What about future? Robustness under vertex-uncertainty in graph-problems

Cecile Murat, Vangelis Th. Paschos

UNIVERSITE PARIS DAUPHINE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision

CNRS UMR 7024

CAHIER DU LAMSADE

236

Avril 2006

What about future?
Robustness under vertex-uncertainty in graph-problems

Cécile Murat, Vangelis Th. Paschos

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

What about future? Robustness under vertex-uncertainty in graph-problems

Cécile Murat Vangelis Th. Paschos

LAMSADE, CNRS UMR 7024 and Université Paris-Dauphine
Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France
{murat,paschos}@lamsade.dauphine.fr

April 12, 2006

Abstract

We study a robustness model for graph-problems under vertex-uncertainty. We assume that any vertex v_i of the input-graph $G(V, E)$ has only a probability p_i to be present in the final graph to be optimized (i.e., the final instance for the problem tackled will be only a sub-graph of the initial graph). Under this model, the original "deterministic" problem gives rise to a new (deterministic) problem on the same input-graph G, having the same set of feasible solutions as the former one, but its objective function can be very different from the original one, the set of its optimal solutions too. Moreover, this objective function is a sum of $2^{|V|}$ terms; hence, its computation is not immediately polynomial. We give sufficient conditions for large classes of graph-problems under which objective functions of the robust counterparts are polynomially computable and optimal solutions are well-characterized. Finally, we apply these general results to natural and well-known representatives of any of the classes considered.

1 Introduction

Very often people has to make decisions under several degrees of uncertainty, i.e., when only probabilistic information about the future is available. We deal with the following robustness model under data uncertainty. Consider a generic instance I of a combinatorial optimization problem Π. Assume that Π is not to be necessarily solved on the whole I, but rather on a (unknown a priori) sub-instance $I' \subset I$. Suppose that any datum d_i in the data-set describing I has a probability p_i, indicating how d_i is likely to be present in the final sub-instance I'. Consider finally that once I' is specified, the solver has no opportunity to solve it directly (for example she/he has to react quasi-immediately, so no sufficient time is given her/him). Concrete examples of such situations dealing with satellite shots planning or with timetabling are given in [24, 25].

What can the solver do in this case? A natural way to proceed is to compute an *anticipatory solution* S for Π, i.e., a solution for the entire instance I, and once I' becomes known, to modify S in order to get a solution S' fitting I'. The objective is to determine an initial solution S for I such that, for any sub-instance $I' \subseteq I$ presented for optimization, the solution S' respects some predefined quality criterion (for example, optimal for I', or achieving, say, constant approximation ratio, or ...).

In what follows we restrict ourselves in graph-problems and consider the following very simple and quick modification[1] of S: *take the restriction of S in the present sub-instance of G*. Consider

[1] As we will see later such a modification strategy does not always produce feasible solutions; in such a case some more work is needed.

a graph-problem Π, a graph $G(V, E)$ of order n, instance of Π, and an n-vector $\mathbf{Pr} = (p_1, \ldots, p_n)$ of vertex-probabilities any of them, say p_i, measuring how likely is for vertex $v_i \in V$, $i = 1, \ldots, n$ to be present in the final subgraph $G' \subseteq G$, on which the problem will be really solved. For any solution S for Π in G and for any $V' \subseteq V$, denote by S' the restriction of S in V', i.e., the set resulting from S after removal of the vertices that do not belong to V'. As we have mentioned, S', can or cannot (depending of the definition of Π) be a feasible solution of Π in the subgraph G' of G induced by V'. Whenever S' is feasible, denote by $m(G', S')$ the objective value of S' in G'. Then, the value of S for G, denoted by $E(G, S)$ (and frequently called *functional*), is the expectation of $m(G', S')$, over all the possible $G' \subseteq G$. Formally, given S, the functional $E(G, S)$ of S is defined by:

$$E(G, S) = \sum_{V' \subseteq V} \Pr\left[V'\right] m\left(G', S'\right) \tag{1}$$

where $\Pr[V']$ is the probability that V' will be finally the real instance to be optimized and is defined by: $\Pr[V'] = \prod_{v_i \in V'} p_i \prod_{v_i \in V \setminus V'} (1 - p_i)$. Obviously, $E(G, S)$ depends also on $\mathbf{Pr}$ but, for simplicity, this dependency will be omitted. Quantity, $E(G, S)$ can be seen as the objective function of a new combinatorial problem, derived from Π and denoted by ROBUST Π in what follows, where we are given an instance G of Π and a probability vector $\mathbf{Pr}$ on the vertices of G and the objective is to determine a solution S^* in G optimizing $E(G, S)$ (optimal anticipatory solution). The optimization goal of ROBUST Π is the same as the one of Π.

This way to tackle robustness in combinatorial optimization is sometimes called *a priori optimization* (this term has been introduced by [7]) and can also be seen as a particular case of stochastic optimization ([9, 26]). In a priori optimization, the combinatorial problem to be solved, being subject to hazards or to inaccuracies, is not defined on a static and clearly specified instance, since the instance to be effectively optimized is not known with absolute certainty from the beginning. Robustness requirements are due to the fact that uncertainty in the presence of data makes that it is not possible to assign unique values to some parameters of the problems. In such an approach, the goal is to compute solutions that behave "well" under any assignment of values to these parameters. Under this model, restrictive versions of routing and network-design robust minimization problems (in complete graphs) have been studied in [2, 4, 5, 6, 7, 13, 14, 15, 16]. Recently, in [8], the analysis of the robust minimum travelling salesman problem, originally performed in [4, 13], has been revisited and refined. In [22, 24] the minimum vertex covering and the minimum coloring are tackled, while in [20, 21] robust maximization problems, namely, the longest path and the maximum independent set, are studied. An early survey about a priori optimization can be found in [3] while, a more recent one appears in [23].

Another way to apprehend robustness under uncertainty is as it is done in [1, 10, 17, 18, 19]. In this framework, one either identifies different feasible scenarii, or associates an interval of values rather than one unique value with a parameter. Finally, a third very interesting face of robustness that model inaccuracy by assigning probabilities to scenarii or to data is apprehended by what has been called two- and multi-stage stochastic optimization (see, for example, [11, 12, 27] for recent results on this area).

Our goal is to go beyond study of robust versions of particular combinatorial problems and to propose a structural study of robustness under uncertainty for the a priori optimization paradigm. Here, the main mathematical issues (assuming that, given an anticipatory solution S, its restriction to $G[V']$ is feasible) are:

- the complexity of the computation of $E(G, S)$ which, carrying over 2^n additive terms, is non-trivially polynomial;
- compact combinatorial characterization (based upon the form of $E(G, S)$) of S^* as optimal solution of ROBUST Π;

- the complexity of computing S^*, at least for particular classes of subproblems of the initial problem.

Notice that, for any problem Π, its combinatorial counterpart ROBUST Π contains Π as subproblem (just consider probability vector $(1, \ldots, 1)$ for Π). Hence, from a complexity point of view, ROBUST Π is at least as hard as Π, that is, if Π is **NP**-hard, then ROBUST Π is also **NP**-hard, while if Π is polynomial, then no immediate indication can be provided for the complexity of ROBUST Π, until this latter problem is explicitly studied.

In what follows, in Sections 2, 3 and 4, we deal with three categories of combinatorial graph-problems exhausting a very large part of the most known ones, namely:

- problems whose solutions are subsets of the input vertex-set verifying some specific property (Section 2);
- problems whose solutions are collections of subsets of the input vertex-set verifying some specified non-trivial hereditary property[2] (Section 3);
- problems whose solutions are subsets of the input edge-set verifying some specific property (Section 4).

For any of these categories, we give sufficient conditions under which functionals are analytically expressible and polynomially computable and anticipatory solutions are well-characterized.

In Section 5, we deal with problems having as solutions subsets of the input edge-set verifying some connectivity properties. For this type of problems, the restriction of S to $G[V']$ is not feasible in general, but some additional work (with low algorithmic complexity) is sufficient in order to render this set feasible. There, as we will see, anticipatory solutions cannot be as well-characterized as previously, neither complexity of computing S^*. However, we give sufficient conditions under which functionals for the robust counterparts of these problems are computable in polynomial time.

These structural results immediately apply to several well-known problems, for instance, MIN VERTEX COVER, MAX INDEPENDENT SET, MIN COLORING, MAX CUT, MAX MATCHING, etc., producing so particular results interesting per se. Furthermore, the scope of our results is even larger than for graph-problems, as problems not originally defined on graphs (e.g., MAX SET PACKING or MIN SET COVER), are also captured. So, this work can provide a framework for a systematic classification of a great number of robust derivatives of well-known graph-problems.

In what follows, we deal with problems in **NPO**. Informally, this class contains optimization problems whose decision versions belong to **NP**. Given a combinatorial problem $\Pi \in \mathbf{NPO}$, we denote by ROBUST Π, its robust counterpart defined as described previously and assume that the vertex-probabilities are independent.

Let $\mathtt{A}$ be a polynomial time approximation algorithm for an **NP**-hard graph-problem Π, let $m(G, S)$ be the value of the solution S provided by $\mathtt{A}$ on an instance G of Π, and $\mathrm{opt}(G)$ be the value of the optimal solution for G (following our notation for ROBUST Π, $\mathrm{opt}(G) = E(G, S^*)$). The approximation ratio $\rho_{\mathtt{A}}(G)$ of the algorithm $\mathtt{A}$ on G is defined by $\rho_{\mathtt{A}}(G) = m(G, S)/\mathrm{opt}(G)$. An approximation algorithm achieving ratio, at most, ρ on any instance G of Π will be called ρ-approximation algorithm.

[2] A property π is *hereditary* if, whenever is satisfied by a graph G, it is satisfied by any subgraph of G; a hereditary property π is non-trivial if it is true (satisfied) for infinitely many graphs and false for infinitely many graphs.

2 Solutions are subsets of the initial vertex-set

In this section, we deal with graph-problems whose solutions are subsets of the vertex-set of the input-graph and where, given such a solution S and a set $V' \subseteq V$, the restriction of S in V', i.e., the set $S' = S \cap V'$ is feasible for $G[V']$. The main result of this section is stated in Theorem 1.

Theorem 1. *Consider a graph-problem* Π *verifying the following assumptions: (i) an instance of* Π *is a vertex-weighted graph* $G(V, E, \vec{w})$*; (ii) solutions of* Π *are subsets of* V*; (iii) for any solution* S *and any subset* $V' \subseteq V$*,* $S' = S \cap V'$ *is feasible for* $G' = G[V']$*; (iv) the value of any solution* $S \subseteq V$ *is defined by:* $m(G, S) = w(S) = \sum_{v_i \in S} w_i$*, where* w_i *is the weight of* $v_i \in V$*. Then, the functional of* ROBUST Π *is expressed as:* $E(G, S) = \sum_{v_i \in S} w_i p_i$ *and can be computed in polynomial time. Furthermore, the complexity of* ROBUST Π *is the same as the one of* Π*.*

Proof. Fix a subset $V' \subseteq V$ and an anticipatory solution S for ROBUST Π on G. According to assumption (iii), S' is feasible for $G[V']$. Its value is given by: $m(G', S') = \sum_{v_i \in S} w_i 1_{\{v_i \in V'\}}$. Then, denoting by 1_F the indicator function of a fact F and using (1) we get:

$$\begin{aligned} E(G,S) &= \sum_{V' \subseteq V} m\left(G', S'\right) \Pr\left[V'\right] = \sum_{V' \subseteq V} \sum_{v_i \in S} w_i 1_{\{v_i \in V'\}} \Pr\left[V'\right] \\ &= \sum_{v_i \in S} w_i \sum_{V' \subseteq V} 1_{\{v_i \in V'\}} \Pr\left[V'\right] \end{aligned} \tag{2}$$

For any vertex $v_i \in V$, let $V_i = V \setminus \{v_i\}$ and $\mathcal{V}'_i = \{V' \subseteq V : V' = \{v_i\} \cup V'', V'' \subseteq V_i\}$. Using also the fact that presence-probabilities of the vertices of V are independent, we get:

$$\begin{aligned} \sum_{V' \subseteq V} 1_{\{v_i \in V'\}} \Pr\left[V'\right] &= \sum_{V' \in \mathcal{V}'_i} \Pr\left[V'\right] = \sum_{V'' \subseteq V_i} \Pr\left[\{v_i\} \cup V''\right] \\ &= \sum_{V'' \subseteq V_i} \Pr\left[v_i\right] \Pr\left[V''\right] = \Pr\left[v_i\right] \sum_{V'' \subseteq V_i} \Pr\left[V''\right] = p_i \end{aligned} \tag{3}$$

Combination of (2) and (3) immediately leads to the expression claimed for $E(G, S)$.

It is easy to see that this functional can be computed in time linear in n. Furthermore, computation of the optimal anticipatory solution for ROBUST Π in G, obviously amounts to the computation of the optimal weighted solution for Π in $G(V, E, \vec{w}')$, where, for any $v_i \in V$, $w'_i = w_i p_i$. Consequently, by this observation and by assumption (iv) in the statement of the theorem, Π and ROBUST Π have the same complexity. ∎

Although computation of the functional is, as we have mentioned, a priori exponential (since it carries over 2^n subgraphs of G), assumptions (i) through (iv) in Theorem 1 allow polynomial computation of its value. This is due to the fact that, under these assumptions, given a sub-graph G' induced by a subset $V' \subseteq V$, the value of the solution for G' is the sum of the weights of the vertices in $S \cap V'$. Furthermore, a vertex not in S will never make part of any solution in any sub-graph of G. Consequently, computation of the functional amounts to determining, for any G', which vertices make part of $S \cap V'$. This is equivalent with the specification, for any $v_i \in S$, of all the subgraphs to which v_i belongs and with a summation of the presence-probabilities of these subgraphs. This sum is equal to p_i (the probability of v_i). This simplification is the main reason that renders functional's computation polynomial, despite of the exponential number of terms in its generic expression.

Notice that Theorem 1 can also be used for getting generic approximation results for ROBUST Π. Indeed, since this problem is a particular weighted version of Π, one immediately concludes that *if* Π *is approximable within approximation ratio* ρ*, so is* ROBUST Π.

Corollary 1. *Under the hypotheses of Theorem 1, whenever* Π *and* ROBUST Π *are* ***NP****-hard, they are equi-approximable.*

Theorem 1 has also the following immediate corollary dealing with the case of robust versions of unweighted combinatorial optimization problems.

Corollary 2. *Consider a robust combinatorial optimization problem* ROBUST Π *verifying assumptions (i) to (iv) of Theorem 1 with* $\vec{w} = \vec{1}$*. Then, the functional of* ROBUST Π*, is expressed as:* $E(G, S) = \sum_{v_i \in S} p_i$ *and can be computed in polynomial time. Furthermore,* ROBUST Π *is equivalent to a weighted version of* Π *where vertex-weights are the vertex-probabilities.*

Corollary 2 is somewhat weaker than Theorem 1 since it does not establish the equivalence between Π and ROBUST Π but rather a kind of reduction from Π to ROBUST Π stating that the latter is a priori harder than the former one. As a consequence, whenever Π is **NP**-hard, so is ROBUST Π whereas if Π is polynomial, the status of ROBUST Π remains unclear by Corollary 2.

2.1 Applications of Theorem 1

Theorem 1 can be applied to a broad class of problems that fit its four conditions, as ROBUST MAX INDEPENDENT SET ([21]), ROBUST MIN VERTEX COVERING ([22]), etc. We describe in what follows two further applications, namely, ROBUST MAX INDUCED SUBGRAPH WITH PROPERTY π and ROBUST MIN FEEDBACK VERTEX-SET.

2.1.1 ROBUST MAX INDUCED SUBGRAPH WITH PROPERTY π

Consider a graph $G(V, E)$ and a non-trivial hereditary property. A feasible solution for MAX INDUCED SUBGRAPH WITH PROPERTY π is a subset $V' \subseteq V$ such that, the subgraph $G[V']$ of G induced by V' satisfies π. The objective for MAX INDUCED SUBGRAPH WITH PROPERTY π is to determine such a set V' of maximum-size. Note that, "*independent set*", "*clique*", "*planar graph*" are hereditary properties. In the weighted version of the problem (i.e., the one where positive weights are associated with the vertices of G), called MAX WEIGHTED INDUCED SUBGRAPH WITH PROPERTY π, we search for maximizing the total weight of V'.

Given a solution S for MAX WEIGHTED INDUCED SUBGRAPH WITH PROPERTY π and an induced subgraph $G[V']$ of the input graph $G(V, E)$, the set $S \cap V'$ is a feasible solution for $G[V']$, since, by the definition of π, if a subset $S \subseteq V$ induces a subgraph verifying it, then any subset of S also induces a subgraph verifying π. Henceforth, MAX WEIGHTED INDUCED SUBGRAPH WITH PROPERTY π fits the conditions of Theorem 1, or of Corollary 2 (for the unweighted case).

2.1.2 ROBUST MIN FEEDBACK VERTEX-SET

Given an oriented graph $G(V, A)$, a *feedback vertex-set* is a subset $V' \subseteq V$ such that V' contains at least a vertex of any directed cycle of G. In MIN FEEDBACK VERTEX-SET, the objective is to determine a feedback vertex-set of minimum size.

Remark that, absence of a vertex v from a feedback vertex-set V', breaks any cycle containing this vertex. If v makes part of an anticipatory solution S then, since no such cycle that contained v exists in G', feasibility of the solution $S \cap V'$ does not suffer from the absence of v. So, Corollary 2 applies for this problem.

Note that the weighted version of this problem can be tackled in a similar way.

2.2 Extensions of Theorem 1 beyond graphs

Theorem 1 can also be used to capture problems that are not originally defined on graphs as ROBUST MAX SET PACKING and ROBUST MIN SET COVER. The robust versions dealt for both

of them consist of considering presence probabilities $p_1, \ldots, p_n$ for the corresponding elements of the ground set C. Extensions to these problems are shown in the following Sections 2.2.1 and 2.2.2.

2.2.1 ROBUST MAX SET PACKING

Given a family $\mathcal{S} = \{S_1, \ldots, S_m\}$ of subsets of a ground set $C = \{c_1, \ldots, c_n\}$, MAX SET PACKING consists of determining a maximum size sub-collection $\mathcal{S}' \subseteq \mathcal{S}$ such that for any $(S_i, S_j) \in \mathcal{S}' \times \mathcal{S}'$, $S_i \cap S_j = \emptyset$.

The robust version assumed for MAX SET PACKING consists of considering presence probabilities $p_1, \ldots, p_n$ for the corresponding elements of C. A set $S_i \in \mathcal{S}$ is present if at least one of its elements is present. So, denoting by $\Pr[S_i]$ the presence probability of the set $S_i = \{c_{i_1}, c_{i_2}, \ldots, c_{i_k}\}$, we get:

$$\Pr[S_i] = 1 - \prod_{j=1}^{k} (1 - p_{i_j}) \tag{4}$$

Consider the following very well-known transformation of an instance $(\mathcal{S}, C)$ of MAX SET PACKING into an instance $G(V, E)$ of MAX INDEPENDENT SET:

- for any $S_i \in \mathcal{S}$, add a new vertex $v_i \in V$;
- for any pair $(S_i, S_j) \in \mathcal{S} \times \mathcal{S}$, if $S_i \cap S_j \neq \emptyset$, then add an edge $(v_i, v_j) \in E$.

It is easy to see that, under the above transformation, any set packing in $(\mathcal{S}, C)$ becomes an independent set of G of the same size and vice-versa.

So, given a robust model for MAX SET PACKING, one can transform, as described just previously, its instance $(\mathcal{S}, C)$ into a graph $G(V, E)$ and one can consider vertex probabilities $p(v_i) = \Pr[S_i]$, $i = 1, \ldots, m$, where v_i is the vertex added for S_i. A feasible MAX INDEPENDENT SET-solution S' in G corresponds to a sub-family $\mathcal{S}'$ consisting of the sets represented by the vertices of S' and is indeed a set packing. On the other hand, an induced subgraph $G[V']$ of G corresponds to a sub-instance I' of $(\mathcal{S}, C)$ induced by the sets represented by the vertices of V' (according to the rule that a set is present if at least one of its elements is in I'). Then, taking the restriction of S' in V' amounts to consider the restriction of $\mathcal{S}'$ in I'. It is easy to see that this restriction is a feasible set packing for I'. So, MAX SET PACKING, being an independent set problem, meets conditions of Corollary 2 (or of Theorem 1 for MAX WEIGHTED SET PACKING). Hence, given an instance $(\mathcal{S}, C)$ of MAX SET PACKING, with element-probabilities p_i, for any $c_i \in C$, and a feasible solution $\mathcal{S}'$ of $(\mathcal{S}, C)$, then $E((\mathcal{S}, C), \mathcal{S}') = \sum_{S_i \in \mathcal{S}'} \Pr[S_i]$, where $\Pr[S_i]$ is as defined in (4). Then, the robust version of MAX SET PACKING amounts to a particular weighted version of MAX SET PACKING where each set $S_i = \{c_{i_1}, \ldots, c_{i_k}\}$ in $\mathcal{S}$ is weighted by $\Pr[S_i]$.

2.2.2 ROBUST MIN SET COVER: when sets become vertices

Given a collection $\mathcal{S} = \{S_1, \ldots S_m\}$ of subsets of a ground set $C = \{c_1, \ldots, c_n\}$ (it is assumed that $\cup_{S_i \in \mathcal{S}} S_i = C$), MIN SET COVERING consists of determining a minimum-size set cover of C, i.e., a minimum-size sub-collection $\mathcal{S}'$ of $\mathcal{S}$ such that $\cup_{S_i \in \mathcal{S}'} S_i = C$.

We use the same robust model for MIN SET COVER, i.e., we consider presence probabilities $p_1, \ldots, p_n$ for the corresponding elements of the ground set C. As previously in Section 2.2.1, a set $S_i \in \mathcal{S}$ is present if at least one of its elements is present. So, $\Pr[S_i]$ is given by (4). We give a formulation of MIN SET COVER as a graph-problem and according this formulation, we show that ROBUST MIN SET COVER fits conditions of Theorem 1.

Consider an instance $(\mathcal{S}, C)$ of MIN SET COVER and construct $G_S(V_S, E_S, \vec{\ell})$, an edge-labelled multi-graph, as follows:

- for any set $S_i \in \mathcal{S}$, add a vertex $v_i \in V_S$;
- for any pair S_i, S_j of sets in $\mathcal{S}$, add a new edge (v_i, v_j) labelled by c_k only if $c_k \in S_i \cap S_j$.

Under this formulation, MIN SET COVER amounts to determine the smallest subset of vertices covering all the labels in G_S. Then, an analysis very similar to the one for ROBUST MAX SET PACKING in Section 2.2.1, concludes that, under the considered formulation, MIN SET COVER meets conditions of Theorem 1 (for the weighted case) or of Corollary 2 (for the unweighted one).

3 Solutions are collections of subsets of the initial vertex-set

We now deal with problems the feasible solutions of which are collections of subsets of the initial vertex-set. Consider a graph $G(V, E)$ and a combinatorial optimization graph-problem Π whose solutions are collections of subsets of V verifying some specified non-trivial hereditary property (e.g., independent set, clique, etc.). The following theorem characterizes functionals and optimal anticipatory solutions for such problems.

Theorem 2. *Consider a graph- problem Π verifying the following assumptions: (i) an instance of Π is a graph $G(V, E)$; (ii) a solution of Π on an instance G is a collection $S = (V_1, \ldots, V_k)$ of subsets of V any of them satisfying some specified non-trivial hereditary property; (iii) for any solution S and any subset $V' \subseteq V$, the restriction S' of S in V', i.e., $S' = (V_1 \cap V', \ldots, V_k \cap V')$, is feasible for $G' = G[V']$; (iv) the value of any solution $S \subseteq V$ of Π is defined by: $m(G, S) = |S| = k$. Then, $E(G, S) = \sum_{j=1}^{k}(1 - \prod_{v_i \in V_j}(1 - p_i))$ and can be computed in polynomial time. ROBUST Π amounts to a particular weighted version of Π, where the weight of any vertex $v_i \in V$ is $1 - p_i$, the weight $w(V_j)$ of a subset $V_j \subseteq V$ is defined by $w(V_j) = 1 - \prod_{v_i \in V_j}(1 - p_i)$ and the objective function to be optimized is equal to $\sum_{V_j \in C} w(V_j)$.*

Proof. Consider an anticipatory solution $S = (V_1, V_2, \ldots, V_k)$ and a subgraph $G' = G[V']$ of G. Denote by $k' = m(G', S')$, the value of the solution obtained on G' as described in assumption (iii). Then, from (1), $E(G, S) = \sum_{V' \subseteq V} \Pr[V']k'$.

Consider the facts F_j: $V_j \cap V' \neq \emptyset$ and $\bar{F}_j$: $V_j \cap V' = \emptyset$. Then, k' can be written as $k' = \sum_{j=1}^{k} 1_{F_j} = \sum_{j=1}^{k}(1 - 1_{\bar{F}_j})$ and $E(G, S)$ becomes:

$$\begin{aligned}
E(G,S) &= \sum_{V' \subseteq V} \Pr\left[V'\right] \left(\sum_{j=1}^{k} \left(1 - 1_{\bar{F}_j}\right) \right) \\
&= \sum_{V' \subseteq V} \Pr\left[V'\right] \sum_{j=1}^{k} 1 - \sum_{V' \subseteq V} \Pr\left[V'\right] \sum_{j=1}^{k} 1_{V_j \cap V' = \emptyset} \\
&= \sum_{j=1}^{k} \sum_{V' \subseteq V} \Pr\left[V'\right] - \sum_{j=1}^{k} \sum_{V' \subseteq V} \Pr\left[V'\right] 1_{V_j \cap V' = \emptyset} \\
&= k - \sum_{j=1}^{k} \prod_{v_i \in V_j} (1 - p_i) \;=\; \sum_{j=1}^{k} \left(1 - \prod_{v_i \in V_j} (1 - p_i) \right) \qquad (5)
\end{aligned}$$

It is easy to see that computation of $E(G, S)$ can be performed in at most $O(n)$ steps; consequently, ROBUST Π is in **NPO**. Furthermore, by (5), the characterization of the feasible solutions for ROBUST Π claimed in the statement of the theorem is immediate. ∎

What does play a central role for yielding result of Theorem 2, is the fact that property satisfied by the sets of the collection is hereditary. This allows to preserve sets $V_1, \ldots, V_k$ in the solution returned by $S \cap V_i$, $i = 1, \ldots, k$, unless they are empty and, consequently, to express $E(G, S)$ as in (5), in terms of facts F_j and $\bar{F}_j$.

Assume that $p_i = 1$, for any $v_i \in V$. Then, by (5), $E(G, S) = k$ and ROBUST Π coincides in this case with Π.

Corollary 3. *If Π is **NP**-hard, then* ROBUST Π *is also **NP**-hard.*

As for Corollary 2, Corollary 3 settles complexity only for the case where Π is **NP**-hard, leaving unclear the status of ROBUST Π when $\Pi \in \mathbf{P}$.

3.1 Applications of Theorem 2

Theorem 2 has also application for numerous combinatorial optimization problems, as ROBUST MIN COLORING ([24]), ROBUST MIN PARTITION INTO CLIQUES, etc. In what follows we describe two further applications, namely, to ROBUST MIN COMPLETE BIPARTITE SUBGRAPH COVER and ROBUST MIN CUT COVER.

3.1.1 ROBUST MIN COMPLETE BIPARTITE SUBGRAPH COVER

Given a graph $G(V, E)$, a solution of MIN COMPLETE BIPARTITE SUBGRAPH COVER is a collection $\mathcal{C} = (V_1, V_2, \ldots, V_k)$ of subsets of V such that the subgraph induced by any of the V_i's, $i = 1, \ldots, k$, is a complete bipartite graph and for any edge $(u, v) \in E$ there exists a V_i containing both u and v. The objective here is to minimize the size $|\mathcal{C}|$ of $\mathcal{C}$.

Consider a solution $S = (V_1, \ldots, V_k)$ MIN COMPLETE BIPARTITE SUBGRAPH COVER and a subset $V' \subseteq V$. The set $S' = (V_1 \cap V', \ldots, V_k \cap V')$, is feasible for $G' = G[V']$. Indeed, if a vertex v disappears from some subset V_i of an anticipatory solution S, the surviving set V_i always induces a complete bipartite graph. Furthermore:

- except the edges that have been disappeared (the ones incident to v) any other edge remain covered by the surviving sets of S;
- property "is a complete bipartite graph" is hereditary.

So, ROBUST MIN COMPLETE BIPARTITE SUBGRAPH COVER meets the conditions of Theorem 2.

3.1.2 ROBUST MIN CUT COVER

Given a graph $G(V, E)$, a feasible solution for MIN CUT COVER is a collection $(V_1, \ldots, V_k)$ of V such that any V_i, $i = 1, \ldots, k$ is a cut, i.e., for any $(u, v) \in E$, there exists a V_i such that either $u \in V_i$ and $v \notin V_i$, or $u \notin V_i$ and $v \in V_i$. The objective is to minimize the size of the collection.

Consider a solution $S = (V_1, \ldots, V_k)$ for MIN CUT COVER. If a vertex $v \in V$ is absent, then any edge incident to v is also absent. But, absence of a vertex together with any edge incident to it, does not affect the edges present to the final graph $G'(V', E')$, that remain feasibly covered by endpoints any of them belonging to distinct cuts. Hence, $S' = (V_1 \cap V', \ldots, V_k \cap V')$ is feasible for MIN CUT COVER, that meets the conditions of Theorem 2, since property "is a cut" is hereditary.

3.2 ROBUST MIN SET COVER: when elements become vertices

As in Section 2, application of Theorem 2 can go beyond graphs. We give in this section another formulation of MIN SET COVER as a graph-problem that can be seen as a kind of dual with respect to the formulation given in Section 2.2.2. We show that, according this new formulation,

ROBUST MIN SET COVER fits conditions of Theorem 2. As we will see, both ways to tackle ROBUST MIN SET COVER lead to the same result.

Consider an instance $(\mathcal{S}, C)$ of MIN SET COVER and this time construct an edge-colored multi-graph $G_C(V_C, E_C, \vec{\ell_{\mathcal{S}}})$ as follows:

- for any $c_i \in C$, add a vertex $v_i \in V_C$;
- for any pair c_i, c_j of elements in C, add a new edge (v_i, v_j) colored with S_k only if $S_k \supseteq \{c_i, c_j\}$.

In the so-constructed graph G_C a set $S_i = \{c_{i_1}, \dots c_{i_k}\} \in \mathcal{S}$ becomes a clique on vertices $v_{i_1}, \dots, v_{i_k} \in V_C$ all the edges of which are colored with the same color S_i; we will call such a clique a unicolored clique. Under this alternative formulation, MIN SET COVER can be viewed as a particular clique-covering problem where the objective is to determine a minimum size covering of V_C by unicolored cliques.

Consider a set cover $\mathcal{S}'$ for the initial instance $(\mathcal{S}, C)$ and a sub-instance I' of $(\mathcal{S}, C)$ consisting of some elements of C and of the subsets of $\mathcal{S}$ including these elements. These objects correspond, in G_C, to a vertex-covering by unicolored cliques and the subgraph G'_C of G_C defined with respect to I'. Restriction of $\mathcal{S}'$ in I', can be viewed, with respect to G_C as restriction of the initial vertex-covering by unicolored cliques to the vertices of G'_C just as described in Section 3. Observe finally that "unicolored clique" is a hereditary property. So, under this formulation, ROBUST MIN SET COVER exactly fits conditions of Theorem 2.

So, according to any of the formulations used for MIN SET COVER, given an instance $(\mathcal{S}, C)$ of MIN SET COVER, with element-probabilities p_i, for any $c_i \in C$, and a feasible solution $\mathcal{S}'$ of $(\mathcal{S}, C)$, then, $E((\mathcal{S}, C), \mathcal{S}') = \sum_{S_i \in \mathcal{S}'}(1 - \prod_{c_j \in S_i}(1 - p_j))$ and can be computed in polynomial time. The robust version of MIN SET COVER amounts to a particular weighted version of the initial problem where each set $S_i = \{c_{i_1}, \dots, c_{i_k}\}$ in $\mathcal{S}$ is weighted as in (4).

Hence, ROBUST MIN SET COVER is indeed a simple weighted version of MIN SET COVER, where one has to determine a set cover minimizing its total weight. In this sense, the problem dealt seems to be simpler than the majority of the problems captured by Theorem 2 as, for instance MIN COLORING. This is due to the fact that, dealing with MIN SET COVER, there is a polynomial number of unicolored cliques in G_C (the sets of $\mathcal{S}$) candidate to be part of any solution, while, for MIN COLORING the number of the independent sets that may be part of a solution is exponential.

3.3 A generic approximation result for the problems fitting conditions of Theorem 2

This section extends an approximation result of [24] for ROBUST MIN COLORING, in order to capture the whole of problems meeting the conditions of Theorem 2.

Consider such an **NPO** problem Π, an instance $G(V, E)$ of Π, set $n = |V|$ and consider a solution $S = (V_1, \dots, V_k)$ of Π on G. Denote by $p_{\min}$ and $p_{\max}$ the minimum and maximum vertex-probabilities, respectively. Then, the following bounds hold for $E(G, S)$:

$$\max\left\{\sum_{i=1}^{n} p_i - \sum_{i=1}^{n}\sum_{j=i+1}^{n} p_i p_j, k p_{\min}\right\} \leqslant E(G, S) \leqslant \min\left\{\sum_{i=1}^{n} p_i \leqslant n p_{\max}, k\right\} \tag{6}$$

Observe first that the rightmost upper bound for $E(G, S)$ in (6) is immediately derived from the expression for $E(G, S)$ in the statement of Theorem 2.

We now prove the leftmost upper bound and lower bound of (6). We first produce a framing for the term $1 - \prod_{v_i \in V_j}(1 - p_i)$. For simplicity, assume $|V_j| = \ell$ and arbitrarily denote vertices

in V_j by $v_1, \ldots, v_\ell$. Then, by induction in ℓ the following holds:

$$\sum_{i=1}^{\ell} p_i - \sum_{i=1}^{\ell} \sum_{j=i+1}^{\ell} p_i p_j \leqslant 1 - \prod_{i=1}^{\ell} (1 - p_i) \leqslant \sum_{i=1}^{\ell} p_i \tag{7}$$

Indeed, for the left-hand side of (7), observe first that it is true for $\ell = 1$ and suppose it true for $\ell = \kappa$, i.e., $\sum_{i=1}^{\kappa} p_i - \sum_{i=1}^{\kappa} \sum_{j=i+1}^{\kappa} p_i p_j \leqslant 1 - \prod_{i=1}^{\kappa}(1 - p_i)$, or:

$$\prod_{i=1}^{\kappa} (1 - p_i) \leqslant 1 - \sum_{i=1}^{\kappa} p_i + \sum_{i=1}^{\kappa} \sum_{j=i+1}^{\kappa} p_i p_j \tag{8}$$

Suppose now that $\ell = \kappa + 1$ and multiply both terms of (8) by $(1 - p_{\kappa+1})$; then:

$$\begin{aligned}
\prod_{i=1}^{\kappa+1} (1 - p_i) &\leqslant \left(1 - \sum_{i=1}^{\kappa} p_i + \sum_{i=1}^{\kappa} \sum_{j=i+1}^{\kappa} p_i p_j\right)(1 - p_{\kappa+1}) \\
&= 1 - \sum_{i=1}^{\kappa} p_i + \sum_{i=1}^{\kappa} \sum_{j=i+1}^{\kappa} p_i p_j - p_{\kappa+1} + p_{\kappa+1} \sum_{i=1}^{\kappa} p_i - p_{\kappa+1} \sum_{i=1}^{\kappa} \sum_{j=i+1}^{\kappa} p_i p_j \\
&= 1 - \sum_{i=1}^{\kappa+1} p_i + \sum_{i=1}^{\kappa+1} \sum_{j=i+1}^{\kappa+1} p_i p_j - p_{\kappa+1} \sum_{i=1}^{\kappa} \sum_{j=i+1}^{\kappa} p_i p_j \leqslant 1 - \sum_{i=1}^{\kappa+1} p_i + \sum_{i=1}^{\kappa+1} \sum_{j=i+1}^{\kappa+1} p_i p_j
\end{aligned}$$

which proves the left-hand side inequality in (7).

For the right-hand side of (7), we show by induction on ℓ that $\prod_{i=1}^{\ell}(1 - p_i) \geqslant 1 - \sum_{i=1}^{\ell} p_i$. This is clearly true for $\ell = 1$. Suppose it also true for any $\ell \leqslant \kappa$, i.e., $\prod_{i=1}^{\kappa}(1 - p_i) \geqslant 1 - \sum_{i=1}^{\kappa} p_i$. Then, by multiplying both members of this inequality by $(1 - p_{\kappa+1})$, we get that the product obtained is equal to $1 - p_{\kappa+1} - \sum_{i=1}^{\kappa} p_i + p_{\kappa+1} \sum_{i=1}^{\kappa} p_i \geqslant 1 - \sum_{i=1}^{\kappa+1} p_i$, q.e.d.

Taking the sums of the members of (7) for $m = 1$ to k, the right-hand side inequality immediately gives $E(G, S) \leqslant \sum_{i=1}^{n} p_i$.

We now prove that $E(G, S) \geqslant \sum_{i=1}^{n} p_i - \sum_{i=1}^{n} \sum_{j=i+1}^{n} p_i p_j$, i.e., the leftmost lower bound claimed in (6). From the left-hand side of (7), we get:

$$\sum_{m=1}^{k} \left(\sum_{i=1}^{\ell} p_i - \sum_{i=1}^{\ell} \sum_{j=i+1}^{\ell} p_i p_j \right) = \sum_{i=1}^{n} p_i - \sum_{m=1}^{k} \sum_{i=1}^{\ell} \sum_{j=i+1}^{\ell} p_i p_j \geqslant \sum_{i=1}^{n} p_i - \sum_{i=1}^{n} \sum_{j=i+1}^{n} p_i p_j \tag{9}$$

Observe that, from the first inequality of (7), we have:

$$\sum_{m=1}^{k} \left(\sum_{i=1}^{\ell} p_i - \sum_{i=1}^{\ell} \sum_{j=i+1}^{\ell} p_i p_j \right) \leqslant \sum_{m=1}^{k} \left(1 - \prod_{i=1}^{\ell} (1 - p_i) \right) \tag{10}$$

The righthand side of (10) is exactly $E(G, S)$. Putting this together with (9), the leftmost lower bound for $E(G, S)$ in (6) is proved.

Finally, in order to derive the rightmost lower bound in (6), observe that $\prod_{v_i \in S_j}(1 - p_i) \leqslant (1 - p_{\min})^{|S_j|} \leqslant 1 - p_{\min}$, i.e., $1 - \prod_{v_i \in S_j}(1 - p_i) \geqslant p_{\min}$. Summing for $j = 1$ to k, we get the bound claimed.

We are ready now to study an approximation algorithm for the whole class of problems meeting Theorem 2. Fix a vertex-probability p', assume that there exists a ρ-approximation polynomial time algorithm `A` for Π, and run the following algorithm, called `RA` for ROBUST Π:

1. partition the vertices of G into three subsets: the first, V_1 including the vertices with probabilities at most $1/n$, the second, V_2, including the vertices with probabilities in the interval $[1/n, p']$ and the third, V_3, including the vertices with probabilities greater than p';
2. feasibly solve Π in $G[V_1]$ and $G[V_2]$ separately;
3. run $\mathtt{A}$ in $G[V_3]$;
4. take the union of the solutions computed in steps 2 and 3 as solution for G.

Theorem 3. *If* $\mathtt{A}$ *achieves approximation ratio* ρ *for* Π*, then* $\mathtt{RA}$ *approximately solves in polynomial time the robust version of* Π *within ratio* $O(\sqrt{\rho n})$*.*

Proof. Denote by $S^* = (V_1^*, \ldots, V_{k^*}^*)$ an optimal anticipatory solution and by $S = (\hat{V}_1, \ldots, \hat{V}_k)$ the approximate solution computed in step 4 and, respectively by $S_i^* = (V_{1,i}^*, \ldots, V_{|S_i^*|,i}^*)$ and $S_i = (\hat{V}_{1,i}, \ldots, \hat{V}_{|S_i|,i})$, the optimal and approximate solutions in $G[V_i]$, $i = 1, 2, 3$. Denote by $S^*[V_1]$, $S^*[V_2]$ and $S^*[V_3]$ the restrictions of S^* in $G[V_1]$, $G[V_2]$ and $G[V_3]$, respectively. Denote finally by n_i, the orders of $G[V_i]$, for $i = 1, 2, 3$, respectively. The proof is based upon the following claims:

1. any feasible polynomial time approximation algorithm for ROBUST Π achieves in $G[V_1]$ approximation ratio bounded above by 2;
2. any feasible polynomial time approximation algorithm for ROBUST Π achieves in $G[V_2]$ approximation ratio bounded above by $O(np')$;
3. assuming that $\mathtt{A}$ achieves approximation ratio ρ for Π, algorithm $\mathtt{RA}$, when running in $G[V_3]$, achieves approximation ratio bounded above by ρ/p' for ROBUST Π.

For Claim 1, using (6) for S_1 and S_1^*, we get: $E(G[V_1], S_1) \leqslant \sum_{i=1}^{n_1} p_i$ and $E(G[V_1], S_1^*) \geqslant \sum_{i=1}^{n_1} p_i - \sum_{i=1}^{n_1} \sum_{j=i+1}^{n_1} p_i p_j$. Combining them, we derive:

$$\begin{aligned}
\frac{E\left(G\left[V_1\right], S_1^*\right)}{E\left(G\left[V_1\right], S_1\right)} &\geqslant 1 - \frac{\sum\limits_{i=1}^{n_1} \sum\limits_{j=i+1}^{n_1} p_i p_j}{\sum\limits_{i=1}^{n_1} p_i} = 1 - \frac{\left(\sum\limits_{i=1}^{n_1} p_i\right)^2 - \sum\limits_{i=1}^{n_1} p_i^2}{2 \sum\limits_{i=1}^{n_1} p_i} \\
&\geqslant 1 - \frac{\sum\limits_{i=1}^{n_1} p_i}{2} + \frac{\sum\limits_{i=1}^{n_1} p_i^2}{2 \sum\limits_{i=1}^{n_1} p_i} \geqslant 1 - \frac{\sum\limits_{i=1}^{n_1} p_i}{2}
\end{aligned} \tag{11}$$

Since p_i's are smaller than $1/n$ and $n_1 \leqslant n$, the right-hand side of (11) is at least as large as $1/2$. Hence, the approximation ratio of any algorithm for Π in $G[V_1]$ is at most 2, and the proof of Claim 1 is complete.

We deal now with Claim 2. Here, for any v_i, $p_i > 1/n$. Consequently, $1 - \prod_{v_i \in V_{j,2}^*}(1 - p_i) \geqslant 1 - (1 - (1/n))^{|V_{j,2}^*|} \geqslant (|V_{j,2}^*|/n) - (|V_{j,2}^*|(|V_{j,2}^*| - 1)/2n^2)$ where the last inequality is an easy application of the left-hand side of (7) with $p_i = 1/n$ for any vertex v_i. Furthermore:

$$\frac{\left|V_{j,2}^*\right|}{n} - \frac{\left|V_{j,2}^*\right|\left(\left|V_{j,2}^*\right| - 1\right)}{2n^2} = \frac{\left|V_{j,2}^*\right|}{n}\left(1 - \frac{\left|V_{j,2}^*\right| - 1}{2n}\right) \geqslant \frac{\left|V_{j,2}^*\right|}{n}\frac{n+1}{2n} \geqslant \frac{\left|V_{j,2}^*\right|}{2n} \tag{12}$$

Summing inequality (12) for $j = 1, \ldots, |S_2^*|$, we get $E(G[V_2], S_2^*) \geqslant n_2/2n$, where n_2 is the order of $G[V_2]$. On the other hand, using the leftmost upper bound in (6) we get $E(G[V_2], S_2) \leqslant n_2 p'$. The bounds for $E(G[V_2], S_2^*)$ and $E(G[V_2], S_2)$ immediately derive approximation ratio at most $2np' = O(np')$ and the proof of Claim 2 is complete.

We now turn to Claim 3. Using the rightmost lower bound of (6), $E(G[V_3], S_3^*) \geqslant |S_3^*|p'$. On the other hand, by the rightmost upper bound of (6), $E(G[V_3], S_3) \leqslant |S_3|$. So, assuming that A achieves ratio ρ for Π, step 3 achieves ratio $|S_3|/|S_3^*|p'$ for $G[V_3]$, that turns out to a ratio bounded above by ρ/p', completing so the proof of Claim 3.

We prove that, for any $k \in \{1, 2, 3\}$: $E(G, S^*) \geqslant E(G[V_k], S^*[V_k]) \geqslant E(G[V_k], S_k^*)$. Remark that $S^*[V_k]$ is a particular feasible solution for $G[V_k]$; hence: $E(G[V_k], S^*[V_k]) \geqslant E(G[V_k], S_k^*)$. In order to prove the first inequality, fix a k and consider a component, say V_j^* of S^*. Then, the contribution of V_j^* in $S^*[V_k]$ is: $1 - \prod_{v_i \in V_j^* \cap V_k}(1 - p_i) \leqslant 1 - \prod_{v_i \in V_j^*}(1 - p_i)$, which is its contribution in S^*. Iterating this argument for all the colors in $C^*[V_k]$, the claim follows.

Algorithm RA solves separately each $G[V_k]$, $k \in \{1, 2, 3\}$ and returns as solution S the union of the colors used. Hence, $E(G, S) = E(G[V_1], S_1) + E(G[V_2], S_2) + E(G[V_3], S_3)$. On the other hand, $E(G, S^*)$ is at least as large as any of $E(G[V_k], S_k^*)$, $k \in \{1, 2, 3\}$. So, the ratio of the algorithm in G is at most the sum of the ratios proved by Claims 1, 2 and 3, i.e., at most $O(2 + np' + (\rho/p'))$.

Remark that the ratio claimed in Claim 2 is increasing with p', while the one in Claim 3 is decreasing with p'. Equality of expressions np' and ρ/p' holds for $p' = \sqrt{\rho/n}$. In this case the value of the ratio obtained is $O(\sqrt{\rho n})$, and the proof of the theorem is now complete. ∎

4 Solutions are subsets of the initial edge-set

We deal in this section with problems for which solutions are sets of edges. Notice that whenever a vertex is absent from some subset $V' \subseteq V$, the edges incident to it are also absent from $G[V']$. So, our assumption is that, given a solution (in terms of a set of edges) S, and a set $V' \subseteq V$ inducing a subgraph $G[V'] = G'(V', E')$ of G, the set $S \cap E'$ is feasible for Π in G'. The main result for this case, is the following theorem.

Theorem 4. *Consider a graph-problem Π verifying the following assumptions: (1) an instance of Π is an edge- (or arc-) valued graph $G(V, E, \vec{\ell})$; (2) any solution of Π on any instance G is a subset of E; (3) for any solution S and any subset $V' \subseteq V$, denoting by $G'(V', E')$ the subgraph of G induced by V', the set $S \cap E'$ is feasible; (4) the value of any solution $S \subseteq E$ of Π is defined by: $m(G, S) = w(S) = \sum_{(v_i, v_j) \in S} \ell(v_i, v_j)$, where $\ell(v_i, v_j)$ is the valuation of the edge (or arc) (v_i, v_j) of G. Then, the functional of* ROBUST Π *is expressed as: $E(G, S) = \sum_{(v_i, v_j) \in S} \ell(v_i, v_j) p_i p_j$ and can be computed in polynomial time. Furthermore, dealing with their respective computational complexities,* ROBUST Π *and Π are equivalent.*

Proof. Set $S' = S \cap E'$; by the assumptions of the theorem, S' is feasible for G'. Furthermore, $m(G', S') = \sum_{(v_i, v_j) \in S} \ell(v_i, v_j) 1_{\{(v_i, v_j) \in E'\}}$. Then, using (1):

$$\begin{aligned} E(G, S) &= \sum_{V' \subseteq V} m\left(G', S'\right) \Pr\left[V'\right] = \sum_{V' \subseteq V} \sum_{(v_i, v_j) \in S} \ell\left(v_i, v_j\right) 1_{\{(v_i, v_j) \in E'\}} \Pr\left[V'\right] \\ &= \sum_{(v_i, v_j) \in S} \ell\left(v_i, v_j\right) \sum_{V' \subseteq V} 1_{\{(v_i, v_j) \in E'\}} \Pr\left[V'\right] \end{aligned} \tag{13}$$

Any edge (or arc) $(v_i, v_j) \in E$ belongs to $G' = G[V']$, if and only if both of its endpoints belong to V'. Let $V_{ij} = V \setminus \{v_i, v_j\}$ and $\mathcal{V}'_{ij} = \{V' \subseteq V : V' = \{v_i\} \cup \{v_j\} \cup V'', V'' \subseteq V_{ij}\}$, the set of

all the subsets of V containing both v_i and v_j. Using also the fact that presence-probabilities of the vertices of V are independent, we get:

$$\begin{aligned}\sum_{V'\subseteq V} 1_{\{(v_i,v_j)\in E'\}} \Pr\left[V'\right] &= \sum_{V'\in\mathcal{V}'_{ij}} \Pr\left[V'\right] = \sum_{V''\subseteq V_{ij}} \Pr\left[\{v_i\}\cup\{v_j\}\cup V''\right] \\ &= \sum_{V''\subseteq V_{ij}} p_ip_j \Pr\left[V''\right] = p_ip_j \sum_{V''\subseteq V_{ij}} \Pr\left[V''\right] = p_ip_j \quad (14)\end{aligned}$$

Combination of (13) and (14) immediately leads to the expression claimed for the functional.

It is easy to see that this functional can be computed in time quadratic with n. Furthermore, computation of an optimal anticipatory solution for ROBUST Π in G obviously amounts to a computation of an optimal solution for Π in an edge- (or arc-) valued graph $G(V,E,\vec{\ell'})$ where, for any $(v_i,v_j)\in E$, $\ell'(v_i,v_j)=\ell(v_i,v_j)p_ip_j$. Consequently, by this observation and by assumption (4), Π and ROBUST Π have the same complexity. ∎

The reasons for which the functional derived in Theorem 4 becomes polynomial are quite analogous to the ones in Theorem 1. Since an edge that does not belong to the anticipatory solution S will never be part of any solution in any subgraph $G'(V',E')$ of G, the computation of the functional amounts to the specification, for any G', of the set $S\cap E'$. This is equivalent to first determining, for any edge $e\in S$, all the subgraphs containing e and next to a summation of the probabilities of these subgraphs.This sum equals to the product of the probabilities of the endpoints of e.

Let us note that, as in Section 2, Theorem 4 can be used for getting generic approximation results for ROBUST Π. Since this problem is a particular weighted version of Π, one immediately concludes that *if* Π *is approximable within approximation ratio* ρ*, so is* ROBUST Π.

Corollary 4. *Under the hypotheses of Theorem 4, whenever* Π *and* ROBUST Π *are* ***NP****-hard, they are equi-approximable.*

Corollary 5. *Consider a robust combinatorial optimization problem* Π *verifying assumptions (1) through (4) of Theorem 4 with* $\vec{\ell}=\vec{1}$*. Then, the functional of* ROBUST Π *is expressed as:* $E(G,S)=\sum_{(v_i,v_j)\in S} p_ip_j$ *and can be computed in polynomial time and* ROBUST Π *is equivalent to an edge- (or arc-) valued version of* Π *where the values of an edge is the product of the probabilities of its endpoints.*

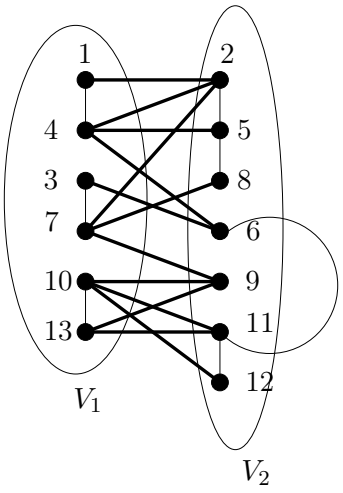
4.1 Application of Theorem 4 to ROBUST MAX MATCHING

In MAX MATCHING, the objective is, given a graph $G(V,E)$ to determine a maximum-size matching, i.e., a maximum-size subset of E such that its edges are pairwise disjoint, in the sense that they have no common endpoint.

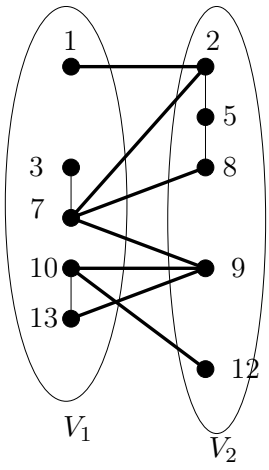
Clearly, MAX MATCHING in both edge-valued and non-valued graphs, fits conditions of Theorem 4 and Corollary 5, respectively. Moreover, since MAX WEIGHTED MATCHING is polynomial, both ROBUST MAX WEIGHTED MATCHING and ROBUST MAX MATCHING are polynomial also.

4.2 Application of Theorem 4 to ROBUST MAX CUT

We deal here with MAX CUT in both edge-valued and unitary edge-valued graphs. Consider a graph $G(V,E)$. In MAX CUT (resp. MAX WEIGHTED CUT) we wish to determine a maximum cardinality (resp., maximum weight) cut, that is to partition V into two subsets V_1 and V_2 such that a maximum number of edges (resp., maximum-weight set of edges) have one of their endpoints in V_1 and the other one in V_2.



(a) A graph G with a cut S (thick edges)



(b) Some "surviving" subgraph and the "surviving" solution

Figure 1: An example for ROBUST MAX CUT.

We can represent an anticipatory cut S as a set of edges in such a way that whenever $(v_i, v_j) \in S$, $v_i \in V_1$ and $v_j \in V_2$. For example, in Figure 1(a), where for simplicity values of edges are not mentioned, the cut partitions V in subsets $V_1 = \{1, 3, 4, 7, 10, 13\}$ and $V_2 = \{2, 5, 6, 8, 9, 11, 12\}$ and the anticipatory cut S (thick edges) can then be written as $S = \{(1, 2), (3, 6), (4, 2), (4, 5), (4, 6), \ldots, (13, 11)\}$ (where edges are ranged in lexicographic order). In Figure 1(b), we present graph's and cut's states assuming that vertices 4, 6 and 11 are absent. The solution S' considered misses in all edges of S having at least one endpoint among $\{4, 6, 11\}$ but it obviously remains a feasible cut for the surviving graph.

Hence, both weighted and cardinality ROBUST MAX CUT meet the conditions of Theorem 4 and Corollary 5, respectively. Consequently, MAX CUT being **NP**-hard, ROBUST MAX WEIGHTED CUT and ROBUST MAX CUT are also **NP**-hard.

5 When things become complicated

In this section we tackle edge-weighted graph-problems where feasible solutions are connected sets of edges (for example, paths, trees, cycles, etc.) but we assume that, given a solution S and a set $V' \subseteq V$ inducing a subgraph $G[V'] = G'(V', E')$ of G, the set $S \cap E'$ is not always feasible for G'.

Formally, consider a problem Π where a feasible solution is a connected set S of edges. Consider also that vertices in S are ordered in some appropriate order. Assume that operation $S \cap E'$ results in a set of connected subsets $C_1, C_2, \ldots, C_k$ of S but that $S'' = \cup_{i=1}^{k} C_i$ is not connected (i.e., S'' does not constitute a feasible solution for Π). Assume also that connected subsets $C_1, C_2, \ldots, C_k$ are also ranged in this order (always following some appropriate ordering implied by the one of S).

We consider a kind of "completion" of S'' by additional edges linking, for $i = 1, \ldots, k-1$, the last vertex (in the ordering considered for S) of C_i with the first vertex of C_{i+1}. In other words, given S (representing a connected set of edges), we apply the following algorithm, denoted by `A` in the sequel:

1. range the vertices of S following some appropriate order;

2. compute $S \cap E'$; let $C_1, C_2, \ldots, C_k$ be the resulting connected components of $S \cap E'$;

3. for $i = 1, \ldots, k-1$, use an edge to link the last vertex of C_i with the first vertex of C_{i+1};

4. output S' the solution so computed.

Obviously, in order that step 3 of A is able to link components C_i and C_{i+1}, an edge must exist between the vertices implied; otherwise, A is definitely unfeasible. So, in order to assure feasibility, we make, for the rest of the section the basic assumption that the input graph for the problems tackled is complete.

In what follows, we denote by $V[S']$ the set of vertices in S' and set $G''(V[S'], E'') = G[V[S']]$. Also, we denote by $[v_i, v_j]$ the set $\{v_{i+1}, v_{i+2}, \ldots, v_{j-1}\}$ ($i < j$ in the ordering assumed for S) such that: (a) for any $\ell = i, i+1, \ldots, j-1$, $(v_\ell, v_{\ell+1}) \in S$ (i.e., $[v_i, v_j]$ is the set of vertices in the path linking v_i to v_j in S, where v_i and v_j themselves are not encountered[3]) and (b) v_i and v_j belong to consecutive[4] connected subsets C_m and C_{m+1}, for some $m < k$.

Theorem 5. *Consider a robust combinatorial optimization problem* ROBUST Π *verifying the following assumptions: (i) instances of* Π *are edge-valued complete graphs* $(K_n, \vec{\ell}) = G(V, E, \vec{\ell})$*; furthermore, in the robust version of* Π *any vertex* $v_i \in V$ *has a presence-probability* p_i*; (ii) a solution of* Π *is a subset* S *of* E *verifying some connectivity property; (iii) given an anticipatory solution* S *(the vertices of which are ranged in some appropriate order), algorithm* A *computes a feasible solution* S'*, for any subgraph* $G'(V', E', \vec{\ell}) = G[V']$ *of* G *(obviously,* G' *is complete); (iv)* $m(G, S) = \sum_{(v_i,v_j)\in S} \ell(v_i, v_j)$*. Then,* $E(G, S)$ *is computable in polynomial time and is expressed by:* $E(G, S) = \sum_{(v_i,v_j)\in S} \ell(v_i, v_j) p_i p_j + \sum_{(v_i,v_j)\in E''\setminus S} \ell(v_i, v_j) p_i p_j \prod_{v_l\in[v_i,v_j]} (1 - p_l)$.

Proof. Denote by $C[E']$, the set of edges added to S'' during the execution of step 3 of A. Obviously, $S' = S'' \cup C[E']$; also, if an edge belongs to $C[E']$, then it necessarily belongs to $E[S]$, the set of edges in E induced by the endpoints of the edges in S. By assumptions (i) to (iii), S' is a feasible set of edges. Furthermore:

$$m\left(G', S'\right) = \sum_{(v_i,v_j)\in E} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in S'\}} = \sum_{(v_i,v_j)\in E} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in S''\cup C[E']\}} \tag{15}$$

By construction, any element of $C[E']$ is an edge (or arc) the initial endpoint of which corresponds to the terminal endpoint of a connected subset C_i of S and the terminal endpoint of which corresponds to the initial endpoint of the "next" connected subset C_{i+1} of S. Then, for any subgraph G' of G, the following two assertions hold: (1) $S' \subseteq E''$, and (2) any edge that does not belong to E'', will never be part of any feasible solution. Indeed, for such an edge, at least one of its endpoints does not belong to $V[S']$; so, $C[E'] \subseteq E''$. We so have from (15):

$$\begin{aligned} m\left(G', S'\right) &= \sum_{(v_i,v_j)\in E} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in S''\cup C[E']\}} = \sum_{(v_i,v_j)\in E''} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in S''\cup C[E']\}} \\ &= \sum_{(v_i,v_j)\in E''} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in S''\}} + \sum_{(v_i,v_j)\in E''} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in C[E']\}} \\ &= \sum_{(v_i,v_j)\in S} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in E'\}} + \sum_{(v_i,v_j)\in E''\setminus S} \ell\left(v_i, v_j\right) 1_{\{(v_i,v_j)\in C[E']\}} \end{aligned} \tag{16}$$

[3]It is assumed that if $[v_i, v_j] = \emptyset$, then $\prod_{v_l\in[v_i,v_j]}(1 - p_l) = 0$.

[4]With respect to the order $C_1, \ldots, C_k$.

Using (16), we get:

$$
\begin{aligned}
E(G,S) &= \sum_{V' \subseteq V} \left(\sum_{(v_i,v_j) \in S} \ell(v_i,v_j)\, 1_{\{(v_i,v_j) \in E'\}} + \sum_{(v_i,v_j) \in E'' \setminus S} \ell(v_i,v_j)\, 1_{\{(v_i,v_j) \in C[E']\}} \right) \Pr\left[V'\right] \\
&= \sum_{(v_i,v_j) \in S} \ell(v_i,v_j) \sum_{V' \subseteq V} 1_{\{(v_i,v_j) \in E'\}} \Pr\left[V'\right] \\
&\quad + \sum_{(v_i,v_j) \in E'' \setminus S} \ell(v_i,v_j) \sum_{V' \subseteq V} 1_{\{(v_i,v_j) \in C[E']\}} \Pr\left[V'\right] \qquad (17)
\end{aligned}
$$

As in the proof of Theorem 4, the first term of (17) can be simplified as follows:

$$
\sum_{(v_i,v_j) \in S} \ell(v_i,v_j) \sum_{V' \subseteq V} 1_{\{(v_i,v_j) \in E'\}} \Pr\left[V'\right] = \sum_{(v_i,v_j) \in S} \ell(v_i,v_j)\, p_i p_j \qquad (18)
$$

Using (18) in (17), we get:

$$
E(G,S) = \sum_{(v_i,v_j) \in S} \ell(v_i,v_j)\, p_i p_j + \sum_{(v_i,v_j) \in E'' \setminus S} \ell(v_i,v_j) \sum_{V' \subseteq V} 1_{\{(v_i,v_j) \in C[E']\}} \Pr\left[V'\right] \qquad (19)
$$

We now deal with the second term of (19) that, in this form seems to be exponential. Consider some edge (v_i, v_j) added during step 3 in order to "patch", say connected components C_l and C_{l+1} of the anticipatory solution S. Since $(v_i, v_j) \notin S$, there exists in S a sequence $\mu = [v_i, v_j]$ of consecutive edges (or arcs) linking v_i to v_j. Assume that this sequence is listed by its vertices and that neither v_i, nor v_j belong to μ. Edge $(v_i, v_j) \in E'' \setminus S'$ is added to S' just because all the vertices in μ are absent. In other words, inclusion $(v_i, v_j) \in C[E']$ holds for any subgraph $G'(V', E')$, where $V' \in \mathcal{U}'_{ij}$ with: $\mathcal{U}'_{ij} = \{V' \subseteq V : v_i \in V', v_j \in V',$ any vertex of $\mu = [v_i, v_j]$ is absent$\}$. Consequently, the second sum in the second term of (19) can be written as:

$$
\sum_{V' \subseteq V} 1_{\{(v_i,v_j) \in C[E']\}} \Pr\left[V'\right] = \sum_{V' \in \mathcal{U}'_{ij}} \Pr\left[V'\right] = p_i p_j \prod_{v_l \in [v_i,v_j]} (1 - p_l) \qquad (20)
$$

Combination of (17), (19) and (20) derives the expression claimed for the functional. It is easy to see that computation of a single term in the second sum of the functional requires $O(n)$ computations (at most $n + 1$ multiplications). Since we may do this for at most $O(n^2)$ times (the edges in E), it ensues that the whole complexity of functional's computation is of $O(n^3)$, that concludes the proof of the theorem. ∎

The fact that $E(G, S)$ is polynomial is partly due to the same reasons as in Theorems 1 and 4. Furthermore, the order of the additional edges-choices in step 3 of `A` is also crucial for this efficience. Indeed, this order is such that one can say a priori under which conditions an edge (or arc) (v_i, v_j) will be added in S'. These conditions carry over, the presence or the absence of the edges initially lying between v_i and v_j in S.

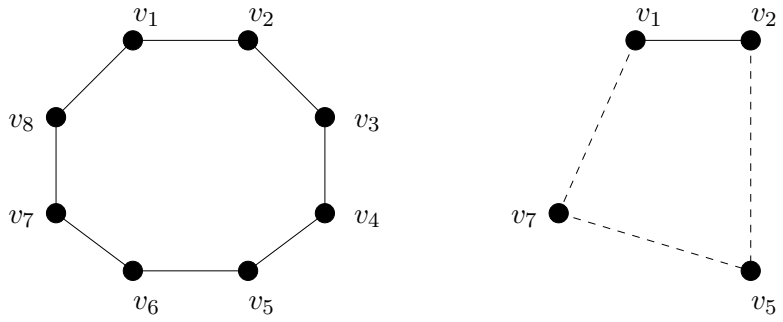
Unfortunately, in the opposite of Theorems 1 and 4, Theorem 5 does not derive a "good" characterization for the optimal anticipatory solutions of the problems meeting the assumptions (i) to (iv). In particular, the form of the functional does not imply solution of some well-defined weighted version of Π (the deterministic support of ROBUST Π). Indeed due to the second term of the expression for $E(G, S)$ in Theorem 5, the "costs" assigned to the edges depend on the structure of the anticipatory solution chosen.

In what follows, we outline some problems dealing with assumptions of Theorem 5. In particular, we tackle cases where feasible solutions are either cycles or trees.

5.1 Application of Theorem 5 when the anticipatory solution is a cycle

In this section, we consider MIN TSP and its robust version. Given a complete graph on n vertices, denoted by K_n, with positive distances on its edges, MIN TSP consists of minimizing the cost of a Hamiltonian cycle (i.e., of an ordering $\langle v_1, v_2, \ldots, v_n \rangle$ of V such that $v_n v_1 \in E$ and, for $1 \leqslant i < n$, $v_i v_{i+1} \in E$), the cost of such a cycle being the sum of the distances of its edges. We shall represent any Hamiltonian cycle T (called also a tour in what follows) as the set of its edges; its value is $m(K_n, T) = \sum_{e \in T} \ell(v_i, v_j)$. Moreover, we arbitrarily number the vertices of K_n in the order that they are visited in T; so, we can set $T = \{(v_1, v_2), \ldots, (v_i, v_{i+1}), \ldots, (v_{n-1}, v_n), (v_n, v_1)\}$.

Consider an anticipatory tour T in an edge-valued complete graph K_n and a set of absent vertices. Then, application of step 2 of A may result in a set $\{P_1, P_2, \ldots, P_k\}$ of paths[5], ranged in the order vertices have been visited in T, that is not feasible for MIN TSP in the surviving graph. In order to render this set feasible, one can link (modulo k) the last vertex of the path P_i to the first vertex of P_{i+1}; this is always possible since the initial graph is complete.



(a) An anticipatory tour

(b) The tour T' computed by A

Figure 2: An example of application of algorithm A for ROBUST MIN TSP.

For example, in Figure 2(a), an anticipatory cycle T, derived from a (symmetric) K_8 is shown. In Figure 2(b), we consider that vertices v_3, v_4, v_6 and v_8 are absent. In a first time, application of Step 2 of A results in a path-set $\{\{(v_1, v_2)\}, \{v_5\}, \{v_7\}\}$. In a second time, we will link vertex v_2 to v_5 (using dotted edge (v_2, v_5)) and vertex v_5 to v_7 (by dotted edge (v_5, v_7)). This creates a Hamiltonian path linking all the surviving vertices of the initial K_8. Finally, we link vertex v_7 to v_1 (by the dotted edge (v_7, v_1)). We so build a new tour feasibly visiting all the present vertices of the remaining graph.

It is easy to see that for the case we deal with, all conditions of Theorem 5 are verified. Consequently, its application for the case of ROBUST MIN TSP gives for $E(K_n, T)$ the expression claimed in the theorem. We so recover the result of [13] about ROBUST MIN TSP. The anticipatory solution minimizing the functional cannot be characterized tightly by means of Theorem 5, since the expression for $E(K_n, T)$ depends on the particular anticipatory tour T considered and by the way this particular tour will be completed in the surviving instance.

5.2 Application of Theorem 5 when the anticipatory solution is a tree

Let us now consider MIN SPANNING TREE. Given an edge-valued graph $G(V, E, \vec{\ell})$, MIN SPANNING TREE consists of determining a tree T spanning V and minimizing quantity $\sum_{e \in T} \ell(e)$. For reasons that will be understood later (see also [6]) we restrict ourselves to complete graphs. As

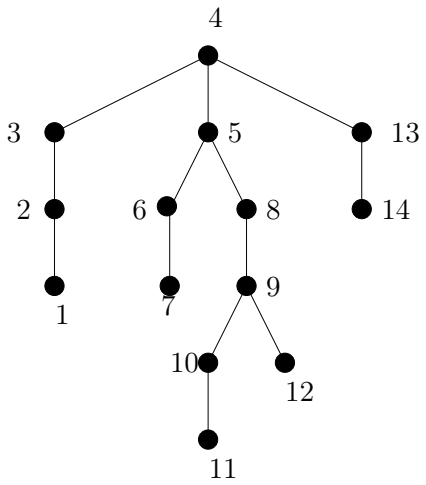
[5]These paths may be sets of edges, or simple edges, or even isolated vertices, any such vertex considered as a path.

previously in Section 5.1, we consider a tree by the set of its edges. For any tree T its value is $m(K_n, T) = \sum_{e \in T} \ell(v_i, v_j)$.

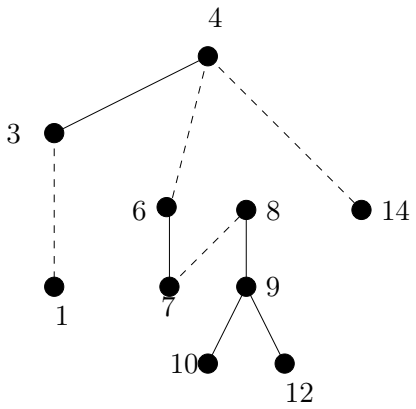
Note that for ROBUST MIN TSP in Section 5.1, its solution induces an implicit and natural ordering of the edges. This is not the case here since various orderings can be considered. We consider the one given by a kind of inverse depth-first-search (dfs) that goes as follows:

- starting from leaf numbered by 1 we first number its incident edge, then one of the edges intersecting it and so on, until we arrive to a second leaf;
- we then return back to the last vertex of the path incident to an edge not numbered yet and we continue the numbering from this edge until a third leaf is encountered;
- we continue in this way until all the edges of T are visited.

This ordering is performed in $O(n)$ for a tree of order n (recall that such a tree has $n-1$ edges).



(a) The ordering of the nodes of an anticipatory solution T



(b) The solution T' derived from application of algorithm A on T

Figure 3: When anticipatory solution is a tree.

For example, consider the tree of Figure 3(a) and assume that it is a minimum spanning tree of some graph on 14 vertices. Starting from the leftmost leaf (numbered by 1) we visit edges $(1,2),(2,3),\ldots,(6,7)$ in this order; node 7 is another leaf. We then return to node 5 that is the last node incident to an edge not yet visited, and we visit edges $(5,8),(8,9)(9,10),(10,11)$. We then return to node numbered by 9 and visit edge $(9,12)$. We return back to node numbered by 4 and visit edges $(4,13),(13,14)$ in this order. The ordering just obtained, partitions the edges of the tree into a number of edge-disjoint paths $P_1, P_2, \ldots$ For instance, dealing with Figure 3(a), the depicted tree T is partitioned into 4 paths: $P_1 = \{(1,2),(2,3),\ldots,(6,7)\}$, $P_2 = \{(5,8),(8,9)(9,10),(10,11)\}$, $P_3 = \{(9,12)\}$ and $P_4 = \{(4,13),(13,14)\}$.

Suppose now that some vertices are absent from the initial graph G. Also, let us refer to the nodes by their numbering in the ordering just described. Then, step 2 of A will produce a non connected set of edges (forming paths, any of them being a subset of some P_i); denote by $\{P'_1, P'_2, \ldots, P'_k\}$ the set of paths so-obtained. Order them paths following the order their edges appear in the inverse dfs paths of T. For any $l = 1, \ldots, k$, we link the last vertex, say i of path P'_l to the first vertex, say j, of the path P'_{l+1}. Since the initial graph is assumed complete, such an edge always exists.

We have now to explicit the path $[i, j]$ associated with the edge (i, j) connecting P'_l and P'_{l+1}. Starting from T the edges of which are ordered as described, one can, w.l.o.g., renumber the vertices of the graph in such a way that T can be rewritten as $T = \{(1, 2), (2, 3), \ldots, (n-k, n)\}$. Then, T can be seen as a sequence of vertices some of them appearing more than once, i.e., $T = (1_1, 2_1, 3_1, \ldots, j_1, i_2, (j+1)_1, \ldots, (n-k)_q, n_1)$, where $i < j$ and i_c represents the c-th time node i is encountered in T. Based upon this representation, one can reconstruct T in the following way: for any pair $(i_c, j_{c'})$ of consecutive vertices, edge (i, j) belongs to T if and only if $i < j$. Note that a leaf appears only once in the list.

Application of step 2 of `A` implies that, whenever an absent vertex v is not a leaf, its removal disconnects T. In order to reconnect it, we must link the vertex preceding it in the sequence to the one succeeding it. By the particular form of the sequence considered, any edge (i, j) that algorithm `A` is likely to add in order to reconnect T is such that $i < j$; the corresponding path $[i, j]$ (i.e., the list of vertices that have to be absent in order that (i, j) is added), is the portion of the list between i_l and j_1, where i_l is the last occurrence of i before the first occurrence j_1 of vertex j.

Let us revisit the example of Figure 3(a). The sequence associated with the tree is $T = (1_1, \ldots, 7_1, 5_2, 8_1, \ldots, 11_1, 9_2, 12_1, 4_2, 13_1, 14_1)$. Assume now that vertices 2, 5, 11 and 13 disappear from the initial graph. Application of step 2 of `A` results in the subsequence $T' = (1_1, 3_1, 4_1, 6_1, 7_1, 8_1, 9_1, 10_1, 9_2, 12_1, 4_2, 14_1)$. The first connected component (path) in this sequence is vertex 1 itself, and the second one is represented by nodes 3 and 4 (3_1 and 4_1 in T'); we so add edge $(1, 3)$ in order to connect these two components. The third component is represented by vertices 6 and 7 (6_1 and 7_1 in T'); edge $(4, 6)$ is consequently added. The fourth connected component is induced by vertices 8, 9, 10 and 12 (8_1, 9_1, 10_1, 9_2, and 12_1 in T'); hence, edge $(7, 8)$ added. The fifth connected component in the sequence is vertex 4 itself (4_2 in T') but edge $(12, 4)$ is not added because $12 > 4$. Finally, the sixth connected component, vertex 14 (14_1 in T'), entails introduction of edge $(4, 14)$ that completes the modification of T by algorithm `A`. Figure 3(b), where slotted edges represent edges added during execution of step 3 of `A`, illustrates what has been just discussed.

Denoting also by T the sequence of vertices representing the anticipatory spanning tree T and denoting any vertex of the initial graph by its numbering in the inverse dfs explained above, $E(K_n, T)$ can be expressed as claimed by Theorem 5.

6 Final remarks

We have drawn a framework for the classification of robust problems under the a priori optimization paradigm. What seems to be of interest in this classification is that when restriction of the initial solution to the "present" subgraph is feasible, then the complexity of determining the optimal anticipatory solution for the problems tackled, amounts to the complexity of solving some weighted version of the deterministic problem, where the weights depend on the vertex-probabilities. These weights do not depend on particular characteristics of the anticipatory solution considered, thing that allows a compact characterization of optimal anticipatory solution. On the contrary, when more-than-one-stage algorithms are needed for building solutions, then the observation above is no more valid. In this case, one also recovers some weighted version of the original problem, but the weights on the data cannot be assigned independently of the structure of a particular anticipatory solution.

References

[1] I. Averbakh. On the complexity of a class of combinatorial optimization problems with uncertainty. *Math. Programming, Ser. A*, 90:263–272, 2001.

[2] I. Averbakh, O. Berman, and D. Simchi-Levi. Probabilistic a priori routing-location problems. *Naval Res. Logistics*, 41:973–989, 1994.

[3] M. Bellalouna, C. Murat, and V. Th. Paschos. Probabilistic combinatorial optimization problems: a new domain in operational research. *European J. Oper. Res.*, 87(3):693–706, 1995.

[4] D. J. Bertsimas. *Probabilistic combinatorial optimization problems*. Phd thesis, Operations Research Center, MIT, Cambridge Mass., USA, 1988.

[5] D. J. Bertsimas. On probabilistic traveling salesman facility location problems. *Transportation Sci.*, 3:184–191, 1989.

[6] D. J. Bertsimas. The probabilistic minimum spanning tree problem. *Networks*, 20:245–275, 1990.

[7] D. J. Bertsimas, P. Jaillet, and A. Odoni. A priori optimization. *Oper. Res.*, 38(6):1019–1033, 1990.

[8] L. Bianchi, J. Knowles, and N. Bowler. Local search for the probabilistic traveling salesman problem: correction to the 2-p-opt and 1-shift algorithms. *European J. Oper. Res.*, 161(1):206–219, 2005.

[9] J. Birge and F. Louveaux. *Introduction to stochastic programming*. Springer, Berlin, 1997.

[10] V. G. Deĭneko and G. J. Woeginger. On the robust assignment problem under a fixed number of cost scenarios. *Operations Research Letters*. To appear.

[11] A. Gupta, M. Pál, R. Ravi, and A. Sinha. Boosted sampling: approximation algorithms for stochastic optimization. In *Proc. STOC'04*, pages 417–426, 2004.

[12] A. Gupta, M. Pál, R. Ravi, and A. Sinha. What about wednesday? approximation algorithms for multistage stochastic optimization. In *Proc. Approximation Algorithms for Combinatorial Optimization Problems, APPROX'05*, Lecture Notes in Computer Science. Springer-Verlag, 2005.

[13] P. Jaillet. Probabilistic traveling salesman problem. Technical Report 185, Operations Research Center, MIT, Cambridge Mass., USA, 1985.

[14] P. Jaillet. A priori solution of a traveling salesman problem in which a random subset of the customers are visited. *Oper. Res.*, 36(6):929–936, 1988.

[15] P. Jaillet. Shortest path problems with node failures. *Networks*, 22:589–605, 1992.

[16] P. Jaillet and A. Odoni. The probabilistic vehicle routing problem. In B. L. Golden and A. A. Assad, editors, *Vehicle routing: methods and studies*. North Holland, Amsterdam, 1988.

[17] P. Kouvelis and G. Yu. *Robust discrete optimization and its applications*. Kluwer Academic Publishers, Boston, 1997.

[18] R. Montemanni, L. M. Gambardella, and A. V. Donati. A branch and bound algorithm for the robust shortest path problem with interval data. *Oper. Res. Lett.*, 32:225–232, 2004.

[19] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios. Robust optimization of large-scale systems. *Oper. Res.*, 43(2):264–281, 1995.

[20] C. Murat and V. Th. Paschos. The probabilistic longest path problem. *Networks*, 33:207–219, 1999.

[21] C. Murat and V. Th. Paschos. A priori optimization for the probabilistic maximum independent set problem. *Theoret. Comput. Sci.*, 270:561–590, 2002. Preliminary version available on `http://www.lamsade.dauphine.fr/~paschos/documents/c166.pdf`.

[22] C. Murat and V. Th. Paschos. The probabilistic minimum vertex-covering problem. *Int. Trans. Opl Res.*, 9(1):19–32, 2002. Preliminary version available on `http://www.lamsade.dauphine.fr/~paschos/documents/c170.pdf`.

[23] C. Murat and V. Th. Paschos. L'optimisation combinatoire probabiliste. In V. Th. Paschos, editor, *Optimisation combinatoire : concepts avancés*, chapter 6, pages 221–247. Hermès Science, Paris, 2005.

[24] C. Murat and V. Th. Paschos. On the probabilistic minimum coloring and minimum $\boldsymbol{k}$-coloring. *Discrete Appl. Math.*, 154:564–586, 2006.

[25] C. Murat and V. Th. Paschos. *Probabilistic combinatorial optimization on graphs.* ISTE and Hermès Science Publishing, London, 2006.

[26] A. Prekopa. *Stochastic programming.* Kluwer Academic Publishers, The Netherlands, 1995.

[27] R. Ravi and A. Sinha. Hedging uncertainty: approximation algorithms for stochastic optimization problems. In *Proc. International Conference on Integer Programming and Combinatorial Optimization, IPCO'04*, volume 3064 of *Lecture Notes in Computer Science*, pages 101–115. Springer-Verlag, 2004.