Approximating min-max (regret) versions of some polynomial problems

Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten

LAMSADE, Université Paris-Dauphine, France
{aissi,bazgan,vdp}@lamsade.dauphine.fr

Abstract. While the complexity of min-max and min-max regret versions of most classical combinatorial optimization problems has been thoroughly investigated, there are very few studies about their approximation. For a bounded number of scenarios, we establish a general approximation scheme which can be used for min-max and min-max regret versions of some polynomial problems. Applying this scheme to shortest path and minimum spanning tree, we obtain fully polynomial-time approximation schemes with much better running times than the ones previously presented in the literature.

Keywords: min-max, min-max regret, approximation, fptas, shortest path, minimum spanning tree.

1 Introduction

The definition of an instance of a combinatorial optimization problem requires to specify parameters, in particular objective function coefficients, which may be uncertain or imprecise. Uncertainty/imprecision can be structured through the concept of *scenario* which corresponds to an assignment of plausible values to parameters. There exist two natural ways of describing the set of all possible scenarios. In the *interval data case*, each numerical parameter can take any value between a lower and an upper bound. In the *discrete scenario case*, which is considered here, the scenario set is described explicitly. Kouvelis and Yu [6] proposed the min-max and min-max regret criteria, stemming from decision theory, to construct solutions hedging against parameters variations. The min-max criterion aims at constructing solutions having a good performance in the worst case. The min-max regret criterion, less conservative, aims at obtaining a solution minimizing the maximum deviation, over all possible scenarios, of the value of the solution from the optimal value of the corresponding scenario.

Complexity of the min-max and min-max regret versions has been studied extensively during the last decade. In [6], for the discrete scenario case, the

[⋆] This work has been partially funded by grant CNRS/CGRI-FNRS number 18227. The second author was partially supported by the ACI Sécurité Informatique grant-TADORNE project 2004.

complexity of min-max (regret) versions of several combinatorial optimization problems was studied, including shortest path and minimum spanning tree. In general, these versions are shown to be harder than the classical versions. More precisely, if the number of scenarios is not constant, these problems become strongly *NP*-hard, even when the classical problems are solvable in polynomial time. On the other hand, for a constant number of scenarios, min-max (regret) versions of these polynomial problems usually become weakly *NP*-hard.

While the complexity of these problems was studied thoroughly, their approximation was not studied until now, except in [2]. That paper investigated the relationships between min-max (regret) and multi-objective versions, and showed the existence, in the case of a bounded number of scenarios, of fully polynomial-time approximation schemes (fptas) for min-max versions of several classical optimization problems (shortest path, minimum spanning tree, knapsack). The interest of studying these relationships is that, unlike for min-max (regret) versions, fptas, which determine an approximation of the non-dominated set (or Pareto set), have been proposed for the multi-objective version (see, e.g., [9, 11]). Approximation algorithms for the min-max version, which basically consist of selecting one min-max solution from an approximation of the non-dominated set, are then easy to derive but critically depend on the running time of the approximation scheme for the multi-objective version.

In this paper, we adopt an alternative perspective and develop a general approximation scheme, using the scaling technique, which can be applied to min-max (regret) versions of some problems, provided that some conditions are satisfied. The advantage of this approach is that the resulting fptas usually have a much better running time than those derived using multi-objective fptas.

After presenting some background concepts in section 2, we introduce in section 3 the general approximation scheme. In section 4 we present applications of this general scheme to shortest path and minimum spanning tree, giving in each case fptas with better running times than previously known fptas based on multi-objective versions.

2 Preliminaries

We consider in this paper the class $\mathcal{C}$ of 0-1 problems with a linear objective function defined as:

$$\begin{cases} \min \sum_{i=1}^{m} c_i x_i & c_i \in \mathbb{N} \\ x \in X \subset \{0,1\}^m \end{cases}$$

This class encompasses a large variety of classical combinatorial problems, some of which are polynomial-time solvable (shortest path problem, minimum spanning tree, ...) and others are *NP*-hard (knapsack, set covering, ...). The size of a solution $x \in X$ is the number of variables x_i which are set to 1.

2.1 Min-max, min-max regret versions

Given a problem $\mathcal{P} \in \mathcal{C}$, the min-max (regret) version associated to $\mathcal{P}$ has for input a finite set of scenarios S where each scenario $s \in S$ is represented by a vector $(c_1^s, \dots, c_m^s)$. We denote by $val(x,s) = \sum_{i=1}^m c_i^s x_i$ the value of solution $x \in X$ under scenario $s \in S$ and by val_s^* the optimal value in scenario s.

The min-max optimization problem corresponding to $\mathcal{P}$, denoted by Min-Max $\mathcal{P}$, consists of finding a solution x having the best worst case value across all scenarios, which can be stated as: $\min_{x \in X} \max_{s \in S} val(x,s)$.

Given a solution $x \in X$, its *regret*, $R(x,s)$, under scenario $s \in S$ is defined as $R(x,s) = val(x,s) - val_s^*$. The *maximum regret* $R_{max}(x)$ of solution x is then defined as $R_{max}(x) = \max_{s \in S} R(x,s)$.

The min-max regret optimization problem corresponding to $\mathcal{P}$, denoted by Min-Max Regret $\mathcal{P}$, consists of finding a solution x minimizing the maximum regret $R_{max}(x)$ which can be stated as: $\min_{x \in X} \max_{s \in S}\{val(x,s) - val_s^*\}$.

When $\mathcal{P}$ is a maximization problem, the max-min and min-max regret versions associated to $\mathcal{P}$ are defined similarly.

2.2 Approximation

Let us consider an instance I, of size $|I|$, of an optimization problem and a solution x of I. We denote by *opt*(I) the optimum value of instance I. The *performance ratio* of x is $r(x) = \max\left\{\frac{val(x)}{opt(I)}, \frac{opt(I)}{val(x)}\right\}$, and its *error* is $\varepsilon(x) = r(x) - 1$. For a function f, an algorithm is an $f(n)$*-approximation algorithm* if, for any instance I of the problem, it returns a solution x such that $r(x) \le f(|I|)$. An optimization problem has a *fully polynomial-time approximation scheme* (an *fptas*, for short) if, for every constant $\varepsilon > 0$, it admits an $(1+\varepsilon)$-approximation algorithm which is polynomial both in the size of the input and in $1/\varepsilon$. The class of problems having an fptas is denoted by *FPTAS*.

2.3 Matrix tree theorem

In this section we describe classical results concerning the *matrix tree theorem* that will enable us to derive approximation schemes for min-max and min-max regret versions of spanning tree.

The matrix tree theorem provides a way of counting all the spanning trees in a graph (see, e.g., [10]). Consider a graph $G = (V,E)$ with $|V| = n$, $|E| = m$ and let c_{ij} denote the cost of edge $(i,j) \in E$.

Define an $n \times n$ matrix A whose entries are given as follows:

$$a_{ij} = \begin{cases} -c_{ij} \text{ if } i \neq j \text{ and } (i,j) \in E \\ \sum_{(i,\ell) \in E} c_{i\ell} \text{ if } i = j \\ 0 \text{ otherwise} \end{cases}$$

Define A_r as the submatrix of A obtained by deleting the r^{th} row and column and $\mathcal{D}(A_r)$ as its determinant. The matrix tree theorem [10] states the following equality:

$$\mathcal{D}(A_r) = \sum_{T \in \mathcal{T}} \prod_{(i,j) \in T} c_{ij} \tag{1}$$

where $\mathcal{T}$ is the set of all spanning trees of G.

As indicated in [3], this theorem can be extended to count the number of spanning trees of value v for each possible value v using a matrix depending on one variable. Following this idea, we can extend the matrix tree theorem to the multiple scenarios case as in [5]. Define the $n \times n$ matrix $A(y_1, \dots, y_k)$ as follows:

$$a_{ij}(y_1, \dots, y_k) = \begin{cases} -\prod_{s=1}^{k} y_s^{c_{ij}^s} \text{ if } i \neq j \text{ and } (i,j) \in E \\ \sum_{(i,\ell)\in E} \prod_{s=1}^{k} y_s^{c_{i\ell}^s} \text{ if } i = j \\ 0 \text{ otherwise} \end{cases}$$

Then, the determinant of the submatrix $A_r(y_1, \dots, y_k)$ obtained by deleting any r^{th} row and column is given by

$$\mathcal{D}(A_r(y_1, \dots, y_k)) = \sum_{v_1, \dots, v_k \in V^{\mathcal{T}}} a_{v_1, \dots, v_k} \prod_{s=1}^{k} y_s^{v_s} \tag{2}$$

where $a_{v_1,\dots,v_k}$ is the number of spanning trees with value v_s in scenario s, for all $s \in S$ and $V^{\mathcal{T}}$ is the set of values reached on all scenarios, for all spanning trees of G.

Equality (2) is obtained by replacing each c_{ij} in (1) by $\prod_{s=1}^{k} y_s^{c_{ij}^s}$. Then each product term in (1) corresponding to tree T becomes $\prod_{s=1}^{k} y_s^{\sum_{(i,j)\in T} c_{ij}^s}$.

3 A general approximation scheme

We establish now a general result giving a sufficient condition for the existence of fptas for min-max (regret) versions of problems $\mathcal{P}$ in $\mathcal{C}$.

Theorem 1. *Given a problem* Min-Max (Regret) $\mathcal{P}$, *if*

1. *for any instance I, a lower and an upper bound L and U of opt can be computed in time $p(|I|)$, such that $U \leq q(|I|)L$, where p and q are two polynomials with q non decreasing and $q(|I|) \geq 1$,*
2. *and there exists an algorithm that finds for any instance I an optimal solution in time $r(|I|, U)$ where r is a non decreasing polynomial,*

then Min-Max (Regret) $\mathcal{P}$ *is in FPTAS.*

Proof. Let I be an instance of Min-Max $\mathcal{P}$ or Min-Max Regret $\mathcal{P}$ defined on a scenario set S where each scenario $s \in S$ is represented by a vector $(c_1^s, \dots, c_m^s)$. We use the technique of *scaling* in order to provide an fptas. In order to obtain a solution with an error bounded by ε, we need a lower bound of *opt*(I). Moreover, for obtaining a polynomial algorithm, we have to use a lower bound that is polynomially related to an upper bound.

When I is an instance of Min-Max $\mathcal{P}$, consider $\overline{I}$ the instance of Min-Max $\mathcal{P}$ derived from I where each scenario $s \in S$ is represented by a vector $(\overline{c}_1^s, \dots, \overline{c}_m^s)$, with $\overline{c}_i^s = \lfloor \frac{tc_i^s}{\varepsilon L} \rfloor$ and t is an upper bound of the size of any feasible solution of I. Let x^* and $\overline{x}^*$ denote respectively an optimal solution of instance I and $\overline{I}$. Let $\overline{val}(x,s)$ denote the value of a solution x in scenario s for $\overline{I}$. We have

$$c_i^s < \frac{\varepsilon L}{t}(\overline{c}_i^s + 1), \text{ for all } s \in S,$$

and thus, $val(\overline{x}^*, s) < \frac{\varepsilon L}{t}\overline{val}(\overline{x}^*, s) + \varepsilon L$, for all $s \in S$,

which implies $\max_{s\in S} val(\overline{x}^*, s) < \frac{\varepsilon L}{t} \max_{s\in S} \overline{val}(\overline{x}^*, s) + \varepsilon L$.

Since $\overline{x}^*$ is an optimal solution in $\overline{I}$, we have

$$opt(\overline{I}) = \max_{s\in S} \overline{val}(\overline{x}^*, s) \leq \max_{s\in S} \overline{val}(x^*, s)$$

and thus, the value of an optimal solution of $\overline{I}$ has, in I, the value

$$\max_{s\in S} val(\overline{x}^*, s) < \frac{\varepsilon L}{t} \max_{s\in S} \overline{val}(x^*, s) + \varepsilon L \leq opt(I) + \varepsilon L \leq opt(I)(1+\varepsilon).$$

A similar result can be obtained for Min-Max Regret $\mathcal{P}$. Let I be an instance of Min-Max Regret $\mathcal{P}$ and let $\overline{I}$ denote the instance derived from I, by scaling each entry c_i^s as follows: $\overline{c}_i^s = \lfloor \frac{2tc_i^s}{\varepsilon L} \rfloor$, where t is an upper bound of the size of any feasible solution of I. Let x^* and $\overline{x}^*$ denote respectively an optimal solution of instance I and $\overline{I}$ and let x_s^*, $\overline{x}_s^*$ denote respectively, an optimal solution of instance I and $\overline{I}$ in scenario s.

Then, we have, for all $s \in S$,

$$\begin{aligned} val(\overline{x}^*, s) - val(x_s^*, s) &< \frac{\varepsilon L}{2t}\overline{val}(\overline{x}^*, s) - val(x_s^*, s) + \frac{\varepsilon}{2}L \\ &\leq \frac{\varepsilon L}{2t}(\overline{val}(\overline{x}^*, s) - \overline{val}(x_s^*, s)) + \frac{\varepsilon}{2}L \\ &\leq \frac{\varepsilon L}{2t}(\overline{val}(\overline{x}^*, s) - \overline{val}(\overline{x}_s^*, s)) + \frac{\varepsilon}{2}L \end{aligned}$$

and thus

$$\begin{aligned} \max_{s\in S}\{val(\overline{x}^* , s) - val(x_s^*, s)\} &< \max_{s\in S}\left\{\frac{\varepsilon L}{2t}(\overline{val}(\overline{x}^*, s) - \overline{val}(\overline{x}_s^*, s))\right\} + \frac{\varepsilon}{2}L \\ &\leq \max_{s\in S}\left\{\frac{\varepsilon L}{2t}(\overline{val}(x^*, s) - \overline{val}(\overline{x}_s^*, s))\right\} + \frac{\varepsilon}{2}L \\ &\leq \max_{s\in S}\{val(x^*, s) - val(x_s^*, s) + val(x_s^*, s) - \frac{\varepsilon L}{2t}\overline{val}(\overline{x}_s^*, s)\} + \frac{\varepsilon}{2}L \\ &\leq \max_{s\in S}\{val(x^*, s) - val(x_s^*, s) + val(\overline{x}_s^*, s) - \frac{\varepsilon L}{2t}\overline{val}(\overline{x}_s^*, s)\} + \frac{\varepsilon}{2}L \\ &\leq \max_{s\in S}\{val(x^*, s) - val(x_s^*, s)\} + \varepsilon L \leq opt(I)(1+\varepsilon) \end{aligned}$$

We show in the following that such a solution $\overline{x}^*$ of instance $\overline{I}$ for Min-Max $\mathcal{P}$ or Min-Max Regret $\mathcal{P}$ can be obtained in polynomial time in $|I|$ and $\frac{1}{\varepsilon}$. The bounds L and U can be computed in time $p(|I|)$ by hypothesis. In order to compute an optimal solution for $\overline{I}$, we apply the algorithm (that exists by hypothesis) that runs in time $r(|\overline{I}|, U(\overline{I}))$.

In the case where $\overline{I}$ is an instance of Min-Max $\mathcal{P}$, since $opt(\overline{I}) \leq \frac{t\,opt(I)}{\varepsilon L} \leq \frac{tU}{\varepsilon L} \leq \frac{tq(|I|)}{\varepsilon}$, and q, r are non decreasing, the total time for computing the $(1+\varepsilon)$-approximation is $p(|I|) + r(|\overline{I}|, U(\overline{I})) \leq p(|I|) + r(|\overline{I}|, q(|\overline{I}|)L(\overline{I})) \leq p(|I|) + r(|I|, q(|I|)\frac{tq(|I|)}{\varepsilon})$.

In the case where $\overline{I}$ is an instance of Min-Max Regret $\mathcal{P}$, since $opt(\overline{I}) \leq \frac{2t\,opt(I)}{\varepsilon L} + t \leq \frac{2tU}{\varepsilon L} + t \leq \frac{2tq(|I|)}{\varepsilon} + t$, and q, r are non decreasing, the total time for computing the $(1+\varepsilon)$-approximation is $(k+1)p(|I|) + r(|\overline{I}|, U(\overline{I})) \leq (k+1)p(|I|) + r(|\overline{I}|, q(|\overline{I}|)L(\overline{I})) \leq (k+1)p(|I|) + r(|I|, q(|I|)(\frac{tq(|I|)}{\varepsilon} + t))$. □

We discuss now the two conditions of the previous theorem. The following result shows that the first condition can be satisfied easily if the underlying problem $\mathcal{P}$ is solvable in polynomial time.

Proposition 1. *If a minimization problem $\mathcal{P}$ is solvable in polynomial time, then for any instance on a set of k scenarios of* Min-Max $\mathcal{P}$ *and* Min-Max Regret $\mathcal{P}$*, there exist a lower and an upper bound L and U of opt computable in polynomial time, such that $U \leq kL$.*

Proof. Consider an instance I of Min-Max $\mathcal{P}$ defined on a set S of k scenarios where each scenario $s \in S$ is represented by $(c_1^s, \dots, c_m^s)$ and let X be the set of feasible solutions of I. We define the following instance I' of a single scenario problem $\min_{x \in X} \sum_{s \in S} \frac{1}{k} val(x,s)$ obtained by taking objective function coefficients $c'_i = \sum_{s=1}^{k} \frac{c_i^s}{k}$, $i = 1, \dots, m$. Let x^* be an optimal solution of I'. We take as lower and upper bounds $L = \sum_{s \in S} \frac{1}{k} val(x^*, s)$ and $U = \max_{s \in S} val(x^*, s)$. Clearly, we have

$$L = \min_{x \in X} \sum_{s \in S} \frac{1}{k} val(x,s) \leq \min_{x \in X} \sum_{s \in S} \frac{1}{k} (\max_{s \in S} val(x,s)) = \min_{x \in X} \max_{s \in S} val(x,s) = opt$$

and

$$\min_{x \in X} \max_{s \in S} val(x,s) \leq \max_{s \in S} val(x^*, s) \leq \sum_{s \in S} val(x^*, s) = k \sum_{s \in S} \frac{1}{k} val(x^*, s) = kL$$

Consider now an instance I of Min-Max Regret $\mathcal{P}$ defined on a set S of k scenarios and let X be the set of feasible solutions of I. Let $x^* \in X$ be an optimal solution of the single scenario instance I' derived from I as for the min-max case. We take as lower and upper bounds $L = \sum_{s \in S} \frac{1}{k}(val(x^*, s) - val_s^*)$ and $U = \max_{s \in S}(val(x^*, s) - val_s^*)$. Clearly, we have

$$L = \min_{x \in X} \frac{1}{k} \sum_{s \in S} (val(x,s) - val_s^*) \leq \min_{x \in X} \frac{1}{k} k \max_{s \in S} (val(x,s) - val_s^*) = opt$$

and

$$\min_{x \in X} \max_{s \in S} (val(x,s) - val_s^*) \leq \max_{s \in S} (val(x^*, s) - val_s^*) \leq \sum_{s \in S} (val(x^*, s) - val_s^*) = kL$$

If any instance of $\mathcal{P}$ of size n is solvable in time $p(n)$, where p is a polynomial, then bounds L and U are computable in $O(p(|I|/k))$. □

If $\mathcal{P}$ is polynomially approximable, then the first condition of Theorem 1 can be satisfied for Min-Max $\mathcal{P}$. More precisely, if $\mathcal{P}$ is $f(n)$-approximable where $f(n)$ is a polynomial, given an instance I of Min-Max $\mathcal{P}$, let $\widetilde{x}$ be an $f(|I|/k)$-approximate solution in I' (defined as in the proof of Proposition 1), then we have $L = \frac{1}{f(|I|/k)} \sum_{s \in S} \frac{1}{k} val(\widetilde{x}, s)$ and $U = \max_{s \in S} val(\widetilde{x}, s)$, and thus $U \leq k f(|I|/k) L$.

The second condition of Theorem 1 can be weakened for Min-Max $\mathcal{P}$ by requiring only a pseudo-polynomial algorithm, that is an algorithm polynomial in $|I|$ and $max(I) = \max_{i,s} c_i^s$. Indeed, knowing an upper bound U, we can eliminate any variable x_i such that $c_i^s > U$ on at least one scenario $s \in S$. Condition 2 is then satisfied applying the pseudo-polynomial algorithm on this modified instance.

MIN-MAX and MIN-MAX REGRET versions of some problems, like shortest path, knapsack, admit pseudo-polynomial time algorithms based on dynamic programming [6]. For some dynamic programming formulations, we can easily obtain algorithms satisfying condition 2, by discarding partial solutions with value more than U on at least one scenario. We illustrate this approach in section 4.1 for the shortest path problem.

For other problems, which are not known to admit pseudo-polynomial algorithms based on dynamic programming, specific algorithms are required. We present an algorithm verifying condition 2 for MIN-MAX SPANNING TREE (section 4.2).

Unfortunately, these algorithms cannot be adapted directly in order to obtain algorithms satisfying condition 2 for min-max regret versions. The basic difficulty here is that, if we can find an algorithm in $r(|I|, U(I))$ for any instance I of MIN-MAX $\mathcal{P}$, the direct extension of this algorithm for the corresponding instance I' of MIN-MAX REGRET $\mathcal{P}$ will be in $r(|I'|, U(I') + opt_{max})$ where $opt_{max} = \max_{s \in S} val_s^*$ is a value which is not necessarily polynomially related to $U(I')$.

However, for problems whose feasible solutions have a fixed size such as spanning tree, we reduced the min-max regret version to a min-max version in [2]. In this context, we need to consider instances where some coefficients are negative and possibly non integral but any feasible solution has a non-negative integral value. For an optimization problem $\mathcal{P}$, we denote by $\mathcal{P}'$ the extension of $\mathcal{P}$ to these instances. More precisely, we proved the following theorem.

Theorem 2. **([2])** *For any polynomial-time solvable minimization problem $\mathcal{P}$ whose feasible solutions have a fixed size and for any function $f : \mathbb{N} \rightarrow (1, \infty)$, if* MIN-MAX $\mathcal{P}'$ *has a polynomial-time $f(n)$-approximation algorithm, then* MIN-MAX REGRET $\mathcal{P}$ *has a polynomial-time $f(n)$-approximation algorithm.*

4 Applications

In this section, we apply the previous results to min-max (regret) shortest path, and minimum spanning tree. We also compare the running time for our algorithms and for the fptas obtained using an approximation of the non-dominated set, and show a significant improvement.

4.1 Shortest Path

In [6], Kouvelis and Yu proved the *NP*-hardness of min-max and min-max regret versions of shortest path, even for two scenarios.

Consider an instance I defined by a directed graph $G = (V, A)$, with $V = \{1, \ldots, n\}$ and $|A| = m$, and a set S of k scenarios giving for each arc $(i, j) \in A$ its cost c_{ij}^s under scenario s. Denote by c_{ij} the vector of size k formed by c_{ij}^s, $s \in S$. We are interested in optimal paths from 1 to any other vertex.

We give now pseudo-polynomial algorithms satisfying condition 2 of Theorem 1 for MIN-MAX (REGRET) SHORTEST PATH.

Proposition 2. *Given U an upper bound on the optimal value, then* Min-Max Shortest Path *and* Min-Max Regret Shortest Path *can be solved in time $O(nmU^k)$.*

Proof. We propose for each problem, an enumeration algorithm based on a dynamic programming formulation, that produces the set of all vectors of values (or regrets), for which all coordinates are less than or equal to U, and selects from this set an optimal vector. Let $u = (U, \ldots, U)$ denote the vector of size k.

Considering first Min-Max Shortest Path, we describe an algorithm that computes at each stage ℓ, the set V_j^ℓ of all possible vectors of values at most U corresponding to paths from 1 to j of length at most ℓ, $\ell = 1, \ldots, n-1$, $j = 2, \ldots, n$. The algorithm starts by initializing $V_1^0 = \{(0, \ldots, 0)\}$, where $(0, \ldots, 0)$ is a vector of size k and computes V_j^ℓ at each stage ℓ for each vertex j, $\ell = 1, \ldots, n-1$, $j = 2, \ldots, n$ as follows:

$$V_j^\ell = \cup_{i \in \Gamma^{-1}(j)} \{v^j = v^i + c_{ij} : v^i \in V_i^{\ell-1} \text{ and } v^j \leq u\} \tag{3}$$

Finally, the algorithm selects, as an optimal vector, a vector in V_j^{n-1} such that its largest coordinate is minimum, for $j = 2, \ldots, n$.

Consider now Min-Max Regret Shortest Path. Let $(val_s^*)^i$, $s \in S$, $i = 1, \ldots, n$, be the value of a shortest path in graph G from 1 to i under scenario s and let $(val^*)^i$ be the vector of size k of these values $(val_s^*)^i$, $s \in S$.

We describe an algorithm that computes at each stage ℓ, the set R_j^ℓ of all possible vectors of regrets at most U corresponding to paths from 1 to j of length at most ℓ, $\ell = 1, \ldots, n-1$, $j = 2, \ldots, n$. Consider arc $(i, j) \in A$ and let P_i be a path in G from 1 to i of regret $r_s^i = val(P_i, s) - (val_s^*)^i$, $s \in S$. Denote by P_j the path constructed from P_i by adding arc (i, j). The regret of P_j is $r_s^j = val(P_i, s) + c_{ij}^s - (val_s^*)^j = r_s^i + (val_s^*)^i + c_{ij}^s - (val_s^*)^j$, $s \in S$. The algorithm starts by initializing $R_1^0 = \{(0, \ldots, 0)\}$ and for $1 \leq \ell \leq n-1$ and $2 \leq j \leq n$ let

$$R_j^\ell = \cup_{i \in \Gamma^{-1}(j)} \{r^j = r^i + (val^*)^i + c_{ij} - (val^*)^j : r^i \in R_i^{\ell-1} \text{ and } r^j \leq u\} \tag{4}$$

Finally, the algorithm selects, as an optimal vector, a vector in R_j^{n-1} such that its largest coordinate is minimum, for $j = 2, \ldots, n$.

We point out that, for both algorithms, any path of interest can be obtained using standard bookkeeping techniques that do not affect the complexity of these algorithms.

In order to prove the correctness of these algorithms, we show that V_j^{n-1}, resp. R_j^{n-1}, contains all vectors of values, resp. regrets, at most U corresponding to paths from 1 to j, $j = 2, \ldots, n$. For this, we need to justify that we can eliminate, at any stage, any vector which violates the upper bound U, without losing any vector at the end.

Indeed, for the min-max version, if such a solution v^i is obtained then any of its extensions computed in (3) would also violate U due to the non-negativity of vectors c_{ij}.

Similarly, for the min-max regret version, if such a solution r^i is obtained then any of its extensions computed in (4) would also violate U since vectors of the form $(val^*)^i + c_{ij} - (val^*)^j$ are non-negative.

Both algorithms can be implemented in time $O(nmU^k)$. □

Corollary 1. Min-Max (Regret) Shortest Path *are in FPTAS.*

Proof. Using Theorem 1, Propositions 1 and 2, we derive an fptas whose running time is $O(\frac{mn^{k+1}}{\varepsilon^k})$. □

Warburton describes in [11] an fptas for approximating the non-dominated set for the multi-objective version of the shortest path problem. From this fptas, Warburton derives an fptas for Min-Max Shortest Path in acyclic graphs with running time $O(\frac{n^{2k+1}}{\varepsilon^{2k-2}})$, whereas our running time, for general graphs, is better.

4.2 Minimum Spanning Tree

In [6], Kouvelis and Yu proved the *NP*-hardness of min-max and min-max regret versions of minimum spanning tree, even for two scenarios. We first describe algorithms for Min-Max Spanning Tree with running time polynomial in a suitably chosen upper bound on the optimal value.

Consider an instance of Min-Max Spanning Tree represented by a graph $G = (V, E)$ where $|V| = n$, $|E| = m$, c^s_{ij} is the cost of edge (i, j) in scenario $s \in S$ and $|S| = k$.

Proposition 3. *Given U an upper bound on the optimal value, then* Min-Max Spanning Tree *can be solved in time $O(mn^4U^k \log U)$.*

Proof. We can solve Min-Max Spanning Tree using an extension of the matrix tree theorem to the multiple scenarios case as presented in section 2.3.

The optimal value *opt* of Min-Max Spanning Tree can be computed by considering, for each monomial in (2), the largest power $v_{max} = \max_{s=1,\ldots,k} v_s$. The minimum value of v_{max} over all monomials corresponds to *opt*.

Actually, instead of computing all monomials, we can use, as suggested in [5], the algorithm presented in [7]. When applied to matrix $A_r(y_1, \ldots, y_k)$, this algorithm can compute the determinant polynomial up to a specified degree in each variable in opposition to the classical method of Edmonds [4]. In this case, it is sufficient to compute the polynomial determinant up to degree U in each variable y_s for $s = 1, \ldots, k$. The algorithm in [7] requires $O(n^4)$ multiplications and additions of polynomials. The time needed to multiply two multivariate polynomials of maximum degree d_s in variable y_s for $s = 1, \ldots, k$ is $\prod_{s=1}^k d_s \log \prod_{s=1}^k d_s$ [1]. Thus, the running time to compute the polynomial determinant is $O(n^4U^k \log U)$.

Once an optimal vector is identified, a corresponding spanning tree can be constructed using self reducibility [8]. It consists of testing iteratively, for each edge if the graph obtained by contracting this edge admits a spanning tree of the required vector of adjusted values on all scenarios (subtracting iteratively the vector of costs c^s_{ij}, $s \in S$, for each edge (i, j) being tested to the required vector of values). In at most $m-(n-1)$ iterations such a spanning tree is obtained. Hence, the self reducibility requires $O(m)$ computations of determinant polynomial. □

Corollary 2. MIN-MAX SPANNING TREE *is in FPTAS.*

Proof. Using Theorem 1, Propositions 1 and 3, we derive an fptas whose running time is $O(\frac{mn^{k+4}}{\varepsilon^k} \log \frac{n}{\varepsilon})$. □

Corollary 3. MIN-MAX REGRET SPANNING TREE *is in FPTAS.*

Proof. Notice that Theorem 1 and Proposition 3 remain true even for the instances of spanning tree where some coefficients are negative but any feasible solution has a non-negative value. Thus, MIN-MAX SPANNING TREE′ is in FPTAS. The result follows from Theorem 2. The running time of the fptas is $O(\frac{mn^{k+4}}{\varepsilon^k} \log \frac{n}{\varepsilon})$. □

In this case, we obtain fptas with better running times for MIN-MAX (REGRET) SPANNING TREE. Indeed, the running time of the fptas obtained in [2] using the general multi-objective approximation scheme presented in [9] is $O(\frac{n^{k+4}}{\varepsilon^{2k}} (\log U)^k \log \frac{n}{\varepsilon})$.

References

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullmann. *The design and analysis of computer algorithms.* Addison-Wesley Reading, 1976.
2. H. Aissi, C. Bazgan, and D. Vanderpooten. Approximation complexity of min-max (regret) versions of shortest path, spanning tree, and knapsack. In *Proceedings of the 13th Annual European Symposium on Algorithms (ESA 2005), Mallorca, Spain*, LNCS 3669, pages 862–873, 2005.
3. F. Barahona and R. Pulleyblank. Exact arborescences, matching and cycles. *Discrete Applied Mathematics*, 16:91–99, 1987.
4. J. Edmonds. System of distinct representatives and linear algebra. *Journal of Research of the National Bureau of Standards*, 71:241–245, 1967.
5. S. P. Hong, S. J. Chung, and B. H. Park. A fully polynomial bicriteria approximation scheme for the constrained spanning tree problem. *Operations Research Letters*, 32(3):233–239, 2004.
6. P. Kouvelis and G. Yu. *Robust discrete optimization and its applications.* Kluwer Academic Publishers, Boston, 1997.
7. M. Mahajan and V. Vinay. A combinatorial algorithm for the determinant. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 1997), New Orleans, USA*, pages 730–738, 1997.
8. C. H. Papadimitriou. *Computational complexity.* Addison Wesley, 1994.
9. C. H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *IEEE Symposium on Foundations of Computer Science (FOCS 2000), Redondo Beach, California, USA*, pages 86–92, 2000.
10. W.T. Tutte. *Graph Theory*, volume 21 of *Encyclopedia of Mathematics and its Applications.* Addison-Wesley, 1984.
11. A. Warburton. Approximation of Pareto optima in multiple-objective, shortest-path problems. *Operations Research*, 35:70–79, 1987.