equivalently

$$x_i \circ z_i = y_i \circ z_i, \ i = 1, 2, \ldots, s.$$

We must show that either $\vec{x} = \vec{y}$ or $\vec{z} = \vec{\phi}$. To exclude the trivial case, let us assume that $\vec{z} \neq \vec{\phi}$, which is equivalent to

$$z_i \neq \phi_i, \ i = 1, 2, \ldots, s.$$

But then, since $\circ$ in P_i has the cancellation property, the above two exposed formulas together imply that

$$x_i = y_i,\ i = 1, 2, \ldots, s.$$

Hence $\vec{x} = \vec{y}$. The other part of the cancellation property (cancellation from left) is proved analogously. □

As already announced, our extended method for solving multi-objective variants of optimization path problems will be based on path algebras of the form $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$, where $s \geq 2$ is an integer, $P_1, P_2, \ldots, P_s$ are extremal path algebras, and $\bar{P}$ is a linguistic path algebra. Propositions 4.1 and 4.2 should guarantee that $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$ is really a path algebra. But note that this guarantee is given only if for each $i = 1, 2, \ldots, s$ the join operation in P_i is a choice operation and multiplication in P_i has the cancellation property. Luckily enough, most extremal path algebras, e.g., most of those from Table 1, satisfy such conditions. An algebra of the form $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$ could be called a *composite vector-set* algebra.

A detailed description of our extended method would look similarly as for the simpler method from Section 3. Here we list only the main differences.

- To enable identification, each arc in our graph G is now also labeled with a distinct letter from a finite alphabet Σ. That letter is interpreted as a single-letter word, and furthermore as a single-word language from a suitable linguistic algebra $\bar{P}$ over Σ.
- The given problem instance is now specified by an adjacency matrix A over a composite vector-set algebra of the form $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$, rather than a vector-set algebra of the form $\mathcal{S}(\mathcal{V}(P_1, P_2, \ldots, P_s))$.
- The (i, j)-th entry of the closure $\widehat{A}$ is not anymore a set of vectors, but rather a set of ordered pairs. The active component within each pair is a vector with the same meaning as before, while the corresponding passive component is a language whose each word identifies one path where the corresponding vector is achieved.
- Depending on the chosen linguistic algebra $\bar{P}$, all efficient paths or only some of them are identified. In order to obtain all paths, a relatively demanding algebra PA or PE from Table 2 must be used. However, if we are pleased with only one path per vector, we can use a simpler algebra PO from Table 2.

The described method is illustrated by the following example.

Example 4.3. We consider again the two-objective path problem instance given in Figure 1, where the objectives are to minimize path length and to maximize path reliability. Arc lengths and reliabilities are still given by numerical arc labels. As in Example 3.3, for any pair of nodes i and j we are looking for paths from i to j that are at the same time as short and as reliable

as possible. But now we want to explicitly identify such paths by algebraic means. In order to enable identification, each arc is additionally labeled with a distinct letter from the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$.

The considered two-objective problem instance is now specified in a more complex form than in Example 3.3, i.e., by the following 5×5 adjacency matrix:

$$A = \begin{bmatrix} \emptyset & \{((3,0.8),\{\mathtt{a}\})\} & \{((2,0.5),\{\mathtt{b}\})\} & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \{((9,0.7),\{\mathtt{c}\})\} & \{((7,0.5),\{\mathtt{d}\})\} \\ \{((6,0.3),\{\mathtt{e}\})\} & \emptyset & \emptyset & \{((4,0.4),\{\mathtt{f}\})\} & \{((8,0.8),\{\mathtt{g}\})\} \\ \emptyset & \emptyset & \emptyset & \emptyset & \{((1,0.6),\{\mathtt{h}\})\} \\ \emptyset & \emptyset & \{((5,0.9),\{\mathtt{i}\})\} & \emptyset & \emptyset \end{bmatrix}$$

The composite vector-set algebra involved here is $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(PS, PR), PA)$. As in Example 3.3, PS and PR are the extremal algebras from Table 1 associated with the conventional shortest path and most reliable path problem, respectively. The newly introduced PA is the linguistic algebra from Table 2 associated with the problem of listing all paths. The corresponding closure matrix computed within $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(PS, PR), PA)$ is:

$$\widehat{A} = \begin{bmatrix} \{((8,0.15),\{\mathtt{be}\})\} & \{((3,0.8),\{\mathtt{a}\})\} & \{((2,0.5),\{\mathtt{b}\})\} & \{((12,0.56),\{\mathtt{ac}\}),\ ((6,0.2),\{\mathtt{bf}\})\} & \{((10,0.4),\{\mathtt{ad},\mathtt{bg}\}),\ ((7,0.12),\{\mathtt{bfh}\})\} \\ \{((18,0.135),\{\mathtt{die}\})\} & \{((21,0.108),\{\mathtt{diea}\})\} & \{((12,0.45),\{\mathtt{di}\})\} & \{((9,0.7),\{\mathtt{c}\})\} & \{((7,0.5),\{\mathtt{d}\})\} \\ \{((6,0.3),\{\mathtt{e}\})\} & \{((9,0.24),\{\mathtt{ea}\})\} & \{((8,0.15),\{\mathtt{eb}\}),\ ((13,0.72),\{\mathtt{gi}\}),\ ((10,0.216),\{\mathtt{fhi}\})\} & \{((4,0.4),\{\mathtt{f}\})\} & \{((8,0.8),\{\mathtt{g}\}),\ ((5,0.24),\{\mathtt{fh}\})\} \\ \{((12,0.162),\{\mathtt{hie}\})\} & \{((15,0.1296),\{\mathtt{hiea}\})\} & \{((6,0.54),\{\mathtt{hi}\})\} & \{((10,0.216),\{\mathtt{hif}\})\} & \{((1,0.6),\{\mathtt{h}\})\} \\ \{((11,0.27),\{\mathtt{ie}\})\} & \{((14,0.216),\{\mathtt{iea}\})\} & \{((5,0.9),\{\mathtt{i}\})\} & \{((9,0.36),\{\mathtt{if}\})\} & \{((13,0.72),\{\mathtt{ig}\}),\ ((10,0.216),\{\mathtt{ifh}\})\} \end{bmatrix}$$

Similarly as in Example 3.3, the matrix $\widehat{A}$ contains the solutions to our problem instance. But now those solutions are more elaborately presented. Indeed, each efficient vector is accompanied with a set of words identifying paths where the values (lengths, reliabilities) from that vector are achieved. For instance, if we are again interested in paths from node 1 to node 5, we must observe the $(1,5)$-th entry of $\widehat{A}$. We can see the same two efficient vectors $(10, 0.4)$ and $(7, 0.12)$ as in Example 3.3. But now it is also visible that $(10, 0.4)$ corresponds either to the path through arcs `a` and `d`, or to the path through `b` and `g`. Similarly, $(7, 0.12)$ corresponds to the path through `b`, `f` and `h`. In this way, the whole set of efficient paths from node 1 to node 5 has been identified.

In the remaining part of this section we will explore applicability of our extended method to different types of path problems. Applicability first of all means that for the chosen integer s and path algebras $P_1, P_2, \ldots, P_s$ and $\bar{P}$ the involved algebraic structure $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$ is really a

path algebra. However, as already emphasized, we can guarantee that $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$ is a path algebra only if, for all $i = 1, 2, \ldots, s$, $\vee$ in P_i is a choice operation and $\circ$ in P_i has the cancellation property. But even then, we still cannot be sure that the method will be feasible and correct. Or more precisely, we still cannot be sure that the constructed adjacency matrix A over $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$ will be stable and that its closure $\widehat{A}$ will really contain the desired results.

As in the previous section, feasibility and correctness can be guaranteed by Theorem 2.1 from Section 2. But to be able to apply Theorem 2.1, we must assure that the adjacency matrix A is absorptive in the sense of $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$. Conditions for such absorptiveness will be established by the following proposition and its corollaries. We assume the usual notation: if P and $\bar{P}$ are two path algebras, then their unit elements are denoted with ϵ and $\bar{\epsilon}$, respectively. Also $l(\,)$ denotes labeling with P, while $\bar{l}(\,)$ is labeling with $\bar{P}$.

PROPOSITION 4.4. *Let P and $\bar{P}$ be two path algebras such that $\widetilde{\mathcal{S}}(P, \bar{P})$ is also a path algebra. Suppose that a graph G is labeled simultaneously with P and $\bar{P}$. Let the labels assigned to the same arc be interpreted as a single-pair set from $\widetilde{\mathcal{S}}(P, \bar{P})$. Then G is absorptive in the sense of $\widetilde{\mathcal{S}}(P, \bar{P})$ if and only if for each elementary cycle γ in G one of the following conditions holds:*

- $l(\gamma) \prec \epsilon$,
- $l(\gamma) = \epsilon$ *and* $\bar{l}(\gamma) \preceq \bar{\epsilon}$.

PROOF. Let $L(\,)$ denote labeling with $\widetilde{\mathcal{S}}(P, \bar{P})$, and let γ be an elementary cycle in G. Absorptiveness in the sense of $\widetilde{\mathcal{S}}(P, \bar{P})$ means that $L(\gamma) \preceq E$, where $E = \{(\epsilon, \bar{\epsilon})\}$ is the unit element in $\widetilde{\mathcal{S}}(P, \bar{P})$. It is easy to check that $L(\gamma) = \{(l(\gamma), \bar{l}(\gamma))\}$, so that absorptiveness reduces to the following equality:

$$\widetilde{\text{eff}}\left(\{(l(\gamma), \bar{l}(\gamma)),\ (\epsilon, \bar{\epsilon})\}\right) = \{(\epsilon, \bar{\epsilon})\}. \tag{4.1}$$

We first prove necessity. The equality (4.1) can be achieved only in two ways:

- The pair $(l(\gamma), \bar{l}(\gamma))$ is dominated by $(\epsilon, \bar{\epsilon})$ in the sense of P. Then it is necessary that $l(\gamma) \prec \epsilon$.
- The pair $(l(\gamma), \bar{l}(\gamma))$ merges with $(\epsilon, \bar{\epsilon})$ so that the result of merging is $(\epsilon, \bar{\epsilon})$. Then it is necessary that $l(\gamma) = \epsilon$ and also $\bar{l}(\gamma) \vee \bar{\epsilon} = \bar{\epsilon}$ (i.e., $\bar{l}(\gamma) \preceq \bar{\epsilon}$).

So at least one of the conditions listed in the proposition statement must hold. Next we prove sufficiency. Suppose that one of the listed conditions holds.

- If it is the first condition (i.e., $l(\gamma) \prec \epsilon$,), then the pair $(l(\gamma), \bar{l}(\gamma))$ is dominated by $(\epsilon, \bar{\epsilon})$ and the left-hand side of (4.1) reduces to $\{(\epsilon, \bar{\epsilon})\}$.
- If it is the second condition, then (because of $l(\gamma) = \epsilon$) the pairs on the left-hand side of (4.1) merge together into $(\epsilon, \bar{l}(\gamma) \vee \bar{\epsilon})$, which is (because of $\bar{l}(\gamma) \preceq \bar{\epsilon}$) equal to $(\epsilon, \bar{\epsilon})$.

Thus in both cases the equality (4.1) is achieved. □

COROLLARY 4.5. *Consider the situation from Proposition 4.4. Suppose additionally that G is absorptive in the sense of both P and $\bar{P}$. Then G must as well be absorptive in the sense of $\widetilde{\mathcal{S}}(P, \bar{P})$.*

PROOF. It follows from Proposition 4.4. Indeed, let γ be any elementary cycle in G. Absorptiveness in the sense of P means that $l(\gamma) \preceq \epsilon$, while absorptiveness in the sense of $\bar{P}$ means that $\bar{l}(\gamma) \preceq \bar{\epsilon}$. If the inequality $l(\gamma) \preceq \epsilon$ is strict, then the first condition from Proposition 4.4 is fulfilled, otherwise the second condition holds. □

COROLLARY 4.6. *Consider again the situation from Proposition 4.4. Suppose additionally that labeling of G with P has the following property: for each elementary cycle γ in G, $l(\gamma) \prec \epsilon$. Then G must be absorptive in the sense of $\widetilde{\mathcal{S}}(P, \bar{P})$ regardless of properties of $\bar{P}$.*

PROOF. It follows directly from Proposition 4.4. Namely the first condition from Proposition 4.4 is fulfilled, so that the second condition does not need to be examined. □

COROLLARY 4.7. *Let a graph G be labeled simultaneously with path algebras $P_1, P_2, \ldots, P_s$ according to s objectives, thereby being absorptive in the sense of P_i for each $i = 1, 2, \ldots, s$. Suppose that for any $i = 1, 2, \ldots, s$ the join operation in P_i is a choice operation and multiplication in P_i has the cancellation property. Let $\bar{P}$ be another path algebra. Suppose that the same graph G is also labeled with $\bar{P}$, being also absorptive in the sense of $\bar{P}$. Let all labels assigned to the same arc be interpreted as a single-pair set from $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$. Then G must be absorptive in the sense of $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$.*

PROOF. It follows directly from Corollary 4.5 by replacing P in Corollary 4.5 with $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$. Thereby, we have in mind Propositions 4.1 and 4.2 and take into account (a version of) Proposition 3.4. □

COROLLARY 4.8. *Let a graph G be labeled simultaneously with path algebras $P_1, P_2, \ldots, P_s$ according to s objectives. Thereby, for each elementary cycle γ in G, the label of γ is $\preceq \epsilon_i$ for any objective i, but $\prec \epsilon_i$ for at least one objective i (here ϵ_i is the unit element in P_i). Suppose that for any i the join operation in P_i is a choice operation and multiplication in P_i has the cancellation property. Let $\bar{P}$ be another path algebra. Suppose that the same graph G is also labeled with $\bar{P}$. Let all labels assigned to the same arc be interpreted as a single-pair set from $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$. Then G must be absorptive in the sense of $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s), \bar{P})$ regardless of properties of $\bar{P}$.*

PROOF. It follows from Corollary 4.6 by replacing P in Corollary 4.6 with $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$. We have again in mind Propositions 4.1 and 4.2 and take into account that ordering in $\widetilde{\mathcal{V}}(P_1, P_2, \ldots, P_s)$ is compatible with orderings in P_i. □

Similarly as the basic method from the previous section, our extended method is primarily intended to solve multi-objective combinations of well-known conventional path problems such as those listed in Table 1. Analogously as in the previous section, careful analysis case-by-case shows that the extended method is applicable to combinations of (at least) the following conventional problems: shortest paths, longest paths, most reliable paths. Feasibility and correctness of the method is thereby guaranteed by Corollary 4.7 or 4.8 combined with Theorem 2.1. The conditions from the corollaries are fulfilled either automatically or trivially or with some natural assumptions regarding the involved graph. If more specific conditions from Corollary 4.8 are fulfilled, then the method works with the linguistic algebra PA, otherwise PE or PO must be used. For instance, the guarantee for computations within Example 4.3 is based on Corollary 4.8, and it is given because the labeled graph in Figure 1 satisfies the conditions from that corollary (i.e., its cycles have strictly positive lengths).

Note that the only path problem from Table 1 whose involvement could prevent applicability of our extended method is the maximum capacity problem. Indeed, the obstacle lies in the associated algebra PC where multiplication does not posses the cancellation property. Thus if our multi-objective problem comprises maximum capacities as one of the objectives, Corollaries 4.7 and 4.8 cannot be applied. At this point, one could argue that the method may still work even with maximum capacities, although we are not able to prove it. However, such possibility is clearly eliminated by the following example.

EXAMPLE 4.9. We consider an instance of a two-objective path problem given in Figure 2, where both objectives refer to capacities. The graph has again the same structure as in Examples 3.3 and 4.3, but now numerical labels on arcs have different meaning. We can imagine that two commodities flow through the graph. The left-hand and right-hand label on an arc specify arc capacity regarding the first and second commodity, respectively. For any pair of nodes i and j, each objective tries to find a path between i and j whose capacity regarding one of the commodities is as large as possible. Similarly as in the previous example, in order to enable identification of paths, each arc is additionally labeled with a distinct letter from the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$.

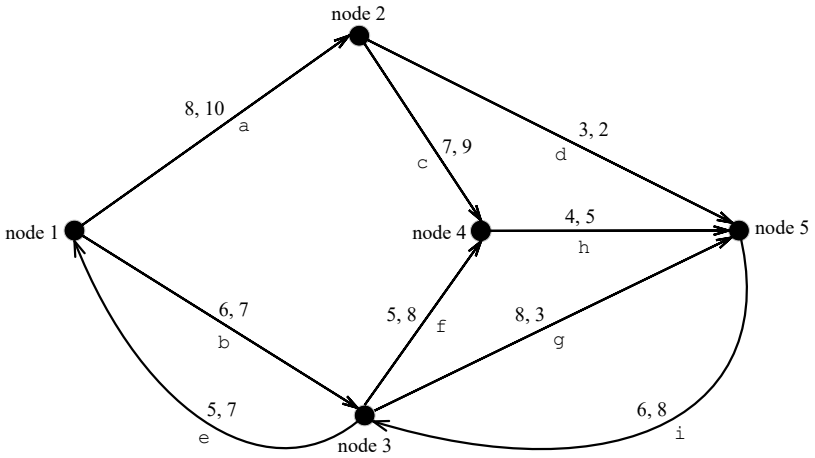

FIGURE 2. An instance of a two-objective maximum-capacity path problem

The considered two-objective problem instance can be specified by the following 5×5 adjacency matrix:

$$A = \begin{bmatrix} \emptyset & \{((8,10),\{\texttt{a}\})\} & \{((6,7),\{\texttt{b}\})\} & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \{((7,9),\{\texttt{c}\})\} & \{((3,2),\{\texttt{d}\})\} \\ \{((5,7),\{\texttt{e}\})\} & \emptyset & \emptyset & \{((5,8),\{\texttt{f}\})\} & \{((8,3),\{\texttt{g}\})\} \\ \emptyset & \emptyset & \emptyset & \emptyset & \{((4,5),\{\texttt{h}\})\} \\ \emptyset & \emptyset & \{((6,8),\{\texttt{i}\})\} & \emptyset & \emptyset \end{bmatrix}$$

This matrix is built over the structure $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(PC, PC), PE)$, where PC is the extremal path algebra from Table 1 associated with the conventional maximum-capacity path problem, and PE is the linguistic algebra from Table 2 for listing elementary paths. By pretending that $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(PC, PC), PE)$ is a path algebra, we try to compute the corresponding closure matrix $\widehat{A}$. The straightforward computation of joins of matrix powers gives the following result:

$$\widehat{A} = \begin{bmatrix} \{((5,7),\{\texttt{be}\})\} & \{((8,10),\{\texttt{a}\})\} & \{((6,7),\{\texttt{b}\})\} & \{((7,9),\{\texttt{ac}\})\} & \{((6,3),\{\texttt{bg}\}),\ ((4,5),\{\texttt{ach}\})\} \\ \{((4,5),\{\texttt{chie}\})\} & \{((4,5),\{\texttt{chiea}\})\} & \{((4,5),\{\texttt{chi}\})\} & \{((7,9),\{\texttt{c}\})\} & \{((4,5),\{\texttt{ch}\})\} \\ \{((5,7),\{\texttt{e}\})\} & \{((5,7),\{\texttt{ea}\})\} & \{((5,7),\{\texttt{eb}\}),\ ((6,3),\{\texttt{gi}\})\} & \{((5,8),\{\texttt{f}\})\} & \{((8,3),\{\texttt{g}\}),\ ((4,5),\{\texttt{fh, each}\})\} \\ \{((4,5),\{\texttt{hie}\})\} & \{((4,5),\{\texttt{hiea}\})\} & \{((4,5),\{\texttt{hi}\})\} & \{((4,5),\{\texttt{hif, hieac}\})\} & \{((4,5),\{\texttt{h}\})\} \\ \{((5,7),\{\texttt{ie}\})\} & \{((5,7),\{\texttt{iea}\})\} & \{((6,8),\{\texttt{i}\})\} & \{((5,8),\{\texttt{if}\})\} & \{((6,3),\{\texttt{ig}\}),\ ((4,5),\{\texttt{ifh, ieach}\})\} \end{bmatrix}$$

According to the $(1,5)$-th entry of $\widehat{A}$, there are only two efficient paths from node 1 to node 5. The first of them goes through arcs `b` and `g`, and its capacities for different objectives are given by the vector $(6,3)$. The second path crosses arcs `a`, `c` and `h` and achieves the capacity vector $(4,5)$. The two listed paths are really efficient, namely the first one is better according the first objective, and the second one according the second objective. Still, the obtained solution set is not complete. Namely, it is easy to check that there exists a third efficient path, going through arcs `b`, `f` and `h`, achieving the same vector $(4,5)$ as the second path. So we have found a situation where the method fails.

The main reason why our method did not produce correct results is simply because $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(PC,PC),PE)$ is not really a path algebra. Some of the path-algebra properties are violated, and therefore the computed results can be different depending on ordering of algebraic operations. In our attempt to compute the closure $\widehat{A}$, matrix powers have been generated one after another and joined together. Thereby A^3 has been obtained by multiplying A^2 with A. Thus paths consisting of three arcs have been generated by extending two-arc paths with an additional arc. In this way, the three-arc path `b`,`f`,`h` should have been generated from the two-arc path `b`,`f` by extension with `h`. But our algorithm discarded `b`,`f` since it connects the same nodes 1 and 4 as `a`,`c` and is dominated by `a`,`c`. This was a mistake since `b`,`f` would again become efficient after extension with `h`.

One may wonder how is it possible that a non-efficient path can become efficient after it is extended with one more arc. The reason for such unexpected behavior is lack of cancellation property, which originates in PC but spreads also to vectors over PC. To see the whole phenomenon more clearly, let us denote the capacity vector of the path `b`,`f` with $\vec{x}=(5,7)$, the vector of `a`,`c` with $\vec{y}=(7,9)$, and the vector of the arc `h` with $\vec{z}=(4,5)$. Then the capacities of the path `b`,`f`,`h` are computed as $\vec{x}\circ\vec{z}=(4,5)$, and the capacities of `a`,`c`,`h` are obtained as $\vec{y}\circ\vec{z}=(4,5)$. The whole situation is summarized as follows:

$$\vec{x}\prec\vec{y},\ \vec{z}\neq\vec{\phi},\ \text{but}\ \ \vec{x}\circ\vec{z}=\vec{y}\circ\vec{z},$$

which is a clear violation of the cancellation property. Namely, if we were allowed to cancel $\vec{z}$ in the last equation, we would obtain $\vec{x}=\vec{y}$, thus a contradiction.

According to Example 4.9, the cancellation property stated in Corollaries 4.7 and 4.8 is really essential for proper working of our extended method. However, from the practical point of view, the method can still be regarded as useful even if such property is missing. Namely, computation of efficient vectors is still correct since it mimics simpler computation from the previous section. Also, each vector will be accompanied with at least one efficient path.

Thus the method can be used if we do not insist on identifying all efficient paths for the same vector and do not care which of several possible paths has been identified.

5. Solving robust path problems

Let us consider a robust path problem where uncertainty is expressed through s scenarios. Obviously, such problem can be interpreted as a multi-objective problem with s objectives. Thereby each objective corresponds to one scenario. So the initial robust problem reduces to a special type of multi-objective problem where all objectives are of the same type, thus labeling the graph repeatedly with the same path algebra.

According to the above interpretation, an instance of a robust path problem can be solved by using the methods from Sections 3 and 4. More precisely, an instance is specified by an adjacency matrix over a vector-set algebra of the form $\mathcal{S}(\mathcal{V}(P, P, \ldots, P))$ or a composite vector-set algebra of the form $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(P, P, \ldots, P), \bar{P})$. The scalar algebra P here is a suitable extremal algebra, and $\bar{P}$ is a linguistic algebra. The problem instance is solved by computing the corresponding closure matrix.

As an example of solving a robust path problem instance we can consider again Example 4.9. Indeed, the graph from Figure 2 can be reinterpreted as an instance of the robust maximum capacity problem. Numerical arc labels then become arc capacities under two scenarios. If we ignore arc identifiers (letters), computation within the vector-set algebra $\mathcal{S}(\mathcal{V}(PC, PC))$ will be correct, and the same sets of efficient vectors will be obtained as already shown in Example 4.9. Each vector corresponds to (at least) one efficient path, and it contains capacities of that path under different scenarios. Another example of the same kind will be given soon as Example 5.1.

In Section 1 we have already noticed that our methodology for solving robust problems by computing whole sets of efficient vectors differs from the usual practice found in the literature. Namely, the usual practice is that only one "robustly optimal" solution is selected according to some more specific criterion. In the next paragraphs we will demonstrate how our approach can be harmonized with the usual approach. More precisely, we will show that a robustly optimal solution can be selected a posteriori, by applying a suitable ranking function to all vectors within a set of solutions. Ranking should allow comparison of all vectors, and the desired single solution should correspond to the best ranked vector. We will define two abstract ranking procedures, which can be expressed in terms of algebraic operations from the corresponding scalar path algebra P. It will turn out that the two procedures are generalizations of the standard min-max and min-max regret criterion, respectively ([1]).

Our first way of ranking vectors is called *abstract min-max*, and it is feasible only if the join operation $\vee$ in P is a choice operation. Obviously, with such $\vee$, any x and y from P are comparable according to the ordering $\preceq$, namely the join $x \vee y$ is either x or y, which means that either $y \preceq x$ or $x \preceq y$. Consequently, any finite set of elements from P is totally ordered, thus containing the smallest and the greatest element according to $\preceq$. Let us denote the corresponding operators for finding the smallest and the greatest element by $\min_{\preceq}$ and $\max_{\preceq}$, respectively. We use such notation in order to distinguish these operators from the standard min and max for real numbers.

The abstract min-max works in the following way. For a vector $\vec{x} = (x_1, x_2, \ldots, x_s)$ from $\mathcal{V}(P, P, \ldots, P)$, its rank $r(\vec{x})$ is computed as

$$r(\vec{x}) = r(x_1, x_2, \ldots, x_s) = \min_{\preceq}\{x_1, x_2, \ldots, x_s\}.$$

The best ranked vector $\vec{z}$ in a finite set X of (efficient) vectors is the one whose rank is the greatest according to $\preceq$, i.e.,

$$r(\vec{z}) = \max_{\preceq}\{r(\vec{x}) \,|\, \vec{x} \in X\}.$$

Note that the proposed ranking $r(\ )$ is sound in the sense that the best rank always belongs to an efficient vector. Indeed, let $\vec{y} = (y_1, y_2, \ldots, y_s)$ be inefficient. Then $\vec{y}$ is dominated by some efficient $\vec{x} = (x_1, x_2, \ldots, x_s)$, i.e., $y_i \preceq x_i$, $i = 1, 2, \ldots, s$. Thus

$$r(\vec{y}) = \min_{\preceq}\{y_1, y_2, \ldots, y_s\} \preceq \min_{\preceq}\{x_1, x_2, \ldots, x_s\} = r(\vec{x}).$$

So it really makes sense to rank only efficient vectors.

Note also that both $\min_{\preceq}$ and $\max_{\preceq}$ can be interpreted algebraically as evaluation of certain algebraic expressions. Indeed,

$$\min_{\preceq}\{x_1, x_2, \ldots, x_s\} = x_1 \wedge x_2 \wedge \ldots \wedge x_s,$$

where $\wedge$ is the choice operation complementary to $\vee$, i.e., for any $x, y \in P$:

$$x \wedge y = \begin{cases} x & \text{if } x \vee y = y \\ y & \text{if } x \vee y = x \end{cases}.$$

Similarly, $\max_{\preceq}$ can be evaluated by using $\vee$, e.g.,

$$\max_{\preceq}\{r(\vec{x}) \,|\, \vec{x} \in X\} = \bigvee_{\vec{x} \in X} r(\vec{x}).$$

Now we define our second ranking, which is called *abstract min-max regret*. As before, the join operation in P is assumed to be a choice operation. Additionally, it is also assumed that multiplication in P allows inversion. Thus for each element $x \in P$, $x \neq \phi$, there is another element x^{-1} such that

$$x \circ x^{-1} = x^{-1} \circ x = \epsilon.$$

The notation here is standard, i.e., ϕ denotes the zero element in P, while ϵ is the unit element in P. Sometimes x^{-1} cannot be found in the original P, but only in a larger path algebra that includes P as its subset.

Prior to starting the ranking procedure, it is necessary to solve s instances of the corresponding conventional path problem. They are obtained from the given robust instance by considering each of its scenarios separately. Let us denote the (optimal) solution value of the i-th conventional instance by ω_i. Assume that $\omega_i \neq \phi$.

The abstract min-max regret ranking can briefly be described as the (previously defined) abstract min-max ranking applied to modified vectors. Modification is done by multiplying each vector entry x_i with the inverse of the corresponding ω_i. The idea is to measure the "regret" for each scenario, i.e., deviation of the actual solution value vs. the optimal value.

Going into more details, the abstract min-max regret ranking can be described as follows. For a vector $\vec{x} = (x_1, x_2, \ldots, x_s)$ from $\mathcal{V}(P, P, \ldots, P)$, its rank $r'(\vec{x})$ is computed as

$$r'(\vec{x}) = r'(x_1, x_2, \ldots, x_s) = \min_{\preceq}\{x_1 \circ \omega_1^{-1}, x_2 \circ \omega_2^{-1}, \ldots, x_s \circ \omega_s^{-1}\}.$$

The best ranked vector $\vec{z}$ in a set X of (efficient) vectors is the one whose rank $r'(\,)$ is the greatest according to $\preceq$, i.e.,

$$r'(\vec{z}) = \max_{\preceq}\{r'(\vec{x}) \,|\, \vec{x} \in X\}.$$

The proposed ranking $r'(\,)$ is again sound in the sense that the best rank always belongs to an efficient vector. Indeed, suppose that $\vec{y} = (y_1, y_2, \ldots, y_s)$ is dominated by $\vec{x} = (x_1, x_2, \ldots, x_s)$. Then $y_i \preceq x_i$ and also $y_i \circ \omega_i^{-1} \preceq x_i \circ \omega_i^{-1}$ for all $i = 1, 2, \ldots, s$. Consequently,

$$\begin{aligned} r'(\vec{y}) &= \min_{\preceq}\{y_1 \circ \omega_1^{-1}, y_2 \circ \omega_2^{-1}, \ldots, y_s \circ \omega_s^{-1}\} \\ &\preceq \min_{\preceq}\{x_1 \circ \omega_1^{-1}, x_2 \circ \omega_2^{-1}, \ldots, x_s \circ \omega_s^{-1}\} = r'(\vec{x}). \end{aligned}$$

So again, there is no need to rank inefficient vectors.

The whole procedure of solving a robust path problem instance with a posteriori ranking of efficient solutions is illustrated by the next example. Both ranking functions are applied and compared.

EXAMPLE 5.1. We consider an instance of the robust shortest-path problem given in Figure 3. The shown graph G has arcs whose numerical labels denote lengths. There are two scenarios for arc lengths given by left-hand and right-hand labels, respectively. For any pair of nodes i and j we would like to find shortest paths between i and j, thus paths whose sum of arc lengths under any scenario is as small as possible. In order to enable identification of paths, each arc is additionally labeled with a distinct letter from the alphabet $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \ldots\}$.

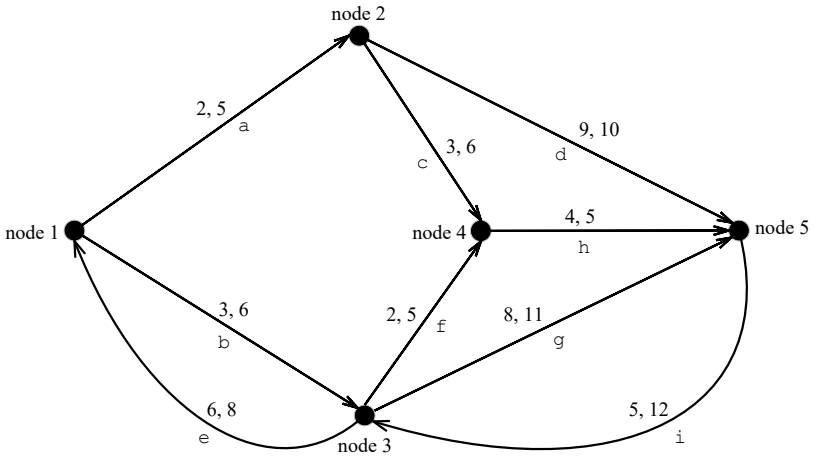

FIGURE 3. An instance of the robust shortest path problem with two scenarios

The considered robust problem instance is specified by the following 5×5 adjacency matrix:

$$A = \begin{bmatrix} \emptyset & \{((2,5),\{\mathtt{a}\})\} & \{((3,6),\{\mathtt{b}\})\} & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \{((3,6),\{\mathtt{c}\})\} & \{((9,10),\{\mathtt{d}\})\} \\ \{((6,8),\{\mathtt{e}\})\} & \emptyset & \emptyset & \{((2,5),\{\mathtt{f}\})\} & \{((8,11),\{\mathtt{g}\})\} \\ \emptyset & \emptyset & \emptyset & \emptyset & \{((4,5),\{\mathtt{h}\})\} \\ \emptyset & \emptyset & \{((5,12),\{\mathtt{i}\})\} & \emptyset & \emptyset \end{bmatrix}$$

The composite vector-set algebra involved here is $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(PS,PS),PA)$, where PS is the extremal algebra from Table 1 associated with the conventional shortest-path problem, and PA is the linguistic algebra from Table 2 associated with the problem of listing all paths. The corresponding closure matrix computed within $\widetilde{\mathcal{S}}(\widetilde{\mathcal{V}}(PS,PS),PA)$ is:

$\widehat{A} =$

$$\begin{bmatrix} \{((9,14),\{\mathtt{be}\})\} & \{((2,5),\{\mathtt{a}\})\} & \{((3,6),\{\mathtt{b}\})\} & \{((5,11),\{\mathtt{ac},\mathtt{bf}\})\} & \{((11,15),\{\mathtt{ad}\}),((9,16),\{\mathtt{ach},\mathtt{bfh}\})\} \\ \{((20,30),\{\mathtt{die}\}),((18,31),\{\mathtt{chie}\})\} & \{((22,35),\{\mathtt{diea}\}),((20,36),\{\mathtt{chiea}\})\} & \{((14,22),\{\mathtt{di}\}),((12,23),\{\mathtt{chi}\})\} & \{((3,6),\{\mathtt{c}\})\} & \{((9,10),\{\mathtt{d}\}),((7,11),\{\mathtt{ch}\})\} \\ \{((6,8),\{\mathtt{e}\})\} & \{((8,13),\{\mathtt{ea}\})\} & \{((9,14),\{\mathtt{eb}\})\} & \{((2,5),\{\mathtt{f}\})\} & \{((6,10),\{\mathtt{fh}\})\} \\ \{((15,25),\{\mathtt{hie}\})\} & \{((17,30),\{\mathtt{hiea}\})\} & \{((9,17),\{\mathtt{hi}\})\} & \{((11,22),\{\mathtt{hif}\})\} & \{((4,5),\{\mathtt{h}\})\} \\ \{((11,20),\{\mathtt{ie}\})\} & \{((13,25),\{\mathtt{iea}\})\} & \{((5,12),\{\mathtt{i}\})\} & \{((7,17),\{\mathtt{if}\})\} & \{((11,22),\{\mathtt{ifh}\})\} \end{bmatrix}$$

The solutions of our problem instance can directly be read from $\widehat{A}$. For instance, from the $(1,5)$-th entry of $\widehat{A}$ we can read that there are exactly three efficient paths from node 1 to node 5. The first path goes through arcs

denoted by `a` and `d`, the second through `a`, `c` and `h`, and the third through `b`, `f` and `h`. The second and the third path are equivalent in the sense that they achieve the same length for any scenario, i.e., 9 (for the first scenario) and 16 (for the second scenario), respectively. The lengths of the first path depending on the scenarios are 11 and 15, respectively, which means that the first path is longer than the other two paths under the first scenario, but shorter under the second scenario.

Next we analyze application of the abstract min-max ranking to the considered robust problem. As it can easily be verified, in the associated scalar algebra PS the operator $\min_{\preceq}$ is equivalent to the standard max, while $\max_{\preceq}$ is equivalent to the standard min. Therefore, in the context of robust shortest paths, the abstract min-max reduces to the standard min-max known from the literature. So the robustly optimal path is the one whose maximal length, measured over all scenarios, is as small as possible.

To be concrete, let us consider again only efficient paths between nodes 1 and 5, described by the set of pairs $\{\,((11,15),\{\mathtt{ad}\}),\,((9,16),\{\mathtt{ach},\mathtt{bfh}\})\,\}$. Let us see how the abstract min-max procedure works on that data. The first phase of ranking replaces each vector with its rank $r(\ )$, so that the set transforms into $\{\,(15,\{\mathtt{ad}\}),\,(16,\{\mathtt{ach},\mathtt{bfh}\})\,\}$. The second phase applies for instance the operator $\widetilde{\mathrm{eff}}(\ )$ to the transformed set, thus extracting the pair with the best rank, i.e., $(15,\{\mathtt{ad}\})$. Consequently, the robustly optimal path goes through arcs `a` and `d`, and its length is at most 15 under any scenario.

Finally, we analyze application of the abstract min-max regret ranking to the considered problem. Since multiplication in the algebra PS is in fact addition of real numbers, the expression of the form $x_i \circ \omega_i^{-1}$ appearing in our ranking is in fact $x_i - \omega_i$. Obviously, $x_i - \omega_i$ can be interpreted as absolute deviation of the actual path length under a certain scenario vs. the optimal length for that scenario. Consequently, in the context of robust shortest paths, the abstract min-max regret reduces to the standard min-max regret known from the literature. The robustly optimal path is the one whose maximal deviation from the optimal length, measured over all scenarios, is as small as possible.

Let us now test how the abstract min-max regret works on efficient paths between nodes 1 and 5, which are described by the previously shown set of pairs. Before starting the ranking procedure, it is necessary to solve the conventional shortest-path problem instances for each scenario separately. As it can easily be verified, the shortest paths under the first and second scenario have lengths 9 and 15, respectively. The first phase of ranking replaces each vector with its rank $r'(\ )$. The rank of the first vector is $\max\{11-9, 15-15\} = 2$, while the rank of the second vector is $\max\{9-9, 16-15\} = 1$. Consequently, the set of pairs transforms into $\{\,(2,\{\mathtt{ad}\}),\,(1,\{\mathtt{ach},\mathtt{bfh}\})\,\}$. The second phase of ranking applies the operator $\widetilde{\mathrm{eff}}(\ )$ to the transformed set, thus leaving only

the second pair having a better rank. Consequently, the robustly optimal path is either a,c,h or b,f,h, and its length differs from the optimal length by at most 1 under any scenario. Note that the paths selected by the min-max regret criterion are not the same as for the min-max criterion.

6. Conclusions

In this paper the algebraic approach from [5] has been applied to multi-objective and robust variants of path problems. More precisely, it has been shown that multi-objective problems are concrete instances of the same abstract algebraic problem as conventional (single-objective) problems. Each problem (either multi-objective or conventional) is characterized by a different path algebra, i.e., a different instance of the same abstract algebraic structure. Thereby, the newly constructed algebras used for multi-objective problems are more complex, and they incorporate algebras associated with conventional problems as their building blocks. The same algebraic construction covers also robust path problems with discrete scenarios since they can be considered as special cases of multi-objective problems. The paper contains proofs showing that all newly constructed path algebras are correctly defined.

According to our results, an instance of a multi-objective or robust path problem, posed in a directed graph, can be specified by the adjacency matrix of that graph over the corresponding path algebra, and solved by computing the closure of that matrix within that algebra. It has been proved that, under some plausible conditions, the involved adjacency matrix is absorptive, which guarantees that its closure is computable and meaningful. For a pair of graph nodes, the solutions are expressed as a set of efficient paths connecting those nodes. Such paths are identified explicitly, or described implicitly by vectors of objective-function values obtained for different objectives/scenarios. In case of a robust problem it is also possible to extract only one "robustly optimal" solution by applying suitable algebraic ranking functions. The rankings proposed in the paper can be regarded as generalizations of the traditional min-max or min-max regret criteria.

The results of the paper are interesting because their provide an elegant and clear setting for all considered problems. Indeed, multi-objective, robust and conventional path problems conform to a common algebraic pattern, i.e., they can be described in the same matrix form and solved by the same general matrix-closure algorithms. Moreover, thanks to absorptiveness, the most-efficient closure algorithms from [5] can be used, which are based on Gaussian elimination or Gauss-Seidel iteration. The main advantage of the considered algorithms is their generality - there is no need to invent a dedicated solution procedure for each particular problem variant.

Computational speed of a general algorithm is measured in terms of the involved abstract algebraic operations. Unfortunately, for a multi-objective or

robust problem, a single algebraic operation is already complex by itself. As a consequence, application of a fast general algorithm to a multi-objective or robust problem can still be extremely demanding in terms of actual computing time. This is in fact not a surprise. Namely, it is well known that a set of Pareto-efficient solutions can have exponential size. Also, it is well known that robust path problems are NP-hard. Thus we cannot expect from the considered algorithms to be really efficient on large problem instances.

An interesting topic for future work would be to develop good computer implementations of the path algebras proposed in this paper. Such implementations should be incorporated into a general-purpose software package for solving path problems. With that package, application of general path-finding algorithms to multi-objective or robust path problems could become more attractive. At least, smaller instances having only few objectives/scenarios could be solved in an ad-hoc fashion without bothering with dedicated algorithms.

Acknowledgements.

This work has been fully supported by Croatian Science Foundation under the project IP-2018-01-5591. The author would like to thank the reviewers for their useful remarks on an earlier version of the paper.

References

[1] H. Aissi, C. Bazgan and D. Vanderpooten, *Min-max and min-max regret versions of combinatorial optimization problems: A survey*, European J. Oper. Res. **197** (2009), 427–438.

[2] R. Backhouse, *Regular algebra applied to language problems*, J. Log. Algebr. Program. **66** (2006), 71–111.

[3] J.S. Baras and G. Theodorakopoulos, Path problems in networks, Synthesis Lectures on Communication Networks, Morgan & Claypool Publishers, San Rafael CA, 2010.

[4] D. Bertsimas, D.B. Brown and C. Caramanis, *Theory and applications of robust optimization*, SIAM Rev. **53** (2011), 464–501.

[5] B. Carré, Graphs and networks, Oxford University Press, Oxford, 1979.

[6] P. de la Torre and C.P. Kruskal, *Fast parallel algorithms for all-sources lexicographic search and path-algebra problems*, J. Algorithms **19** (1995), 1–24.

[7] C.T. Djamégni, P. Quinton, S. Rajopadhye and T. Risset, *Derivation of systolic algorithms for the algebraic path problem by recurrence transformations*, Parallel Comput. **26** (2000), 1429–1445.

[8] M. Ehrgott, Multicritera optimization, 2nd Edition, Springer, Berlin, 2010.

[9] J.S. Golan, Semirings and their applications, Kluwer Academic Publishers, Dortrecht, 1999.

[10] M. Gondran and M. Minoux, Graphs, dioids and semirings. New models and algorithms, Springer, Berlin, 2008.

[11] A.J.T. Gurney and T.G. Griffin, *Lexicographic products in metarouting*, in: Proceedings of the International Conference on Network Protocols (ICNP), Beijing, China, October 16-19, 2007, (eds. K.L. Calvert and D. Yau), IEEE Computer Society, Piscataway NJ, 2007, 113–122.

[12] T.G. Griffin, *Exploring the stratisfied shortest-paths problem*, Netw. Sci. **1** (2012), 2–14.

[13] U. Huckenbeck, Extremal paths in graphs, Akademie Verlag, Berlin, 1997.

[14] D. Jungnickel, Graphs, networks and algorithms, Fourth Edition, Springer, Berlin, 2013.
[15] A. Kasperski and P. Zielinski, *Robust discrete optimization under discrete and interval uncertainty: A survey*, in: Robustness Analysis in Decision Aiding, Optimization, and Analytics (eds. M, Doumpos, C. Zopounidis and E. Grigoroudis), Springer, Cham CH, 2016, 113–143.
[16] V.N. Kolokoltsov, *Idempotent structures in optimization*, Journal of Mathematical Sciences **104** (2001), 847–880.
[17] P. Kouvelis and G. Yu, Robust discrete optimization and its applications, Springer, Berlin, 1997.
[18] J-Y. Le Boudec and P. Thiran, *Network calculus. A theory of deterministic queuing systems for the internet*, Lecture Notes in Computer Science **2050**, Springer, Berlin, Online Version April 26, 2012.
[19] F.S. Lobato and V. Steffen Jr., Multi-objective optimization problems: concepts and self-adaptive parameters with mathematical and engineering applications, Springer, Cham, 2017.
[20] R. Manger, *Gaussian block algorithms for solving path problems*, Math. Commun. **3** (1998), 67–81.
[21] R. Manger, *Solving path problems on a network of computers*, Informatica (Ljubl.) **26** (2002) 91–100.
[22] R. Manger, *A new path algebra for finding paths in graphs*, in: Proceedings of the 26th International Conference on Information Technology Interfaces - ITI 2004, Cavtat, Croatia, June 7-10, 2004. (eds. V. Lužar-Stiffler, and V. Hljuz Dobrić), University Computing Centre, Zagreb, 2004, 657–662.
[23] R. Manger, *Composite path algebras for solving path problems in graphs*, Ars Combin. **78** (2006), 137–150.
[24] M. Mohri, *Semiring frameworks and algorithms for shortest-distance problems*, J. Autom. Lang. Comb. **7** (2002), 321–350.
[25] G. Rote, *Path problems in graphs*, Comput. Suppl. **7** (1990), 155–189.
[26] M. Russling, *Deriving a class of layer-oriented graph algorithms*, Sci. Comput. Programming **26** (1996), 117–132.
[27] K.K. Somasundaram and J.S. Baras, *Solving multi-metric network problems: An interplay between idempotent semiring rules*, Linear Algebra Appl. **435** (2011), 1494–1512.
[28] A. Wongseelashote, *Semirings and path spaces*, Discrete Math. **26** (1979), 55–78.

R. Manger
Department of Mathematics
Faculty of Science
University of Zagreb
Bijenička cesta 30, 10 000 Zagreb
Croatia
E-mail: manger@math.hr

Received: 7.3.2019.
Revised: 30.8.2019.