Uncertain Constraint Optimisation 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. It resolution is a *closure*, a set of potential solutions. This paper extends the UCSP model to account for optimisation criteria, by defining the *uncertain CSOP*. The challenge is to combine optimisation (preferences over individual solutions) with a closure of a certain type (preference over sets of solutions) to a UCSOP. Unlike traditional CSOPs we need to compare closures (i.e. families of solutions) rather than just single solutions. We address this problem in a two stage process. First, non-dominated closures present the choice of solutions to a UCSOP; once one is chosen, second, its refinement to a redundancy-free or optimal closure balances reliability and optimality as the user specifies. We describe means to effectively perform these derivations by leveraging decision analysis under uncertainty and multi-criteria optimisation theory.

1 Introduction

Data uncertainties are inherent in the real world. Across numerous applications, real-world Large Scale Combinatorial Optimisation problems (LSCOs) are permeated by data uncertainty. Despite its successes, and its extensions to account for soft constraints, for example, the classical constraint satisfaction and optimisation problem (CSOP) is recognised as inadequate as a model of LSCOs with uncertain data.

The *uncertain CSP* (UCSP) was introduced in [19] to model LSCO problems with incomplete and erroneous data, without approximation of data or potential solutions. The resolution of a UCSP is a set of its potential solutions, called a *closure*. Depending on her application and the nature of the uncertainty, the user may be interested in one or more aspects of the potential solutions. For planning the control of aerospace components, for instance, which was modelled as a UCSP in [20], the resolution sought is a plan of operation for each anticipated environmental uncertainty. This corresponds to a *covering set closure*: a set of solutions that contains at least one solution (not necessarily all potential solutions) for each anticipated *realisation* of the data parameters.

Previous work on the UCSP does not account for preferences or soft constraints; or optimisation other than over the size of the closure or the number of realisations it covers. Preferences to maximise solution quality in the aerospace planning problem, for instance, were translated into hard satisfiability constraints on a minimum preference level. For such problems, where the user demands not only a reliable solution, but also one that meets specified, numerical objectives, the UCSP is incomplete as a model.

This paper introduces the *uncertain CSOP* to confront LSCOs with optimisation criteria. The key challenge is to define the semantics of a reliable and relevant resolution to the extended model, given preferences on individual potential solutions. On one hand there is the resolution sought in respect of the data uncertainty; on the other hand there are the user's optimisation criteria orthogonal to the uncertainty. Thus multiple and possibly conflicting criteria arise from the definition of a closure (in terms of supported potential solutions) and the value of the closure (in terms of the preferences). Further, this latter notion of the valuation of preferences over a closure must itself be defined.

After reviewing necessary background in Section 2, we extend the uncertain CSP to define the uncertain CSOP in Section 3, and present our approach to resolving a UCSOP: selecting first a type of closure, then the 'best' closure of that type (according to the user's optimisation criteria), and then, possibly, the 'best' elements of that closure.

The next two sections thus suppose the type of closure has been selected. To compare different closures of the chosen type (discussed second, in Section 5) we define objective functions *over closures,* based on the user's objective functions on potential solutions. We adapt criteria from classical decision making under uncertainty to make this definition. Then, when the sought closure is other than the full closure, comparing closures may be a multi-criteria problem, with criteria arising from the definition of the closure and from the objective functions over closures. Our approach is to compute the *non-dominated* closures, which form a Pareto frontier over closures, by adapting methods from multi-criteria optimisation theory. Once a non-dominated closure is chosen, its refinement to a *redundancy-free* or *optimal* closure (discussed in Section 4) balances reliability and optimality as the user specifies.

The certainty closure framework, developed for reliable inference around the UCSP model and closures, is distinguished by its enclosure approach to data uncertainty and solutions. In contrast to a CSOP, where the 'best' solution is sought, for a UCSOP the enclosure approach seeks the 'best' closure, i.e. we must reason about (sets of) sets of solutions. While reasoning with a UCSOP is thus distinguished from CSOPs and classical decision making under uncertainty, there are parallels between reasoning over closures and multi-criteria optimisation. Where possible we exploit these parallels, and adapt also CSOP algorithms as solving components. An overview of classical optimisation under uncertainty is in [17]; while [9] discuss multi-criteria optimisation.

Related work in CP to uncertainty includes approximation models and stochastic models [11] (where various metrics, including expected value, can be maximised for a single solution or policy), and robust single solutions [7]. Closest to our work are possibilistic approaches that simultaneously consider preferences and uncertainty [4,14]. The enclosure approach has been used within Operational Research, where optimisation in the context of uncertain data is addressed by [10], for instance.

2 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). 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 *realisation* of the data is a fixing of the parameters to values from their uncertainty sets. We say that any certain constraint corresponding to a realisation is a *realised* constraint.

In a CSOP, solutions are ordered, partially or totally, by optimisation criteria. It is usual for each criterion to be expressed as a (partial) function, the objective function $f_i : \mathcal{S} \rightarrow A$, where A is a partially ordered set of values. Without loss of generality, solutions are sought that satisfy all hard constraints and *minimise* the objective functions.

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 [19]:

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* $\mathcal{U}$*.*

We say that any certain CSP $\widehat{P}$, corresponding to a realisation of the parameters of P, is a *realised CSP*. 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. For reasons of space, in this paper we restrict ourselves to UCSPs with discrete data and logically independent parameters. Our examples are mostly arithmetic constraints. The UCSP model encompasses both continuous data and dependent parameters; as discussed in [18], their impact is mostly orthogonal to the optimisation issues considered here.

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_1 and c_3 are both uncertain constraints. □

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*.

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 state of the real world. In concrete terms, the form that reliable inference takes depends on two, linked issues: the requirements of the user and the nature of the 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.

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. An adequate closure includes all solutions relevant to the user's interest; reliability is unaffected when we disregard irrelevant potential solutions. Commonly-useful types of closures include [19]:

1. The *full closure*: the set of all solutions that each cover at least one realisation. Example usage: behaviour guarantee across all possible solutions; diagnosis of the reliability of other methods.
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, as in contingent planning.
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 [11].

Example 2. Let P be the UCSP of Example 1. The full closure of P in tuple notation is $(X_1, X_2) \in \{(3,1), (3,5), (4,2), (5,3)\}$; a covering set closure of minimal size is $(X_1, X_2) \in \{(3,1), (5,3)\}$, since this solution set covers all three realisations. $\square$

Remark. The most robust solution closure is a familiar concept in frameworks for uncertainty. For example, it is the solution sought to a no-observability mixed CSP [5]. Maximising robustness — whether by the metric of number of covered realisations (*coverage*), or by another, such as maximal expectation — is a common idea. Despite the attraction of robustness, it is not uncommon for the robust set closure to be empty, because of its strong requirement for solutions that cover *all* realisations.

A *support operator* tells us which realisations support which solutions. Its inverse tells us, dually, which realisations are covered by which solutions. Knowledge of such support information — the relationship between realisations and potential solutions — not only formally defines the different types of closures, but is essential for deriving them, which can be achieved by a variety of means, including one from another, transformation of the UCSP, and enumeration over realisations [18].

With respect to a constraint domain $\mathfrak{D}$, 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 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 2 (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*. If a support operator provides all realisations that support a set of solutions* S*, we say* Σ *is* complete *for* S. $\square$

Example 3. For Example 1, a support operator Σ_1 is defined by: $(3,1) \mapsto 2$, $(3,5) \mapsto 2$, $(4,2) \mapsto \{2,3\}$, and $(5,3) \mapsto \{3,4\}$. 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 $(5,3)$. $\square$

3 Uncertain CSOP

Not in every LSCO are all solutions equal. In a planning problem, for instance, the user might judge plans of shorter length to be preferable. In an uncertain CSP, this discrimination between resolutions to the model is manifest as a preference for some elements of a closure over others, or for some closures over others. Although we present a principled approach, much of the discussion cannot be specific, because optimisation criteria tie in so closely to the user's decision-making objective for a given LSCO problem. While we restrict ourselves mostly to a single optimisation criterion in this paper, moving from one to many criteria is a relatively smaller step. We first extend the definition of a UCSP in the natural way from a pure satisfaction problem to an optimisation problem:

Definition 3 (UCSOP). *An* uncertain constraint satisfaction and optimisation problem $\langle \mathcal{V}, \mathcal{D}, \Lambda, \mathcal{U}, \mathcal{C} \rangle$ *is a classical CSOP* $\langle \mathcal{V}, \mathcal{D}, \mathcal{C}, A \rangle$ *in which some of the constraints may be uncertain. That is, it is a UCSP with an objective function* $f : \mathcal{S} \times \mathcal{R} \to A$ *to be minimised, and a partially ordered set* A*.* $\square$

As in a CSOP, the objective function is a soft constraint; if we ignore it, a UCSOP is a UCSP, and therefore the results known for UCSPs apply immediately for UCSOPs. In particular, the definitions and derivations of the different closures apply. The complexity of solving a UCSP depends on the closure sought: e.g. deriving the full closure is Σ_2^p-hard [19]. Solving a UCSOP involves comparing closures, which increases the complexity up the polynomial hierarchy to the class PSPACE. Because of the uncertainty, it is to be expected that UCSOPs are computationally more challenging than CSOPs, which are in the class NP optimisation [3].

The user's assessment of the value of a solution might depend not only on the solution itself, but also on the realisations it covers. Thus the objective function f in Definition 3 is defined on $\mathcal{S} \times \mathcal{R}$, i.e. over both solution and realisation spaces.

Example 4. Let us consider adding an optimisation criterion to the discrete UCSP of Example 1. Consider the criterion of minimising the value of X_2. For a solution $s = (X_1, X_2)$, this gives the simple objective function $f(s) = X_2$. Note this f involves $\mathcal{S}$ only: it does not involve the realisations covered by s. The elements of the full closure are (in this case) totally ordered by the objective function: $(3,1) < (4,2) < (5,3) < (3,5)$. The solution with the best objective value is $\hat{s} = (3,1)$. Observe that $\hat{s}$ covers only one realisation ($\lambda_1 = 2$). There are two single solutions that cover the greatest number of realisations, $(4,2)$ and $(5,3)$; they have support $|\Sigma_1((4,2))| = |\Sigma_1((5,3))| = 2$. For them, observe that $(4,2) < (5,3)$. $\square$

The objective function of a CSOP is defined on variables and constants. For a UCSOP it may also include parameters. In this paper we will restrict ourselves to objective functions without parameters, and assume all values occurring in the objective are ground once the decision variables are chosen. However, it is worth noting that uncertainty in the objective function of a UCSOP can sometimes be rewritten as an (uncertain) constraint, so reducing the problem to one with certain objective function. As one instance, consider an interval linear system, a UCSOP with linear constraints, and with uncertainty sets given by real intervals. If we have a linear objective function $\min \sum_i \lambda_i X_i$, uncertainty in the objective is easily removed by adding the additional constraint $\sum_i \lambda_i X_i \leq Z$ for an auxiliary variable Z, and optimising $\min Z$. The general case, of course, will not reduce in this simple way.

3.1 Resolving a UCSOP

Given a single optimisation criterion, classical decision making [17] seeks one single solution, chosen by the rationale of minimising the objective function. The central tenant of solving a UCSP is that, unless specified by the user, no potential solution is a priori excluded. Since a closure is thus the resolution of a UCSP, given a single optimisation criterion, a rational approach is to seek a modified closure.

Deriving some types of closures is by itself already an optimisation problem: a minimal covering set, a maximal robust set, and a most robust solution. These closures have in common a criterion based on the amount of support; we say they are *optimisation-dependent*. For minimal covering sets, the support criterion is to minimise the cardinality of the closure. For maximal robust sets and most robust solutions, the criterion is to maximise the support of each element.

Thus, if we desire an optimisation-dependent closure, with even a single objective function, a UCSOP has the potential to be a multi-criteria optimisation problem. The two at best orthogonal and at worst competing criteria are: cardinality (the size of the closure) or support (measured by the support operator), and optimality (measured by the objective function). Analogously, multiple criteria are seen when seeking robust 'super' solutions to a CSOP [7]. The challenge is how to balance these two criteria.

Further, whether optimisation-dependent or not, all closures are defined formally in terms of support operators: e.g. the full closure is the set of solutions *each supported by* at least one realisation. Thus for *any* closure, in general any optimisation criterion may compete with reliability, which is defined in terms of support.

Our approach is the following: we give precedence to reliability, since, as part of the definition of a closure, it is the more fundamental. In analogy with a classical CSOP, firstly we desire solutions to the problem and only secondly do we evaluate them for optimality; so with a UCSOP, firstly we derive closures and only secondly do we evaluate them. We give greater precedence to the optimality criterion only if the user deliberately specifies a greater desire for optimality; only then is such a closure adequate.

In analogy, consider a soft temporal CSP with contingent events and preferences. Here a suitable notion of controllability [15] might put precedence for reliability over optimality. That is, we require hard temporal constraints to be satisfied, and secondly prefer solutions of higher quality according to the soft constraints.

What this means in practice is that we first select the appropriate type of closure for the problem, as if it were a UCSP with no optimisation criterion. We then consider the impact of the optimisation criterion:

- Suppose we have derived a closure of the desired type. The optimisation criterion means we can refine it by removing some elements. The more the user desires optimality over reliability, the more elements can be removed.
- Suppose instead we have only selected the desired type of closure. The optimisation criterion means we have a principled way to prefer some closures of this type to others of the type, if we extend the criterion to closures rather than individual potential solutions. However, if the type of closure is optimisation-dependent, such as a minimal covering set, then multiple criteria may arise when comparing closures.

A merit of this approach to optimisation under uncertainty is that we balance the two extremes: on one hand, deterministic approaches based on the worst case, and on the other, stochastic approaches based on probabilistic assumptions. As [1] point out, the former favours robustness over optimality, while the latter favours optimality over robustness. A relevant closure, to be established in the sequel, ensures a reliable solution (subsuming the benefits of a robust single solution when one exists); and it ensures an optimal solution, in the sense to be described.

4 Refining a Given Closure

In this section we suppose the type of closure has been chosen and one closure of the type has been derived, and now we are given an optimisation criterion. The resulting objective function means there is now a reason to prefer some elements of a closure to others. Thus we describe how to refine a closure with respect to an objective function.

Unless the user specifies it, we cannot simply pick the most preferred element of a closure according to the objective function. The reason is that it is the whole closure that provides a reliable solution to the problem; any one element (or more generally, any subset) need not necessarily be a reliable solution.

Nonetheless, the optimisation criterion still gives a potential reason to prune a closure: when one element makes another *redundant*. Definition 4 says that one solution is made redundant by another if the latter covers at least the same realisations (support) and is preferred according to the objective function (optimality).

Definition 4 (Redundant solution). *Let $S \subseteq \mathrm{Cl}(P)$ be a closure of UCSOP P, and $s_1, s_2 \in S$ be elements of S. Let Σ denote a complete support operator for P. s_2 is made* redundant *by s_1 if $\Sigma(s_2) \subseteq \Sigma(s_1)$ and $f(s_1) < f(s_2)$.* □

Example 5. In Example 4 the solutions $(3, 1)$ and $(3, 5)$ cover the same realisations. Thus, with respect to the objective function $f(s) = X_2$, $(3, 1)$ makes $(3, 5)$ redundant. Any covering set that includes $(3, 1)$ gains nothing by also including $(3, 5)$. Thus we can refine such a covering set closure by removing $(3, 5)$ without compromising reliability.

Redundancy applies to any closure, although for singleton closures clearly it is trivial. Note that, as a consequence of their definitions, both a most robust solution and a minimal covering set are redundancy-free. There is no need to retain redundant solutions in a closure, unless the user specifies that regardless she wants all single solutions. In the absence of any specification by the user to the contrary, we say that a closure is an adequate solution to a UCSOP only if it contains no redundant elements:

Definition 5 (Redundancy-free). *In the context of a UCSOP, we say that a closure is* redundancy-free *iff it contains no redundant elements; otherwise it is* redundant. □

Example 6. In Example 4, the covering set $\{(3, 1), (3, 5), (5, 3)\}$ is a redundant closure, since $(3, 5)$ is made redundant by $(3, 1)$; $\{(3, 1), (5, 3)\}$ is redundancy-free. □

Hence, given a closure S and an objective function, we prune the redundant elements from the closure to yield the redundancy-free refined closure $S' \subseteq S$. S' is the smallest subset of S that a priori is a reliable solution to the problem. Nonetheless, the user may specify her primary desire for an optimal single solution (even though it might not cover every realisation), in the same way as she might ask for a most robust solution closure (even though it might not cover every realisation). As stated earlier, only with such a specification can we give optimisation precedence over reliability, and say such a solution is adequate. In particular, suppose the user desires the minimal elements of the full closure according to the objective function, even though this minimal set will not necessarily cover all realisations. Here is the trade-off between robustness and optimality: between probability of covering all realisations and the value of the solution.

We can translate the user's restriction on the full closure into a closure of another type: the *optimal closure* $\{s \in \mathrm{Cl}(P) : f(s) \text{ minimal}\}$. The optimal closure prefers elements of the full closure with respect to the objective function, parallel to how a robust set closure prefers elements with respect to their support.[3] Generalising, if the user requires the optimal elements of any closure, we can translate this requirement into a demand for the optimal closure of that type:

Definition 6 (Optimal closure). *Given a closure S of type $\mathfrak{t}$, an* optimal $\mathfrak{t}$ closure *is the subset of S of elements minimal under an objective function f.* □

An optimal closure of a given type need not be a closure of that type. For example, from a covering set closure S comes an optimal covering set closure S', but S' need not be a covering set. Note also that every optimal closure is redundancy-free, but not every redundancy-free closure is optimal. In contrast to an optimal closure, which places optimality before support, a general redundancy-free closure places support before optimality: it prunes only those elements whose omission does not ameliorate the coverage of the closure, i.e. the number of realisations covered.

Example 7. In Example 4, the covering set $\{(3, 1), (5, 3)\}$ is redundancy-free, but is not an optimal covering set closure because $f((5, 3)) = 3 > 1$ is not minimal. An optimal covering set closure is $\{(3, 1)\}$, which is a singleton closure in this case. Since it does not cover all realisations, it is not a covering set closure. □

Example 8. Consider a problem arising in routing of uncertain traffic demands in a network [1], suitable for modelling as a UCSOP. Here, the desired closure is a robust set — each proposed routing must hold for all realisations of the demands within the uncertainty set — and the routing should be of minimum cost. Thus an optimal robust set is adequate to the user as a reliable solution for the problem. Operationally, [1] compute one member of such a closure directly, using column generation. □

5 Choosing Between Different Closures

To begin with in this section we again suppose the type of closure has been chosen. The last section assumed one closure of the chosen type had been selected. We now ask which closure should be selected: which closure of the type is 'best' given an optimisation criterion? In other words, having considered preferring some elements of a closure to others, we now consider preferring some closures (as subsets of $\mathrm{Cl}(P)$) to others.

There are two aspects to address: (1) how to define a criterion to evaluate a closure, given the user's preferences over individual potential solutions; and (2) how to compare closures, given this criterion, which might be in conflict to the criterion that comes from the definition of the type of closure, i.e. how to approach the multi-criteria optimisation problem of choosing between closures.

[3] Analogously, consider super solutions to a classical CSOP: an optimal closure corresponds to the most robust optimal super solution, and a (non-optimal) redundancy-free closure corresponds to the optimal robust super solution [7].

Example 9 (Example 4 continued). In Example 4 the two minimal (and so redundancy-free) covering sets are $S_1 = \{(3,1),(5,3)\}$ and $S_2 = \{(4,2),(5,3)\}$. First, let us compare them by the sum of the number of realisations covered. $(3,1)$ covers one realisation ($\lambda_1 = 2$); $(4,2)$ covers two realisations ($\lambda_1 = 1, \lambda_1 = 2$); and $(5,3)$ covers two realisations ($\lambda_1 = 2, \lambda_1 = 3$). Thus S_1 has a cumulative coverage (the number of covered realisations) of $1+2=3$ and S_2 of $2+2=4$, so S_2 is better. This comparison focuses on the heuristic of maximising the amount of support.

Second, compare the minimal covering sets by the sum of the objective function f on their elements. Then S_1 has a cumulative value of $1+3=4$ and S_2 of $2+3=5$. In contrast to the first, by this second ordering, S_1 is better (since we minimise f). This comparison focuses on the optimality criterion.

Third, compare the closures by the sum of the best solution they give for each realisation. S_1 covers $\lambda_1 = 2$ by $(3,1)$ and the other realisations by $(5,3)$, scoring $1+3+3=7$. S_2 covers $\lambda_1 = 2$ by $(4,2)$, $\lambda_1 = 3$ by $(4,2)$ and $(5,3)$, and $\lambda_1 = 4$ by $(5,3)$, scoring $2+2+3=7$. Now the two minimal covering sets are incomparable. This comparison seeks to balance both reliability and optimality.

Lastly, consider the covering set $S_3 = \{(3,1),(4,2),(5,3)\}$. This set is not minimal, since its cardinality is three. However, according to the last metric, it scores $1+2+3=6$, which makes it better than S_1 and S_2. We call S_3 the *optimal for each realisation* closure, since it contains the best solution for each closure with respect to the objective function. It shows that the support criterion (minimise cardinality) and the optimality criterion (minimise some lifted function of f) are opposed, and so we have a multi-criteria problem. □

5.1 Extending an Objective Function to a Closure

We first must define means to ascribe numerical values to closures, so that we have means to compare them with respect to the objective function f specified by the user. That is, we must lift f from single solutions to closures, i.e. from $\mathcal{S} \times \mathcal{R}$ to $\mathcal{P}(\mathcal{S}) \times \mathcal{R}$, and project it from $\mathcal{P}(\mathcal{S}) \times \mathcal{R}$ to $\mathcal{S}$. The basis for doing so is found by reviewing the criteria known in decision making under uncertainty. We recall them briefly and then present different means to define f on closures based upon them.

Recall from [17] that a *valuation matrix* that associates a value v_{ij} to each action a_i and each future outcome Θ_j. The decision problem is to decide among the actions (which we can assume are known) in the presence of a lack of knowledge about which outcome will occur. In terms of a UCSOP, the actions a_i are consistent tuples for the variables, and the outcomes Θ_j are the feasible realisations of the parameters. Thus the valuation matrix is nothing more than the objective function enumerated over $\mathcal{S}$ and $\mathcal{R}$.

The literature contains many criteria for decisions under uncertainty, when seeking a single solution rather than a set of solutions. For an action a_i, the criteria specify the value to assign to the action with respect to the objective function. Since we are minimising f, the optimal action is the one that minimises this value, i.e. argmin_{a_i}. The criteria, first, specify what value to give to a single solution in the light of the uncertainty. In our notation, they specify how to project f from $\mathcal{S} \times \mathcal{R}$ to $\mathcal{S}$. Second, they specify how to select a single solution s that optimises the projected objective function $f(s)$. The criteria differ most importantly in how conservative they are. Beginning with the most optimistic, simply suppose the most favourable outcome will occur. That is,

$$\min_{a_i} \min_{\Theta_j} v_{ij} \tag{1}$$

The *Laplace criterion* [17] is also optimistic. It assumes the outcomes are equally likely, and converts the problem to a decision under risk, computing expected utility:

$$\min_{a_i} \big(\frac{1}{m} \sum_{j=1}^{m} v_{ij} \big) \tag{2}$$

The most pessimistic criterion supposes that the least favourable outcome will occur. The *minimax criterion* acts conservatively to avoid the worst actions:

$$\min_{a_i} \max_{\Theta_j} v_{ij} \tag{3}$$

The *spread* takes a middle ground, as the difference of the most pessimistic and most optimistic criteria:

$$\min_{a_i} \big(\max_{\Theta_j} v_{ij} - \min_{\Theta_j} v_{ij} \big) \tag{4}$$

Also neither purely optimistic nor pessimistic, the *minimax regret criterion* computes the *regret matrix* that associates the opportunity cost of an action: $r_{ij} = v_{ij} - \min_{a_k} v_{kj}$. Regret expresses the difference, in hindsight, between the best decision and the decision taken. The decision criterion is then to apply minimax to the regret matrix:

$$\min_{a_i} \max_{\Theta_j} r_{ij} \tag{5}$$

Variants of regret, such as percentage regret, are defined in robust optimisation [10].

Finally, the *Hurwicz criterion* [17] is parametrised by an index of optimism $\alpha \in [0, 1]$: 0 is pessimistic, 1 is optimistic:

$$\min_{a_i} \big(\alpha \min_{\Theta_j} v_{ij} + (1 - \alpha) \max_{\Theta_j} v_{ij} \big) \tag{6}$$

We must extend these criteria in order to apply them to closures, because in general a closure will have more than one element. Thus we need to lift the evaluation of f from a single solution s to a set of solutions, a closure S. It is clear there is more than one answer: for example, as in Example 9, we could sum the values for the elements or we could take the least value. The most suitable choice of the above means to adopt depends on the criteria of user for the problem; we present eight such alternatives. Let $S = \{s_1, \ldots, s_N\}$ be a set of potential solutions, and Σ be a complete support operator.

1. Take the minimum value of f over the individual elements:

   $$f(S) = \min_{i=1,\ldots,N} f(s_i) \tag{7}$$

   This is $\times$ from the fuzzy semiring [2]; the egalitarian definition of welfare in utility theory [13]. Since we minimise f, this means of lifting f onto S is the most optimistic; it corresponds to the most favourable criterion (1) above.

2. Take the maximum value of f over the individual elements:

$$f(S) = \max_{i=1,\ldots,N} f(s_i) \tag{8}$$

This is the least optimistic alternative; it corresponds to the minimax criterion (3) and gives us a hard upper bound on the optimum.

3. Take the spread of the values:

$$f(S) = \min_{i=1,\ldots,N} f(s_i) - \max_{i=1,\ldots,N} f(s_i) \tag{9}$$

This corresponds to (4), and also (shown by rearranging the equation) to the regret criterion (5).

4. Use the Hurwicz criterion with an index of optimism $\alpha \in [0, 1]$:

$$f(S) = \alpha \min_{i=1,\ldots,N} f(s_i) + (1 - \alpha) \max_{i=1,\ldots,N} f(s_i) \tag{10}$$

This corresponds to (6).

5. Sum the support of the individual elements:

$$f(S) = \sum_{i=1,\ldots,N} \frac{1}{|\Sigma(s_i)| + 1} \tag{11}$$

Since we are minimising f but support is usually maximised, we use the reciprocal of $|\Sigma(s_i)|$, the number of realisations that support solution s_i. Note the $+1$ in the denominator to give correct results if s has a support metric 0. If lesser support is preferred, we simply use $|\Sigma(s_i)|$ rather than its reciprocal.

6. Sum the values of f on the individual elements:

$$f(S) = \sum_{i=1,\ldots,N} f(s_i) \tag{12}$$

This is $\times$ from the *weighted semiring* [2]; the utilitarian definition of welfare in utility theory [13]. It corresponds to the Laplace criterion (2).

7. Sum the values of f on the individual elements, weighted by their support:

$$f(S) = \sum_{i=1,\ldots,N} \frac{f(s_i)}{|\Sigma(s_i)| + 1} \tag{13}$$

If we view the amount of support as defining a likelihood of occurrence (a possibility distribution function), then (13) corresponds to an expected value criterion.

8. Take the best value of f on the elements that cover each realisation:

$$f(S) = \sum_{j=1,\ldots,M} \max_{s_i \text{ covers } r_j} f(s_i) \tag{14}$$

We call this *optimal for each realisation*. It defines an extension of the covering set closure, where not only does the closure contain at least one solution for each realisation, but at least one optimal solution for each.

Table 1. Comparison of closures by various metrics

	cardinality	min	max	spread	Hurwicz ($\alpha = 0.5$)	support	sum	weighted	best
S_1	2	1	3	2	2	$\frac{1}{3}$	4	$\frac{5}{2}$	7
S_2	2	2	3	1	$\frac{3}{2}$	$\frac{1}{4}$	5	$\frac{5}{2}$	7
S_3	3	1	3	2	2	$\frac{1}{5}$	6	$\frac{7}{2}$	6

To evaluate (11) and (13) we require $\Sigma(s_i)$ for each s_i, known as *enumeration support information* [18]. To evaluate (14) we require $\Sigma^{-1}(r_j)$ for each realisation $r_j \in \Sigma(S)$, where Σ^{-1} is a relation inverse of Σ; enumeration support information is certainly enough for this.

Example 10 (Example 9 revisited). In Example 9 we compared by different metrics the three covering set closures: $S_1 = \{(3,1),(5,3)\}$ and $S_2 = \{(4,2),(5,3)\}$ (both minimal), and $S_3 = \{(3,1),(4,2),(5,3)\}$. Table 1 evaluates the three closures by all of the above metrics, and compares them also with the support criterion of the cardinality of the sets. We see that there are metrics by which each of the closures are strictly best. A decision between the three closures will depend on the criteria of the user. Moreover, if we seek a minimal covering set, which is an optimisation-dependent closure, then we have multiple criteria. If the optimality criterion is *best*, for instance, then the support criterion (*cardinality*) and optimality criterion are opposed to each other. □

5.2 Comparing Closures of the Same Type

Summarising, based on the objective function of a UCSOP, we have defined means of numerically comparing closures of any one type. This enables us to choose a 'best' closure, by deriving one or all closures of the type that minimise the corresponding objective function (7)–(14). This is analogous to looking for the elements of a closure that minimise the original objective function: it is the closure equivalent of the optimal elements. As we stated, the most suitable choice of (7)–(14) depends on the criteria of user for the specific LSCO problem at hand.

However, choosing the 'best' closure requires more than just minimising $f(S)$. Example 10 illustrates, for minimal covering sets, how the addition of even one optimisation criterion to a UCSP can lead to a multi-criteria optimisation problem. The two, essentially orthogonal, sources of criteria arise from support (or cardinality) and optimality (defined by the chosen $f(S)$). As the example showed, when optimality and support objectives are opposed, they generate a trade-off, resulting in a multi-criteria optimisation problem to choose a closure. The multiple criteria are reflected by multiple objective functions, which we write as f_i, reserving f_0 for the support criterion.

Of the approaches to multi-criteria optimisation [16], the Pareto frontier fits naturally with the UCSP, because it is based on providing the user with information to enable her to take an informed decision. A Pareto frontier *of closures* is a plausible set of closures which the user might examine for the trade-off of the criteria. The selected closures can then be refined to their redundancy-free or optimal versions.

For a given type of closure, a *Pareto frontier of closures* is a set of non-dominated closures. One closure S *dominates* another S' iff $f_i(S) \leq f_i(S')$ for each objective function f_i and there exists at least one f_k s.t. $f_k(S) < f_k(S')$; a closure is *non-*

dominated if there exists no closure that dominates it. A closure in the frontier cannot be improved with respect to any criterion without deteriorating it with respect to another.[4]

The alternatives to the Pareto frontier translate the multi-objective problem into a single-objective problem or problems. Widely used for instance is a weighted sum of the criteria. It is perhaps less natural than a frontier, because (1) deciding the weights beforehand is often unclear; and (2) it gives extremal solutions whereas the frontier provides a range of balanced solutions.

Example 11 (Example 9 concluded). Let us say the cardinality $f_0(S) = |S|$, the minimum value $f_1(S) = \min_{s_i} f(s_i)$, and the sum $f_2(S) = \sum_{s_i} f(s_i)$ are the three criteria the user is concerned with in Example 9. Note that the former comes from the support criterion, while the latter two come from explicit optimisation criteria. Referring to Table 1, observe that S_2 is dominated by S_1 but neither S_1 nor S_3 dominate each other. Thus the Pareto frontier is the set $\{S_1, S_3\}$. Refining both with respect to f, we see that S_1 is the redundancy-free form of S_3. Hence we offer the user the closure S_1 as the resolution of the UCSOP. □

Computing the Frontier. To make the discussion more concrete, we now consider how to perform an efficient comparison. We can compose the problem of evaluating $f(S)$ as a meta CSP. The sole variable is the closure S sought; its domain is the set of all closures of the UCSP. The constraints specify that S is a closure of the sought type, and there is an objective function according to the criterion on closures, i.e. $f(S)$.

Without domain-specific knowledge, the natural algorithm to use is branch-and-bound. The search must be complete to ensure we find an $f(S)$-minimal closure; it may be modified to give one or all such closures. To reduce the computational cost, we can integrate problem decomposition methods: e.g. the hybrid of branch-and-bound and tree decomposition [8]. If the cost is still too great, we may optionally give up completeness (and so optimality) by using heuristics, or incomplete methods such as local search. Below, we discuss further the minimal covering set and most robust solution closures.

Once we can compute $f_i(S)$ for each i, methods in the literature to compute Pareto frontiers [16] apply directly, if we replace ‘solution’ by ‘closure’. A common approach is to generate a sample of points on the frontier, by either defining a parameterised, scalar objective (such as a weighted sum) called a *generator*, and varying its parameters; or by finding non-dominated points by local search; both are surveyed in [12]. Sampling the frontier of closures leads to approximation, a topic for future work.

Since approximation might not be desired, we also highlight two methods that compute the whole frontier. The first method is to employ generators in a CSP. Under suitable restrictions on the solution space, some carefully chosen generators are complete: they generate the whole Pareto frontier as their parameters vary. Further, some generators have analytical form which can be expressed as a constraint. Thus if we add such a generator constraint to our meta CSP defined above, we use the generator directly as part of the CSP solving. The second set of methods are specific for CSPs; they can be viewed as extensions of branch-and-bound. In particular, [6] combine branch-and-bound and an efficient representation of the frontier with quadtrees.

[4] Our definition of dominance is in line with the standard definition; it requires that the objective functions be scalar and monotone [9]. The idea of non-dominated closures is similar to redundancy-free solutions (Definition 4), but differs in that there is no mention of support. In fact, support is implicit because the definition is parametrised by the type of closure.

Covering set closures Example 11 indicates that for covering set closures, the principle trade-off is between the size of the set and its optimality. At one extreme is a covering set closure of minimal size. This is favourable because: (1) it requires less space to store; (2) fewer elements must mean each is more robust (on average); and (3) the closure changes less when it is refined as knowledge about the realisations is acquired.

At the other extreme is a covering set that contains an optimal solution for each realisation. This is favourable by the metric (14), and if the closure is refined as the realisations are reduced to a single possibility, it gives us an optimal solution. However, such an *optimal for each realisation* closure is often likely to be too large to be useful. The aerospace planning problem is a case study of balancing the criteria [20].

Most robust solution closures A most robust solution closure is a singleton closure (a single potential solution) and as such there is work in the literature. First consider the case where there is no objective function, i.e. a UCSP. To derive a most robust solution closure is a single-criteria optimisation problem where the objective is to maximise robustness. In the discrete case, existing branch-and-bound and forward-checking algorithms can be readily adapted by removing probabilities [5,11].

Second, the main case is a UCSOP where there is an objective function. This gives a multi-criteria optimisation problem in which the criterion arising from support is the number of covered realisations. Since the sought closure is a singleton, classical multi-criteria optimisation methods directly apply: the Pareto frontier of closures reduces to the classical Pareto frontier of solutions. Local search methods such as multi-objective simulated annealing are known to be effective.

5.3 Comparing Closures of Different Types

So far we have supposed the type of closure has been chosen. We now briefly discuss comparing closures of different types. From an optimisation criterion, the objective function assigns a numerical value to each closure, according to one of the above means. Closures of different types can be compared with respect to their values, just as closures of the same type. Moreover, we can go on to define domination between closures.

The advantage is a well-founded, quantitative comparison of heterogeneous types of closure. It means we can resolve a UCSOP without deciding what types of closure might best meet the user's requirements and then trying each; by analogy, rather than generate-and-test we integrate the evaluation with the generation. However, the important caveat is that different types of closure provide very different types of reliable solution to a LSCO, in general, and care must be taken that their comparison is coherent.

6 Conclusion and Future Work

The uncertain CSP extends the classical CSP to model incomplete and erroneous data. Its resolution is a closure, a set of potential solutions. In this paper we extended the UCSP model to the uncertain CSOP, to account for user preferences and other criteria that can be modelled with an objective function. To do so, we extended the notion of a closure to confront LSCOs with optimisation criteria. Non-dominated closures present the choice of solutions to a UCSOP; once one is chosen, its refinement to a redundancy-free or optimal closure balances reliability and optimality as the user specifies. Consequently, we can model problems where the user demands not only a reliable solution, but also one that meets specified, numerical objectives.

As an extension of the classical CSP, rather than e.g. valued CSP [2], the UCSOP model presented is focused on hard constraints. Future work is to consider soft constraints within a UCSOP. However, the UCSOP can already accommodate softness in as far as it can be described by an objective function, e.g. minimising the weight of violated constraints. Similarly, future work includes uncertainty in the objective function of a UCSOP, beyond what can be rewritten out of the objective into a constraint.

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

References

1. W. Ben-Ameur and H. Kerivin. Routing of uncertain demands. *Optimization and Engineering*, 3:283–313, 2005.
2. S. Bistarelli, H. Fargier, U. Montanari, F. Rossi, T. Schiex, and G. Verfaillie. Semiring-based CSPs and valued CSPs: Basic properties and comparison. In *LNCS 1106*. 1996.
3. N. Creignou, S. Khanna, and M. Sudan. *Complexity classifications of Boolean constraint satisfaction problems*. SIAM Press, Philadelphia, PA, 2001.
4. D. Dubois, H. Fargier, and H. Prade. Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty. *Applied Intelligence*, 6:287–309, 1996.
5. 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.
6. M. Gavanelli. An algorithm for multi-criteria optimization in CSPs. In *Proc. of ECAI-02*, pages 136–140, Lyon, France, July 2002.
7. 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.
8. P. Jégou and C. Terrioux. Hybrid backtracking bounded by tree-decomposition of constraint networks. *Artificial Intelligence*, 146(1):43–75, 2003.
9. R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives*. Cambridge, 1993.
10. P. Kouvelis and G. Yu. *Robust Discrete Optimization and its Applications*. Kluwer, 1996.
11. S. Manandhar, A. Tarim, and T. Walsh. Scenario-based stochastic constraint programming. In *Proc. of IJCAI'03*, pages 257–262, Acapulco, Mexico, Aug. 2003.
12. P. Meseguer, N. Bouhmala, T. Bouzoubaa, M. Irgens, and M. Sánchez. Current approaches for solving over-constrained problems. *Constraints*, 8(1):9–39, 2003.
13. H. Moulin. *Axioms for Cooperative Decision Making*. Cambridge University Press, 1988.
14. M. S. Pini, F. Rossi, and K. B. Venable. Possibility theory for reasoning about uncertain soft constraints. In *Proc. of ECSQARU 2005*, Barcelona, Spain, July 2005.
15. F. Rossi, K. B. Venable, and N. Yorke-Smith. Controllability of soft temporal constraint problems. In *Proc. of CP'04*, LNCS 3258, pages 588–603, Toronto, Canada, Sept. 2004.
16. R. Steuer. *Mulitple Criteria Optimization*. Wiley, New York, 1986.
17. H. Taha. *Operations Research: An Introduction*. Prentice Hall, New Jersey, 1997.
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. 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.