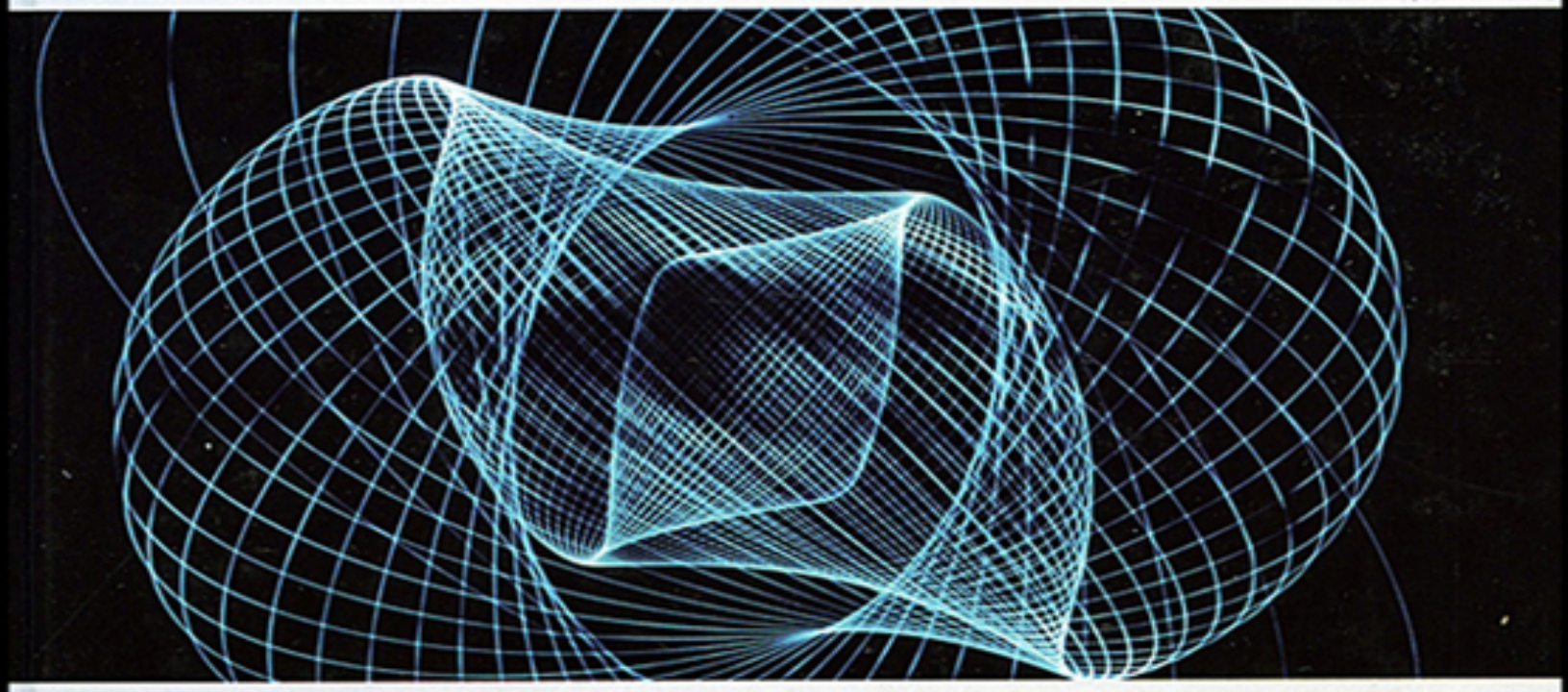
Probabilistic Combinatorial Optimization on Graphs

Cécile Murat and Vangelis Th. Paschos



ISTE

Table of Contents

Probabilistic Combinatorial Optimization on Graphs

Cécile Murat
and
Vangelis Th. Paschos

iSTE

First published in Great Britain and the United States in 2006 by ISTE Ltd

ISTE Ltd
6 Fitzroy Square
London W1T 5DX
UK

ISTE USA
4308 Patrice Road
Newport Beach, CA 92663
USA

www.iste.co.uk

Library of Congress Cataloging-in-Publication Data

Murat, Cecile.

Probabilistic combinatorial optimization on graphs / Cécile Murat and Vangelis Th. Paschos.

p. cm.

Includes bibliographical references and index.

ISBN-13: 978-1-905209-33-0

ISBN-10: 1-905209-33-9

1. Combinatorial probabilities. 2. Combinatorial optimization. 3. Random graphs. I. Paschos, Vangelis Th. II. Title.

QA273.45.M87 2006

519.2--dc22

2006000868

British Library Cataloguing-in-Publication Data

A CIP record for this book is available from the British Library

ISBN 10: 1-905209-33-9

ISBN 13: 978-1-905209-33-0

Preface

This monograph is the outcome of our work on probabilistic combinatorial optimization since 1994. The first time we heard about it, it seemed to us to be a quite strange scientific area, mainly because randomness in graphs is traditionally expressed by considering probabilities on the edges rather than on the vertices. This strangeness was our first motivation to deal with probabilistic combinatorial optimization. As our study progressed, we have discovered nice mathematical problems, connections with other domains of combinatorial optimization and of theoretical computer science, as well as powerful ways to model real-world situations in terms of graphs, by representing reality much more faithfully than if we do not use probabilities on the basic data describing them, i.e., the vertices.

What is probabilistic combinatorial optimization? Basically, it is a way to deal with aspects of robustness in combinatorial optimization. The basic problematic is the following. We are given a graph (let us denote it by $G(V, E)$, where V is the set of its points, called vertices, and E is a set of straight lines, called edges, linking some pairs of vertices in V), on which we have to solve some optimization problem Π. But, for some reasons depending on the reality modelled by G, Π is only going to be solved for some subgraph G' of G (determined by the vertices that will finally be present) rather than for the whole of G. The measure of how likely it is that a vertex $v_i \in V$ will belong to G' (i.e., will be present for the final optimization) is expressed by a probability p_i associated with v_i. How we can proceed in order to solve Π under this kind of uncertainty?

A first very natural idea that comes to mind is that one waits until G' is specified (i.e., it is present and ready for

optimization) and, at this time, one solves Π in G'. This is what is called *re-optimization*. But what if there remains very little time for such a computation? We arrive here at the basic problematic of the book. If there is no time for re-optimization, another way to proceed is the following. One solves Π in the whole of G in order to get a feasible solution (denoted by S), called *a priori solution*, which will serve her/him as a kind of benchmark for the solution on the effectively present subgraph G'. One has also to be provided with an algorithm that modifies S in order to fit G'. This algorithm is called *modification strategy* (let us denote it by **M**). The objective now becomes to compute an *a priori* solution that, when modified by **M**, remains "good" for any subgraph of G (if this subgraph is the one where Π will be finally solved). This amounts to computing a solution that optimizes a kind of expectation of the size of the modification of S over all the possible subgraphs of G, i.e., the sum of the products of the probability that G' is the finally present graph multiplied by the value of the modification of S in order to fit G' over any subgraph G' of G. This expectation, depending on both the instance of the deterministic problem Π, the vertex-probabilities, and the modification strategy adopted, will be called the *functional*. Obviously, the presence-probability of G' is the probability that all of its vertices are present.

Seen in this way, the probabilistic version PΠ of a (deterministic) combinatorial optimization problem Π becomes another equally deterministic problem Π′, the solutions of which have the same feasibility constraints as those of Π but with a different objective function where vertex-probabilities intervene. In this sense, probabilistic combinatorial optimization is very close to what in the last couple of years has been called "one stage optimisation under independent decision models", an area very popular in the stochastic optimization community.

What are the main mathematical problems dealing with probabilistic consideration of a problem Π in the sense discussed above? We can identify at least five interesting mathematical and computational problems dealing with probabilistic combinatorial optimization:

1) write the functional down in an analytical closed form;

2) if such an expression of the functional is possible, prove that its value is polynomially computable (this amounts to proving that the modified problem Π′ belongs to **NP**);

3) determine the complexity of the computation of the optimal *a priori* solution, i.e., of the solution optimizing the functional (in other words, determine the computational complexity of Π′);

4) if Π′ is **NP**-hard, study polynomial approximation issues;

5) always, under the hypothesis of the **NP**-hardness of Π′, determine its complexity in the special cases where Π is polynomial, and in the case of **NP**-hardness, study approximation issues.

Let us note that, although curious, point 2 in the above list in neither trivial nor senseless. Simply consider that the summation for the functional includes, in a graph of order n, 2^n terms (one for each subgraph of G). So, polynomiality of the computation of the functional is, in general, not immediate.

Dealing with the contents of the book, in Chapter 1 probabilistic combinatorial optimization is formally introduced and some old relative results are quickly presented.

The rest of the book is subdivided into two parts. The first one (Part I) is more computational, while the second (Part II) is rather “structural”. In Part I, after formally introducing probabilistic combinatorial optimization and presenting some older results (Chapter 1), we deal with probabilistic versions of four paradigmatic combinatorial problems, namely, PROBABILISTIC MAX INDEPENDENT SET,

PROBABILISTIC MIN VERTEX COVER, PROBABILISTIC LONGEST PATH and PROBABILISTIC MIN COLORING (Chapters 2, 3, 4 and 5, respectively). For any of them, we try, more or less, to solve the five types of problems just mentioned.

As the reader will see in what follows, even if, mainly in Chapters 2 and 3, several modification strategies are used and analyzed, the strategy that comes back for all the problems covered is the one consisting of moving absent vertices out of the *a priori* solution (it is denoted by MS for the rest of the book). Such a strategy is very quick, simple and intuitive but it does not always produce feasible solutions for any of the possible subgraphs (i.e., it is not always feasible). For instance, if it is feasible for PROBABILISTIC MAX INDEPENDENT SET, PROBABILISTIC MIN VERTEX COVER and PROBABILISTIC MIN COLORING, this is not the case for PROBABILISTIC LONGEST PATH, unless particular structure is assumed for the input graph. So, in Part II, we restrict ourselves to this particular strategy and assume that either MS is feasible, or, in case of unfeasibility, very little additional work is required in order to achieve feasible solutions. Then, for large classes of problems (e.g., problems where feasible solutions are subsets of the initial vertex-set or edge-set satisfying particular properties, such as stability, etc.), we investigate relations between these problems and their probabilistic counterparts (under MS). Such relations very frequently derive answers to the above mentioned five types of problems. Chapter 7 goes along the same lines as Chapter 6. We present a small compendium of probabilistic graph-problems (under MS). More precisely we revisit the most well-known and well-studied graph-problems and we investigate if strategy MS is feasible for any of them. For the problems for which this statement holds, we express the functional associated with it and, when

possible, we characterize the optimal *a priori* solution and the complexity of its computation.

The book should be considered to be a monograph as in general it presents the work of its authors on probabilistic combinatorial optimization graph-problems. Nevertheless, we think that when the interested readers finish reading, they will be perfectly aware of the principles and the main issues of the whole subject area. Moreover, the book aims at being a self-contained work, requiring only some mathematical maturity and some knowledge about complexity and approximation theoretic notions. For help, some appendices have been added, dealing, on the one hand, with some mathematical preliminaries: on sets, relations and functions, on basic concepts from graph-theory and on some elements from discrete probabilities and, on the other hand, with elements of the complexity and the polynomial approximation theory: notorious complexity classes, reductions and **NP**-completeness and basics about the polynomial approximation of **NP**-hard problems. We hope that with all that, the reader will be able to read the book without much preliminary effort. Let us finally note that, for simplifying reading of the book, technical proofs are placed at the end of each chapter.

As we have mentioned in the beginning of this preface, we have worked in this domain since 1994. During all these years many colleagues have read, commented, improved and contributed to the topics of the book. In particular, we wish to thank Bruno Escoffier, Federico Della Croce and Christophe Picouleau for having working with, and encouraged us to write this book. The second author warmly thanks Elias Koutsoupias and Vassilis Zissimopoulos for frequent invitations to the University of Athens, allowing full-time work on the book, and for very fruitful discussions. Many thanks to Stratos Paschos for valuable help on LaTeX.

Tender and grateful thanks to our families for generous and plentiful support and encouragement during the task.

Finally, it is always a pleasure to work with Chantal and Sami Menasce, Jon Lloyd and their colleagues at ISTE.

Cécile Murat and Vangelis Th. Paschos

Athens and Paris, October 2005

Chapter 1

A Short Insight into Probabilistic Combinatorial Optimization

1.1. Motivations and applications

The most common way in which probabilities are associated with combinatorial optimization problems is to consider that the data of the problem are deterministic (always present) and randomness carries over the relation between these data (for example, randomness on the existence of an edge linking two vertices in the framework of a random graph theory problem ([BOL 85]) or randomness on the fact that an element is included to a set or not, when dealing with optimization problems on set-systems or, even, randomness on the execution time of a task in scheduling problems). Then, in order to solve an optimization problem, algorithms (probabilistic or, more frequently, deterministic) are devised, and the mathematical expectation of the obtained solution is measured. A main characteristic of this approach is that probabilities do not intervene in the mathematical formulation of the problems but only in the mathematical analysis performed in order to get results.

More recently, in the late 1980s, another approach to the randomness of combinatorial optimization problems was developed: probabilities are associated with the data

describing an optimization problem (for a particular datum, we can see the probability associated with it as a measure of how much this datum is likely to be present in the instance to be finally optimized) and, in this sense, probabilistic elements are explicitly included in the formulations of these problems. Such formulations give rise to what we will call *probabilistic combinatorial optimization problems*. Here, the objective function is a form of carefully defined mathematical expectation over all possible instances of size less than, or equal to, a given initial size.

The fact that, when dealing with probabilistic combinatorial problems, randomness lies in the presence of the data means that the underlying models are very suitable for the modelling of natural problems, where randomness is the quantification of uncertainty, or fuzzy information, or inability to forecast phenomena, etc.

For instance, in several versions of satellite shot planning problems, the uncertainty concerning meteorological conditions can be quantified by a system of probabilities. The optimization problems derived are, as we will see later in this chapter, clearly of probabilistic nature. If, on the other hand, during a salesman's tour, some clients need not to be visited, he should omit them from his tour and if the fact that a client has to be visited or not is modelled in terms of probabilities-systems, then a probabilistic traveling salesman problem arises[1]. For similar or other reasons, starting from a transportation, or computer, or any other kind of network, we encounter problems like probabilistic shortest path problem[2] or probabilistic longest path problem[3], probabilistic minimum spanning tree problem[4], etc. Also, in industrial automation, the systems for foreseeing workshops' production give rise to probabilistic scheduling, or probabilistic set covering or probabilistic set packing, etc. Finally, in computer science, mainly when

dealing with parallelism or distributed computation, probabilistic combinatorial optimization problems very frequently have to be solved. For instance, modeling load-balancing with non-uniform processors and failures possibility becomes a probabilistic graph partitioning problem; also in network reliability theory, many probabilistic routing problems are met ([BER 90b]).

In all, models of probabilistic nature are very suitable and appropriate real-life problems where randomness is a constant source of concern and, on the other hand, the study of the problems derived from these models are very attractive as mathematical abstraction of real systems. Another reason motivating work on probabilistic combinatorial optimization is the study and the analysis of the stability of the optimal solutions of deterministic combinatorial optimization problems when the considered instances are perturbed. For problems defined on graphs, more particularly, these perturbations are simulated by the occurrence, or the absence, of subsets of vertices (see, for example, [PEE 99] where probabilistic combinatorial optimization approaches and concepts are used to yield robust solutions for an on-line traffic-assignment problem).

Informally, given a combinatorial optimization graph[5]-problem Π, defined on a graph $G(V, E)$, an instance of its probabilistic counterpart, denoted by PΠ, is built by associating a probability p_i with any vertex v_i in V. This probability is considered as the presence-probability of v_i in the subgraph of G on which Π will be finally solved. Problem PΠ expresses the fact that Π will, eventually, have to be solved not on the whole G, but rather on some of its subgraphs that will be specified very shortly before the solution in this subgraph is required.

In order to illustrate the issue outlined above, we will consider in what follows four examples of models that give rise to probabilistic combinatorial optimization problems.

EXAMPLE 1.1.- *Probabilistic traveling salesman*. A repair company has to perform a minimum-length daily tour visiting n potential clients. This is the classical (deterministic) traveling salesman problem, denoted by MIN TSP in what follows. It is formally defined as follows: given a set C of n cities and distances $d(c_i, c_j) \in \mathbb{Q}$, for any pair $(c_i, c_j) \in C \times C$, $i \neq j$, MIN TSP consists of determining a tour of C, i.e., a permutation σ: {1,...,n} → {1,...,n}, minimizing the length of the tour, i.e., the quantity $d(c_{\sigma(n)}, c_{\sigma(1)}) + \sum_{i=1}^{n-1} d(c_{\sigma(i)}, c_{\sigma(i+1)})$. MIN TSP is commonly modeled in terms of a complete graph, denoted by K_n (see section A.2 of Appendix A) on n vertices (representing the cities). Edge (v_i, v_j) is weighted by $d(c_i, c_j)$ and an optimal solution is a Hamiltonian cycle (section A.2 of Appendix A) of minimum total length (or distance), the length of a cycle being the sum of the distances on its edges. But, if we assume that any client will not need to be repaired every day, then this implies that, a given date, only a subset of clients need to be effectively visited; this subset changes from day to day. What can be done is that a client i can be assigned, for a random day, with a repairing-probability p_i; this probability is independent from the probabilities dealing with the other clients. We thus get a version of the probabilistic traveling salesman problem (initially introduced and studied in [JAI 85, JAI 88a]).

EXAMPLE 1.2.- *Probabilistic coloring*. Consider for a given University-fall a list of potential classes that students can follow: any student has to choose a sublist of such classes. For any of them, one knows the title, the teaching professor and the time slot assigned to it, each such slot being proposed by the professor in charge. A class will only take place if the number of students having chosen it is above a given threshold. So, nobody knows *a priori* if a particular class will take place before the closing of students'

registrations (we can reasonably assume that the choice of any student is dependent on the contents of the course and of the teacher). On the other hand, one can, for example, by looking at statistics on the behavior of the students during past years, assign probabilities on the fact that a particular class will really take place, the mandatory courses being assigned with probability 1. The problem for the University planning services is how many rooms need to be assigned to the set of courses. This problem is typically an instance of probabilistic coloring if one considers courses as vertices and if one links two such vertices if the corresponding classes cannot take place in the same room (because they are planned with the same professor, or are assigned with overlapping time slots). This type of graph is known by the term incompatibility graph. Here, an independent set, i.e., a potential color, corresponds to a set of “compatible classes”, i.e., to classes that can be assigned with the same room. The number of colors used in such a graph represents the total number of rooms assigned to the set of classes considered. The probabilities resulting from the statistical analysis on the former students’ behavior are the presence probabilities for the vertices (i.e., the probabilities that the corresponding classes will really take place).

EXAMPLE 1.3.- *Probabilistic independent set*. Consider a planning aiding process for realizing satellite shots. One associates a vertex with any shot requested and one links two vertices if they correspond to shots that cannot be realized on the same orbit. But a shot realized under, for example, strong cloud cover cannot be used for the purposes for which it has been requested. Using meteorological forecasting, one can assign to any shot requested a probability that it will be usable. This problem has been modelled in [GAB 97] (see also [GAB 94]) as a maximum independent set and if one takes into account

probabilities associated with meteorological forecasting, then one has to solve a probabilistic version of it.

EXAMPLE 1.4.- *Probabilistic longest path*. For the satellite shot planning problem dealt in Example 1.3, one can use another graph-representation (see [GAB 97] for details) where an arc models the possibility of successive realization of its two endpoints. Then, the satellite shot planning can be represented as a particular longest path problem. Integration of probabilities associated, for instance, with meteorological forecasting to this model gives rise to a probabilistic longest path.

1.2. A formalism for probabilistic combinatorial optimization

We have already mentioned that the probabilistic version of an optimization problem models the fact that, given an instance of the problem, only a subinstance of it will eventually be solved. Since we do not know which is this subinstance, the most natural approach that comes in mind is to optimally solve any particular subinstance of the problem at hand (following the probabilities on its vertices, any such subinstance is more or less likely to be the one where optimization has to be effectively performed). Such an approach, called reoptimization in [BER 90b, BER 93, JAI 93], can be very much time- and space-consuming, in particular when the initial problem is **NP**-hard. Indeed, given a graph $G(V, E)$ of order n (i.e., $|V| = n$), there exist 2^n subsets of V and consequently 2^n subgraphs, induced by these subsets, any of them candidate to be the instance effectively under consideration. For an **NP**-hard problem (this remains true even for a polynomial problem), the

amount of time needed to solve any of these instances to the optimum can be huge so that reoptimization becomes practically unrealistic.

This is the main reason for which another, more realistic, approach is used and this will be dealt in this book. It is called an *a priori optimization* and has been introduced in [JAI 85, BER 88]. Informally, instead of reoptimizing any subinstance, the underlying idea of an *a priori* optimization consists of determining a solution of the whole (initial) instance, i.e., the one where all data are present, called an *a priori solution*, and to apply a strategy, called a *modification strategy*, making it possible to adapt as quickly as possible the *a priori* solution to the subinstance that must effectively be solved. The choice of this strategy depends strongly on the application modelled by the problem.

Consider a graph $G(V, E)$ instance of a combinatorial optimization problem Π, a feasible solution S, for Π in G, a subset V' of V and the subgraph $G[V']$ of G induced by V'. A modification strategy M is an algorithm transforming S in order to get a feasible Π-solution for any such $G[V']$. Obviously, it is assumed that if M is applied on G (i.e., if $V' = V$), then S remains unchanged. Also, one can suppose that application of M in the final instance is possible, in the sense that there exists sufficient time for its achievement.

EXAMPLE 1.1. (CONTINUED) — Revisit the repairing company dealt in Example 1.1. Assume that for several material reasons, its staff do not wish to reoptimize the daily tours for its vehicles. A possible way to plan these tours is the following. One computes firstly a feasible tour T including the whole set of the clients is computed; this is the *a priori* solution mentioned just above. A possible modification strategy in order to compute the effective tour for a given day is to drop absent clients (i.e., clients that do not ask for intervention during this day); then, it suffices to

visit the present ones following the order induced by the *a priori* tour *T*.

In practice, the *a priori* approach corresponds to a behavior observed for the real problem; the modification strategy algorithmically models this behavior. The choice of a modification strategy depends strongly on the real-world application modelled. In order to illustrate this, consider the following example inspired from a vehicle routing problem studied in [BER 90b, BER 92, BER 96].

EXAMPLE 1.5.- The problem studied in [BER 90b, BER 92, BER 96] consists of determining a shortest distance tour through *n* clients under several constraints, for example, limits on vehicle capacities together with assumptions that any of the vehicles have to retrieve different quantities of different objects from any client, etc. If any on day d only a subset of the clients has to be visited, then for modifying an *a priori* tour to fit these present clients, two modification strategies could be used depending on when clients' demands become available:

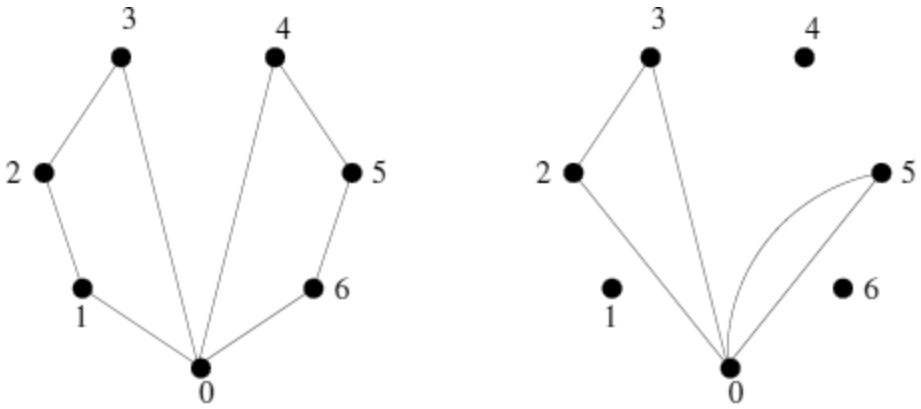
– In the first strategy, denoted by `M1`, a vehicle, following the *a priori* tour, visits all the clients but it only serves the ones having asked for service on day *d*. When the vehicle is saturated, i.e., its capacity is attained and it returns to the depot before continuing with the next client.

– The second strategy, denoted by `M2`, differs from `M1` by the fact that the vehicle only visits (following the *a priori* tour) clients having asked for services on day *d* (returning to the depot when saturated and then continuing with the next client).

In order to illustrate differences between the two strategies, consider an *a priori* tour (0, 1, 2, 3, 4, 5, 6, 0) and assume that depot is vertex 0 and that vehicle has capacity 30. At day *d*, the clients 1, 4 and 6 need not to be visited and that the demands for clients 2, 3 and 5 are 20, 10 and 20,

respectively. The results for the two strategies above are shown in Figure 1.1.

Figure 1.1. *Application of modification strategies* M1 *and* M2 *for the probabilistic vehicle routing problem with capacity constraints*



As one can see, under strategy M1 (Figure 1.1(a)), the route realized by the vehicle will be (0, 1, 2,3,0), then (0,4, 5,6,0), while the route under M2 (Figure 1.1(b)) will be (0, 2, 3, 0), then (0,5,0).

There exists an important difference between these two strategies:

– M1 models situations where demand of a particular client becomes clear (or known) only once it has been visited;

– M2 corresponds to situations where clients' demands are known in advance, i.e., before the vehicle starts the route.

A basic operational and computational feature of the *a priori* optimization approach is that the optimization problem considered has to be solved only once; next, the only "tool" needed is a quick modification strategy which is able to adapt the *a priori* solution to the subinstance to be effectively optimized. In this way, computational time is not really a serious problem.

The question now is: "what is the measure of an *a priori* solution?". Let S be a feasible solution for Π on $G(V, E)$, M be a modification strategy for S and V be a subset of V. Denote

by $S(V, \mathrm{M})$ the solution for Π in $G[V]$, obtained from S by applying M and by $m(G[V], S(V, \mathrm{M}))$ its value. A reasonable requirement for $S(V, \mathrm{M})$ is that $m(G[V], S(V,\mathrm{M}))$ is as close as possible to the value of an optimal solution for Π in $G[V]$, denoted by $\mathrm{opt}(G[V])$. Since, on the other hand, we do not know *a priori*, which will be the subinstance to be solved, we will use as evaluation-measure for S its expectation. Denote by $\Pr[V]$ the probability of presence of the vertices of V, hence the probability of $G[V]$ and set $\Pr[v_i] = p_i$, the presence-probability of $v_i \in V$; then:

$$\Pr[V'] = \prod_{v_i \in V'} p_i \prod_{v_j \notin V'} (1 - p_j) \quad [1.1]$$

In particular, when $p_i = p$, for any $v_i \in V$, then [1.1] becomes:

$$\Pr[V'] = p^{|V'|} (1-p)^{|V|-|V'|}$$

The measure (i.e., the objective function) of S for PΠ, also called *functional* in what follows, is defined as:

$$m(G, S, \mathrm{M}) = E(G, S, \mathrm{M}) = \sum_{V' \subseteq V} m(G[V'], S(V', \mathrm{M})) \Pr[V'] \quad [1.2]$$

where $\Pr[V]$ is defined by [1.1].

In standard complexity-theoretic language, the problems studied in this book belong to the class **NPO**. Informally, an **NPO** problem is an optimization problem, the decision versions of which is in **NP** (see also Appendix B). More formally now, an **NPO** problem can be defined as follows.

DEFINITION 1.1.- *A problem* Π *in* ***NPO*** *is a quadruple* $(\mathcal{I}_\Pi, Sol_\Pi, m_\Pi, \mathrm{goal}(\Pi))$ *where:*

– $\mathcal{I}_\Pi$ *is the set of instances of* Π *(and can be recognized in polynomial time);*

– *given* $I \in \mathcal{I}_\Pi$, $Sol_\Pi(I)$ *is the set of feasible solutions of* I*; the size of a feasible solution of* I *is polynomial in the size* $|I|$ *of the instance; moreover, one can determine in polynomial time if a solution is feasible or not;*

– *given* $I \in \mathcal{I}_\Pi$ *and* $S \in Sol_\Pi(I)$, $m_\Pi(I, S)$ *denotes the value of the solution S of the instance I;* m_Π *is called the objective function, and is computable in polynomial time;*

– goal(Π) ∈ {min, max}.

We can now give a formal definition for probabilistic combinatorial optimization problems (under the *a priori* optimization assumption), derived from Definition 1.1.

DEFINITION 1.2.- *Let* Π = ($\mathcal{I}_\Pi$, Sol_Π, m_Π, goal(Π)) *be an* ***NPO*** *problem as in Definition 1.1. The probabilistic version of* Π, *denoted by* PΠ, *is a six-tuple* (($\mathcal{I}_\Pi$, **Pr**), Sol_Π, goal(Π), M, E_Π), *where:*

– $\mathcal{I}_\Pi$ *is as in Definition 1.1 and* **Pr** *is the set of all the vectors* Pr *of the presence-probabilities of the data representing* $I \in \mathcal{I}$; *the pair* ($\mathcal{I}_\Pi$, **Pr**) *is the instance-set of* PΠ *and the couple I,* Pr[*I*], $I \in \mathcal{I}$, Pr ∈ **Pr** *is an instance of* PΠ; Sol_Π *and* goal(Π) *are as in the corresponding items of Definition 1.1;*

– *M is an algorithm, called modification strategy, such that, given an instance* (*I*, Pr[*I*]) *of* Π, *a solution* $S \in Sol(I$, Pr[*I*]) *and any subinstance I′ of I, it modifies S in order to produce a feasible solution S*(*I′*, M);

– E_Π *is the functional of S and is defined (analogously to [1.2]) as:*

$$E_{\Pi\Pi}(I, S, \mathtt{M}) = \sum_{I' \subseteq I} m\left(I', S\left(I', \mathtt{M}\right)\right) \Pr\left[I'\right] \quad [1.3]$$

where Pr[*I′*] *is defined (analogously to [1.1]) as:*

$$\Pr\left[I'\right] = \prod_{d_i \in I} \Pr\left[d_i\right] \prod_{d_j \notin I} \left(1 - \Pr\left[d_j\right]\right)$$

where d_i, d_j *draw data of I and* Pr[d_i] *and* Pr[d_j] *their presence probabilities respectively.*

One can see that Definition 1.2 implies that modification strategy M is part of the definition of the problem. In this sense, two distinct strategies `M1` and `M2`, associated with the

same **NPO** problem Π, give rise to two distinct probabilistic problems $P\Pi_1$ and $P\Pi_2$, respectively, since changing a modification strategy changes the functional. In other words, distinct modification strategies lead to distinct objective functions.

The modification strategy used most frequently until now is the one consisting of dropping absent data out of the *a priori* solution and of taking the remaining elements of it as a solution for the effective instance. This simple strategy, denoted by MS for the rest of this chapter, is feasible for numerous problems (this is the case of all the problems dealt in this monograph and for the ones dealt in [AVE 94, AVE 95, BEL 93, BER 88, BER 89, BER 90b, JAI 85, JAI 88a, JAI 88b, JAI 92, SÉG 93]) but not for any problem. Let us take for example the case of the probabilistic minimum independent dominating set (also called the minimum maximal independent set). Here, given an *a priori* maximal independent set S, dropping the absent vertices out from S does not necessarily result in a maximal independent set for the present subgraph.

As we will see in the next chapters, in particular under strategy MS and in the cases where the optimum *a priori* solution has a closed combinatorial characterization, the derived probabilistic problems can be equivalently stated as "deterministic combinatorial optimization problems" under particular and sometimes rather non-standard objective functions.

Let us note also that *a priori* optimization under strategy MS corresponds to the following robustness model for combinatorial optimization. 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*) subinstance $I' \subset I$. Suppose that any datum d_j in the data-set describing I has a probability p_j, indicating how d_j is likely to be present in the final

subinstance I'. Consider finally that once instance I' is specified, the solver has no opportunity to solve directly instance I'. In this case, there certainly exist many ways to proceed. Here we deal with a simple and natural way where one computes an initial solution S for Π in the entire instance I and, once I' becomes known, one removes from S those elements of S that do not belong to I' (providing that this deletion results in a feasible solution for I') thus giving a solution S' fitting I'. The objective is to determine an initial solution S for I such that, for any subinstance $I' \subseteq I$ presented for optimization, the solution S' respects some predefined quality criterion (for example, optimal for I', or achieving, say, constant approximation ratio, etc.).

Let us note that a measure analogous to the ones of [1.2] or, more generally, of [1.3] can be obtained also for the reoptimization approach. Consider a probabilistic combinatorial optimization graph-problem PΠ, derived from an optimization graph-problem Π and let $G(V, E)$ be a generic instance for the latter problem. Set $n = |V|$ and consider a vector $(p_1,\ldots, p_n)$ of presence-probabilities on the vertices of V. Then the functional $E^*(G)$ of the reoptimization for PΠ is defined as:

$$E^*(G) = \sum_{V' \subseteq V} m\left(G[V'], S^*(V')\right) \Pr[V'] \quad [1.4]$$

where $S^*(V)$ is an optimal solution for Π in $C[V]$, and $\Pr[V]$ is as in [1.1].

1.3. The main methodological issues dealing with probabilistic combinatorial optimization

1.3.1. *Complexity issues*

1.3.1.1. *Membership in NPO is not always obvious*

As one can see from [1.3] computation of functional's value is not *a priori* polynomial, since this expectation carries over all the possible subsets of the initial data-set. So, with respect to Definition 1.1, probabilistic versions of **NPO** problems do not trivially belong to **NPO** too. As we will see in the next chapters, when dealing with strategy MS sketched at the end of section 1.2, we succeed by more or less simple algebraic manipulations to show that functionals associated with it can be polynomially computed. This is the case for the problems dealt with in the next chapters as well as for the problems studied in [AVE 95, AVE 94, BEL 93, BER 88, BER 89, BER 90a, BER 90b, JAI 85, JAI 88a, JAI 92, JAI 88b, SÉG 93]. The basic idea underlying such a simplification is the following: instead of computing the value of the solution induced by any subinstance (recall that there exist an exponential number of subinstances of a given initial instance), one tries to determine, for any element of the *a priori* solution, the number of subinstances for which this element remains part of this solution. Even if this simplification technique works for numerous problems (associated with strategy MS), we will see in Chapters 2 and 3 that it quickly attains its limits once one tries to enrich MS with elementary operations improving its result. In particular, we will see that matching MS with natural greedy improvement techniques largely complicates the corresponding functionals in such a way that it is not obvious that their computation can be performed in polynomial time.

1.3.1.2. *Complexity of deterministic vs. complexity of probabilistic optimization problems*

Obviously, for any probabilistic combinatorial optimization graph-problem PΠ defined on a graph $G(V, E)$, if $p_i = 1$, for

any $v_i \in V$, then PΠ coincides with Π in the sense that for any *a priori* solution S for PΠ, its functional has the same value as the objective value of S seen as solution of Π. This remark implies that if the functional[6] is computable in polynomial time and if Π is **NP**-hard, then PΠ, being a generalization of Π, is also **NP**-hard. Conversely, if Π is polynomial, then no immediate result can be deduced for PΠ.

Consider for instance two classical polynomial problems, the shortest path problem for fixed departure- and arrival-vertices s and t, respectively, and the minimum spanning tree problem. A probabilistic version for the former one is defined and studied in [JAI 92]. There, the input graph is complete, its vertices are independent and uniformly distributed in the unit square, some vertices are always present (i.e., they have probability 1), in particular s and t, and the rest of the vertices all have the same presence probability. Given an *a priori* solution, the adopted modification strategy consists of removing the absent vertices from this solution (this is not a problem since the input graph is assumed complete). As it is proved in [JAI 92], this version of probabilistic shortest path problem is **NP**-hard. The same holds for the minimum spanning tree problem ([BER 88, BER 90a]). For this problem, the input is the same as for shortest path. The modification strategy considered is the following: given an *a priori* tree T and a subgraph $G[V']$ of the input-graph, we consider the subtree of T restricted to the vertices of V' together with some vertices of $V \setminus V'$ (and the edges of T incident to these vertices) in order to guarantee connectivity of the induced subtree. This probabilistic version of minimum spanning tree is **NP**-hard.

When the deterministic version Π of a probabilistic problem PΠ is **NP**-hard, an interesting mathematical problem is to determine the complexity of PΠ for the classes

of instances where Π is polynomial. Here also, results for the probabilistic problem are, very frequently, opposite to the ones for its (deterministic) support. For instance, as we will see in Chapter 5, the probabilistic versions of many coloring problems studied there are **NP**-hard, even for graph-classes for which deterministic coloring is polynomial.

Another interesting fact that will be clear in the next chapters (mainly in Chapters 4 and 5) is the role that the specific probability-system considered plays in complexity or approximation behaviors of the problems dealt. For instance, the fact that one assumes identical or distinct vertex-probabilities can completely change the complexity of a problem or its approximability.

Notice that an analogous fact can be established for the probabilistic traveling salesman, even when the input-graph K_n (the complete graph on n vertices) has identical vertex-probabilities. Denote by T^* an optimal *a priori* tour in K_n (i.e., an optimal solution of probabilistic traveling salesman under MS) and by T_0^* an optimal tour for the deterministic counterpart. In [JAI 85], counter-examples are given showing that $T^* \neq T_0^*$. In [BEL 93], it is shown that if n is odd ($n = 2k + 1$), then:

$$E(K_n, T^*, \mathrm{MS}) \geqslant p^2 m(K_n, T^*) \frac{1-(1-p)^{n-1}}{1-(1-p)^k} \quad [1.5]$$

From [1.5], one can deduce the following two estimations:

$$\frac{m(K_n, T^*) - m(K_n, T_0^*)}{m(K_n, T_0^*)} \leqslant \frac{1-p^2}{p^2} \quad [1.6]$$

$$\frac{E(K_n, T_0^*, \mathrm{MS}) - E(K_n, T^*, \mathrm{MS})}{E(K_n, T^*, \mathrm{MS})} \leqslant \frac{1-p^2}{p^2} \quad [1.7]$$

The bounds given in [1.6] and [1.7] show that T_0^* constitutes a good approximation for T^* only in the case where p is large, i.e., when the probabilistic version becomes “close” to the deterministic one.