Closures of Uncertain Constraint Satisfaction Problems

Neil Yorke-Smith[1] and Carmen Gervet[2]

[1] Artificial Intelligence Center, SRI International, USA. nysmith@ai.sri.com
[2] IC–Parc, Imperial College London, UK. cg6@icparc.ic.ac.uk

Abstract Data uncertainties are inherent in the real world. The *uncertain CSP* (UCSP) is an extension of classical CSP that models incomplete and erroneous data by coefficients in the constraints whose values are unknown but bounded, for instance by an interval. Formally, the UCSP is a tractable restriction of the quantified CSP. The resolution of a UCSP, a set of its potential solutions called a *closure*, can take different forms according to the user's requirements and the nature of data uncertainty in the problem specification. In a former paper we presented mainly the full closure, the set of all potential solutions, which finds application in diagnosis problems where the existence of any potential solutions is an objective by itself. In this paper, we develop other commonly-useful closures such as the covering set (found in contingent planning problems) and the most robust solution (found in conformant planning problems). We formally define different closures as solutions to a UCSP model, relate them in a hierarchy, and outline means to derive them from one another.

1 Introduction

Across numerous applications, real-world Large Scale Combinatorial Optimisation problems (LSCOs) are permeated by data uncertainty. Despite its successes, it is recognised that the classical constraint satisfaction problem (CSP) is inadequate as a model of LSCOs with uncertain data. For bounded data uncertainty (as opposed to stochastic [14]), increasing attention has turned to modelling these problems with variations of *quantified CSP* (QCSP), particularly over the reals [2,5,15]. Applications with bounded uncertain data include camera control [5], engineering design [1], network inference [19], and robot motion [11]. In such applications, where a probabilistic description of the data uncertainty is not available or not meaningful, bounding the uncertainty models data where values are unknown but can be bounded with definite available knowledge.

Although solving the general quantified CSP is substantially more difficult, compared to the classical CSP, its full expressive power is, fortunately, not necessary for all forms of uncertainty. The *uncertain CSP* (UCSP) was introduced in [19] to model LSCO problems with incomplete and erroneous data, without approximation of data or potential solutions. Driven by practical considerations, a UCSP provides a formal model as a tractable restriction of the general QCSP; efficient solving of a UCSP exploits its restricted quantification.

The resolution of a UCSP can be the set of all potential solutions (previously introduced as its *full closure*), or a subset of it. The resolution sought depends on the problem. In [19] we presented the full closure, which is required in diagnosis problems such as some forms of network inference subject to data measurement errors. However, depending on her application, the user may be interested in one or more aspects of the potential solutions. For planning the control of aerospace components [21], for

instance, a UCSP model is suitable but the resolution sought is a plan of operation for each anticipated environmental uncertainty. The UCSP reliably captures the nature of uncertainty in the data and the solution sought. This corresponds to a *covering set*: a set of solutions that contains at least one solution (not necessarily all potential solutions) for each anticipated *realisation* of the data parameters.

Contrasting with alternative approaches to uncertainty like the stochastic CSP [14] — where with a probabilistic model one would naturally seek robustness since by essence the representation of uncertainty goes in hand with seeking a solution state that is most likely to occur, given the knowledge of the data — we show that the UCSP model is not bound to a certain form of solution. One can seek to enclose data errors with certainty (full closure), or to reason alternatively with uncertainty (another closure), according to the user's requirements.

To meet the need for a relevant, reliable resolution to UCSPs in different applications, this paper develops further the concept of a closure. We give a systematic study of different closures for UCSP models, depending on the application requirement, as well as generic approaches to derive them independently of the problem specifics. After introducing the necessary background (Sections 2 and 3), we introduce the different closures, and formalise a hierarchy of closures, showing how they relate (Section 4). This enables us to develop resolution forms to derive them (Section 5).

2 Related Work

Existing generic approaches to uncertain data in CP propose models and methods for robust solutions to the problem. The *mixed CSP* framework [7], defined for discrete data and variables, seeks a solution that holds under the most possible realisations of the data (a most robust single solution) or a conditional decision that depends on the observed data realisation. The *stochastic CSP* framework [14] attaches a probability distribution to parameters and seeks a solution (which may be a policy) that maximises expectation or a related objective. When exploited for uncertain data, the *fuzzy CSP* [6] seeks a solution most satisfied "whatever the value of [the data] turns out to be". In a related vein, the super CSP framework [10] seeks solutions requiring little repair when the data changes. In all these cases, the resolution of the model is defined (if implicitly) in terms of the *support* relationship between data realisations and potential solutions.

The purpose of computing robust solutions is to ensure that whatever the real world situation, the solution holds under most cases. Whether a robust solution is adequate for resolving an uncertain LSCO depends on the nature of the uncertainty and the sought outcome [19]. For instance, when handling data errors one is certainly not looking for a solution that satisfies as many erroneous models as possible.

In contrast to approximation or stochastic approaches, the UCSP model is based on the paradigm of *convex modelling* (of which interval analysis over the reals is a simple instance), as used in control theory and operational research on continuous problems to obtain an enclosure guaranteed to contain the true solution [12]. In these applications, convex modelling essentially derives the full closure, since the uncertainty chiefly arises due to bounded measurement errors in the data, thus entailing the full closure as the sought solution. Within CP, the UCSP approach is most closely related to work such as [2, 5], who seek a reliable solution set in the presence of uncertain data, by using quantified constraints over the reals.

3 Background

A classical CSP is a tuple $\langle \mathcal{V}, \mathcal{D}, \mathcal{C} \rangle$, where $\mathcal{V}$ is a finite set of variables, $\mathcal{D}$ is the set of corresponding domains, and $\mathcal{C} = \{c_1, \ldots, c_m\}$ is a finite set of constraints. A solution is a complete consistent value assignment. We represent a CSP by a conjunction of its constraints $\bigwedge_i c_i$ (as opposed to the set of its allowed tuples); hence we will define functions that operate equally on a single constraint or a CSP. Similarly, we represent a solution or set of solutions to a CSP by a conjunction of constraints.

A constraint is a relation between constants, variables and function symbols. The constants we refer to as *coefficients*. A coefficient may be *certain* (its value is known) or *uncertain* (value not known). In a classical CSP, all the coefficients are certain. We call an uncertain coefficient a *parameter*. The set of possible values of a parameter λ_i is its *uncertainty set*, denoted U_i. We say an *uncertain constraint* is one in which some coefficients are uncertain. Observe that the coefficients in an uncertain constraint are still constants; merely as parameters their exact values are unknown. For example, if the parameter λ_1 has uncertainty set $U_1 = \{0, 1, 2\}$, the constraint $X < \lambda_1$ is uncertain.

A *data realisation* fixes the parameters to values from their uncertainty sets. Any certain constraint corresponding to a realisation is a *realised* constraint, denoted $\hat{c} \in c$. An uncertain constraint can have many realisations, as many as the size of the Cartesian product of uncertainty sets involved. We write $\mathbb{C}$ for the set of all (uncertain) constraints in a constraint domain, and $\widehat{\mathbb{C}} \subset \mathbb{C}$ for the set of all (certain) realised constraints.

3.1 Uncertain CSP

The *uncertain CSP* extends a classical CSP with an explicit description of the data that allows us to reason with the uncertainty to derive reliable solution enclosures.

Definition 1 (UCSP). *An* uncertain constraint satisfaction problem $\langle \mathcal{V}, \mathcal{D}, \Lambda, \mathcal{U}, \mathcal{C} \rangle$ *is a classical CSP* $\langle \mathcal{V}, \mathcal{D}, \mathcal{C} \rangle$ *in which some of the constraints may be uncertain. The finite set of parameters is denoted by* Λ*, and the set of corresponding uncertainty sets by* $\mathcal{U}$*.*

Example 1. Let X_1 and X_2 both have domains $D_1 = D_2 = [1, 5] \subseteq \mathbb{Z}$. Let λ_1 and λ_2 be parameters with uncertainty sets $U_1 = \{2, 3, 4\}$ and $U_2 = \{2\}$ respectively. Consider three constraints: $c_1 : X_1 > \lambda_1$, and $c_2 : |X_1 - X_2| = \lambda_2$, and $c_3 : X_2 - \lambda_1 \neq 1$. Writing $\mathcal{V} = \{X_1, X_2\}$, $\mathcal{D} = \{D_1, D_2\}$, $\Lambda = \{\lambda_1, \lambda_2\}$, $\mathcal{U} = \{U_1, U_2\}$, and $\mathcal{C} = \{c_1, c_2, c_3\}$, then $\langle \mathcal{V}, \mathcal{D}, \Lambda, \mathcal{U}, \mathcal{C} \rangle$ is a UCSP. Note that c_2 is a certain constraint; c_1 and c_3 are both uncertain constraints. □

We say that any certain CSP $\widehat{P}$, corresponding to a realisation of the parameters of P, is a *realised CSP*, and write $\widehat{P} \in P$. Observe that a UCSP is the disjunction of its realised CSPs. If r is a realisation and $\widehat{P}$ is a corresponding realised CSP, for a solution s of $\widehat{P}$, we say that the realisation r *supports* s, and s *covers* r.

The *complete solution set* $\mathrm{Cl}(P)$ of a UCSP P is the set of all solutions supported by *at least one* realisation. Each element of $\mathrm{Cl}(P)$ is called a *potential solution*. The resolution to a UCSP model is a *closure*: a set of potential solutions, i.e. a subset of $\mathrm{Cl}(P)$. If the closure is the entire solution space, we say it is the *full closure*. The task we address in this paper is formalising the nature of and relation between closures, and developing methods to derive meaningful closures other than the full closure. For reasons of space, our examples are mostly arithmetic constraints.

Example 2. Let P be the UCSP of Example 1. The full closure of P in tuple notation is $(X_1, X_2) \in \{(3,1), (3,5), (4,2), (5,3)\}$. For comparison, a covering set closure of minimal size (which we formally define in the next section) is $(X_1, X_2) \in \{(3,1), (5,3)\}$, since this solution set covers all three realisations $(\lambda_1 = 2 \vee 3 \vee 4)$. □

An uncertain CSP is a restriction of a general QCSP, if we view parameters as existentially quantified variables. In a QCSP variables may be arbitrarily existentially ($\exists$) and universally ($\forall$) quantified. This means the complexity of solving a QCSP is significantly greater than solving a classical CSP. The general problem is PSPACE-complete, and for many constraint classes where a CSP is tractable, a QCSP is not [3].

Fortunately, the complexity of solving a UCSP is much less: deriving one potential solution of a UCSP is NP-complete. The complexity of deriving the full closure (in contrast to one potential solution) is Σ_2^p-complete since the set may be exponentially large. Indeed, a potential solution to a UCSP P corresponds to an instantiation to the variables such that there exists a realisation of the parameters, such that all the constraints of P hold; hence the quantification in a UCSP is $\exists\exists$ for the full closure. In principle, by such a reduction to a classical CSP we can derive the full closure; in practice, less naive methods are more effective.

3.2 Deriving the Full Closure

We briefly recall the different approaches to derive the full closure first presented in [19] since in the sequel we will adopt them to derive other types of closures.

Since a UCSP is an instance of QCSP, one approach to deriving the full closure (and indeed other closures) is to employ general QCSP solution methods. With the increasing interest in QCSPs, propagation-, search-, and transformation-based methods exist for continuous and, to a lesser extent, for discrete QCSPs [2, 4, 5, 8, 9, 13, 15].

However, prior work on UCSPs has focused on developing two dedicated but generic resolution forms for UCSPs. This is because, firstly, UCSPs are a restricted form of QCSP and the restricted quantification in queries should be exploited where possible; and secondly because we seek sets of solutions not individual solutions. But one is free to exploit elements of generic QCSP methods where advantageous. For instance, we can use quantified AC [4, 13] as a preprocessing step.

The first specific means to derive the full closure of a discrete UCSP (i.e. with discrete uncertainty sets) is to enumerate the realisations. Under each realisation, the UCSP is a classical CSP. By finding all solutions to each such realised CSP, and combining them, the full closure of the UCSP is obtained. As observed in [19], naive enumeration can be improved, for example by exploiting algorithms for constructive disjunction and by interleaving quantified AC checking.

A second means to derive the full closure is to reformulate the UCSP into a tractable classical CSP such that the problems are equivalent. An *equivalent CSP*, denoted $\tau(P)$ with respect to a UCSP P, is such that its complete solution set coincides with the full closure of P. Since the complexity of enumeration can grow rapidly with the size of the uncertainty sets, where feasible, a reformulation approach can be expected to have much lower complexity than enumeration.

To make precise equivalence of solution sets, we define the partial order $\preceq$ such that: $c_1 \preceq c_2$ iff $\mathrm{Cl}(c_1) \subseteq \mathrm{Cl}(c_2)$. The equivalent CSP should satisfy $P = \tau(P)$ under $\preceq$. Formally, the equivalent CSP is found using a *certain equivalence transform* (CET):

Definition 2 (Certain Equivalence Transform). *A map $\tau : \mathbb{C} \rightarrow \widehat{\mathbb{C}}$ is a* certain equivalence transform *if it (1) preserves certainty, i.e. $\tau(\hat{c}) = \hat{c} \ \ \forall \hat{c} \in \widehat{\mathbb{C}}$; and (2) is a closure operator, i.e. is increasing, monotone and idempotent. Increasing means $c \preceq \tau(c)$.*

The equivalent problem obtained by such a transformation may be solved with any classical CSP methods. The process we encapsulate as a *solution operator*:

Definition 3 (Solution operator). *Let $\widehat{P}$ be a certain CSP. Let $\phi : \widehat{\mathbb{C}} \rightarrow \widehat{\mathbb{C}}$ be a map such that $\phi(\mathcal{C})$ describes a set of solutions to $\widehat{P}$. If ϕ obeys the contraction, monotone and idempotence properties, then we say that ϕ is a (certain)* solution operator. *If further $\phi(\mathcal{C})$ describes the set of all solutions to $\widehat{P}$, we say ϕ is* complete *for $\widehat{P}$.* □

Whether by direct computation, enumeration, transformation, or other means, the derivation of a closure is captured formally as an *uncertain solution operator*:

Definition 4 (Uncertain solution operator). *Let P be a UCSP. An* uncertain solution operator *is a map $\rho : \mathbb{C} \rightarrow \widehat{\mathbb{C}}$ such that $\rho(\mathcal{C}) \preceq \mathrm{Cl}(P)$. An uncertain solution operator ρ must obey the contraction, monotone and idempotence properties. If further $\rho(\mathcal{C}) = \mathrm{Cl}(P)$, we say ρ is* complete *for P.* □

4 Different Types of Closures

At the heart of the UCSP model and its resolution is a demand for *reliable* solutions, by which, informally, we mean faithful relative to our knowledge of the real world state. A perquisite for a reliable solution is a reliable model, i.e. one that does not approximate the data, but bounds it with the definite available knowledge. In concrete terms, the form that reliable inference takes depends on two, linked issues: the requirements of the user and the nature of the data uncertainty. In a diagnosis problem, for example, the user simply might want to know whether there exist any realisations at all with solutions (any *good* realisations); while in a planning problem, she might want to know which realisations support which solutions. We meet the varying forms of reliable inference by providing closures of various types in relationship with UCSP data representation.

More specifically, suppose the user specifies that she is interested in particular information as the resolution of a UCSP. This corresponds to a particular aspect of the potential solutions. An *adequate* solution is then one that (1) comprises at least this information, and (2) faithfully reflects our knowledge about the real-world. Hence, we say that an *adequate closure* is a subset of the full closure that provides at least the information the user requires as the resolution of a UCSP. The key questions are: what properties make a closure often adequate, with respect to the data representation, and how do we derive these closures?

We consider four different closures of practical significance and will show how they can be derived when a UCSP model is used:

1. The *full closure*: the set of all solutions that each cover at least one realisation. Example usage: detecting whether the model has any potential solution (thus differentiating model inconsistency from data errors), diagnosis of the reliability of approximation models.
2. A *covering set*: a set of solutions that together cover all realisations. A covering set closure is *minimal* if the cardinality of this set is minimal among all such sets. Example usage: 'robust' solution covering every eventuality, e.g. contingent planning.

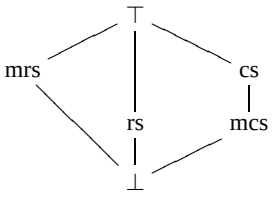

Figure 1. Simple hierarchy of closures. At the top of the lattice is the full closure, at the bottom the empty closure. Illustrated in the middle are a covering set (cs), a minimal covering set (mcs), a robust set (rs), and the most robust solution (mrs).

$\widehat{P}_1$	$a\ b\quad d$
$\widehat{P}_2$	$a\quad c$
$\widehat{P}_3$	$\quad b\qquad e$
$\widehat{P}_4$	$\quad b\ c$

Figure 2. For Example 3, realised CSPs (denoted $\widehat{P}_1$–$\widehat{P}_4$) and their feasible solutions (denoted a–e).

3. A *robust set*: a set of solutions such that each cover *all* realisations (not just at least one). A robust set closure is maximal if the cardinality of this set is maximal among all such sets. Example usage: conformant planning.
4. A *most robust solution*: a single solution that covers the maximal number of realisations, of all single solutions. Example usage: robust solution that must be a single solution and not a set of solutions, e.g. schedule for a staff roster [14].

We next characterise these closures formally in terms of the space of realisations and the space of potential solutions. A simple hierarchy of closures is shown in Figure 1. The closures form a lattice under set inclusion. The full closure is the top of the lattice, and the empty closure (the empty set) is the bottom. Examining the hierarchy of closures, we then will address how to derive closures, including one closure from another.

Example 3. Consider the abstract UCSP depicted in Figure 2. There are four realised CSPs, denoted $\widehat{P}_1$–$\widehat{P}_4$, and five potential solutions, denoted a–e. The top of the lattice hierarchy is the set $\top = \{a, b, c, d, e\}$. The most robust solution is b; while uniquely maximal, it does not cover all realisations ($\widehat{P}_2$ is uncovered). Since there is no solution that covers all realisations, the (maximal) robust set is empty. However, there are two covering sets of minimal cardinality, $\{a, b\}$ and $\{b, c\}$. □

4.1 Support Operators

In order to define formally and derive closures, we now introduce *support operators*. A *support operator* tells us which realisations support which potential solutions; we call this *support information*.[3] Its inverse tells us, dually, which realisations are covered by which solutions. The relationship between potential solutions and realisations is the heart of the definition of closures and central to the distinction between which closure is adequate as the resolution for a given UCSP. Moreover, support information is precisely what we need for reasoning about elements of a closure, for instance in optimisation (although outside the remit of this paper).

Let $\mathcal{R}$ be the *space of realisations*, the set of all possible realisations; and let $\mathcal{S}$ be the *space of solutions*. Observe that for any UCSP P, its complete solution set $\mathcal{S}_P$ is a subspace of $\mathcal{S}$. Similarly, we define the complete realisation set $\mathcal{R}_P$ of P; it is a

[3] Similar concepts are found in related frameworks, although not always explicitly formulated.

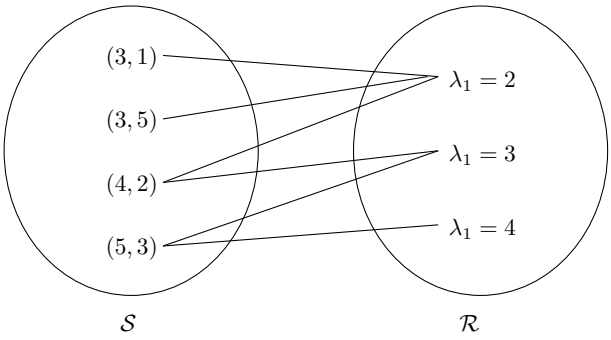

Figure 3. Solutions and realisations for Example 1.

subspace of $\mathcal{R}$. Recall that the power set $\mathcal{P}(S)$ of a set S is the set of all subsets of S: for example, $\mathcal{P}(\{1,2\}) = \{\,\emptyset, \{1\}, \{2\}, \{1,2\}\,\}$.

Definition 5 (Support operator). *A* support operator *is a map $\Sigma : \mathcal{P}(\mathcal{S}) \to \mathcal{P}(\mathcal{R})$ such that $\forall S \subseteq \mathcal{S},\ \Sigma(S) = R$, where $R \subseteq \mathcal{R}$ is a set of realisations s.t. each supports* at least one *solution in S. We write a relational inverse of Σ as $\Sigma^{\text{-}1} \subseteq \mathcal{P}(\mathcal{R}) \times \mathcal{P}(\mathcal{S})$, and write $\Sigma^{\text{-}1}(R)$ to be the projection of a tuple $\Sigma^{\text{-}1}(R, S)$ onto $\mathcal{P}(\mathcal{S})$.*

If a support operator provides all realisations that support a set of solutions, i.e. $\Sigma(S) = R$ where R is the set of all *realisations such that each supports at least one solution in S, we say Σ is* complete*. Likewise, if $\Sigma^{\text{-}1}(R) = S$ where S is the set of all solutions that cover at least one realisation in R, we say $\Sigma^{\text{-}1}$ is* complete*.* □

Example 4. For Example 1, let us represent both $\mathcal{S}$ and $\mathcal{R}$ in the natural way with the constraint class of tuples; the two spaces are shown pictorially in Figure 3. In the figure we see that the full closure to the UCSP is $\mathcal{S}_P = \{(3,1), (3,5), (4,2), (5,3)\}$, and the set of all realisations that each support at least one solution in $\mathcal{S}_P$ is $\mathcal{R}_P = \{2,3,4\}$.

A support operator Σ_1 is defined by: $(3,1) \mapsto 2$, $(3,5) \mapsto 2$, $(4,2) \mapsto 3$, and $(5,3) \mapsto 4$. Σ_1 is complete: one can verify that, for instance, $\Sigma_1(\mathcal{S}_P) = \mathcal{R}_P$.

A second support operator Σ_2 is defined by: $(3,1) \mapsto 2$, $(3,5) \mapsto 2$, $(4,2) \mapsto 2$, and $(5,3) \mapsto 3$. Σ_2 is not complete, because it never includes $\lambda_1 = 4$ (for example) but this realisation supports solution $(5,3)$.

One relation inverse $\Sigma_1^{\text{-}1}$ of Σ_1 is defined by the tuples: $2 \times (3,1)$, $3 \times (4,2)$, and $4 \times (5,3)$; it is not complete. Another relation inverse $\Sigma_2^{\text{-}1}$ of Σ_1 is defined by: $2 \times \{(3,1), (3,5), (4,2)\}$, $3 \times \{(4,2), (5,3)\}$, and $4 \times (5,3)$; it is complete. □

Since $\Sigma^{\text{-}1}$ maps realisations to solutions, we can view an uncertain solution operator ρ in terms of a support operator inverse. Declaratively, ρ maps a UCSP P to a closure, a subset of $\mathrm{Cl}(P) \equiv \mathcal{S}_P$. Since P is the disjunction of all its realised CSPs, and these are in one-to-one correspondence with realisations, we can write $\rho(P) = \Sigma_\rho^{\text{-}1}$ for some $\Sigma_\rho^{\text{-}1}$, linking the operational and declarative views.

Example 5. Let ρ be an uncertain solution operator that computes the full closure for the UCSP P of the last Example. $\rho(P)$ corresponds to $\Sigma_2^{\text{-}1}(\mathcal{R}_P)$, i.e. to $\Sigma_2^{\text{-}1}$ acting on the space of the realisations of P. □

Table 1. Levels of support information.

	LEVEL	INFORMATION
strongest	full	$\Sigma(S)$ known for all $S \subseteq \mathcal{S}_P$
	enumeration	$\Sigma(s)$ known for all $s \in \mathcal{S}_P$
	badness	whether $\Sigma^{-1}(r)$ is empty or not for all $r \in \mathcal{R}_P$
weakest	none	only $\Sigma(\mathcal{S}_P)$ known

Remark. Note first that Σ depends not only on the problem but also the type of closure; just as there may be multiple ρ for a problem (even for the same closure), there may be multiple Σ. Second, in general Σ need not be a surjective function and so need not have a functional inverse. As a relation, however, Σ may have many relational inverses.

4.2 Calculating Support Information

To derive uncertain solution operators requires we have knowledge, in general, of which realisations support which potential elements of the sought closure, which amounts to knowing and having a means to compute Σ on subsets of $\mathcal{S}_P$. We describe the amount of information known about Σ as the *level* of support information, summarised in Table 1. In this subsection we outline how to compute support operators; for the full descriptions, the reader is referred to [18].

At one extreme, if $\Sigma(S)$ is known for all $S \subseteq \mathcal{S}_P$, then we have *full* support information. This is crucial in some applications, such as planning problems where the user wants to know under what circumstances a plan is valid. At the other extreme, if only $\Sigma(\mathcal{S}_P)$ is known (i.e. not $\Sigma(S)$ for any $S \subseteq \mathcal{S}_P$), then we have no support information. This suits other applications, such as diagnosis problems where the user wants to know only what potential solutions might arise. The intermediate cases trade off knowledge of support with the effort of computing the information. If any support information is required by an application, the enumeration level is frequently the least information that is useful; thus we concentrate on this case. The main exception is when the user wants to know only about bad realisations, which we can meet with a simple list.

In the case of discrete data uncertainty, the option of exhaustive enumeration is available since the realisation space is finite. It provides a natural way to calculate enumeration or badness levels of support information. In fact, search through the realisation space $\mathcal{R}_P$, for those realisations that support a given single solution s of P, can be interleaved with deriving a closure by enumeration. This search corresponds to solving a CSP P^∂ dual to the original UCSP P. The variables in the dual CSP P^∂ are the realisations of P with domains in $\mathcal{R}_P$, and the constraints of P^∂ state that a solution r must support s as a solution to P.

In the case of continuous data uncertainty, the question of finding a realisation that supports a given solution s — solving the dual CSP — can be thought of as exhibiting a *witness*. That is, from the continuum of realisations, find one (the witness) that shows that s is supported. In contrast with the discrete case, analytical derivation of support

information is more feasible, because of the continuity of the realisation space: for instance, for real-valued UCSPs with linear constraints [18].[4]

Whereas exhaustive enumeration is clearly undesirable with continuous data, we can use numerical CSP methods. The question to answer, namely: find a witness or prove none exists, is addressed readily by these methods: see the literature in [15].

4.3 Formal Definition of Types of UCSP Closures

Summarising, we have defined support operators and outlined means to compute them. Now, with these tools, we are able to define closures of different types and give methods to derive them. Recall that the motivation to consider closures of different types is that a UCSP can be used for different purposes; a bounded data representation does not limit the type of resolution sought. Depending on the user requirements and the data representation, reliable inference can be achieved with solutions which are subsets of the full closure. Thus four types of closure were informally introduced earlier, and illustrated in Figure 1. The last subsection and the coming next section present methods to reason upon these closures:

Definition 6 (Types of closure). *A closure* $\Sigma_\rho^{\text{-}1}(\mathcal{R}_P) \equiv \rho(P) \preceq \mathrm{Cl}(P)$ *is:*

1. *the* full *closure iff* $\Sigma^{\text{-}1}$ *is surjective on* $\mathcal{S}_P$*, i.e.* $\Sigma^{\text{-}1}(\mathcal{R}_P) = \mathcal{S}_P$*,*
2. *a* covering set *if* $\Sigma \circ \Sigma_\rho^{\text{-}1}(\mathcal{R}_P) = \mathcal{R}_P$*; and minimal iff* $|\Sigma_\rho^{\text{-}1}(\mathcal{R}_P)|$ *is minimal,*
3. *a* robust set *iff* $\Sigma(s) = \mathcal{R}_P, \forall s \in \Sigma_\rho^{\text{-}1}(\mathcal{R}_P)$*; and maximal iff* $|\Sigma_\rho^{\text{-}1}(\mathcal{R}_P)|$ *is maximal,*
4. *a* most robust solution *iff* $|\Sigma_\rho^{\text{-}1}(\mathcal{R}_P)| = 1$ *and* $|\Sigma \circ \Sigma_\rho^{\text{-}1}(\mathcal{R}_P)|$ *is maximal.* □

While the full closure is unique for any UCSP, other closures need not be; there may be multiple closures of one type. For instance, as we saw earlier, there are two minimal covering sets of the UCSP of Example 3. We call *singleton closures* those that are necessarily formed of one element, such as a most robust solution closure.

Example 6. Returning once more to the abstract CSP of Example 3, the different closures for the problem are:

- The full closure is $\mathcal{S}_P = \{a, b, c, d, e\}$.
- The covering set closures are any subset of $\mathcal{S}_P$ containing at least one of: $\{a, b\}$, $\{a, c, e\}$, $\{b, c\}$, or $\{c, d, e\}$.
- The minimal covering set closures are two: $\{a, b\}$ and $\{b, c\}$.
- There is no robust set closure, since no solution covers all realisations.
- The most robust solution closure is b, which covers three realisations. □

Because the criteria for elements of a robust set closure are strong — each element must cover all realisations — it is common for no robust set closure to exist, as in the example. This said, because of the strength of its definition, when such a *fully robust* solution does exist, the robust set closure comes to life:

Proposition 1 (Robust set closure). *If* S *is a non-empty robust set closure, every element of* S *is both a most robust solution closure and a minimal covering set closure.* □

[4] An analogy might be drawn with mathematical programming, where continuous problems (e.g. LP) are more amenable to analysis than discrete problems (e.g. IP).

The result, explaining the relation between a robust set closure and other types of closures, can be used to derive other closures from a robust set. Firstly, since every element of a robust set closure is a fully robust single solution, by definition, each is a most robust solution closure. As a trivial corollary, every most robust solution is contained in some maximal robust set closure. Secondly, since every element of a robust set covers all realisations, each is a covering set closure; in fact, a minimal covering set.

4.4 Complexity

The complexity of UCSP resolution depends on the decision question, which varies for different closures. We analyze the complexity by expressing these closures in first order logic, extended for convenience with set semantics. Supposing a tuple of values $\boldsymbol{v}$, the support of $\boldsymbol{v}$ as a potential solution is expressed by the formula:

$$\exists \boldsymbol{\lambda}_{\boldsymbol{v}} : \mathcal{C}(\boldsymbol{v};\, \boldsymbol{\lambda}_{\boldsymbol{v}}) \tag{1}$$

We can then describe these closures:

- full closure: the set of all $\boldsymbol{v}$ satisfying (1), i.e.:

$$\exists \boldsymbol{v}\, \exists \boldsymbol{\lambda}_{\boldsymbol{v}} : \mathcal{C}(\boldsymbol{v};\, \boldsymbol{\lambda}_{\boldsymbol{v}}) \tag{2}$$

- covering set: let S be a set of $\boldsymbol{v}$ satisfying (1); a covering set closure is:

$$\forall \boldsymbol{\lambda}\, \exists \boldsymbol{v}_{\boldsymbol{\lambda}} \in S : \mathcal{C}(\boldsymbol{v}_{\boldsymbol{\lambda}};\, \boldsymbol{\lambda}) \tag{3}$$

- robust set: let S be a set of $\boldsymbol{v}$ satisfying (1); a robust set closure is:

$$\exists \boldsymbol{v} \in S\, \forall \boldsymbol{\lambda} : \mathcal{C}(\boldsymbol{v};\, \boldsymbol{\lambda}) \tag{4}$$

A most robust solution is more awkward to write in this way because the maximality of $|\Sigma \circ \rho(P)|$ is integral to the definition. In contrast, for the covering and robust set closures, minimality or maximality is only required to give us the minimal covering set or maximal robust set respectively.

For the full closure, in Section 3.1 we noted how the decision question for a UCSP ('Does there exist a potential solution?') reduces to that for a classical CSP ('Does there exist a solution?'), if we treat the parameters as variables.[5] Thus the decision problem for one potential solution of a UCSP, i.e. (1), is NP-complete.

The full closure corresponds to the set of all potential solutions, i.e. (2). For a classical CSP, finding all solutions is Σ_2^p-complete, since we guess a complete solution set and check it in NP time; it cannot be checked in P time in general. By a similar argument to the above, finding $\mathcal{S}_P$ for a UCSP is Σ_2^p-complete. In contrast, for a general QCSP, finding all solutions is PSPACE-complete.

Remark. For mixed CSP, deciding consistency is Π_2^p-complete [7], where Π_2^p = co-Σ_2^p is the complementary complexity class to Σ_2^p. Deciding strong consistency is in Σ_2^p,

[5] Provided these pseudo-variables are instantiated before the actual decision variables. While this reduction is helpful from a theoretical perspective, operationally, as we also noted, in general it is not the best means to derive the full closure.

unless parameters are logically independent, when it is NP-complete. For stochastic CSP, deciding consistency is PSPACE-complete [14]. For a one-stage problem, it is Σ_2^p-complete. Thus we can say that finding the complete solution set of a UCSP has the same worst-case complexity as deciding consistency of a mixed CSP, or deciding consistency of a one-stage stochastic CSP. It follows also that deriving a most robust solution closure to a UCSP is Σ_2^p-complete.

Neither (3) nor (4) have the double existential quantification of (2). Hence the reduction to a classical CSP does not extend to covering or robust set closures. However, similar arguments to the above show that the covering set and robust set closures have the same complexity as the full closure. In contrast, deriving the minimal covering set and maximal robust set closures is an optimisation problem over subsets of the full closure. The complexity thus increases up the polynomial hierarchy to the class PSPACE, the same complexity class as the general QCSP.

5 Deriving Closures

We now show how to derive other closures, besides the full closure, by computing the relevant uncertain solution operators ρ. The derivations rely, either implicitly or explicitly, on computation of the relevant support information.

5.1 Deriving One Closure From Another

In the last section we noted that the simplest means to calculate the full closure, naive reduction to a classical CSP, is not feasible for some of the other types of closures. However, from the lattice of closures in Figure 1 we can design efficient methods to compute one closure from another in some cases.

By definition every closure is a subset of the full closure. Thus in principle one can derive any closure by first deriving the full closure, then winnowing its elements according to the distinguishing property of the closure sought (Proposition 6). A more common situation is when we know some covering set closure and wish to compute a smaller covering set if one exists. This leads to examining the support of each element. However, perhaps contrary to intuition, it is not always possible to derive a minimal covering set closure (call it MCS) from a covering set closure (call it CS). The reason is that MCS need not be a subset of CS. It is a member of the set of all covering sets, by definition, but any given CS might omit elements of MCS.

Even when two closures are not partially ordered in the hierarchy, it is still possible to gain from the knowledge of one. For example, knowing a covering set is a reasonable starting point for deriving a robust set. This is because single solutions in a covering set are likely to cover many realisations, and so are likely to be more robust than solutions chosen at random.

There are some situations when it is particularly simple to derive one closure from another. For instance, Proposition 1 tells us that a robust set closure, when one exists, provides immediately a most robust solution and also a minimal covering set. In contrast to *coverage* (the number of realisations covered by a closure), *robustness* (whether a singleton closure covers all realisations) is a monotone property. That is, a maximal robust set closure necessarily contains every robust set closure. It is also true that a most

robust solution closure is contained in a maximal robust set, but it is not necessarily true that it need be contained in a minimal covering set. Consequently, because of the monotonicity, from the full closure it is tractable to compute a maximal robust set closure and also a most robust solution closure (but not to compute a minimal covering set closure). We class this as an enumeration algorithm, and list it below.

5.2 Transformation Resolution Form

Transformation proved to be a powerful means of deriving the full closure for some constraint classes [19, 20]. To use it for other types of closures, we must reconsider the definition of a Certain Equivalence Transform. Analysing the correctness proof in [19] reveals that the increasing property of τ, i.e. $c \preceq \tau(c)$, is used only to establish that $\mathrm{Cl}(P) \preceq \rho(P)$. Removing it generalises the definition of a CET:

Definition 7 (Generalised Certain Equivalence Transform). *A lattice homomorphism $\tau : \mathbb{C} \to \widehat{\mathbb{C}}$ is a* generalised certain equivalence transform *if it preserves certainty, and is monotone and idempotent (but not necessarily increasing).* □

Given the desired type of closure, a generalised CET is combined with a suitable property for the closure type, to give a type-specific *specialised CET*. Thus, for any UCSP, we have a family of CETs. Each specialised CET in the family gives rise to an uncertain solution operator that computes a closure of the specific type.

An initial observation is that every uncertain solution operator computes a subset of the full closure, i.e. satisfies (1).

Definition 8 (Closure-contained). *We say a map $\rho : \mathbb{C} \to \widehat{\mathbb{C}}$ is* closure-contained *if $\rho(P) \preceq \mathrm{Cl}(P)$ for any UCSP P.* □

The main result states the features of the specialised CET for each type of closure:

Proposition 2 (Specialised CETs). *Let P be a UCSP and let τ_0 be a generalised CET for P. Suppose $\mathfrak{t}$ is a type of closure. Let $\tau_\mathfrak{t}$ be a specialised CET from τ_0, let $\phi_\mathfrak{t}$ be a solution operator complete for $\tau_\mathfrak{t}(P)$, and let $\rho_\mathfrak{t} = \phi_\mathfrak{t} \circ \tau_\mathfrak{t}$.*

Then for each specialised CET $\tau_\mathfrak{t}$ formed by adding the following properties, the $\rho_\mathfrak{t}$ to which it gives rise is an uncertain solution operator that computes a closure of the following types:

1. *the* full *closure if τ_{full} is tight and increasing, i.e. $\rho_{\mathrm{full}}(P) \preceq \mathrm{Cl}(P)$ and $c \preceq \tau_{\mathrm{full}}(c)\, \forall c$,*
2. *a* covering set *if ρ_{cs} is closure-contained, and: $\Sigma \circ \rho_{\mathrm{cs}}(P) = \mathcal{R}_P$; and minimal iff $|\rho_{\mathrm{cs}}(P)|$ is minimal,*
3. *a* robust set *if ρ_{rs} is closure-contained, and: $\Sigma(s) = \mathcal{R}_P$, $\forall s \in \rho_{\mathrm{rs}}(P)$; and maximal iff $|\rho_{\mathrm{rs}}(P)|$ is maximal,*
4. *a* most robust solution *if ρ_{mrs} is closure-contained, and: $|\rho_{\mathrm{mrs}}(P)| = 1$ and $|\Sigma \circ \rho_{\mathrm{mrs}}(P)|$ is maximal.* □

Proposition 2 yields a method in principle to derive any closure by transformation. First, define a suitable $\tau_\mathfrak{t}$. Second, compute $P' = \tau_\mathfrak{t}(P)$. Third, find the complete solution set of P' using a certain solution operator ϕ. Note that, especially if the closure definition contains an optimisation facet, such as a most robust solution closure, then P'

may be a CSOP, i.e. feature an objective function. In this case, in line with the standard notion for CSOPs, the complete solution set of P' is the set of all optimal solutions.

The key issue in practice is to find a suitable transformation for a given constraint class, just as it was for the case of the full closure [20]. As before, the complexity of the resolution form hinges on the complexity of the transformation, and of solving the equivalent problem.

Example 7. Continuing Example 3, we will derive a covering set closure by transformation. Let us suppose the UCSP P has single variable X, with domain $\{a, b, c, d, e\}$, and single parameter λ, with uncertainty set $\{1, \ldots 4\}$, and P has the sole constraint that gives realisations and solutions as in Figure 2.

Let τ map the UCSP P to the CSP P' given by: $\mathcal{V} = \{X\}$, $\mathcal{D} = \{D_X\}$ where $D_X = \{a, b, c\}$, and $\mathcal{C} = \emptyset$. The complete solution set P' is (trivially) $X \in \{a, b, c\}$; and $\{a, b, c\}$ is a covering set closure for P. Moreover, $\Sigma \circ \rho(P) = \{1, \ldots 4\} = \mathcal{R}_P$, as required by the last Proposition. This τ is by no means unique. The choice of which P' to transform the UCSP into greatly impacts the complexity of deriving the closure. □

5.3 Enumeration Resolution Form

When transformation is not suited for a constraint domain, enumeration is usually an alternative. Again we would like to use it for other types of closures. For lack of space, we can only sketch the algorithms we have developed in each case. For the full descriptions, we again refer to [18].

- covering set closure. Derived by searching through the space of solutions $\mathcal{S}$ or realisations $\mathcal{R}$.
- minimal covering set closure. Approximate derivation by refining a covering set closure, or exact (but expensive) computation of all minimal covering sets for a discrete data UCSP, inspired by Russian Doll Search [17].
- robust set closure. Derived by refinement from the full closure, as described above.
- most robust solution closure. Derived by branch-and-bound with forward checking.

In various forms, the most robust solution is studied in the literature for reasoning about CSPs under uncertainty: as we noted earlier, a single robust solution is the most common solution sought. The branch-and-bound method listed above is a variant of an algorithm for mixed CSPs [7]. Since a most robust solution is a singleton closure, we can apply other methods for constraint optimisation problems. On one hand, there are improvements and hybrids of branch-and-bound (e.g. [16]), and on the other hand there are algorithms based on alternative approaches, such as dynamic programming (e.g. [17]). The gain these more advanced methods have in common is that they exploit the structure of the problem, particularly to reuse computation on subproblems.

5.4 Case Study: Aerospace Component Control

For the aerospace planning problem outlined earlier, the UCSP is a reliable model. It captures both the uncertain but bounded evolution of e.g. temperature, and the model of anticipated contingencies. The full closure does not present an adequate resolution of the model, however, because the properties sought of the solution are that it should

provide one course of action for every anticipated contingency (a complete conditional plan). While reliable since it does cover all realisations, the full closure potentially contains many additional solutions for each realisation. This is a disadvantage because of the effort needed to compute and, crucially, store such a potentially large conditional plan. Instead, a covering set closure provides the sought resolution: it covers every realisation with (at least) one realisation with one course of action for that realisation.

For this application, the instance of the UCSP and the sought resolution combine to give an instance of no-observability mixed CSP [7]. While the two models differ in general, they converge when this type of resolution is sought. Thus, in [21] where different enumeration methods are presented to compute the covering set closure for this UCSP instance, an efficient decomposition method based on [7] can be adopted. The idea is to first derive a feasible plan for a given realisation, and then decompose the remainder of the UCSP by removing from future consideration all realisations covered by this plan. Although the minimal covering set is not guaranteed, heuristics guide the algorithm towards a covering set of smaller cardinality. Moreover, the algorithm can be made anytime in regard to the initial steps of the plan (variables in the UCSP) as well as coverage of the realisations.

Note that to execute the plan we must know which element of the closure to select, given knowledge of the occurred realisation r^*; this is precisely enumeration support information, i.e. $\Sigma(r^*)$. Fortunately, this support information is computed when deriving the covering set closure: the decomposition algorithm terminates with a list of plans (in the closure) that cover a given realisation r. In fact, this list contains at least one but may not contain all such plans, i.e. the support operator computed is not complete.

6 Conclusion and Future Work

The uncertain CSP extends the classical CSP to model incomplete and erroneous data, bounding it with a form of quantification. Its resolution is a closure, a set of potential solutions. While the full closure is suitable for UCSPs that model erroneous data (and where the user seeks a guarantee that the resolution contains at least one reliable solution), we showed that a UCSP model can be combined with different requirements on the solution in a meaningful and effective way, by deriving alternative closures such as the covering set, robust set, and most robust solution closures. We gave a formal definition of such closures in the UCSP framework and outlined means to derive them.

Extending the classical CSP, the UCSP is a restriction of the general quantified CSP. While the full closure corresponds to simple double existential quantification, other closures correspond to more complex queries. Nonetheless, by exploiting the restricted quantification, we can obtain efficient means to derive closures in various constraint domains. An interesting open question is whether specialised techniques for UCSPs can be used to enhance the resolution of generic QCSPs, just as the latter benefits the former. For the future, we are exploring which properties of constraint classes can be leveraged for the resolution forms given here for different closures: for instance, to derive an efficient specialised CET operator, as done for the full closure [20].

Lastly, for all the closures we have introduced, the form of the resolution derived for the UCSP corresponds to 'one-step' data uncertainty: *all* variable values must be decided either before or after (depending on the application) *all* the parameter values are known. Some applications have multiple-stage of decision interleaved with acquisition

of knowledge of the parameters. This leads to the concept of a 'multi-step' closure, which corresponds to a generalisation of the number and pattern of quantifiers seen in Section 4.4, moving such multi-step UCSPs closer to the general QCSP.

Acknowledgement. The authors thank Mark Wallace for much helpful discussion, and the reviewers for their suggestions. This work was performed while the first author was at IC–Parc, partially supported by the EPSRC under grant GR/N64373/01.

References

1. Y. Ben-Haim. Set-models of information-gap uncertainty: Axioms and an inference scheme. *J. Franklin Institute*, 336:1093–1117, 1999.
2. F. Benhamou and F. Goualard. Universally quantified interval constraints. In *Proc. of CP-2000*, LNCS 1894, pages 67–82, Singapore, Sept. 2000.
3. F. Boerner, A. Bulatov, P. Jeavons, and A. Krokhin. Quantified constraints: algorithms and complexity. In *Proc. of 17th Workshop on Computer Science Logic*, pages 58–70, 2003.
4. L. Bordeaux and E. Monfroy. Beyond NP: Arc-consistency for quantified constraints. In *Proc. of CP'02*, LNCS 2470, pages 371–386, Ithaca, NY, Sept. 2002.
5. M. Christie, E. Languénou, and L. Granvilliers. Modeling camera control with constrained hypertubes. In *Proc. of CP'02*, LNCS 2470, pages 618–632, Ithaca, NY, Sept. 2002.
6. D. Dubois, H. Fargier, and H. Prade. Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty. *Applied Intelligence*, 6:287–309, 1996.
7. H. Fargier, J. Lang, and T. Schiex. Mixed constraint satisfaction: A framework for decision problems under incomplete knowledge. In *Proc. of AAAI-96*, pages 175–180, Aug. 1996.
8. I. Gent, P. Nightingale, and K. Stergiou. QCSP-Solve: A solver for quantified constraint satisfaction problems. In *Proc. of IJCAI'05*, Edinburgh, UK, Aug. 2005.
9. I. P. Gent, P. Nightingale, and A. G. Rowley. Encoding quantified CSPs as quantified boolean formulae. In *Proc. of ECAI-04*, pages 176–180, Valencia, Spain, Aug. 2004.
10. E. Hebrard, B. Hnich, and T. Walsh. Robust solutions for constraint satisfaction and optimization. In *Proc. of ECAI-04*, pages 186–190, Valencia, Spain, 2004.
11. L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis*. Springer, 2001.
12. P. Kouvelis and G. Yu. *Robust Discrete Optimization and its Applications*. Kluwer, 1996.
13. N. Mamoulis and K. Stergiou. Algorithms for quantified constraint satisfaction problems. In *Proc. of CP'04*, LNCS 3258, pages 752–756, Toronto, Canada, Sept. 2004.
14. S. Manandhar, A. Tarim, and T. Walsh. Scenario-based stochastic constraint programming. In *Proc. of IJCAI'03*, pages 257–262, Acapulco, Mexico, Aug. 2003.
15. S. Ratschan. Solving existentially quantified constraints with one equality and arbitrary many inequalities. In *Proc. of CP'03*, LNCS 2833, pages 615–633, Sept. 2003.
16. C. Terrioux and P. Jégou. Bounded backtracking for the valued constraint satisfaction problems. In *Proc. of CP'03*, LNCS 2833, pages 709–723, Kinsale, Ireland, Sept. 2003.
17. G. Verfaillie, M. Lemaître, and T. Schiex. Russian doll search for solving constraint optimization problems. In *Proc. of AAAI-96*, pages 181–187, Portland, OR, Aug. 1996.
18. N. Yorke-Smith. *Reliable Constraint Reasoning with Uncertain Data*. PhD thesis, IC-Parc, Imperial College London, June 2004.
19. N. Yorke-Smith and C. Gervet. Certainty closure: A framework for reliable constraint reasoning with uncertainty. In *Proc. of CP'03*, LNCS 2833, pages 769–783, Sept. 2003.
20. N. Yorke-Smith and C. Gervet. Tight and tractable reformulations for uncertain CSPs. In *Proc. of CP'04 Workshop on Modelling and Reformulating Constraint Satisfaction Problems*, pages 142–158, Toronto, Canada, Sept. 2004.
21. N. Yorke-Smith and C. Guettier. Towards automatic robust planning for the discrete commanding of aerospace equipment. In *Proc. of 18th IEEE Intl. Symposium on Intelligent Control (ISIC'03)*, pages 328–333, Houston, TX, Oct. 2003.