Ann Math Artif Intell (2018) 83:165–181
https://doi.org/10.1007/s10472-018-9572-3

Introducing statistical consistency for infinite chance constraints

Imen Zghidi[1] · **Brahim Hnich**[2,3] · **Abdelwaheb Rebai**[1]

Published online: 11 January 2018

Abstract In this paper, we propose a novel notion of *statistical consistency* for single-stage Stochastic Constraint Satisfaction Problems (SCSPs) in which some of the random variables' support set is infinite. The essence of this novel notion of local consistency is to be able to make an inference in the presence of infinite scenarios in an uncertain environment. This inference would be based on a restricted finite subset of scenarios with a certain confidence level and a threshold tolerance error. The confidence level is the probability that characterizes the extend to which our inference — based on a subset of scenarios — is correct. The threshold tolerance error is the error range that we can tolerate while making such an inference. We propose a novel statistical consistency enforcing algorithm that is based on sound statistical inference; and experimentally show how to prune inconsistent values in the presence of an infinite set of scenarios.

Keywords Infinite chance constraints · Statistical consistency · Constraint propagation

Mathematics Subject Classification (2010) 62F25

The original version of this article was revised: The name "Abdelwahad Rebaii" should be corrected to "Abdelwaheb Rebai". The correct name is now shown here.

✉ Imen Zghidi
zghidi.imen@gmail.com

Brahim Hnich
hnich.brahim@gmail.com

Abdelwaheb Rebai
Abdelwaheb.Rebai@fsegs.rnu.tn

1 Modils Research Lab, FSEG, University of Sfax, Sfax, Tunisia

2 CES, ENIS, University of Sfax, Sfax, Tunisia

3 Department of CS, Monastir University, Monastir, Tunisia

1 Introduction

In most industrial contexts, decisions are made based on incomplete information. Stochastic Constraint Satisfaction Problems (SCSPs) [8, 15, 17] provide a powerful modeling framework for problems in which we are required to take decisions under uncertainty.

An m-stage SCSP [8, 15, 17] is defined as a 7-tuple $\langle V, S, D, P, C, \beta, L\rangle$, where V is a set of decision variables and S is a set of random variables, D is a function mapping each element of V (respectively, S) to a domain (respectively, support) of potential values. In classical SCSPs both decision variable domains and random variable supports are assumed to be finite. P is a function mapping each element of S to a probability distribution for its associated support. C is a set of chance-constraints over a non-empty subset of decision variables and a subset of random variables. β is a function mapping each chance-constraint $h \in C$ to β_h which is a threshold value in the interval $(0, 1]$. $L = [\langle V_1, S_1\rangle, \ldots, \langle V_i, S_i\rangle, \ldots, \langle V_m, S_m\rangle]$ is a list of *decision stages* such that each $V_i \subseteq V$, each $S_i \subseteq S$, the V_i form a partition of V, and the S_i form a partition of S.

For the rest of the paper, we restrict ourselves to single-stage SCSPs. To solve a (single-stage) SCSP, an assignment to the decision variables must be found such that, given random values for random variables, the chance constraints are satisfied in the specified fraction of all possible scenarios. Under the assumption that random variable supports are finite, the solution of a SCSP is, in general, represented by means of a *policy tree* [15]. The arcs in such a policy tree represent values observed for random variables; whereas nodes at each level represent the decisions associated with the different stages.

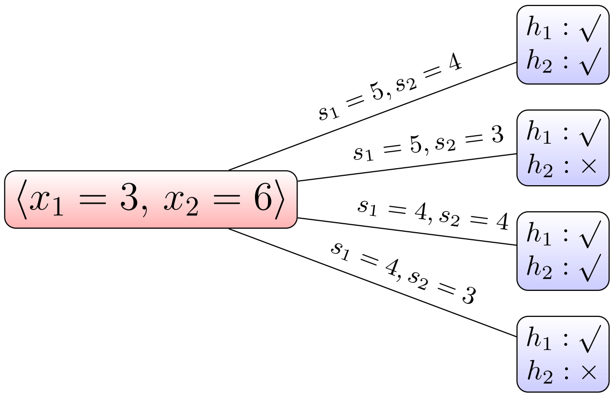
If the random variables' support set is infinite, the number of scenarios would be infinite and the policy tree itself would be infinite. Hence, finding a satisfying policy tree in such cases is impossible in general. We refer to such SCPS as *infinite* SCSPs.

In this paper, we propose a novel notion of *statistical consistency* for single-stage SCSPs in which some of the random variables' support set is infinite. This statistical consistency is crucial if one wants to lift the inference about value consistency in the presence of uncertainty, especially when the set of possible scenarios is infinite. Any constraint solver or search algorithm can use the stochastic inference (through the statistical consistency enforcing algorithm) to directly reason and solve an infinite SCSP. The new notion of statistical consistency is parameterized by a confidence probability, α_c, and an error threshold ϑ. The goal is to be able to make an inference, with confidence probability α_c, about whether a value is consistent or not with respect to the uncertain environment without the error exceeding threshold value ϑ.

The rest of this paper is organized as follows. First, we review the concept of an (α_c, ϑ)-solution to an infinite SCSP in Section 2. Then, We formalize the notion of (α_c, ϑ)-consistency in Section 3. Next, in Section 4, we present our approach based on confidence intervals to infer whether or not a value v is (α_c, ϑ)-consistent; and we empirically validate our approach in Section 5. Then, in Section 6 we show how to enforce (α_c, ϑ)-consistency for a whole infinite SCSP. Before we conclude in Section 8, we present a comparison to related works in Section 7.

2 (α_c, ϑ)-solutions

Let us consider a single-stage SCSP in which $V = \{x_1, x_2\}$ and $S = \{s_1, s_2\}$. The random variable s_1 may take two possible values 4 and 5, each with a probability of 0.5. The random variable s_2 may also take two possible values 3 and 4, each with a probability of 0.5. The



$\langle x_1 = 3, x_2 = 6 \rangle$
$s_1 = 5, s_2 = 4$
$s_1 = 5, s_2 = 3$
$s_1 = 4, s_2 = 4$
$s_1 = 4, s_2 = 3$
$h_1 : \surd$ $h_2 : \surd$
$h_1 : \surd$ $h_2 : \times$
$h_1 : \surd$ $h_2 : \surd$
$h_1 : \surd$ $h_2 : \times$

Fig. 1 A satisfying policy tree with $x_1 = 3$ and $x_2 = 6$

domain of decision variable x_1 is $\{1, 2, 3, 4\}$ whereas that of x_2 is $\{3, 4, 5, 6\}$. We have two chance constraints h_1 and h_2 defined as follows:

$$h_1 : pr\{s_1 \cdot x_1 + s_2 \cdot x_2 \geq 30\} \geq 0.75$$

$$h_2 : pr\{s_2 \cdot x_2 = 12\} \geq 0.5$$

Since this SCSP is a single-stage one, then decision variables x_1 and x_2 must be set to unique values before the random variables are observed. A satisfying policy tree to this SCSP is shown in Fig. 1, in which $x_1 = 3$ and $x_2 = 6$. The first chance constraint is satisfied in all four scenarios and hence the satisfaction probability (i.e. $pr\{s_1 \cdot x_1 + s_2 \cdot x_2 \geq 30\}$) is 1 which is larger than the needed 0.75. The second chance constraint is only satisfied in two scenarios out of four with a satisfaction probability of 0.5, which is just enough to satisfy chance constraint h_2.

Now, assume instead that the random variables s_1 and s_2 are two *continuous* random variables. Assume s_1 takes values following a uniform distribution over the interval $[4, 5]$ and s_2 takes values following a uniform distribution over the interval $[3, 4]$. In this case, the policy tree for the same assignment $x_1 = 3$ and $x_2 = 6$ becomes *infinite*. This renders it impossible, in general, to check whether or not such a policy tree is a satisfying policy tree in a finite amount of time, since we need to check an infinite number of scenarios.

The authors in [12, 13] introduce the notion of (α_c, ϑ)-solutions where α_c[1] is a confidence level and ϑ is a threshold tolerance. Instead of looking for an exact solution, as in the classical m-stage SCSPs, which may not even be possible, we search for a solution that, *with confidence level* α_c, guarantees a satisfaction probability that is no lower than $\beta_h - \vartheta$ for each chance constraint h. The (α_c, ϑ)-solution to an infinite SCSP P is indeed a solution (i.e., a satisfying policy tree) to a restricted version $\hat{P}$ of P in which we only consider a finite subset of the scenarios.

[1]In the original paper α_c is referred to as α. But, to differentiate it later on from the significance level α, we rename it here as α_c.

3 Introducing (α_c, ϑ)-consistency

Constraint propagation techniques are inference methods that help reducing the original CSP into another which is smaller in size [2]. The propagator associated with every constraint, removes — from the domains of the variables in the scope of that constraint — inconsistent values. Inconsistent values are values that do not appear in any solution to that particular constraint. For a constraint c, a value v in the domain of x_i is *generalized arc consistent* (GAC) iff there exist values in the domains of all other variables such that c is satisfied. A constraint c is GAC iff every value in the domain of every variable is GAC. A CSP is GAC iff all constraints are GAC.

The authors in [8] extend the notion of GAC for chance constraints in SCSPs. Since, in this paper we restrict ourselves to single-stage SCSPs, we present here a simplified definition of GAC for chance constraints. Let h be a chance constraint constraining a subset of decision variables $X_h \subseteq V$ and a non-empty subset of random variables $S_h \subseteq S$:

$$h : pr\{C\} \geq \beta_h$$

Let Ω_h denote the set of scenarios constructed from the random variables S_h. Let A be an assignment of the decision variables in X_h. Let T_h^A be the policy tree restricted to h in which the decision variables X_h take the values in assignment A. In other words, each arc in this policy tree is a possible scenario $s \in \Omega_h$ whose probability is denoted by $pr(s)$. Let C_h^s denote the constraint C in which the random variables are replaced with the actual values in s. Let $\overline{C_h^s}$ denote the expression in which the decision variables in C_h^s take the values specified in assignment A. Let the boolean B_h^s be 1 if expression $\overline{C_h^s}$ is satisfied, 0 otherwise.

Definition 1 Given a chance constraint h. A value v in the domain of $x \in X_h$ is GAC iff there exists an assignment A in which $x = v$ and

$$\sum_{s \in \Omega_h} B_h^s \cdot pr(s) \geq \beta_h$$

Example 1 Consider the following chance constraint involving two binary decision variables (x and y) and a random variable r with finite support set:

$$h : pr\{max(2x, 1 - y) \leq r\} \geq \beta_h = 0.75$$

where the support set of r is $\{(0, 0.25), (1, 0.25), (2, 0.25), (3, 0.25)\}$, i.e., r can take values between 0 and 3 with equal probabilities. The policy tree associated with h comprises 4 arcs, one for each scenarios s where each scenario is a possible realization of the random variable r with probability $pr(s) = 0.25$.

Consider the assignment $A_1 : \langle x = 0, y = 0\rangle$. The expression $max(2.0, 1 - 0)$ is equal to 1. So, $1 \leq r$ is satisfied in two out of the four scenarios. Hence, the policy tree in which decision variables take the values in assignment A_1 is not a satisfying policy tree. Indeed the satisfaction probability of each assignment is shown in Fig. 2.

Based on Fig. 2 and Definition 1, value 1 in the domain of x is GAC and so are all values in the domain of y whereas value 0 in the domain of x is inconsistent.

In other words, a value v in the domain of $x \in X_h$ is GAC iff there exists an assignment A in which $x = v$ and T_h^A is a satisfying policy tree. A chance constraint h is GAC iff every value in the domain of every variable in X_h is GAC. A SCSP is GAC iff every chance constraint is GAC.

Fig. 2 The true mean of assignments of h wrt to $\beta_h = 0.75$

Assignment	True mean
$A_1 : \langle x = 0, y = 0\rangle$	$pr\{C(x = 0, y = 0, r)\} = 0.5$
$A_2 : \langle x = 0, y = 1\rangle$	$pr\{C(x = 0, y = 1, r)\} = 0.25$
$A_3 : \langle x = 1, y = 0\rangle$	$pr\{C(x = 1, y = 0, r)\} = 0.75$
$A_4 : \langle x = 1, y = 1\rangle$	$pr\{C(x = 1, y = 1, r)\} = 0.75$

Note that the above consistency definition for chance constraints assumes that all the random variables in S_h have *discrete finite* support. If at least one random variable in S_h has an *infinite* support, then Ω_h would become an infinite set of scenarios and therefore, for any assignment A, the corresponding policy tree T_h^A would have an infinite number of arcs. Indeed, the sum that tests whether a value v is GAC or not would be an infinite sum. So, how to proceed, in such a situation, is a core question that this paper tries to address.

Inspired by the concept of (α_c, ϑ)-solutions proposed in [12], we introduce the notion of *(α_c, ϑ)-consistency* for *infinite chance constraints* of single-stage SCSPs.

Due to the infinite number of scenarios that we are required to consider , it becomes practically impossible to establish whether or not a value v is GAC. An alternate solution would be to establish whether a value v is consistent, with confidence α_c, and error threshold ϑ. By having the parameter α_c, we control how confident we want to be in our judgment; whereas ϑ controls the level of error we are willing to tolerate. Thus, we are now ready to introduce the novel definition of (α_c, ϑ)-consistency as follows.

Definition 2 Given an infinite chance constraint h. A value v in the domain of $x \in X_h$ is (α_c, ϑ)-GAC iff there exists an assignment A in which $x = v$ and, with confidence level α_c,

$$\sum_{s \in \Omega_h} B_h^s \cdot pr(s) \geq \beta_h - \vartheta$$

Note that in $100\alpha_c\%$ of the times, a value v that is truly consistent is detected as so according to this definition with tolerated error ϑ.

Example 2 Consider the following infinite chance constraint involving two binary decision variables (x and y) and a random variable r with infinite support:

$$h : pr\{C(x, y, r)\} \geq \beta_h$$

The policy tree associated with h comprises an infinite number of scenarios where each scenario is a possible realization of the random variable r. It is impossible, for a given assignment A, to find a satisfying policy tree in which decision variables take values in A or show that none exists since we need to compute an infinite sum as stated in Definition 1. Suppose, nevertheless, that somehow we know the true mean of each assignment as shown in Fig. 3.

Based on Fig. 3 and Definition 1, value 0 in the domain of x and value 1 in the domain of y are GAC whereas value 1 in the domain of x and value 0 in the domain of y are inconsistent. Note, however, that it is impossible in general to know the true mean of each assignment since the policy tree is infinite. Therefore, what Definition 2 proposes is to be able to infer, with confidence probability α_c and an error threshold ϑ, whether value v is consistent or not. I.e., there exists an assignment A in which $x = v$ that — with confidence probability α_c — verifies the sum expression in Definition 2. Another way to express what it means for a value to be (α_c, ϑ)-GAC is as follows. Suppose we generate a sampled policy tree for h in which the arcs take sampled values from the random variable using some

Fig. 3 The true mean of assignments of h wrt to $\beta_h = 0.5$

Assignment	True mean
$A_1 : \langle x = 0, y = 0\rangle$	$pr\{C(x = 0, y = 0, r)\} = 0.48$
$A_2 : \langle x = 0, y = 1\rangle$	$pr\{C(x = 0, y = 1, r)\} = 0.5$
$A_3 : \langle x = 1, y = 0\rangle$	$pr\{C(x = 1, y = 0, r)\} = 0.42$
$A_4 : \langle x = 1, y = 1\rangle$	$pr\{C(x = 1, y = 1, r)\} = 0.4$

sampling strategy[2]. Now, value v is (α_c, ϑ)-GAC iff there exists a satisfying policy tree to the sampled infinite chance constraint h in which decision variables take the values in A and, with probability α_c, is truly a satisfying policy tree to the infinite chance constraint h.

Now consider two values v and w in the domain of $x \in X_h$. Suppose v is (α_c, ϑ)-GAC and w is also (α_c, ϑ)-GAC. So, there exists a satisfying policy tree (P_v) to a sampled infinite chance constraint h in which decision variables take the values in some assignment A (in which $x = v$) and with probability α_c is truly a satisfying policy tree to the infinite chance constraint h. Similarly, there exists a satisfying policy tree (P_w) to a sampled infinite chance constraint h in which decision variables take the values in some assignment B (in which $x = w$) and with probability α_c is truly a satisfying policy tree to the infinite chance constraint h. However, the confidence probability of simultaneously having P_v and P_w as truly satisfying policy trees with respects to the infinite chance constraint h is reduced to $\alpha_c \cdot \alpha_c$. Thus, v and w are not simultaneously (α_c, ϑ)-GAC, since the errors accumulate; they are only guaranteed to be (α_c^2, ϑ)-GAC.

Therefore, in our case, we need to worry about multiple statements since our definition of (α_c, ϑ)-consistency is probabilistic in nature. Thus, for an infinite chance constraint we propose the following novel definition:

Definition 3 An infinite chance constraint h is (α_c, ϑ)-GAC iff *simultaneously* every value in the domain of every variable in X_h is (α_c, ϑ)-GAC.

Similarly, when we consider a SCSP composed of multiple infinite chance constraints, we propose the following definition:

Definition 4 An infinite SCSP is (α_c, ϑ)-GAC iff *simultaneously* every infinite chance constraint is (α_c, ϑ)-GAC

The new statistical consistency is inspired by (α_c, ϑ)-solutions but is fundamentally different in two aspects: (1) It is a local consistency roperty as opposed to the concept of (α_c, ϑ)-solution which is a global property of the solution. Thus, one can reason about each infinite chance constraint independently in order to make a global inference; and (2) such a statistical consistency notion can pave the way for a new generation of search algorithms that reason directly about infinite SCSPs instead of relying on reformulating the infinite SCSP into an approximate finite SCSP to be solved using the techniques developed for finite SCSPs.

4 Statistical inference rules via confidence intervals

For the rest of this section, we suppose we are given a chance constraint $h : pr\{C\} \geq \beta_h$ over an infinite set of scenarios Ω_h, a specific assignment A in which $x = v$, a confidence

[2]To be discussed later.

level α_c, and an error threshold ϑ. Our goal is to answer the question of whether or not v is (α_c, ϑ)-consistent with respect to assignment A only. Recall that a value v in the domain of $x \in X_h$ is consistent iff there exists an assignment A in which $x = v$ and

$$\sum_{s\in\Omega_h} B_h^s \cdot pr(s) \geq \beta_h$$

Without loss of generality, assume that the probability of each scenario s, $pr(s)$, is the same. Then, the left hand side of the equation becomes:

$$\frac{\sum_{s\in\Omega_h} B_h^s}{|\Omega_h|} = \mu$$

Which can be viewed as the proportion of 1's (i.e., $B_h^s = 1$) in an *infinite sequence* of zeros and ones. This proportion p, where $p = \mu$, is indeed unknown. But, how can we estimate such an unknown quantity?

Let us consider the following statistical experiment $Y = (Y_1, Y_2, \ldots, Y_n)$ consisting of n repeated trials. In each trial we select a random scenario from Ω_h using a sampling strategy like simple random sampling with replacement. Each trial Y_i of the experiment has two possible outcomes labeled "success" when $B_h^s = 1$, "failure" otherwise. The probability of success μ is the same for each trial (but unknown), and the trials are independent and identically distributed. Such an experiment is known as the *binomial experiment* [10]. Thus, the satisfaction probability (μ) can be estimated by repeatedly observing the behavior of the random variable in a sequence of Bernoulli trials. An exact confidence interval (CI) associated with the binomial distribution can be constructed from the observed data of a specific number of trials in such a way that it covers μ with a certain confidence level α. There are several ways to compute a CI for a binomial distribution [1, 5, 16]. But, as in [12, 13] we opt for the exact Clopper-Pearson CI [5] because this method uses the Binomial distribution in order to build the interval from the observed data, rather than an approximation. Furthermore, given a confidence probability, an important property of CIs — related to the estimation of the "success" probability associated with a Bernoulli trial — is the ability to mathematically derive the minimum number of samples that should be observed in order to produce a confidence interval of a given width.

Now, given a scenario sample $S \subseteq \Omega_h$ of size n generated as described above, let the sample mean κ be $\frac{\sum_{s\in S} B_h^s}{n}$. Let us consider the binomial proportion CI $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$, constructed using sample S, with confidence level α_c and width being ϑ. We will discuss later how to compute the sample size needed to guarantee such a CI with confidence level α_c and width ϑ. Thus, μ is covered by CI in $\alpha_c\%$ of the times.

Let $min = \kappa - \frac{\vartheta}{2}$ and $max = \kappa + \frac{\vartheta}{2}$. Recall that value v is (α_c, ϑ)-consistent iff, with confidence level α_c, we have $\mu \geq \beta_h - \vartheta$. We are now ready to make the following observations.

Observation 1 If $\beta_h < min$, then it is also true that $\beta_h - \vartheta < min$. Since μ is covered by the CI $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$ with coverage probability α_c and in the worst case (when $\mu = min$) we still have $\mu \geq \beta_h - \vartheta$, then v is (α_c, ϑ)-consistent in this case.

Observation 2 If $\beta_h \in [min, max]$, then $\beta_h - \vartheta \leq min$ since $min = max - \vartheta$. This means that $\mu \geq \beta_h - \vartheta$ in $\alpha_c\%$ of the times because the coverage probability of μ by CI is α_c. Thus, v is (α_c, ϑ)-consistent in this case as well.

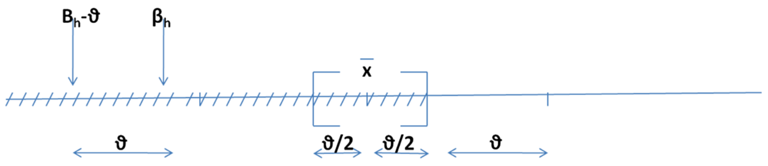


Fig. 4 The ”clear accept” case

So, based on Observation 1 and Observation 2, when $\beta_h \le max$ and the width of the CI is ϑ, we can safely classify v as being (α_c, ϑ)-consistent. See Fig. 4 for an illustration.

Theorem 1 *For any chance constraint h, for any assignment A in which $x = v$, for any ϑ, and for any α_c, for all values of $\beta_h \le \kappa + \frac{\vartheta}{2}$, using a CI $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$ one can properly classify v as being (α_c, ϑ)-consistent.*

Proof Follows immediately from Observation 1 and Observation 2 □

Our third observation is as follows:

Observation 3 If $\beta_h > max + \vartheta$, then it is also true that $\beta_h - \vartheta > max$. Thus, since μ is covered by the CI $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$ with coverage probability α_c and even in the worst case when $\mu = max$, we still have $\mu < \beta_h - \vartheta$, one can properly classify v as being (α_c, ϑ)-inconsistent in this case. See Fig. 5 for an illustration.

Theorem 2 *For any chance constraint h, for any assignment A in which $x = v$, for any ϑ, and for any α_c, for all values of $\beta_h > \kappa + \frac{3\vartheta}{2}$, using a CI $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$ one can properly classify v as being (α_c, ϑ)-inconsistent.*

Proof Follows immediately from Observation 3 □

Finally, we identify the case where we fail to make an inference with confidence probability at least α_c:

Observation 4 If $max < \beta_h \le max + \vartheta$, then $\beta_h - \vartheta \in]\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$. Since, with coverage probability α_c, μ is covered by $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$, it is impossible in this case to make a correct classification with probability α_c because we can either have $\mu \ge \beta_h - \vartheta$ or $\mu < \beta_h - \vartheta$. This is the critical case in which we fail to make a correct inference with probability α_c or higher. See Fig. 6 for an illustration.

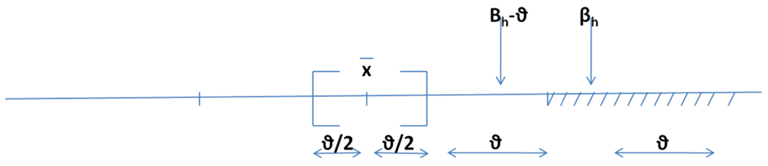


Fig. 5 The ”clear reject” case

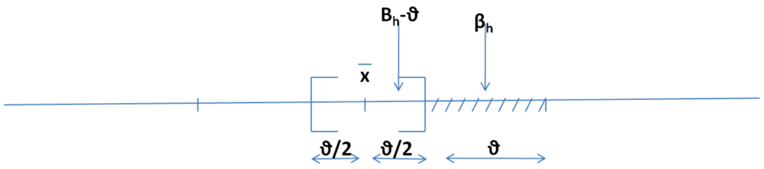


Fig. 6 The critical case

In fact, the following theorem shows that there exists always a situation in which we are unable to make a correct inference.

Theorem 3 *For any chance constraint h, for any assignment A in which $x = v$, for any ϑ, and for any α_c, there exists always a value for β_h such that it is impossible, using the CI approach, to firmly decide whether or not (α_c, ϑ)-consistent.*

Proof Follows immediately from Observation 4 by setting $\beta_h \in]\kappa + \frac{\vartheta}{2}, \kappa + \frac{3\vartheta}{2}]$ □

In this critical case, one possible way to solve the problem is to either: (1) classify v as being (α_c, ϑ)-consistent while it may be inconsistent (increase the chances of type II errors); or (2) classify v as being (α_c, ϑ)-inconsistent while it may be consistent (increase the chances of type I errors). However, this approach makes the inference not completely sound. So is there a better way to remedy this situation?

In fact, in the critical case one might still guarantee a sound inference but at the cost of increasing ϑ. Indeed, we can classify v as being $(\alpha_c, 2 \cdot \vartheta)$-consistent as explained in the following Observation:

Observation 5 If $max < \beta_h \leq max + \vartheta$, then $\beta_h - 2\vartheta \leq \kappa - \frac{\vartheta}{2}$. Since, with coverage probability α_c, μ is covered by $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$, then we can correctly classify v as $(\alpha_c, 2 \cdot \vartheta)$-consistent.

Theorem 4 *For any chance constraint h, for any assignment A in which $x = v$, for any ϑ, and for any α_c, for all values of $max < \beta_h \leq max + \vartheta$, using a CI $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$ one can properly classify v as being $(\alpha_c, 2 \cdot \vartheta)$-consistent.*

Proof Follows immediately from Observation 5 □

Thus, a sound inference method based on CIs is as follows:

Step 1: Find a large enough sample size n and construct a Clopper-Pearson CI $[\kappa - \frac{\vartheta}{2}, \kappa + \frac{\vartheta}{2}]$ so that its coverage probability is α_c;
Step 2: We have three mutually exclusive cases:

Consistent: If $\beta_h \leq \kappa + \frac{\vartheta}{2}$, then classify v as (α_c, ϑ)-consistent;
Inconsistent: If $\beta_h > \kappa + \frac{3\vartheta}{2}$, then classify v as (α_c, ϑ)-inconsistent;
Critical: If $\beta_h \in]\kappa + \frac{\vartheta}{2}, \kappa + \frac{3\vartheta}{2}]$, then classify v as $(\alpha_c, 2 \cdot \vartheta)$-consistent.

The above inference algorithm is able to make a sound statistical inference which guarantees a value v being correctly classified with confidence α_c and an error threshold varying between ϑ and $2 \cdot \vartheta$. So, if we wish to *always* guarantee an (α_c, ϑ)-consistency level, we reset the value ϑ to $\frac{\vartheta}{2}$ instead.

We are still left with the problem of computing a sample size n that is large enough to guarantee a CI of width ϑ or less. The solution we propose is to start with an initial theoretical sample size, say n_1 computed as per Definition 2 in [12], which we restate here:

Definition 5 ([12]) n_1 is computed as the minimum value for which

$$\max(p_{\text{ub}}^{\beta_h} - \beta_h, \beta_h - p_{\text{lb}}^{\beta_h}) \leq \vartheta,$$

where $p_{\text{lb}}^{\beta_h}$ and $p_{\text{ub}}^{\beta_h}$ are the single-sided Clopper-Pearson confidence interval bounds for a confidence probability α_c, and *round*$(\beta_h n)$ "successes" in n trials; *round*() approximates the value to the nearest integer.

If the CI's width is still larger than ϑ in practice, we increase the sample size by one and keep doing so untill we reach a sample size, say $n_k > n_1$, after k iterations in which the width is ϑ or smaller.

The inference algorithm is computationally efficient. The first step is the one that consumes most of the computation since the second one takes a constant time. Now let us consider the first step and assume we start with an initial theoretical sample size n_1 — computed in constant time — and compute the large enough sample size n_k in k iterations where $k = n_k - n_1$. In each iteration t, after we generate a sample of size t (in order $O(t)$-time) we compute a CI and check its width in constant time. Thus, each step is in $O(n_k)$ in the worst-case. So overall and since we have k iterations this step is in $O(k \cdot n_k)$. Since the initial theoretical sample size n_1 is not that far off from the large enough sample size n_k (as will be shown in the experiments section), this step in practice is quite efficient. Thus, our inference method is indeed a suitable light-weight one that can be used effectively to achieve the long-term goal of designing and implementing a new breed of constraint solvers: *statistical constraint solvers*. These solvers would have the following novel key features:

1. *Infinite SCSPs* are *directly modeled and solved* as such instead of being reformulated into finite SCSPs to be solved using traditional constraint solvers designed to handle deterministic CSPs;
2. A new library of *infinite chance constraint types* each with its own dedicated *statistical propagator*. Each propagator would enforces some form of *statistical consistency*, say (α_c, ϑ)-consistency locally by relying an inference methods like the one we propose in this paper;

Fig. 7 Sample size

ϑ	Initial theoretical n	Large enough n
0.01	28000	39000
0.009	34000	48000
0.008	42500	61000
0.007	55500	79000
0.005	109000	155000
0.001	2750000	3900000

Fig. 8 The correct classification rate (CCR) for $\mu = 0.5$, $\alpha_c = 0.95$, and $\vartheta = 0.01$

	CCR	True classification
$\beta_h = 0.49$	**100%**	Consistent
$\beta_h = 0.5$	**98.3%**	Consistent
$\beta_h = 0.51$	**97.2%**	Consistent (Critical case)
$\beta_h = 0.52$	**97.8%**	Inconsistent

3. Integrating statistical components that would allow for sampling and building CIs such as the $\mathcal{R}$ system which is an environment within which many classical and modern statistical techniques have been implemented.
4. A tree search algorithm that solves directly the infinite SCSPs by removing inconsistent values deduced using the statistical propagators at each node of the search tree.

Such statistical constraint solvers do not exist currently. The novel notion of statistical consistency, alongside the statistical inference method presented in this paper, represent the first step toward the larger goal of designing and implementing such statistical constraint solvers in the future.

5 Validating the approach based on confidence intervals

To implement our inference method, we mainly used the $\mathcal{R}$ language and environment for statistical computing and graphics [11]. It is a an environment which was developed at Bell Laboratories by John Chambers and colleagues. $\mathcal{R}$ is an environment within which many classical and modern statistical techniques have been implemented. To test our method, we set our true mean μ to 0.5 of a Bernoulli trial. We vary β_h to take values in

$$\{0.48, 0.49, 0.5, 0.51, 0.52, 0.53\}$$

and

$$\vartheta \in \{0.01, 0.009, 0.008, 0.007, 0.005, 0.001\}.$$

For each configuration $\langle \beta_h, \alpha_c, \vartheta \rangle$ we run 1000 experiments and record the number of times c we classify value v as (α_c, ϑ)-consistent and the number of times f we classify v as (α_c, ϑ)-inconsistent. Now, if the true mean μ is greater than or equal to $\beta_h - \vartheta$, we compute the Correct Classification Rate (CCR) of our method as

$$\frac{c}{c+f} = \frac{c}{1000}$$

Note that, $\frac{f}{f+c}$ represents the ratio of wrong classification in this case. This is the ratio of type I errors since we are rejecting a consistent value v. If, however, the true mean μ is strictly smaller than $\beta_h - \vartheta$, then CCR is computed as

$$\frac{f}{c+f} = \frac{f}{1000}$$

Fig. 9 The correct classification rate (CCR) for $\mu = 0.5$, $\alpha_c = 0.95$, and $\vartheta = 0.009$

	CCR	True classification
$\beta_h = 0.49$	**100%**	Consistent
$\beta_h = 0.5$	**97.6%**	Consistent
$\beta_h = 0.51$	**98.7%**	Consistent (Critical case)
$\beta_h = 0.52$	**99%**	Inconsistent

Fig. 10 The correct classification rate (CCR) for $\mu = 0.5$, $\alpha_c = 0.95$, and $\vartheta = 0.008$

	CCR	True classification
$\beta_h = 0.49$	**100%**	Consistent
$\beta_h = 0.5$	**97.8%**	Consistent
$\beta_h = 0.51$	**99.7%**	Consistent (Critical case)
$\beta_h = 0.52$	**100%**	Inconsistent

Note that, $\frac{c}{f+c}$ represents the ratio of wrong classification in this case. This is the ratio of type II errors since we are accepting an inconsistent value v.

Indeed, CCR is the most important indicator of whether or not the confidence interval approach is effective in correctly classifying (α_c, ϑ)-consistent values or not.

In Fig. 7, we show the theoretical sample size n needed for different values of ϑ to guarantee a 0.95 confidence level as per Definition 2 in [12]. We also show the sample size dynamically found that guarantees the desired width of the CI. We notice that, in practice, the sample size which is large enough to achieve the desired CI width is larger than the theoretical one found as per Definition 2 in [12]; but not by more than 42% on average.

In Figs. 8, 9, 10, 11, 12, and 13, we present the results of our experiments when $\alpha_c = 0.95$ for various values of μ, β_h, and ϑ. When for a given β_h, and ϑ we are in the critical case (i.e., $\beta_h \in]\kappa + \frac{\vartheta}{2}, \kappa + \frac{3\vartheta}{2}]$), we annotate that in the corresponding row in the table of results as ”critical case”.

- Figs. 8, 9, 10, and 11 do represent situations in which we experience the three cases: ”Consistent”, ”Inconsistent”, and ”Critical”. The results are inline with the theory, and we do achieve a 95% or above CCR in all cases of ”Consistent” and ”Inconsistent”.
- Figs. 12 and 13 do represent situations in which we experience only the two cases: ”Consistent” and ”Inconsistent”. The results are inline with the theory and we do achieve a 95% or above CCR in all cases of ”Consistent” and ”Inconsistent”.
- As the error threshold value ϑ gets smaller, the sample size required by our method, and computed dynamically, increases significantly. As a positive consequence, however, the width of the CIs get smaller; and hence we avoid being in the ”Critical” case.

In summary, the experimental results confirm that the proposed method does achieve indeed a 95% CCR or above as expected in theory and does so by guaranteeing an error threshold of no more than $2 \cdot \vartheta$ (in critical case) or ϑ otherwise.

6 Enforcing (α_c, ϑ)-consistency

So far, we have restricted our analysis to just one assignment A in which $x = v$ of a single infinite chance constraint. With respect to this assignment, we looked at how to infer whether value v is (α_c, ϑ)-consistent or not. In this section, we consider a single-stage infinite SCSP and use it to showcase how it would be solved directly. Our method is based on the techniques presented in this paper rather than on reformulation, done by all other

Fig. 11 The correct classification rate (CCR) for $\mu = 0.5$, $\alpha_c = 0.95$, and $\vartheta = 0.007$

	CCR	True classification
$\beta_h = 0.49$	**100%**	Consistent
$\beta_h = 0.5$	**97.6%**	Consistent
$\beta_h = 0.51$	**99.8%**	Consistent (Critical case)
$\beta_h = 0.52$	**100%**	Inconsistent

Fig. 12 The correct classification rate (CCR) for $\mu = 0.5$, $\alpha_c = 0.95$, and $\vartheta = 0.005$

	CCR	True classification
$\beta_h = 0.49$	**100%**	Consistent
$\beta_h = 0.5$	**97.5%**	Consistent
$\beta_h = 0.51$	**97.8%**	Inconsistent
$\beta_h = 0.52$	**100%**	Inconsistent

existing methods in the literature. We also point out to possible challenges and drawbacks of the proposed method.

In particular, we consider the question of how to make a single infinite chance constraint (α_c, ϑ)-GAC, and also how to make a whole infinite SCSP (α_c, ϑ)-GAC. This would simulate the process of constraint propagation that would be performed if we had a statistical constraint solver that directly solves an infinite SCSP.

The approach we have presented in the previous sections treats each value v separately wrt to a given assignment A. It is indeed able to detect whether each value is (α_c, ϑ)-consistent wrt a given assignment. But, if each value is (α_c, ϑ)-consistent, how about all values considered simultaneously in an infinite chance constraint, in more than one infinite chance constraint, and in the whole problem?

Recall that an infinite chance constraint h is (α_c, ϑ)-GAC iff *simultaneously* every value in the domain of every variable in X_h is (α_c, ϑ)-GAC. Let us consider the cross product of all the domains of the decision variables in X_h, i.e. our assignment space for chance constraint h denoted by $\mathcal{A}$. Using, the approach outlined in the previous section, for each value v in the domain of every decision variable $x \in X_h$, one of these two outcomes is possible:

- There exists an assignment $A \in \mathcal{A}$ for which value v is (α_c, ϑ)-consistent;
- For every assignment $A \in \mathcal{A}$, value v is (α_c, ϑ)-inconsistent.

A naive approach is to simply consider each value separately and if it is (α_c, ϑ)-inconsistent, we prune it. But, the problem is that errors will accumulate. Overall, we may not achieve a confidence level of α_c when we consider multiple inferences. Let us illustrate this situation by the following experiment. Consider the single-stage infinite SCSP shown in Fig. 14 where we have three binary decision variables, and two continuous random variables r_1 and r_2, and three infinite chance constraints.

Assume that $\beta_1 = \beta_2 = \beta_3 = 0.5$, $\vartheta = 0.01$, and $\alpha_c = 0.95$. Assume further that, for each chance constraint, we know the true mean for each assignment. Hence, we are able to precisely know whether or not a certain assignment satisfies the chance constraint. In Fig. 15, we show the consistent and inconsistent assignments for chance constraint h_1.

Based on Fig. 15, value 0 in the domain of x and value 1 in the domain of y are truly consistent; but value 1 in the domain of x and value 0 in the domain of y are truly inconsistent. Similarly we set the consistent and inconsistent assignments for h_2 and h_3 so that for h_2 value 1 in the domain of y and value 0 in the domain of z are truly consistent, but value

Fig. 13 The correct classification rate (CCR) for $\mu = 0.5$, $\alpha_c = 0.95$, and $\vartheta = 0.001$

	CCR	True classification
$\beta_h = 0.49$	**100%**	Consistent
$\beta_h = 0.5$	**97.2%**	Consistent
$\beta_h = 0.51$	**100%**	Inconsistent
$\beta_h = 0.52$	**100%**	Inconsistent

Fig. 14 A single-stage SCSP

Constraints:
$h_1 : pr\{C(x, y, r_1, r_2)\} \geq \beta_1$
$h_2 : pr\{C(y, z, r_1, r_2)\} \geq \beta_2$
$h_3 : pr\{C(x, z, r_1, r_2)\} \geq \beta_3$
Decision variables:
$x, y, z \in \{0, 1\}$
Random variables:
r_1, r_2: has infinite support
Stage structure:
$V_1 = \{x, y, z\}$
$S_1 = \{r_1, r_2\}$
$L = [\langle V_1, S_1 \rangle]$

0 in the domain of y and value 1 in the domain of z are truly inconsistent, and for h_3 value 0 in the domain of x and value 0 in the domain of z are truly consistent, but value 1 in the domain of x and value 1 in the domain of z are truly inconsistent.

Therefore, any sound algorithm that enforces (α_c, ϑ)-consistency should *simultaneously* prune 1 from the domain of x, value 0 from the domain of y, and value 1 from the domain of z in α_c% of the times while allowing an error threshold of ϑ. But, if we enforce (α_c, ϑ)-consistency for each value independently, what is the overall CCR achieved for one chance constraint (namely h_1), two chance constraints (say h_1 and h_2 simultaneously), and the whole problem (namely h_1, h_2, and h_3 simultaneously)?

Our experimental setup is as follows. We run 1000 experiments where each experiment is as follows:

1. For each assignment $A_i \in \{A_1, \ldots, A_{12}\}$, since $\alpha_c = 0.95$ and $\vartheta = 0.01$, we generate an independent sample of size $n = 39000$[3].
2. For each value v in the domain of every decision variable (x, y, and z), if, for any assignment A_i in which v appears, our method in the previous section classifies v as (α_c, ϑ)-inconsistent, we prune v from the domain of the decision variable.
3. The overall number of correct classification is updated as follows:

Correct classification wrt to h_1: we increment the number of correct classification wrt to h_1 by one if we simultaneously prune value 1 from the domain of x and value 0 from the domain of y.

Correct classification wrt to h_1 and h_2: we increment the number of correct classification wrt to h_1 and h_2 by one if we simultaneously prune value 1 from the domain of x, value 0 from the domain of y, and value 1 from the domain of z.

Correct classification wrt to h_1, h_2, and h_3: we increment the number of correct classification wrt to h_1, h_2, and h_3 by one if we simultaneously prune value 1 from the domain of x, value 0 from the domain of y, and value 1 from the domain of z. Note that this case is different from the previous case because in the previous case, only

[3]Determined from the previous experiments

Assignment	True mean	True classification
$A_1 : \langle x = 0, y = 0\rangle$	$pr\{C(x = 0, y = 0, r_1, r_2)\} = 0.48$	Inconsistent
$A_2 : \langle x = 0, y = 1\rangle$	$pr\{C(x = 0, y = 1, r_1, r_2)\} = 0.5$	Consistent
$A_3 : \langle x = 1, y = 0\rangle$	$pr\{C(x = 1, y = 0, r_1, r_2)\} = 0.42$	Inconsistent
$A_4 : \langle x = 1, y = 1\rangle$	$pr\{C(x = 1, y = 1, r_1, r_2)\} = 0.4$	Inconsistent

Fig. 15 Consistent and inconsistent assignments in h_1 wrt to $\beta_1 = 0.5$ and $\vartheta = 0.01$

assignments of h_1 and h_2 are used to make the inference; whereas, in this case, all assignments are used in the inference process.

Finally, the CCR wrt to h_1 is the total number of correct classification wrt to h_1 divided by 1000. Similarly, we compute the CCR wrt h_1 and h_2 and the CCR wrt h_1, h_2, and h_3.

The results of the experiments are shown in Fig. 16. Like expected, as $|\mathcal{A}|$ increases, so does the probability of misclassifying values. So, how to cure this problem? A conservative approach is to apply the Bonferoni correction to increase the individual confidence level [6]. Indeed, if we wish to achieve an overall confidence level of 0.95, the adjusted individual confidence level with 10 assignments — using the Bonferroni correction — would be 0.995. Thus, we need to compute our sample size for this adjusted confidence level. In practice, however, it seems that by slightly increasing our sample size for an individual value, we reduce the errors significantly and hence improve the overall confidence level. The reasons are: (1) the Clopper-Pearson CI is a conservative one and achieves a higher confidence level in practice; and (2) since we insist on computing CI's whose width is no larger than ϑ, we end up using a sample size larger than needed to achieve a confidence level α_c, and hence in practice we do achieve a higher confidence level in practice. Furthermore, since there is very little computational cost for using a higher sample size in our method, it is still practical to do so. For instance, in the previous experiments where for each value we maintain (α_c, ϑ)-consistency for $\alpha_c = 0.95$ and $\vartheta = 0.01$, the sample size found and used by our method was $n = 39000$; but if we increase it to $n = 50000$, we get the results shown in Fig. 17, where the desired CCR is achieved in all cases.

7 Related works

Confidence-based reasoning (CBR) was originally introduced in [14] and further developed in [12] which proposes novel concepts and techniques that can be employed to find approximate solutions that feature given statistical properties. In [3, 4, 7, 15] the authors used sampling in order to reduce the number of scenarios considered for a given SCSP and produce a solution within a reasonable time, but with no guarantees of the statistical properties of the approximate solutions found. In stochastic programming, the Sample Average Approximation (SAA) method in [9]. SSA is a Monte Carlo simulation-based approach to stochastic discrete optimization problem which reformulates the problem by replacing the actual distribution of random variables in the problem by an empirical distribution obtained

CCR wrt to h_1	CCR wrt h_1 and h_2	CCR wrt h_1, h_2, and h_3
97.1%	91.4%	88.8%

Fig. 16 The correct classification rate (CCR) for $\mu = 0.5$, $\alpha_c = 0.95$, and $\vartheta = 0.001$

	CCR wrt to h_1	CCR wrt h_1 and h_2	CCR wrt h_1, h_2, and h_3
$n = 50000$	99.6%	96.1%	95.7%

Fig. 17 CCR for multiple chance constraints in practice with increased sample size

via sampling. The reformulated problem is then solved and the procedure is repeated multiple times until a given termination criterion is satisfied. As pointed out by [12], the concepts of confidence-based reasoning ((α, ϑ)-solution and (α, ϑ)-solution set) are novel and differ from those of SAA in which one can not fix any a priori tolerated error threshold ϑ. Furthermore, while CBR is based on the exact Clopper-Pearson confidence interval, SAA uses conservative bounds such as Chernoff's or Hoeffding's inequalities. In addition, determining the sample size using the CBR approach (Definition 2 in [12]) is independent of the number of assignments in the feasible region.

All these methods, however, reformulate the problem via sampling in order to find an approximate solution. All these methods, in order to solve the sampled SCSP, rely on the whole policy tree of the entire sampled SCSP. In fact the policy tree comprises a number of scenarios that is exponential in the size of random variable domains. Hence, as the size of the policy tree grows larger, all these methods face a scalability problem (mainly memory usage one) and thus hindering the application of these methods to solve large-scale SCSPs [12]. This paper proposes the first step toward a completely novel and orthogonal approach toward solving infinite SCSPs by showing how to lift the local inference methods — the core component of most constraint solvers — from finite to infinite chance constraints. Thus, opening the doors toward implementing statistical constraint solvers that would directly solve infinite SCSPs instead of relying on reformulation. The statistical inference methods proposed here are *local*, i.e., each statistical propagator associated with each infinite chance constraint would use them to reason locally using only samples of random variables within its scope. Hence, the size of the "local" policy tree — constructed from the samples— would not face any memory problems due to: (1) the size of the "local" policy tree being much smaller than the size of the whole policy tree since it only depends on the random variables within the scope of the individual infinite chance constraint, as opposed to all random variables within the SCSP; and (2) since these methods are local, we apply them one by one through the propagation process at each node of the search tree and, after the inference is performed, the memory is freed up.

8 Conclusion

Sound and complete reasoning about infinite chance constraints in the presence of random variables with infinite support sets requires considering an infinite set of scenarios which in turn is impossible in practice. In this paper, inspired by the concept of (α_c, ϑ)-solutions proposed in [12], we introduce , for the first time, the notion of statistical consistency and in particular (α_c, ϑ)-consistency for infinite chance constraints for single-stage SCSPs in which at least one random variable has an infinite support. This statistical consistency is based on two key parameters α_c and ϑ in order to relax the completeness and precision of such an inference in practice. With α_c, the definition sets a level of confidence about the inference one wishes to achieve from a subset of scenarios, i.e., from incomplete information about the scenarios whereas the ϑ parameter sets a margin of error that can be tolerated while making such an inference. Another important property of (α_c, ϑ)-consistency is that the two parameters are to be set by the decision-maker depending on the problem at hand, thus making it a flexible notion that may be adjusted based on the particular needs of the decision maker.

We proposed a statistical inference method that uses confidence intervals for enforcing (α_c, ϑ)-consistency. Our empirical study confirms and validates our theoretical properties of the proposed method. Finally, we have demonstrated for the first time how to *directly* solve a whole infinite SCSP without reformulation. This highlights the possibility and potential of using the proposed notions and statistical inference methods in order to be able to design and implement new families of statistical constraint solvers that reason directly about and solve infinite SCSPs.

In future, we plan to: (1) further study how to better determine the appropriate sample size needed to enforce (α_c, ϑ)-GAC; (2) implement a statistical constraint solver that would directly solve infinite SCSPs; and (3) solve few benchmark problems and compare our approach against SAA and CBR to be able to properly assess the value of such an approach.

References

1. Agresti, A., Coull, B.A.: Approximate is better than "exact" for interval estimation of binomial proportions. Am. Stat. **52**(2), 119–126 (1998)
2. Apt, K.: Principles of Constraint Programming. Cambridge University Press, Cambridge (2003)
3. Beck, J.C., Wilson, N.: Proactive algorithms for job shop scheduling with probabilistic durations. J. Artif Intell. Res. (JAIR) **28**, 183–232 (2007)
4. Benoist, T., Bourreau, E., Caseau, Y., Rottembourg, B.: Towards stochastic constraint programming: a study of online multi-choice knapsack with deadlines. In: Walsh, T. (ed.) CP 2001, Proceedings, volume 2239 of LNCS, pp. 61–76. Springer (2001)
5. Clopper, C., Pearson, E.: The use of confidence or fiducial limits illustrated in the case of the binomial. Biometrika **26**(4), 404–413 (1934)
6. Green, S.B., Babyak, M.A.: Control of type i errors with multiple tests of constraints in structural equation modeling. Multivar. Behav. Res. **32**(1), 39–51 (1997)
7. Van Hentenryck, P., Bent, R., Vergados, Y.: Online stochastic reservation systems. In: Beck, J.C., Smith, B.M. (eds.) CPAIOR 2006, Cork, Ireland, May 31 - June 2 Christopher Proceedings, volume 3990 of Lecture Notes in Computer Science, pp. 212–227. Springer (2006)
8. Hnich, B., Rossi, R., Armagan, T.S., Prestwich, S.D.: Filtering algorithms for global chance constraints. Artif. Intell. **189**, 69–94 (2012)
9. Kleywegt, A.J., Shapiro, A., Homem-De-Mello, T.: The sample average approximation method for stochastic discrete optimization. SIAM J. Optim. **12**(2), 479–502 (2001)
10. Papoulis, A. Probability, Random Variables and Stochastic Processes, 2nd ed. McGraw-Hill, New York (1984)
11. R Core Team: R: a language amd Environment for Statistical Computing R Foundation for Statistical Computing (2013)
12. Rossi, R., Hnich, B., Armagan Tarim, S., Prestwich, S.: Confidence-based reasoning in stochastic constraint programming. Artif. Intell. **228**, 129–152 (2015)
13. Rossi, R., Hnich, B., Armagan Tarim, S., Prestwich, S.teven.D.avid.: Finding (α, ϑ)-solutions via sampled Scsps. In: IJCAI, pp. 2172–2177 (2011)
14. Rossi, R., Prestwich, S., Armagan Tarim, S., Hnich, B.: Confidence-based optimisation for the newsvendor problem under binomial, poisson and exponential demand European Journal of Operational Research (2014)
15. Tarim, S.A., Manandhar, S., Walsh, T.: Stochastic constraint programming: a scenario-based approach. Constraints **11**(1), 53–80 (2006)
16. Vollset, S.E.: Confidence intervals for a binomial proportion. Stat. Med. **12**(9), 809–824 (1993)
17. Walsh, T.: Stochastic Constraint Programming. In: Proceedings of the ECAI'2002, pp. 111–115 (2002)