I²SDS
The Institute for Integrating Statistics in Decision Sciences

Technical Report TR-2010-11
May 15, 2010

Pattern-Based Modeling and Solution of Probabilistically Constrained Optimization Problems

Miguel A. Lejeune
Department of Decision Sciences
The George Washington University

Pattern-Based Modeling and Solution of Probabilistically Constrained Optimization Problems

MIGUEL A. LEJEUNE*

George Washington University, Washington, DC, USA; mlejeune@gwu.edu

Abstract

We propose a new modeling and solution method for probabilistically constrained optimization problems. The methodology is based on the integration of the stochastic programming and combinatorial pattern recognition fields. It permits the very fast solution of stochastic optimization problems in which the random variables are represented by an extremely large number of scenarios. The method involves the binarization of the probability distribution, and the generation of a consistent partially defined Boolean function (pdBf) representing the combination (F, p) of the binarized probability distribution F and the enforced probability level p. We show that the pdBf representing (F, p) can be compactly extended as a disjunctive normal form (DNF). The DNF is a collection of combinatorial p-patterns, each of which defining sufficient conditions for a probabilistic constraint to hold. We propose two linear programming formulations for the generation of p-patterns which can be subsequently used to derive a linear programming inner approximation of the original stochastic problem. A formulation allowing for the concurrent generation of a p-pattern and the solution of the deterministic equivalent of the stochastic problem is also proposed. Results show that large-scale stochastic problems, in which up to 50,000 scenarios are used to describe the stochastic variables, can be consistently solved to optimality within a few seconds.

Subject classifications: Stochastic Programming, Combinatorial Pattern, Probabilistic Constraint, Boolean Function

1 Problem Formulation, Literature Review, and Contributions

In this paper, we propose a new modeling and numerical solution framework for stochastic programming problems [4, 42, 47]. The methodology is based on *pattern recognition* [19, 55] and, in particular, on the derivation of *logical* and *combinatorial* patterns [20, 36, 45, 52, 53]. The proposed framework allows for the deterministic reformulation and solution of probabilistically constrained programming problems of the form:

$$\begin{aligned} &\min c^T x \\ &\text{subject to } Ax \geq b \\ &\qquad \mathcal{P}(T_j x \geq \xi_j,\ j \in J) \geq p \\ &\qquad x \geq 0 \end{aligned} \quad . \tag{1}$$

*The author is supported by Grant # W911NF-09-1-0497 from the Army Research Office.

The notation $|J|$ refers to the cardinality of the set J, ξ is a $|J|$-dimensional random vector which has a multivariate probability distribution with finite support, x is the m-dimensional vector of decision variables, $c \in \mathcal{R}^m, b \in \mathcal{R}^d$, $A \in \mathcal{R}^{d\times m}$ and $T \in \mathcal{R}^{|J|\times m}$ are deterministic parameters, p is a prescribed probability or reliability level, and the symbol $\mathcal{P}$ refers to a probability measure. We consider the most general and challenging case in which there is no independence restriction between the components ξ_j of ξ. Thus,

$$\mathcal{P}(T_j x \geq \xi_j,\ j \in J) \geq p \tag{2}$$

is a *joint* probabilistic constraint which enforces that the combined fulfillment of a system of $|J|$ linear inequalities $\sum_{k=1}^{m} T_{jk}x_k \geq \xi_j$ must hold with a $|J|$-variate joint probability. Stochastic programming problems of this form are non-convex and very complex to solve. The example that follows is used throughout the manuscript to illustrate different aspects of the proposed approach.

Example 1 *Consider the probabilistically constrained problem*

$$\begin{aligned} &\min\ x_1 + 2x_2 \\ &\textit{subject to}\ \mathcal{P}\left\{ \begin{array}{l} 8 - x_1 - 2x_2 \geq \xi_1 \\ 8x_1 + 6x_2 \geq \xi_2 \end{array} \right\} \geq 0.7 \\ &x_1, x_2 \geq 0 \end{aligned} \tag{3}$$

where the random vector $\xi = [\xi_1, \xi_2]$ accepts ten equally likely ($p_k = 0.1, k = 1, \ldots, 10$) realizations k represented by $\omega^k = [\omega_1^k, \omega_2^k]$ and has the following two-variate probability distribution:

Table 1: Probability Distribution

k	ω_1^k	ω_2^k	$F(\omega^k)$
1	6	3	0.2
2	2	3	0.1
3	1	4	0.1
4	4	5	0.3
5	3	6	0.3
6	4	8	0.5
7	6	8	0.7
8	1	9	0.2
9	4	9	0.7
10	5	10	0.8

The feasibility set defined by the probabilistic constraint is non-convex. It is the union of the polyhedra $\{(x_1, x_2) \in \mathcal{R}^2_+ : 8 - x_1 - 2x_2 \geq 6,\ 8x_1 + 6x_2 \geq 8\}$*, and* $\{(x_1, x_2) \in \mathcal{R}^2_+ : 8 - x_1 - 2x_2 \geq 4,\ 8x_1 + 6x_2 \geq 9\}$.

1.1 Literature Review

Programming under probabilistic constraints has been extensively studied (see [43] for a review) and has been applied for many different purposes ranging from the replenishment process in military operations [30], the enforcement of cycle service levels in a multi-stage supply chain [32, 33], the construction of pension funds [25], the monitoring of pollution level [18], etc. *Probabilistic constraints with random*

right-hand side have a deterministic technology matrix T in (1), while the stochastic component is in the right-hand side of the inequality $Tx \geq \xi$ subject to the probabilistic requirement. Stochastic optimization problems with individual [9] probabilistic constraints (i.e., ξ is a one-dimensional vector) have a deterministic equivalent, whose continuous relaxation is straightforward to derive using the quantile of the one-dimensional random variable. However, the modeling of the reliability of a system through a set of individual probabilistic constraints does not allow the attainment of a system-wide reliability level [42], but instead enforces a certain reliability level for each individual part of the system. To that end, *joint probabilistic constraints*, first analyzed in [37] under the assumption of independence between each component of the random vector ξ, are needed. Prékopa [39] considered the most general setting by removing the independence assumption between the components of the system.

A key factor for the computational tractability of stochastic problems with joint probabilistic constraints concerns the convexity properties of the feasible set. Prékopa [40] showed that, if the functions $T_j x - \xi_j \geq 0$ are concave in x and ξ, and ξ is continuously distributed with logarithmically concave probability density function, then the set of vectors x satisfying the joint probabilistic constraint is convex, allowing therefore to resort to a solution method based on convex programming techniques. However, such convexity properties do not apply when the random variables are discretely distributed. The corresponding optimization programming problems are well-known to be non-convex and NP-hard, and have been receiving particular attention lately [10, 12, 31, 32, 33, 34, 35, 46, 48].

Three main families of solution approaches can be found in the literature for the above probabilistically constrained optimization problems. The first one relies on the concept of p-efficiency [41] which requires the a priori uncovering of the finite set of p-efficient points. This allows the transformation of the stochastic problem into a disjunctive one which can be solved through a convexification process and the cone generation algorithm [12] or the use of a specialized column generation algorithm [33].

The second family of solution methods associates a binary variable with each possible realization of the random vector and, then, substitutes a mixed-integer programming (MIP) problem of very large dimensionality (i.e., one binary variable per possible realization) for the original stochastic one [10]. To solve the resulting MIP problem, which contains a cover and "big-M" constraints, Ruszczyński [46] developed specialized cutting planes that he embedded in a branch-and-cut algorithm. Cheon et al. [10] proposed a branch-reduce-cut algorithm that iteratively partitions the feasible region and uses bounds to fathom inferior partitions. Luedtke et al. [35] proposed stronger MIP formulations for which they generate a family of valid inequalities, which are subsumed by the facet-defining family of cuts derived in [31] . In a set of very recent related studies, a sample approximation problem [34] is used to generate feasible solutions and optimality bounds for problems with joint probabilistic constraints. It was also shown that MIP reformulations [48, 49] of the probabilistic set covering problem can be solved in a very computationally efficient way [48].

The third type of approaches consists in deriving safe, conservative approximations [8, 38] that take the form of convex optimization problems whose optimal solution is not always very close to the true optimal solution. In fact, the probability level $\hat{p}$ enforced by these techniques can be much larger than

the prescribed one p. If the decision-maker is willing to trade some safety level for lower costs, and sets accordingly the reliability level p to moderately high values (say $p = 0.95$ or 0.9), then the robust approximation might not always be suitable [34].

1.2 Motivation and Contributions

The fundamental contribution of this paper resides in the development of a novel solution methodology for stochastic programming problems. To the best of our knowledge, this is the first time that techniques from the pattern recognition field [15, 17, 19, 55] are employed for the optimization of probabilistically constrained optimization problems. Pattern recognition has been primarily used for feature selection, unsupervised classification, clustering, data mining or image processing purposes. The expected outcomes of pattern-based methods differ depending on whether they are used for classification or for optimization. With classification objectives in mind, logical / combinatorial pattern methods [5, 14, 20, 45, 52, 53], are primarily used to derive "rules" that separate data points belonging to different categories. In the stochastic optimization context focused upon in this paper, the extracted patterns provide a compact representation of sets of conditions that are sufficient for the satisfaction of a probabilistic constraint, and can be used to derive deterministic reformulations of the stochastic problem. Besides its novelty, a crucial factor of the proposed framework is that it allows the very fast exact solution of stochastic optimization problems in which the random variables are represented by an extremely large number of scenarios. We describe below the main elements of the proposed methodology and discuss the organization of the paper.

In Section 2, we introduce the concepts of *p-sufficient* and *p-insufficient* realization, define a *binarization* method for a probability distribution, propose a method for selecting *relevant* realizations, and represent the combination (F, p) of the binarized probability distribution F of the random variable ξ and the enforced probability level p as a *partially defined Boolean function* (pdBf). In Section 3, we extend the pdBf representing (F, p) as a *disjunctive normal form* (DNF), which is a collection of combinatorial p-patterns. Each of those defines sufficient conditions for the probabilistic constraint (2) to hold. Then, we propose a new *mathematical programming* method for the derivation of combinatorial patterns. Two integer programming and two linear programming formulations are presented. Besides its novelty, the interest of the proposed method is that it offers a remedy to an issue associated with enumerative methods, which are highly efficient for the generation of patterns of small degrees, but turn out to be of lower efficacy when large-degree patterns need to be extracted [6]. In Section 4, we show how we can use the combinatorial patterns to derive a linear programming inner approximation and a mixed-integer programming deterministic equivalent of the probabilistically constrained problem (1). Section 5 discusses the numerical implementation of the proposed methodology. Section 6 provides concluding remarks.

2 Representation of (F, p) as a Partially Defined Boolean Function

In this section, we shall first discuss the binarization process of the probability distribution and show how this allows the representation of the combination (F, p) of the probability distribution F and the prescribed probability level p as a partially defined Boolean function (pdBf). We shall then discuss the

required properties of the set of cut points used for the binarization process and define the set of relevant realizations considered for the pattern generation process.

2.1 Binarization of Probability Distributions

We develop an approach to *binarize* probability distributions with finite support. We denote by Ω the finite set of the possible *realizations* $k \in \Omega$ of the $|J|$-dimensional random vector ξ with cumulative distribution function F. Each realization k is represented by the $|J|$-dimensional deterministic vector: $\omega^k = \left[\omega_1^k, \ldots, \omega_{|J|}^k\right]$. We first introduce the concepts of *p-sufficient* and *p-insufficient* realizations.

Definition 1 *A realization k is called p-sufficient if and only if $\mathcal{P}(\xi \leq \omega^k) = F(\omega^k) \geq p$ and is p-insufficient if $F(\omega^k) < p$.*

The inequality sign in $\xi \leq \omega^k$ must be understood componentwise.

The introduction of the Boolean parameter I^k indicating whether k is p-sufficient or not

$$I^k = \begin{cases} 1 & \text{if } F(\omega^k) \geq p \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

generates a partition of the set Ω of realizations into two disjoint sets of p-sufficient Ω^+ ($I^k = 1$) and p-insufficient Ω^- ($I^k = 0$) realizations such that: $\Omega = \Omega^+ \bigcup \Omega^-$ with $\Omega^+ \bigcap \Omega^- = \emptyset$.

The *binarization* process of a probability distribution consists in the introduction of several binary attributes β_{ij} for each component ξ_j. The notation β_{ij} denotes the i^{th} binary attribute associated with component ξ_j. Each binary attribute β_{ij}^k takes value 1 (resp., 0) if the value ω_j^k taken by ξ_j in realization k is larger than or equal to (resp., strictly smaller than) a defined threshold value c_{ij}, called *cut point*:

$$\beta_{ij}^k = \begin{cases} 1 & \text{if } \omega_j^k \geq c_{ij} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

with

$$c_{i'j} < c_{ij} \;\Rightarrow\; \beta_{ij}^k \leq \beta_{i'j}^k \quad \text{for any} \quad i' < i,\ j \in J,\ k \in \Omega\,. \tag{6}$$

As a result of the binarization of the probability distribution, each numerical vector ω^k is mapped to an n-dimensional binary vector

$$\beta^k = \left[\beta_{11}^k, \ldots, \beta_{n_1 1}^k, \ldots, \beta_{ij}^k, \ldots, \beta_{n_j j}^k, \ldots\right] \tag{7}$$

which is a vertex of $\{0,1\}^n$, where $n = \sum_{j \in J} n_j$ is the sum of the number n_j of cut points associated with each component j.

As an illustration, we consider the set of cut points

$$C = \{c_{11} = 4;\ c_{21} = 5;\ c_{31} = 6;\ c_{12} = 8;\ c_{22} = 9;\ c_{32} = 10\} \tag{8}$$

to binarize the numerical components ω_1 and ω_2. The set (8) includes three cut points defined with respect to each component of the vector ξ. The central part of Table 2 displays the binarization of the probability distribution of ξ (see Example 1) with the set of cut points (8).

The set of cut points is used to generate a binary image β^k of each realization initially represented by the numerical vector ω^k. The association of the Boolean parameter $\mathcal{I}^k$ with the binary image β^k of the realization defines the binary projection $\Omega_B = \Omega_B^+ \bigcup \Omega_B^-$ of Ω, where Ω_B^+ and Ω_B^- respectively denote the sets of binarized p-sufficient and p-insufficient realizations. It permits the representation of the *combination* (F, p) of a probability distribution F and a probability level p as a pdBf $g(\Omega_B^+, \Omega_B^-)$ that is defined by the pair of sets (Ω_B^+, Ω_B^-) such that $\Omega_B^+, \Omega_B^- \subseteq \{0,1\}^n$. The right part of Table 2 displays the truth table of the pdBf obtained with the set of cut points (8).

Table 2: Realizations, Binary Images, and Truth Table of Partially Defined Boolean Function

	Numerical Representations		Truth Table of Partially Defined Boolean Function							
			Binarized Images						Indicator	
k	ω_1^k	ω_2^k	β_{11}^k	β_{21}^k	β_{31}^k	β_{12}^k	β_{22}^k	β_{32}^k	$\mathcal{I}^k$	
1	6	3	1	1	1	0	0	0	0	
2	2	3	0	0	0	0	0	0	0	
3	1	4	0	0	0	0	0	0	0	Set Ω_B^- of
4	4	5	1	0	0	0	0	0	0	p-insufficient
5	3	6	0	0	0	0	0	0	0	realizations
6	4	8	1	0	0	1	0	0	0	
8	1	9	0	0	0	1	1	0	0	
7	6	8	1	1	1	1	0	0	1	Set Ω_B^+ of
9	4	9	1	0	0	1	1	0	1	p-sufficient
10	5	10	1	1	0	1	1	1	1	realizations

2.2 Properties of Set of Cut Points

In Example 1, the binarization of the probability distribution with respect to the six cut points in (8) gives a pdBf such that the sets Ω_B^+ and Ω_B^- do not intersect. However, not any set of cut points allows this. Consider for example the set of cut points $\{c_{11} = 5;\ c_{12} = 4;\ c_{22} = 6\}$ which generates the same binary image $(0, 1, 1)$ (Figure 1) for the p-sufficient realization 9 and the p-insufficient ones 5, 6 and 8. Such a set of cut points (and the associated pdBf) does not preserve the disjointedness between the sets of p-sufficient and p-insufficient realizations. Indeed, it results in p-sufficient and p-insufficient realizations having the same binary projection and impedes the derivation of the conditions that are necessary for $\mathcal{P}(T_j x \geq \xi_j,\ j \in J) \geq p$ to hold. Clearly, the ability to accurately separate p-sufficient from p-insufficient realizations is a prerequisite for the derivation of a reformulation to the stochastic problem (1). This requires the generation of a *consistent* set of cut points.

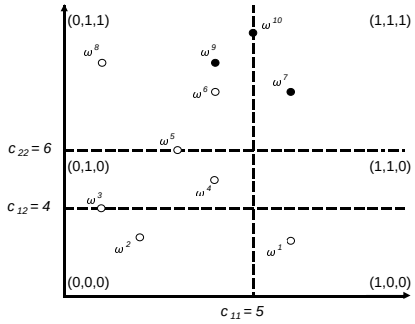
Definition 2 *A set of cut points is consistent if the sets Ω_B^+ and Ω_B^- associated with the pdBf $g(\Omega_B^+, \Omega_B^-)$ are disjoint. If this is the case, $g(\Omega_B^+, \Omega_B^-)$ is a consistent pdBf.*

We introduce the concept of *sufficient-equivalent* set of cut points.

Definition 3 *A sufficient-equivalent set of cut points C^e comprises a cut point c_{ij} for any value ω_j^k taken by any of the p-sufficient realizations on any component j:*

$$C^e = \{c_{ij} : c_{ij} = \omega_j^k,\ j \in J, k \in \Omega^+\}\ . \tag{9}$$

Figure 1: Inconsistent Set of Cut Points



The pdBf $g(\Omega_B^+, \Omega_B^-)$ associated with the sufficient-equivalent set of cut points is called sufficient-equivalent pdBf. Proposition 1 is obvious and a direct consequence of Definition 3.

Proposition 1 *A sufficient-equivalent set of cut points is consistent.*

The construction of the sufficient-equivalent set of cut points is immediate. In our example, the sufficient-equivalent set of cut points is the one defined in (8). Note that the combinatorial pattern literature [5, 21, 28] describes several techniques (polynomial-time algorithm, set covering formulation) to build consistent set of cut points with special features (master or minimal set of cut points).

2.3 Set of Relevant Realizations

The objective is to derive a combinatorial pattern that defines sufficient conditions, possibly the minimal ones, for the probabilistic constraint (2) to be satisfied. In order to do so, we cannot only take into consideration the realizations $k \in \Omega$ of the random vector. In addition to those, we shall consider and generate all points or realizations that could be p-sufficient. For k to be p-sufficient (i.e., $F(\omega^k \geq p)$, the $|J|$ following conditions must hold:

$$F_j(\omega_j^k) \geq p, j = 1, \ldots, |J| \,, \tag{10}$$

where F_j is the marginal probability distribution of ξ_j. Thus, for every j, we create the set of values Z_j

$$Z_j = \{\omega_j^k : F_j(\omega_j^k) \geq p, k \in \Omega, \;\; j = 1, \ldots, |J|\} \,, \tag{11}$$

define the direct product [43]

$$Z = Z_1 \times \ldots \times Z_j \times \ldots \times Z_{|J|} \,, \tag{12}$$

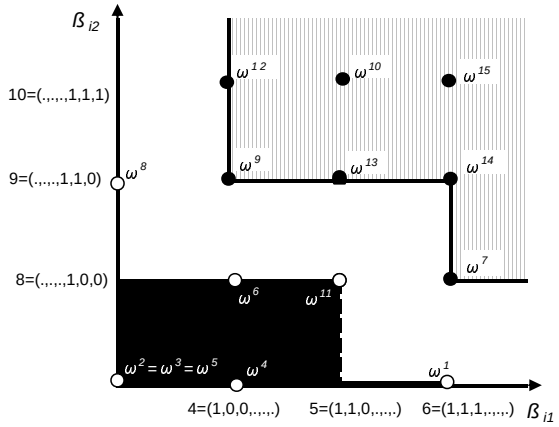
and obtain the extended set $\Omega \bigcup Z$ of realizations.

The application of the binarization process to the additional points included in Z provides their binarized images. In Example 1, the set Z comprises five realizations ($k = 11, \ldots, 15$ in Table 3).

Figure 2 shows that each p-sufficient realization is mapped into a binary vector which differs from all the binary vectors associated with p-insufficient realizations. The gray (resp., black) area in Figure 2 is the integer hull of the p-sufficient (resp., p-insufficient) realizations. All the points in the area between the two integer hulls correspond to vectors β with fractional values, which, by virtue of the binarization

process (5), correspond to numerical values that ξ_j cannot take and that do not belong to Z_j (11). The binarization process and the construction of the extended set of realizations enable the representation of the upper (resp., lower) envelope of the integer hull of the p-insufficient (resp., p-sufficient) points. Note that, if we do not consider realization 11 (which belongs to set Z) with binary image $\beta^{11} = (1, 1, 0, 1, 0, 0)$, we are not able to obtain the upper envelope of the integer hull of the p-insufficient points. This would be a problem for generating patterns sufficient conditions for the constraint (2) to hold.

Figure 2: Integer Hull of p-Sufficient and p-Insufficient Realizations



The binarization phase allows the *elimination* of a number of points and the derivation of the set $\bar{\Omega}$ of *relevant* realizations. Several realizations have the same binary image (e.g., realizations 2 and 3) and we only include one of them in $\bar{\Omega}$. Recall that the objective is to derive patterns defining sufficient conditions for the satisfaction of (2). A well known set of necessary conditions for p-sufficiency is given by (10), which, based on the definition of the sufficient-equivalent set of cut points (9) used for the binarization process (5), can be rewritten as: $\beta^k_{1j} = 1, j \in J$. Thus, each realization k such that $\beta^k_{1j} = 0$ for any $j \in J$ does not meet the basic preliminary condition, is a priori known to be p-insufficient, is not needed to generate patterns separating p-sufficient realizations from p-insufficient ones, and is not included in the set of relevant realizations. Table 3 gives the set of relevant realizations $\bar{\Omega}$ for Example 1.

Table 3: Set of Relevant Realizations $\bar{\Omega}_B$

	Numerical Representations		Binarized Images					
k	ω^k_1	ω^k_2	β^k_{11}	β^k_{21}	β^k_{31}	β^k_{12}	β^k_{22}	β^k_{32}
6	4	8	1	0	0	1	0	0
7	6	8	1	1	1	1	0	0
9	4	9	1	0	0	1	1	0
10	5	10	1	1	0	1	1	1
11	5	8	1	1	0	1	0	0
12	4	10	1	0	0	1	1	1
13	5	9	1	1	0	1	1	0
14	6	9	1	1	1	1	1	0
15	6	10	1	1	1	1	1	1

3 Mathematical Programming Approach for Combinatorial Patterns

In this section, we shall develop a mathematical programming approach allowing for the construction of combinatorial patterns that define sufficient conditions for the probabilistic constraint to hold. Prior to generating combinatorial patterns, we introduce the terminology and explain the rationale for the use of mathematical programming in order to derive patterns.

3.1 Extension of the Partially Defined Boolean Function of (F, p)

Section 2 details how the binarization process permits the derivation of a pdBf that represents the combination (F, p) of the probability distribution F with the probability level p. The question that arises now is whether a compact extension [54] of the pdBf representing (F, p) can be derived.

Definition 4 *[7] Let $\mathcal{B} = \{0, 1\}$ and consider the pdBf g defined by the pair of sets $(\mathcal{T}, \mathcal{F})$: $\mathcal{T}, \mathcal{F} \subseteq \mathcal{B}^n$. A function $f : \mathcal{B}^n \rightarrow \mathcal{B}$ is called an extension of the pdBf $g(\mathcal{T}, \mathcal{F})$ if $\mathcal{T} \subseteq \mathcal{T}(f)$ and $\mathcal{F} \subseteq \mathcal{F}(f)$.*

It was shown [5] that a pdBf $g(\mathcal{T}, \mathcal{F})$ has a Boolean extension [7] if and only if $\mathcal{T} \bigcap \mathcal{F} = \emptyset$, which is equivalent to saying that any consistent pdBf can be extended by a Boolean function. Therefore, from Proposition 1, we know that the sufficient-equivalent pdBf representing (F, p) can be extended as a Boolean function. With the existence of a Boolean extension for the pdBf ensured, the objective is to find an extension f that is defined on the same support set as $g(\bar{\Omega}_B^+, \bar{\Omega}_B^-)$ and that is as simple and compact as possible. Since every Boolean function can be represented by a DNF, we shall extend $g(\bar{\Omega}_B^+, \bar{\Omega}_B^-)$ as a DNF which is a disjunction of a finite number of *combinatorial patterns*.

3.2 Terminology

Before defining the DNF that extends $g(\bar{\Omega}_B^+, \bar{\Omega}_B^-)$, we introduce the key Boolean concepts and notations that are used in the remaining part of this paper and illustrate them with Example 1.

The Boolean variables $\beta_{ij}, i = 1, \ldots, n_j, j \in J$ and their *negations* or *complements* $\bar{\beta}_{ij}$ are called *literals*. A conjunction of literals $t = \bigwedge\limits_{ij \in P} \beta_{ij} \bigwedge\limits_{ij \in N} \bar{\beta}_{ij}$, $P \bigcap N = \emptyset$ constitutes a *term* [5, 28] or *clause* [52, 53] whose degree d is the number ($|P| + |N| = d$) of literals in it. The set P (resp., N) defines the set of non-complemented (resp., complemented) literals involved in the definition of the term t. A disjunction $\bigvee\limits_{s=1}^{S} t_s$ of terms t_s is called a *disjunctive normal form* (DNF) which has degree d if $|P_s \bigcup N_s| \leq d, s = 1, \ldots, S$, i.e., if the maximum number of literals included in any of the terms of the DNF is d.

Definition 5 *A term t is said to cover a realization k, which is denoted by $t(k) = 1$, if the products of the values β_{ij}^k taken k on the literals β_{ij} defining the term is equal to 1:*

$$t(k) = 1 \Leftrightarrow \bigwedge_{ij \in P} \beta_{ij}^k \bigwedge_{ij \in N} \bar{\beta}_{ij}^k = 1 \, .$$

The *coverage* of a term, pattern, or DNF is the number of realizations covered by it. In the above example, $t = \beta_{11}\, \bar{\beta}_{12}$ is a term of degree 2 covering the two negative realizations 1 and 4, and $f =$

$\beta_{11}\bar{\beta}_{12} \bigvee \beta_{31}\bar{\beta}_{32}$ is a DNF that contains two terms of degree 2: f covers the two p-insufficient realizations 1 and 4 and the two p-sufficient ones 7 and 14.

It follows from Definition 4 that the DNF f extending the pdBf $g(\bar{\Omega}_B^+, \bar{\Omega}_B^-)$ must be such that each realization defined as p-sufficient (resp., p-insufficient) by the pdBf $g(\bar{\Omega}_B^+, \bar{\Omega}_B^-)$ must also be considered as p-sufficient $\bar{\Omega}_B^+(f)$ (resp., p-insufficient $\bar{\Omega}_B^-(f)$) by the DNF f. This is equivalent to requiring that the DNF f covers all p-sufficient realizations and does not cover any p-insufficient ones:

$$\begin{cases} f(k) \geq 1, \ k \in \bar{\Omega}_B^+ \\ f(k) = 0, \ k \in \bar{\Omega}_B^- \end{cases} .$$

The DNF $f = \bigvee\limits_{s \in S} t_s$ includes a number $|S|$ of p-patterns which defines sufficient conditions for the probabilistic constraint (2) to hold.

Definition 6 *A term t is a p-pattern if it covers at least one p-sufficient realization and does not cover any p-insufficient one:*

$$\bigvee_{k \in \bar{\Omega}_B^+} t(k) \geq 1 \quad \textit{and} \quad \bigwedge_{k \in \bar{\Omega}_B^-} t(k) = 0 .$$

Broadly defined, a p-pattern is a logical rule that imposes upper and lower bounds on the values of a subset of the input variables and takes the form of a conjunction of literals. It can be interpreted as a subcube of the n-dimensional unit cube $\{0, 1\}^n$ that intersects $\bar{\Omega}_B^+$ (i.e., one or more p-sufficient realization satisfies its conditions) but do not intersect $\bar{\Omega}_B^-$ (i.e., no p-insufficient realization satisfies its conditions). Corollary 1 immediate follows from the construction of the extended set of realizations (11)-(12) and the use of the sufficient-equivalent set of cut points (Definition 3).

Corollary 1 *Consider a sufficient-equivalent set of cut points. A term that does not cover any p-insufficient realization necessarily covers at least one p-sufficient realization.*

3.3 Pattern Properties: Rationale for Mathematical Programming Generation

3.3.1 Properties

In order to derive patterns that can be conveniently used for computational purposes, we shall attempt to derive *prime* [24] patterns.

Definition 7 *A pattern is prime if the removal of one of its literals transforms it into a term which is not a pattern.*

Basically, it means that a prime pattern does not include any redundant literals. We also observe that, for a probabilistic constraint to hold, (at least) one condition must be imposed on each component ξ_j of the $|J|$-dimensional random vector. Proposition 2 follows:

Proposition 2 *The degree of a pattern defining sufficient conditions for (2) to hold is of degree at least equal to $|J|$.*

We shall now investigate whether the pdBf representing (F, p) can take some particular functional form facilitating its computational handling. We first consider the *monotonicity* property which, for Boolean functions, provides crucial computational advantages [11, 50].

Definition 8 *[44] A Boolean function f is positive (increasing) monotone, also called isotone, if $x \leq y$ implies $f(x) \leq f(y)$.*

The inequality sign is understood componentwise. The conditions under which a pdBf can be extended as a positive Boolean function is given in [5] as:

Lemma 1 *A pdBf $g(\mathcal{T}, \mathcal{F})$ has a positive Boolean extension if and only if there is no $x \in \mathcal{T}$ and $y \in \mathcal{F}$ such that $x \leq y$.*

Lemma 1 is used to derive Theorem 1 which applies to the type of extension (i.e., extension of a pdBf representing the combination of a probability distribution and of a probability level) studied in this paper.

Theorem 1 *Any Boolean extension of a consistent pdBf $g(\bar{\Omega}_B^+, \bar{\Omega}_B^-)$ representing (F, p) is a positive Boolean function.*

Proof. Each ω^k (resp., ω_j^k, $j \in J$) is a positive monotone variable in the multivariate (resp., marginal) cumulative probability distribution F (resp., F_j) of ξ (resp., ξ_j, $j \in J$):

$$\mathcal{P}\left(\xi \leq \omega^k\right) \leq \mathcal{P}\left(\xi \leq \omega^{k'}\right) \text{ if } \omega^k \leq \omega^{k'} \quad \text{and} \quad \mathcal{P}\left(\xi_j \leq \omega_j^k\right) \leq \mathcal{P}\left(\xi_j \leq \omega_j^{k'}\right) \text{ if } \omega_j^k \leq \omega_j^{k'},\ j \in J\ .$$

Definition 1 states that $k \in \bar{\Omega}^+$ if and only if $\mathcal{P}(\xi \leq \omega^k) \geq p$, and $k' \in \bar{\Omega}^-$ if and only if $\mathcal{P}(\xi \leq \omega^{k'}) < p$. Therefore, there is no $k \in \bar{\Omega}^+, k' \in \bar{\Omega}^-$ such that $\omega^{k'} \geq \omega^k$. Along with the consistency of the set of cut points used for the binarization process (5), this ensures that the pdBf g representing (F, p) is monotone increasing [6] in the value of each β_{ij}

$$g\left(\beta_{11}, \beta_{21}, \ldots, \beta_{i-1\,j}, 0, \beta_{i+1\,j}, \ldots, \beta_{n_{|J|}|J|}\right) \leq g\left(\beta_{11}, \beta_{21}, \ldots, \beta_{i-1\,j}, 1, \beta_{i+1\,j}, \ldots, \beta_{n_{|J|}|J|}\right)\ .$$

The consistency of g preserves the disjointedness between the sets $\bar{\Omega}_B^+$ and $\bar{\Omega}_B^-$, which implies that $k \in \bar{\Omega}_B^+$ if and only if $g(\beta^k) = 1$ and $k' \in \bar{\Omega}_B^-$ if and only if $g(\beta^{k'}) = 0$.
The consistency and monotonicity properties of g imply that there is no $k \in \bar{\Omega}_B^+, k' \in \bar{\Omega}_B^-$ such that $\beta^{k'} \geq \beta^k$. This, along with Lemma 1, completes the proof. □
Theorem 1 is very important, since it was shown [51] that patterns included in a DNF that constitutes a positive Boolean function do not need to contain complemented literals. The monotonicity property implies that prime patterns (Definition 7) included in a DNF that is an isotone Boolean function do not contain complemented literals [6], and, for the problem at hand, leads to the following Lemma

Lemma 2 *Prime p-patterns do not contain any complemented literal $\bar{\beta}_{ij}$.*

which, combined with Proposition 2, indicates that

Lemma 3 *Prime p-patterns for realizations of a $|J|$-variate random variable are of degree $|J|$.*

Proof. Let $t = \bigwedge_{ij \in P} \beta_{ij} \bigwedge_{ij \in N} \bar{\beta}_{ij}$ be a p-pattern of degree $d = |P| + |N|$. From Lemma 2, we know that $N = \emptyset$ if t is a prime pattern.

Consider that t includes two literals β_{ij} and $\beta_{i'j}$ associated with the same component ξ_j. Let $i < i'$ which implies that $c_{ij} < c_{i'j}$. It follows from (5) and (6) that the requirement imposed by β_{ij} is always satisfied by meeting the one imposed by $\beta_{i'j}$. If the removal of $\beta_{i'j}$ transforms t into a term that is not a pattern, then $\beta_{i'j}$ must be kept among the literals included in t. This makes β_{ij} redundant and the definition of a prime pattern requires its removal. On the other hand, if the removal of $\beta_{i'j}$ does not result in t covering any p-insufficient realization, then it means that $\beta_{i'j}$ is not required and should be removed. This shows that prime p-patterns contain at most one literal per component ξ_j, and are thus of degree $|J|$. □

Lemma 4 follows immediately:

Lemma 4 *A pdBf $g(\bar{\Omega}_B^+, \bar{\Omega}_B^-)$ representing (F, p), where F is a $|J|$-variate probability distribution, can be extended as a DNF containing prime p-patterns of degree $|J|$ that do not include any complemented literal $\bar{\beta}_{ij}$.*

3.3.2 Rationale

Combinatorial patterns are usually generated using term enumeration methods in the combinatorial data mining literature [2, 3, 6, 13, 21]. Recent research related to the combinatorial methodology called logical analysis of data [20] has led to major development in this area and has shown that enumeration methods are very efficient [2, 22, 23] when used for the generation of patterns of small degree (up to 4). The `LAD - Datascope 2.01` software package [1] implements a variety of enumeration algorithms. However, enumerative techniques are extremely computationally expensive [5] when they are used to generate terms of larger degree. Indeed, the number of terms of degree up to d is equal to $\sum_{d'=1}^{d} 2^{d'} \binom{n}{d'}$ and increases very fast with the number n of Boolean variables (and cut points). This is a concern, since, as indicated by Lemma 3, prime p-patterns are of degree $|J|$, which is equal to the dimensionality of the multivariate probability distribution of ξ and potentially large. This motivates the development of a mathematical programming approach for the generation of patterns.

In the combinatorial data mining, a set covering formulation was proposed in [5] for the generation of patterns. While the data mining literature derives patterns to classify data, the objective pursued in this paper is to use combinatorial patterns for the solution of probabilistically constrained optimization problems. Namely, the generated patterns permit the formulation of a tight linear programming inner approximation as well as that of the deterministic equivalent of probabilistically constrained problems. Besides the difference in objective, the mathematical programming formulations proposed in this paper substantially differ from those that can be found in the data mining literature. In particular, we propose two linear programming formulations for the derivation of patterns. The reader is referred to [11, 26, 27, 29] for studies of the interplay between logic, Boolean mathematics, and optimization.

3.4 Mathematical Programming Derivation of p-Pattern

In this section, we propose four mathematical programming formulations for the generation of a p-pattern. Definition 6 shows that a p-pattern defines sufficient conditions for the probabilistic constraint (2) to hold. The *optimal* p-pattern is the one that enforces the minimal conditions for (2) to hold. However, it is not possible to identify some specific properties that an optimal p-pattern has and to accordingly propose a tailored formulation for its generation. Thus, we shall focus on the derivation of a p-pattern that defines sufficient conditions that are "close to" the minimal ones. The proposed formulations account for the following aspects. The optimal p-pattern as well as those defining close-to-minimal conditions represent faces of the lower envelope of the integer hull of the set of p-sufficient realizations and are thus likely to have "large" coverage (see Figure 2).

3.4.1 Integer Programming Formulations

The first integer programming formulation `IP1` is such that its optimal solution defines the p-pattern with *maximal* coverage.

The following notations are used. The decision variables u_{ij} and y^k, respectively associated to the literals β_{ij} and to the p-sufficient realizations k, are binary (17)-(18). The value taken by u_{ij} defines the literals that are included in the p-pattern t: u_{ij} takes value 1 if β_{ij} is included in t, and is equal to 0 otherwise. The binary variable y^k is used to identify which p-sufficient realizations are covered by t as defined by the feasible solution of `IP1`: y^k takes value 1 if the p-sufficient realization k is not covered by t, and can take value 0 otherwise.

The objective function (13) minimizes the number of p-sufficient realizations not covered by the pattern t. Each constraint in (14) forces y^k to take value 1 if the p-sufficient realization k is not covered by t. Each constraint in (15) does not permit t to cover any p-insufficient realization. Constraints (16) force the inclusion in t of one non-complemented literal (and no complemented literal) per component j. We denote by $\mathbf{z}^*$ the optimal value of the objective function. Recall that the parameter β^k_{ij} indicates whether ω^k_j is at least equal to c_{ij} (5) and that we use a sufficient-equivalent set of n cut points.

Theorem 2 *Any feasible solution* $(\mathbf{u}, \mathbf{y})$ *of the integer programming problem* `IP1`

$$z = \min \sum_{k \in \bar{\Omega}^+_B} y^k \tag{13}$$

$$\textit{subject to} \quad \sum_{j \in J} \sum_{i=1}^{n_j} \beta^k_{ij} u_{ij} + |J| y^k \geq |J|, \quad k \in \bar{\Omega}^+_B \tag{14}$$

$$\sum_{j \in J} \sum_{i=1}^{n_j} \beta^k_{ij} u_{ij} \leq |J| - 1, \quad k \in \bar{\Omega}^-_B \tag{15}$$

$$\sum_{i=1}^{n_j} u_{ij} = 1, \quad j \in J \tag{16}$$

$$u_{ij} \in \{0, 1\}, \quad j \in J, i = 1, \ldots, n_j \tag{17}$$

$$y^k \in \{0, 1\}, \quad k \in \bar{\Omega}^+_B \tag{18}$$

(i) defines a prime p-pattern

$$t = \bigwedge_{\substack{\mathbf{u}_{ij}=1 \\ j\in J, i=1,\ldots,n_j}} \beta_{ij}$$

of degree $|J|$*; (ii)* IP1 *has an upper bound equal to* $|\bar{\Omega}_B^+| - 1$*; and (iii) its optimal solution* $(\mathbf{u}^*, \mathbf{y}^*)$ *defines the p-pattern with maximal coverage equal to* $\left(\left|\bar{\Omega}_B^+\right| - \mathbf{z}^*\right)$.

Proof. *(i)* p-pattern: Let $P = \{ij : \mathbf{u}_{ij} = 1, j \in J, i = 1, \ldots, n_j\}$ the set of (non-complemented) literals in t. From Definition 5, we have: $t(k) = 1 \Leftrightarrow \prod_{ij\in P} \beta_{ij}^k = 1$. Thus, (15) ensures that there is no $k \in \bar{\Omega}_B^-$ that can be covered by t, which, combined with Corollary 1, is enough to show that t is a p-pattern. Constraints (16) ensure the inclusion of exactly one uncomplemented literal in t. Thus, t is a pattern of degree $|J|$ and, from Lemma (3), is prime.
(ii) Upper bound: Consider $k \in \bar{\Omega}_B^+$: (14) allows y^k to take value 0 if $t(k) = 1$. Otherwise, y^k is forced to take value 1. Corollary 1 indicates that any pattern not covering any p-insufficient realization covers one or more p-sufficient one. Thus, the number of uncovered p-sufficient realizations $\sum_{k\in\bar{\Omega}_B^+} y^k$ is $\leq |\bar{\Omega}_B^+| - 1$, which is a valid upper bound on the objective value of IP1.
(ii) Coverage: The objective function maximizes the number $\sum_{k\in\bar{\Omega}_B^+} (1 - y^k)$ of p-sufficient realizations covered by t. Thus, the pattern t^* defined by the optimal solution $(\mathbf{u}^*, \mathbf{y}^*)$ has maximal coverage equal to the difference between the number ($|\bar{\Omega}_B^+|$) of p-sufficient realizations and the number ($\mathbf{z}^* = \sum_{k\in\bar{\Omega}_B^+} \mathbf{y}^{k^*}$) of those that are not covered by t.

The number of binary variables in the above MIP is equal to $n + |\bar{\Omega}_B^+|$, and increases with the number of cut points and with the number of p-sufficient realizations, which monotonically decreases with the probability level p. Note that the above MIP does not need to be solved to optimality, since any feasible solution defines a p-pattern and that a pattern with maximal coverage is called a strong pattern [24].

Next, we formulate a mixed-integer programming (MIP) problem IP2 that contains a significantly smaller number of binary variables than IP1 and that also allows for the derivation of a p-pattern. The generated prime p-pattern t contains $|J|$ literals $\beta_{ij}, ij \in P$, and each literal defines a specific condition ($\omega_j^k \geq c_{ij}$) for a realization k to be covered by t. Instead of minimizing the number of p-sufficient realizations not covered by the pattern (see IP1), we shall now minimize the number of conditions imposed by the literals involved in t that are not satisfied by the p-sufficient realizations. If k is covered by t, then $\sum_{j\in J}\sum_{i=1}^{n_j} \beta_{ij}^k u_{ij} = |J|$. Otherwise, (20) forces y^k to be equal to the number ($|J| - \sum_{j\in J}\sum_{i=1}^{n_j} \beta_{ij}^k u_{ij}$) of conditions defined by the literals included in t that k does not satisfy. The resulting MIP problem IP2 contains n binary variables instead of ($n + |\bar{\Omega}_B^+|$) in IP1. The variables y^k are now continuous (21).

Theorem 3 *Any feasible solution* $(\mathbf{u}, \mathbf{y})$ *of the mixed-integer programming problem* IP2

$$z = \min \sum_{k \in \bar{\Omega}_B^+} y^k \tag{19}$$

$$\textit{subject to} \quad \sum_{j \in J} \sum_{i=1}^{n_j} \beta_{ij}^k u_{ij} + y^k = |J|, \quad k \in \bar{\Omega}_B^+ \tag{20}$$

$$0 \leq y^k \leq |J|, \qquad k \in \bar{\Omega}_B^+ \tag{21}$$

$$(15) - (17)$$

(i) defines a prime p-pattern

$$t = \bigwedge_{\substack{\mathbf{u}_{ij}=1 \\ j \in J, i=1,\ldots,n_j}} \beta_{ij}$$

of degree $|J|$ *and coverage* $|V|$ *with* $V = \{k : \mathbf{y}^k = 0, k \in \bar{\Omega}_B^+\}$*; and (ii)* `IP2` *has an upper bound equal to* $|J| \cdot (|\bar{\Omega}_B^+| - 1)$.

Proof. *(i) p*-pattern: We have that $t(k) = 0, k \in \bar{\Omega}_B^+$ (15). Thus, from Corollary 1, t is a p-pattern and is of degree $|J|$ (16), thus prime (Lemma 3). Since $y^k = 0$ if and only if $t(k) = 1, k \in \bar{\Omega}_B^+$, the coverage of t is thus equal to the cardinality of the set V defined above.
(ii) Upper bound: The number of literals included in t, thus the number of conditions that must be satisfied by k to be covered by t, is equal to $|J|$. Thus, over the set $\bar{\Omega}_B^+$, this represents $|J| \cdot |\bar{\Omega}_B^+|$ conditions. Since the set of cut points is consistent, preserving the disjointedness between $\bar{\Omega}_B^+$ and $\bar{\Omega}_B^-$, it is always possible to derive a prime p-pattern t with degree $|J|$. Such a pattern covers at least one p-sufficient realization k, thus $|J|$ of the above conditions always hold. The upper bound on z in `IP2` is $|J| \cdot (|\bar{\Omega}_B^+| - 1)$.
□

In Example 1, the optimal solutions of `IP1` and `IP2` provide both the same p-pattern $t = \beta_{11}\beta_{22}$ with coverage equal to $|V| = 6$ (t does not cover realization 7), and $\mathbf{z}^*$ is equal to 1 for `IP1` and `IP2`.

3.4.2 Linear Programming Formulations

We shall now propose two linear programming formulations for the generation of p-patterns.

Theorem 4 *Any feasible solution* $(\mathbf{u}, \mathbf{y})$ *of the linear programming problem* `LP1`

$$z = \min \sum_{k \in \bar{\Omega}_B^+} y^k \tag{22}$$

$$\textit{subject to} \quad (14) - (16)$$

$$0 \leq u_{ij} \leq 1, \qquad j \in J, i = 1, \ldots, n_j \tag{23}$$

$$0 \leq y^k \leq 1, \qquad k \in \bar{\Omega}_B^+ \tag{24}$$

defines a p-pattern

$$t = \bigwedge_{\substack{\mathbf{u}_{ij}>0 \\ j \in J, i=1,\ldots,n_j}} \beta_{ij} \tag{25}$$

with coverage $|V|$ *with* $V = \{k : \mathbf{y}^k = 0, k \in \bar{\Omega}_B^+\}$.

Proof. Constraints (15) prevent t from covering any $k \in \bar{\Omega}_B^-$ and Corollary 1 implies that t is a p-pattern. From (14), we have that $y^k = 0$ if and only if $t(k) = 1 \Leftrightarrow \beta_{ij}^k = 1,\ ij \in P$. Thus, the coverage of t is $|V|$. □

Problem LP1 is a linear program and is obviously simpler to solve than IP1 and IP2. The "cost" of removing the integrality restrictions on the variables is twofold. First, although the objective function is still related to the coverage of the generated pattern, it cannot be interpreted anymore as representing the number of p-sufficient realizations covered by t (IP1) or as the number of conditions imposed by t that are not met by the p-sufficient realizations (IP2). Second, the pattern t defined by a feasible solution of LP1 is not necessarily prime and can contain a number of literals much larger than $|J|$, which could be inconvenient from a computational point of view. This can be easily remedied. Indeed, from the knowledge of the pattern t generated by LP1, one can immediately derive a prime p-pattern.

Corollary 2 *A prime p-pattern*

$$t = \bigwedge_{\substack{\bar{u}_{i_j j}=1 \\ j\in J, i=1,\ldots,n_j}} \beta_{ij} \tag{26}$$

with $\bar{i}_j = \underset{i}{\operatorname{argmax}}\, \mathbf{u}_{ij} > 0, j \in J \quad , \quad \bar{u}_{ij} \begin{cases} 1 & \text{if} \quad i = \bar{i}_j \\ 0 & \text{otherwise} \end{cases}$

can immediately be derived from any feasible solution $(\mathbf{u}, \mathbf{y})$ *of the linear programming problem* LP1.

Proof. Any component $j \in J$ requires, for k to be covered by t defined by (25), that: $\beta_{ij}^k = 1, ij \in P$. All these conditions can be subsumed by: $\beta_{\bar{i}_j j}^k = 1$. Indeed, from the binarization process (5)-(6), we know that: $\beta_{\bar{i}_j j}^k \leq \beta_{ij}^k,\ (ij) \in P$. Thus, $\beta_{\bar{i}_j j}^k = 1$ implies that $\beta_{ij}^k = 1,\ (ij) \in P$. This means that t defined by (26), which includes only literal $\beta_{i_j j}$ per component j, and is thus prime, defines the same conditions as the pattern defined by (25), and is a p-pattern. □

In Example 1, the optimal solution of LP1 allows the derivation of a prime p-pattern $t = \beta_{11}\beta_{22}$ with coverage equal to $|V| = 6$ and $\mathbf{z}^* = 0.5$.

In the second linear programming formulation LP2, we have a reduced set of $n + |J| + |\bar{\Omega}_B^-|$ constraints and only n continuous decision variables u. We introduce a set of parameters b_{ij} which can be viewed as the price of including the literal β_{ij} in the definition of the pattern t.

Theorem 5 *Any feasible solution* $(\mathbf{u})$ *of the linear programming problem* LP2

$$z = \min \sum_{j\in J} \sum_{i=1}^{n_j} b_{ij} u_{ij} \tag{27}$$

subject to (15) − (16)

$$0 \leq u_{ij} \leq 1, \qquad j \in J, i = 1, \ldots, n_j \tag{28}$$

defines a p-pattern

$$t = \bigwedge_{\substack{\mathbf{u}_{ij}>0 \\ j\in J, i=1,\ldots,n_j}} \beta_{ij}\ .$$

The proof is the same as for LP1. As for LP1, a feasible solution of LP2 does not necessarily define a prime p-pattern, but we can apply Corollary 2 to construct a prime p-pattern. In Example 1, the optimal solution of LP2 gives the p-pattern $t = \beta_{11}\beta_{22}$ with coverage equal to $|V| = 6$ and $\mathbf{z}^* = \mathbf{3}$.

The optimal solution $(\mathbf{u})^*$ of LP2 defines the "least costly" p-pattern. Several approaches can be used to price the inclusion of a literal β_{ij} in the pattern. This is done through the definition of the weights b_{ij} for which we propose the two following guidelines:

- intra-component pricing: We differentiate the weights b_{ij} assigned to the literals associated with the same component j. The goal is to generate a p-pattern that defines the minimal (or close to minimal) conditions for the attainment of the probability level p. Accordingly, we want to include in the pattern t the literals imposing the least demanding conditions. Thus, for any given j and $i > i'$, it is preferable, when possible, to include $\beta_{i'j}$ than β_{ij} in t and we accordingly price $\beta_{i'j}$ cheaper than β_{ij} by setting $b_{ij} > b_{i'j}$, $j \in J$
- inter-component pricing: The value of $b_{ij}, i = 1, \ldots n_j$ associated with component j is an increasing function of the cost associated with j in the objective function of the stochastic problem (1).

In Section 5, we shall evaluate the numerical efficiency of the four proposed mathematical programming formulations and the time needed to solve them to optimality.

4 Linear Reformulation of Probabilistic Problem

4.1 Linear Programming Inner Approximation of Probabilistic Problems

In this section, we derive an inner approximation, taking the form of a *linear program*, for the probabilistically constrained problem (1). The construction of the inner approximation problem is based on the generation of a p-pattern using one of the formulations proposed in Section 3.4.

Theorem 6 *Consider a p-pattern* $t = \bigwedge_{ij \in P} \beta_{ij}$, *with P denoting the set of literals included in t. The linear programming problem* IALP

$$\begin{aligned} &\min c^T x \\ &\textit{subject to } Ax \geq b \\ &\qquad T_j x \geq c_{ij},\ ij \in P \\ &\qquad x \geq 0 \end{aligned} \tag{29}$$

is an inner approximation of the probabilistic problem (1).

Proof. $t(k) = 1 \Leftrightarrow \beta_{ij}^k = 1,\ ij \in P$, which is equivalent to

$$\omega_j^k \geq c_{ij},\ ij \in P\,. \tag{30}$$

Besides, $t(k) = 1$ requires that k is p-sufficient (Definition 6), which, in turn, implies that $\mathcal{P}(\omega^k \geq \xi) \geq p$ (Definition 1). Substituting Tx for ω^k in (30) provides the result that was set out to prove. □

The above linear programming problem can be obtained by using any of the four formulations proposed for the generation of p-patterns. The key question that is addressed in Section 5 pertains to the *tightness* of the inner approximation obtained with the four proposed models.

4.2 Linear Deterministic Equivalent of Probabilistic Problems

We shall now derive a linear deterministic equivalent, taking the form of an MIP problem, called DEIP, for the probabilistically constrained program (1). The derivation of problem DEIP is based on the generation of a p-pattern. The solution of DEIP allows for the *concurrent (i)* generation of the prime p-pattern defining the minimal conditions for the probabilistic constraint (2) to hold and for the *(ii)* reformulation and exact solution of the probabilistic programming problem (1).

Theorem 7 *The mixed-integer programming problem* DEIP

$$
\begin{aligned}
\min \quad & c^T x \\
\textit{subject to} \quad & Ax \geq b \\
& \sum_{j \in J} \sum_{i=1}^{n_j} \beta_{ij}^k \, u_{ij} \leq |J| - 1 \,, \quad k \in \bar{\Omega}_B^- && (31) \\
& T_j x \geq \sum_{i=1}^{n_j} c_{ij} \, u_{ij} \,, \quad j \in J && (32) \\
& \sum_{i=1}^{n_j} u_{ij} = 1, \quad j \in J && (33) \\
& u_{ij} \in \{0,1\}, \quad j \in J, i = 1, \ldots, n_j \\
& x \geq 0
\end{aligned}
$$

is a deterministic equivalent of the probabilistically constrained problem (1). The optimal solution $(\mathbf{u}^*, \mathbf{x}^*)$ *of* DEIP *gives the prime p-pattern*

$$
t \;=\; \bigwedge_{\substack{\mathbf{u_{ij}^*}=\mathbf{1} \\ j \in J, i = 1, \ldots, n_j}} \beta_{ij}, \tag{34}
$$

that defines the minimal conditions for the probabilistic constraint (2) to be satisfied.

Proof. t defined by (34) is a p-pattern, since (31) implies that $t(k) = 0, k \in \bar{\Omega}_B^-$, and is prime, with degree $|J|$ (33). Thus, k can only be covered by t if k is p-sufficient. Besides, $t(k) = 1 \Leftrightarrow \omega_j^k \geq c_{ij}, \; ij \in P$ (30), which is equivalent to

$$
\omega_j^k \geq \sum_{i=1}^{n_j} c_{ij} \, u_{ij}, j \in J \,,
$$

since (33) ensures that only one literal per component j is included in t, i.e., that only one term in the right part of (32) is non-zero. Replacing ω^k by Tx in the above inequality gives (32) which ensures that $\mathcal{P}(Tx \geq \xi) \geq p$ and that Tx satisfies (2). It follows that the optimal solution $(\mathbf{u}^*, \mathbf{x}^*)$ defines the minimal

value that x can take to satisfy (2) and that $\mathbf{u}^*$ allows the generation of the pattern t defining the minimal conditions for (2) to hold. □

Other MIP reformulation approaches have been proposed to derive a deterministic equivalent for (1). The MIP deterministic equivalent reformulations [12, 32, 33, 43] obtained by using the p-efficiency concept [41] associates one binary variable with each p-efficient point which must be found a priori. In [35], several MIP formulations are proposed in which a binary variable is associated to each possible realization. In contrast to this, the number of binary variables in the proposed reformulation `DEIP` is not an increasing function of the number of scenarios used to describe the uncertain variables. It contains a significantly lower number (n) of binary variables, equal to the cardinality of the sufficient-equivalent set of cut points (Definition 3) used for the binarization process.

5 Numerical Implementation

This section evaluates the computational efficiency of the proposed combinatorial pattern approach. The first part compares the speed of the four mathematical programming methods for the generation of p-patterns and analyzes the tightness of the inner approximation obtained with the four methods. The second part pertains to the computational times needed to solve the deterministic equivalent reformulation of the probabilistic problem.

The tests are conducted on a stochastic version of a supply chainoproblem problem in which a set K of distributors must satisfy the random demand ξ of a set J of customers. The decision variables x_{kj} are the supply quantities delivered by a distributor k to a customer j. The model reads:

$$\begin{aligned}
\min \quad & \sum_{k\in K}\sum_{j\in J} c_{kj}x_{kj} && && (35)\\
\text{subject to} \quad & \sum_{j\in J} x_{kj} \le M_k\,, && k\in K && (36)\\
& x_{kj} \le V_{kj}\,, && k\in K, j\in J && (37)\\
& \mathcal{P}(\sum_{k\in K} x_{kj} \ge \xi_j, j\in J) \ge p && && (38)\\
& x \ge 0
\end{aligned}$$

The parameter c_{kj} is the cost of supplying one unit from k to j. The objective function (35) minimizes the sum of the distribution costs. Constraints (36) upper-bound (M_k) the aggregated supply quantity delivered by k to all its customers. Constraints (37) upper-bound (V_{kj}) the individual supply quantity delivered by k to each customer j. Constraints (38) require that the distributors satisfy the demand of all of their customers with a large probability level p.

For the problem instances used in this section, the parameters c_{kj}, M_k and V_{kj} of the above model were randomly generated from uniform distributions. The probability distribution of ξ is described with a finite set of Ω realizations defined as equally likely and sampled from a uniform distribution.

We have created 32 types of problem instances characterized by the tuple $(|J|, |\Omega|, p)$. The instances differ in terms of the dimension ($|J| = 10, 20$) of the random vector, the number of realizations

($|\Omega| = 5000, 10000, 20000, 50000$), and the enforced probability level ($p = 0.875, 0.9, 0.925, 0.95$). For each instance type, we generate five problem instances. Table 5 reports the (time and gap) averages over the five instances of an instance type. The binarization process is carried out with Matlab. The AMPL modeling language [16] is used to formulate the mathematical programming problems which are solved with the CPLEX 11.1 solver. Each problem instance is solved on a 64-bit Dell Precision T5400 Workstation with Quad Core Xeon Processor X5460 3.16GHz CPU, and 4X2GB of RAM.

5.1 Pattern Generation and Solution of Inner Approximation

The fourth (resp., sixth, eighth, and tenth) column in Table 5 reports, for each type of family (see the first three columns of Table 5), the sum of the average computational times needed *(i)* to generate a pattern with the `IP1` (resp., `IP2`, `LP1`, and `LP2`) formulation and *(ii)* to solve the resulting linear programming inner approximation. It can be seen that the four approaches are very fast and this even for problems in which the multivariate random variable is described by a very large number of scenarios. The two linear programming formulations are obviously the fastest (at most 1.8 sec, and most often much less), but the IP formulations are also fast to solve (at most 29 seconds for `IP1` and 5 seconds for `IP2`). It is not surprising to observe that the solution times for the `IP2` formulation are consistently smaller than those for `IP1`, since the former formulation contains a significantly lower of binary variables. For the IP formulations, the average computational time is an increasing function of the dimension of the random vector and a decreasing function of the probability level p.

The next, and most important question, to settle pertains to the *tightness* of the inner approximation that is derived using the patterns obtained with the four proposed formulations. We measure the tightness of the approximation by the relative optimality gap between the optimal value of the inner approximation and the optimal value of the original stochastic problem (1). The fifth (resp., seventh, ninth, and eleventh) column in Table 5 reports, for each type of family the average tightness of the inner approximation obtained by using the `IP1` (resp., `IP2`, `LP1`, and `LP2`) formulation. Table 4 reports the number M of instance types on which each formulation provides the tightest approximation. It appears that, besides being the fastest, the linear programming approach `LP2` is also the one providing the tightest inner approximations. The `LP2` model provides the most conservative ones.

Table 4: Tightness of Inner Approximation Approaches

	IP1	IP2	LP1	LP2
M	8	8	1	21

5.2 Concurrent Pattern Generation and Solution of Deterministic Equivalent

For the 160 problem instances, we solved problem `DEIP` that allows for the simultaneous generation of the prime p-pattern defining the minimal conditions for the probabilistic constraint (2) to hold and for the solution of the deterministic equivalent formulation of (1). Problem `DEIP` contains $|J|$ set partitioning constraints (33) that can be explicitly defined as special ordered set constraints of type one (SOS1).

The twelfth column in Table 5 shows that the deterministic equivalent formulation, for each family instance, can be solved extremely fast, in at most 3 seconds. The number of integer variables in problem DEIP is equal to the number of cut points, which, everything else being equal, increases as the probability level decreases (see Definition 3). Thus, it is logical that the computing time increases as the value of p decreases. We observe that the computing time increases at a very moderate rhythm which suggests that the method could be used for values of p even lower than those considered here.

A key feature of this approach is that the number of binary variables does not increase with the number of realizations. This is what allows the application of the proposed method for cases in which the random variables are subject to a very fine discretization and are characterized by an extra large number of scenarios. The computational results attest that the computing time is not monotone increasing in the number of realizations used to represent the random variables. To our knowledge, none of the existing methods reports numerical results for problem instances in which the number of scenarios is larger than 3000 (e.g., up to 3000 in [35] and 500 in [31]). We have implemented the method proposed in [35] and we could not obtained the optimal solution for any of the problem instances containing 20,000 or more scenarios in one hour of CPU time. The proposed method is thus a very good alternative to the existing algorithmic techniques.

6 Conclusion

We propose a novel methodology to solve probabilistically constrained optimization problems by using concepts from the combinatorial pattern recognition discipline [5, 20, 36, 45, 52, 53]. Combinatorial patterns are able to capture the ”interactions” between the components of a multi-dimensional random vector and their impact on making it possible to reach a predefined reliability level. They have the capability not only to identify the variables that individually influences the probability level p, but also to capture the *collective effect* of the values of those variables on the attainment of the prescribed probability level. Besides its novelty, the proposed method scales extremely well. It permits the very fast solution of stochastic optimization problems in which the random variables are represented by an unprecedently large number of scenarios.

The presented framework introduces the concepts of p-sufficient and p-insufficient realization of a random variable, and describes a binarization method for a probability distribution with finite support. We represent the combination of the binarized version of a probability distribution and a prescribed probability level by a consistent pdBf which can then be compactly extended as an isotone Boolean functional form. The Boolean extension represents the sufficient requirements for the satisfiability of a probabilistic constraint and is modelled as a DNF including p-patterns. Each of those are conjunctions of literals and define sufficient conditions for the satisfaction of a probabilistic constraint.

Enumerative methods, that are most often used for pattern generation purposes, are not very efficient for the construction of patterns with large degree. This motivates the design of a mathematical programming method for pattern generation. Four formulations (2 LPs and 2 IPs) are proposed for the generation of p-patterns which in turn allows for the derivation of a linear programming tight inner approximation

Table 5: Solution Times and Optimality Gap

Instance Type			IP1		IP2		LP1		LP2		DEIP
$\|J\|$	$\|\Omega\|$	p	Time	Opt. Gap	Time	Opt. Gap	Time	Opt. Gap	Time	Opt. Gap	Time
10	5000	0.875	1.594	1.316%	0.406	1.449%	0.030	1.698%	0.031	0.802%	2.212
10	5000	0.9	0.109	0.458%	0.063	0.253%	0.016	0.485%	0.016	0.000%	0.063
10	5000	0.925	0.031	0.734%	0.031	0.971%	0.034	0.856%	0.031	0.560%	0.016
10	5000	0.95	0.031	0.325%	0.031	0.346%	0.015	0.457%	0.016	0.116%	0.005
10	10000	0.875	1.625	0.986%	0.391	0.960%	0.020	1.442%	0.016	1.292%	0.141
10	10000	0.9	0.297	0.396%	0.125	0.432%	0.031	0.421%	0.031	0.868%	0.078
10	10000	0.925	0.031	0.778%	0.016	0.939%	0.036	0.967%	0.031	0.000%	0.016
10	10000	0.95	0.031	0.001%	0.016	0.001%	0.040	0.001%	0.031	0.001%	0.016
10	20000	0.875	1.594	0.292%	0.406	0.487%	0.015	0.541%	0.016	1.631%	0.109
10	20000	0.9	0.281	1.532%	0.094	1.509%	0.030	2.360%	0.031	1.341%	0.078
10	20000	0.925	0.031	0.352%	0.031	0.369%	0.014	0.796%	0.016	0.004%	0.031
10	20000	0.95	0.031	0.775%	0.031	1.031%	0.031	1.029%	0.031	0.002%	0.031
10	50000	0.875	1.578	0.231%	0.406	0.472%	0.016	0.237%	0.016	0.000%	0.141
10	50000	0.9	0.391	1.241%	0.141	1.525%	0.032	2.254%	0.031	1.033%	0.094
10	50000	0.925	0.031	0.381%	0.031	0.815%	0.015	1.259%	0.016	0.137%	0.016
10	50000	0.95	0.031	0.918%	0.031	0.846%	0.034	2.638%	0.031	0.853%	0.008
20	5000	0.875	14.563	0.796%	2.427	0.801%	0.221	1.234%	0.236	0.800%	1.036
20	5000	0.9	3.938	0.610%	1.313	0.608%	0.035	0.896%	0.031	0.459%	0.500
20	5000	0.925	0.047	0.260%	0.047	0.257%	0.036	0.364%	0.031	0.000%	0.032
20	5000	0.95	0.016	0.001%	0.031	0.000%	0.016	0.011%	0.016	0.126%	0.015
20	10000	0.875	15.698	0.924%	4.702	0.922%	0.256	1.264%	0.303	0.925%	1.874
20	10000	0.9	11.609	0.401%	3.688	0.406%	0.050	0.856%	0.031	0.001%	1.391
20	10000	0.925	0.047	0.205%	0.047	0.207%	0.020	0.745%	0.016	0.069%	0.031
20	10000	0.95	0.031	0.332%	0.016	0.340%	0.062	0.867%	0.031	0.068%	0.016
20	20000	0.875	26.539	0.904%	4.269	0.896%	0.126	1.762%	0.129	0.896%	2.891
20	20000	0.9	12.703	0.404%	2.281	0.408%	0.045	2.697%	0.047	0.878%	2.297
20	20000	0.925	0.016	0.175%	0.000	0.175%	0.015	1.671%	0.016	0.000%	0.031
20	20000	0.95	0.031	0.142%	0.016	0.137%	0.031	2.012%	0.031	0.137%	0.031
20	50000	0.875	28.963	0.862%	4.982	0.862%	1.784	3.002%	1.625	0.862%	2.652
20	50000	0.9	21.172	0.666%	3.578	0.678%	1.365	3.216%	1.438	0.956%	1.891
20	50000	0.925	0.047	0.459%	0.047	0.479%	0.010	2.891%	0.016	0.275%	0.031
20	50000	0.95	0.031	0.181%	0.047	0.189%	0.010	2.421%	0.016	0.252%	0.016

of the probabilistic problem. Finally, we propose a model that allows for the simultaneous *(i)* derivation of the p-pattern defining the minimal conditions for the probabilistic constraint to hold and *(ii)* solution of the deterministic equivalent problem.

Results for complex stochastic problems, in which a very fine discretization involving up to 50,000 scenarios is applied to represent the random variables, highlight the computational efficiency of the approach. All problem instances can be solved to optimality in less than three seconds. Moreover, the solution time is not an increasing function of the number of realizations used to describe the random variables. The linear programming formulation LP2 is solved the fastest and provides the tightest inner approximations. This study offers the possibility to solve either a tight inner approximation or the deterministic equivalent of the probabilistic programming problem (1), the choice between both could depend on the specifics of the problem and of the decision-making process and stage.

The proposed approach can be applied in the exact same fashion if the stochastic problem (1) includes integer decision variables, contains non-linear constraint(s) and/or objective function, and if the inequality to which the probabilistic requirement is imposed is nonlinear. Extensions of the proposed approach will concern the handling of probabilistic constraints with random technology matrix and two-stage stochastic programming problems.

References

[1] Alexe G. 2007. *LAD - Datascope V2.1 Software Package*.

[2] Alexe G., Hammer P.L. 2006. Spanned Patterns for the Logical Analysis of Data. *Discrete Applied Mathematics* 154 (7), 1039-1049.

[3] Alexe S., Hammer P.L. 2007. Accelerated Algorithm for Pattern Detection in Logical Analysis of Data. *Discrete Applied Mathematics* 154 (7), 1050-1063.

[4] Birge J.R., Louveaux F. 1997. *Introduction to Stochastic Programming*. Springer Verlag, New York, NY.

[5] Boros E., Hammer P.L., Ibaraki T., Kogan A. 1997. Logical Analysis of Numerical Data. *Mathematical Programming* 79, 163-190.

[6] Boros E., Hammer P.L., Ibaraki T., Kogan A., Mayoraz E., Muchnik I. 2000. An Implementation of Logical Analysis of Data. *IEEE Transactions on Knowledge and Data Engineering* 12(2), 292-306.

[7] Boros E., Ibaraki T., Makino K. 1997. Monotone Extensions of Boolean Data Sets. In: *Algorithmic Learning Theory*, Lecture Notes in Computer Science. Springer, Berlin-Heidelberg.

[8] Calafiore G.C., Campi M.C. 2005. Uncertain Convex Programs: Randomized Solutions and Confidence Levels. *Mathematical Programming* 102, 25-46.

[9] Charnes A., Cooper W.W., Symonds G.H. 1958. Cost Horizons and Certainty Equivalents: An Approach to Stochastic Programming of Heating Oil. *Management Science* 4, 235-263.

[10] Cheon M.-S., Ahmed S., Al-Khayyal F. 2006. A Branch-Reduce-Cut Algorithm for the Global Optimization of Probabilistically Constrained Linear Programs. *Mathematical Programming* 108, 617-634.

[11] Crama Y., Hammer P.L. 2009. *Boolean Functions - Theory, Algorithms, and Applications.*. Eds: Cambridge Press University. In Press.

[12] Dentcheva D., Prékopa A., Ruszczyński A. 2001. Concavity and Efficient Points of Discrete Distributions in Probabilistic Programming. *Mathematical Programming* 47 (3), 1997-2009.

[13] Djukova E.V., Inyakin A.S., Peskov N.V., Sakharov A.A. 2006. Increasing the Efficiency of Combinatorial Logical Data Analysis in Recognition and Classification Problems. *Pattern Recognition and Image Analysis* 16 (4), 707-711.

[14] Djukova E.V., Zhuravlev Y.I. 2000. Discrete Analysis of Feature Descriptions in Recognition Problems of High Dimensionality. *Computational Mathematics and Mathematical Physics* 40, 1214-1227.

[15] Duda R. O., Stork D.G., Hart P.E. 2001. *Pattern Classification*, Second Edition. Wiley & Sons.

[16] Fourer R., Gay D.M., Kernighan B.W. 2003. *AMPL: A Modeling Language for Mathematical Programming*. Second Edition, Duxbury Press Brooks Cole Publishing Co.

[17] Fukunaga K. 1990. *Introduction to Statistical Pattern Recognition*. Academic Press. London, UK.

[18] Gren I.-M. 2008. Adaptation and Mitigation Strategies for Controlling Stochastic Water Pollution: An Application to the Baltic Sea. *Ecological Economics* 66 (2-3), 337-347.

[19] Grenander U., Miller M.I. 2007. *Pattern Theory: From Representation to Inference*. Oxford University Press, NY, USA.

[20] Hammer P.L. 1986. Partially Defined Boolean Functions and Cause-Effect Relationships. *International Conference on Multi-Attribute Decision Making Via OR-Based Expert Systems*. University of Passau, Passau, Germany.

[21] Hammer P.L., Bonates T.O. 2006. Logical Analysis of Data: From Combinatorial Optimization to Medical Applications. *Annals of Operations Research* 148 (1), 203-225.

[22] Hammer P.L., Kogan A., Lejeune M.A. 2006. Modeling Country Risk Ratings Using Partial Orders. *European Journal of Operational Research* 175 (2), 836-859.

[23] Hammer P.L., Kogan A., Lejeune M.A. 2009. Reverse Engineering Country Risk Ratings: Statistical and Combinatorial Non-Recursive Models. Forthcoming in *Annals of Operations Research*.

[24] Hammer P.L., Kogan A., Simeone B., Szedmak S. 2004. Pareto-Optimal Patterns in Logical Analysis of Data. *Discrete Applied Mathematics* 144, 79-102.

[25] Henrion R. 2004. Introduction to Chance-Constrained Programming. *Tutorial Paper for the Stochastic Programming Community Home Page*.

[26] Hooker J. 2000. *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. Wiley & Sons.

[27] Hooker J.N. 2007. *Integrated Methods for Optimization*. Springer. New York, NY, USA.

[28] Ibaraki T. 2009. *Partially Defined Boolean Functions*. In: *Boolean Functions - Theory, Algorithms, and Applications.*. Eds: Crama Y., Hammer P.L. Cambridge Press University. In Press.

[29] Jeroslow R.G. 1989. *Logic-Based Decision Support: Mixed Integer Model Formulation*. Annals of Discrete Mathematics. North-Holland. Amsterdam, The Netherlands.

[30] Kress M., Penn M., Polukarov M. 2007. The Minmax Multidimensional Knapsack Problem with Application to a Chance-Constrained Problem. *Naval Research Logistics* 54, 656-666.

[31] Küçükyavuz. 2009. On Mixing Sets Arising in Probabilistic Programming. *Working Paper*: `http://www.optimization-online.org/DB_HTML/2009/03/2255.html`.

[32] Lejeune M.A. 2008. Preprocessing Techniques and Column Generation Algorithms for *p*-Efficiency. *Journal of Operational Research Society* 59, 1239-1252.

[33] Lejeune M.A., Ruszczyński A. 2007. An Efficient Trajectory Method for Probabilistic Inventory-Production-Distribution Problems. *Operations Research* 55 (2), 378-394.

[34] Luedtke J., Ahmed S. 2008. A Sample Approximation Approach for Optimization with Probabilistic Constraints. *SIAM Journal of Optimization* 19, 674-699.

[35] Luedtke J., Ahmed S., Nemhauser G. 2010. An Integer Programming Approach for Linear Programs with Probabilistic Constraints. *Mathematical Programming* 122 (2), 247-272.

[36] Martinez-Trinidad J. F., Guzmán-Arenas A. 2001. The Logical Combinatorial Approach to Pattern Recognition, an Overview Through Selected Works. *Pattern Recognition* 34 (4), 741-751.

[37] Miller B.L., Wagner H.M. 1965. Chance Constrained Programming with Joint Constraints. *Operations Research* 13, 930-945.

[38] Nemirovski A., Shapiro A. 2006. Convex Approximations of Chance Constrained Programs. *SIAM Journal on Optimization* 17, 969-996.

[39] Prékopa A. 1970. On Probabilistic Constrained Programming. *Proceedings of the Princeton Symposium on Mathematical Programming*. Princeton University Press, 113-138.

[40] Prékopa A. 1973. Contributions to the Theory of Stochastic Programming. *Mathematical Programming* 4, 202-221.

[41] Prékopa A. 1990. Dual Method for a One-Stage Stochastic Programming with Random rhs Obeying a Discrete Probability Distribution. *Zeitschrift of Operations Research* 34, 441-461.

[42] Prékopa A. 1995. *Stochastic Programming*. Kluwer. Boston, MA.

[43] Prékopa A. 2003. Probabilistic Programming Models. Chapter 5 in: *Stochastic Programming: Handbook in Operations Research and Management Science 10.* Eds: Ruszczyński A., Shapiro A. Elsevier Science Ltd, 267-351.

[44] Radeanu S. 1974. *Boolean Functions and Equations*. North-Holland, Amsterdam.

[45] Ruiz-Shulcloper J., Abidi M.A. 2002. Logical Combinatorial Pattern Recognition: A Review. *Transworld Research Networks* 3, 133-176.

[46] Ruszczyński A. 2002. Probabilistic Programming with Discrete Distribution and Precedence Constrained Knapsack Polyhedra. *Mathematical Programming* 93, 195-215.

[47] Ruszczyński A., Shapiro A. 2003. *Stochastic Programming: Handbook in Operations Research and Management Science* 10. Elsevier Science Ltd.

[48] Saxena A., Goyal V., Lejeune M.A. 2010. MIP Reformulations of the Probabilistic Set Covering Problem. *Mathematical Programming* 121 (1), 1-31.

[49] Sen S. 1992. Relaxations for Probabilistically Constrained Programs with Discrete Random Variables. *Operations Research Letters* 11, 81-86.

[50] Torvik V.I., Triantaphyllou E. 2006. Discovering Rules that Govern Monotone Phenomena. In: *Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques.* Eds: Triantaphyllou E., Felici G., Springer, Heidelberg, Germany, 149-192.

[51] Torvik V.I., Triantaphyllou E. 2009. Inference of Monotone Boolean Functions. *Encyclopedia of Optimization*, 1591-1598.

[52] Triantaphyllou E., Felici G. 2006. *Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques.* Springer, Heidelberg, Germany.

[53] Truemper K. 2004. *Design of Logic-Based Intelligent Systems*. Wiley & Sons, Hoboken, NJ.

[54] Urbano R.H., Mueller R.K. 1956. A Topological Method for the Determination of the Minimal Forms of a Boolean Function. *IRE Transactions on Electronic Computers* EC-5, 126-132.

[55] Vapnik V.N. 1998. *Statistical Learning Theory*. Wiley & Sons, USA.