Valued Constraint Satisfaction Problems: Hard and Easy Problems*

Thomas Schiex
INRA
Chemin de Borde Rouge
31326 Castanet-Tolosan Cedex
France
tschiex@toulouse.inra.fr

Hélène Fargier
IRIT
118, Route de Narbonne
31068 Toulouse Cedex
France
fargier@irit.fr

Gérard Verfaillie
CERT-ONERA
2 Av. Édouard Belin
31055 Toulouse Cedex
France
verfail@cert.fr

Abstract

In order to deal with over-constrained *Constraint Satisfaction Problems*, various extensions of the CSP framework have been considered by taking into account costs, uncertainties, preferences, priorities…Each extension uses a specific mathematical operator ($+, \max\ldots$) to aggregate constraint violations.

In this paper, we consider a simple algebraic framework, related to Partial Constraint Satisfaction, which subsumes most of these proposals and use it to characterize existing proposals in terms of rationality and computational complexity. We exhibit simple relationships between these proposals, try to extend some traditional CSP algorithms and prove that some of these extensions may be computationally expensive.

1 Introduction and related works

The CSP framework provides a very convenient framework for representing and solving various problems related to AI and OR (scheduling, assignment, design…). When a real problem is casted in the CSP framework, different types of knowledge have to be dealt with:

- **Hard constraints**: physical properties (eg. spatial or temporal constraints), which have to be necessarily satisfied, are naturally represented as constraints;
- **Preferences**: properties which should be satisfied "*when possible*" (due dates, user preferences, cost…) are either represented as constraints or simply ignored;
- **Uncertainties**: properties that are relevant in some situations which cannot be predicted with certainty; such properties are then ignored, or represented as constraints.

Thus, such soft constraints can be either ignored, which naturally leads to a poor mean quality of the solutions or represented as hard constraints which may yield an inconsistent CSP. A better solution is to take the violation of these constraints into account in a *specific* criterion that should be minimized.

Various proposals have been made in this direction, extending classical CSP in order to express "*soft*" constraints with a dedicated semantics in terms of priorities [Schiex, 1992; Borning *et al.*, 1989], preference degrees [Rosenfeld *et al.*, 1976; Martin-Clouaire, 1992; Dubois *et al.*, 1993; Ruttkay, 1994], costs [Shapiro and Haralick, 1981; Dechter *et al.*, 1990; Freuder and Wallace, 1992] or probabilities [Rosenfeld *et al.*, 1976; Fargier and Lang, 1993]. The specific nature of the criterion optimized allows dedicated branch and bound algorithms to be defined.

In these approaches, hard constraints and preferably satisfied/uncertain constraints are expressed as constraints but a valuation (usually a number) is associated to each constraint c, or each tuple t of a constraint. This valuation expresses the impact of violating the constraint c or using the tuple t on the quality of the solution. These valuations are combined using an operator that gives them a specific semantics. For example, in [Schiex, 1992], the valuations of violated constraints are combined using a $\max$ operator, which gives the valuations an interpretation in terms of priorities, while in [Shapiro and Haralick, 1981], the valuations are numbers combined using addition, with an obvious interpretation as costs.

In this paper, rather than choosing a specific set for expressing valuations and a specific operator, we observe that an ordered commutative monoïd (an ordered set with an operator satisfying some properties), is enough to encompass most existing CSP extensions. The valuations are taken from the set of the monoïd, combined using its operator and compared using the order. For the sake of simplicity, we consider that valuations are only associated to constraints.

Our aim is then to use this abstract framework to provide general algorithms and properties, to bring to light relations between previous proposals and to identify where the difficult problems are, and what property makes them difficult[1].

The next section defines the valued CSP (VCSP) framework, rapidly justifies the algebraic structure used and gives some simple properties. Section 3 describes previous proposals as VCSP and shows how they relate to each other in terms of rationality and complexity. Section 4 considers the generic extension of different *Backtrack* based algorithms and ends with preliminary experiments on the related observed complexities of some types of VCSP.

*This work has been partially funded by the French Centre National d'Études Spatiales and by the European Euclid project CALMA.

[1] This work is therefore related to [Shafer, 1991] which considers an axiomatic framework where hyper-tree structured problems are solved efficiently and to [Minoux, 1976; Bistarelli *et al.*, 1995] where a semi-ring is used to study the possible generalization of shortest path and k-consistency enforcing algorithms respectively.

2 Towards valued CSP

A classical CSP is defined by a set $X = \{x_1, \ldots, x_n\}$ of variables, each variable x_i having an associated finite domain d_i. A constraint $c = (X_c, R_c)$ is defined by a set of variables $X_c \subset X$ and a relation R_c between the variables of X_c *i.e.*, a subset of the Cartesian product $\prod_{x_i \in X_c} d_i$. A CSP is noted (X, D, C), where D is the set of the domains and C the set of the constraints. A solution of the CSP is an assignment of values to the variables in X such that all the constraints are satisfied: for each constraint $c = (X_c, R_c)$, the tuple of the values taken by the variables of X_c belongs to R_c.

To express the fact that a constraint may eventually be violated, we annotate each constraint with a valuation taken from a set of valuations E equipped with the following structure:

Definition 1 *A valuation structure* $(E, \circledast, \succ)$ *verifies:*

- E *is a set, whose elements are called valuations, which is totally ordered by* $\succ$*, with a maximum element noted* $\top$ *and a minimum element noted* $\bot$*;*
- $\circledast$ *is a commutative, associative closed binary operation on* E *that verifies:*
  - Identity*:* $\forall a \in E, a \circledast \bot = a$*;*
  - Monotonicity*:* $\forall a, b, c \in E,$
    $(a \succcurlyeq b) \Rightarrow \big((a \circledast c) \succcurlyeq (b \circledast c)\big)$*;*
  - Absorbing element[2]*:* $\forall a \in E, (a \circledast \top) = \top$.

This structure of totally ordered commutative monoïd with a monotonic operator is also known in uncertain reasoning, E being restricted to $[0, 1]$, as a “triangular co-norm” [Dubois and Prade, 1982]. In the rest of the paper, we implicitly suppose that the computation of $\succ$ and $\circledast$ are always polynomial in the size of their arguments.

2.1 Justification and properties

The ordered set E allows different levels of violations to be expressed. Commutativity and associativity guarantee that the valuation of an assignment depends only on the set of the valuations of the violated constraints, and not on the way they are aggregated. The element $\top$ corresponds to unacceptable violation and is used to express *hard* constraints. The element $\bot$ corresponds to complete satisfaction. These maximum and minimum elements can be added to any totally ordered set, and their existence is supposed without any loss of generality. Monotonicity guarantees that the valuation of an assignment that satisfies a set B of constraints will always be as good as the valuation of any assignment which satisfies a subset of B. Two additional properties will be considered later because of their influence on algorithms and computation:

Strict monotonicity ($\forall a, b, c \in E$, if $(a \succ c), (b \neq \top)$ then $(a \circledast b) \succ (c \circledast b)$) guarantees that any modification in a set of valuations that does not contain $\top$ passes on the aggregation, via $\circledast$, of these valuations: the fact that something can be locally improved can not be globally ignored. This type of property is usual in multi-criteria theory, namely in social welfare theory [Moulin, 1988].

Idempotency ($\forall a \in E, a \circledast a = a$) is fundamental in all CSP algorithms that enforce k-consistency since it guarantees that a constraint that is satisfied by all the solutions of a CSP can be added to the CSP without changing its meaning. Idempotency is incompatible with strict monotonicity as soon as E has more than two elements[3]. It should be noted that the only idempotent operator in a valuation structure is $\max$[4].

2.2 Valued CSP

A valued CSP is then simply obtained by annotating each constraint of a classical CSP with a valuation denoting the impact of its violation[5] or, equivalently, of its rejection from the set of constraints.

Definition 2 *A valued CSP is defined by a classical CSP* (X, D, C)*, a valuation structure* $S = (E, \circledast, \succ)$*, and an application* φ *from* C *to* E*. It is noted* (X, D, C, S, φ)*.* $\varphi(c)$ *is called the valuation of c.*

An assignment A of values to some variables $Y \subset X$ can now be simply evaluated by combining the valuations of all the violated constraints using $\circledast$:

Definition 3 *Given a VCSP* $\mathcal{P} = (X, D, C, S, \varphi)$ *and an assignment* A *of the variables of* $Y \subset X$*, the valuation of* A *with respect to the VCSP* $\mathcal{P}$ *is defined by:*

$$\mathcal{V}_{\mathcal{P}}(A) = \mathop{\circledast}_{\substack{c \in C, X_c \subset Y \\ A \text{ violates } c}} [\varphi(c)]$$

The semantics of a VCSP is a distribution of valuation on the assignments of X (potential solutions). The problem considered is to find an assignment A with a *minimum* valuation. The valuation of such an optimal solution will be called the CSP valuation. It provides a *gradual* notion of inconsistency, from $\bot$, which corresponds to consistency, to $\top$, for complete inconsistency.

Our notion of VCSP is equivalent to [Freuder and Wallace, 1992] view of partial consistency. Indeed, a VCSP defines a relaxation lattice equipped with a distance measure:

Definition 4 *Given a VCSP* $\mathcal{P} = (X, D, C, S, \varphi)$*, a relaxation of* $\mathcal{P}$ *is a classical CSP* (X, D, C')*, where* $C' \subset C$*.*

Relaxations are naturally ordered by inclusion of constraint sets. Obviously, the consistent inclusion-maximal relaxations are the classical CSP which can not get closer to the original problem without loosing consistency. It is also possible to order relaxations by extending the valuation distribution to relaxations:

Definition 5 *Given a VCSP* $\mathcal{P} = (X, D, C, S, \varphi)$*, and a relaxation* (X, D, C') *of* $\mathcal{P}$*, the valuation of this relaxation is:*

$$\mathcal{V}_{\mathcal{P}}(X, D, C') = \mathop{\circledast}_{c \in C - C'} [\varphi(c)]$$

---

[2] Actually, the “absorbing element” property can be inferred from the other axioms: since $\bot$ is the identity, $(\bot \circledast \top) = \top$; since $\bot$ is minimum, $\forall a \in E, (a \circledast \top) \succcurlyeq (\bot \circledast \top) = \top$; since, $\top$ is maximum, $\forall a \in E, (a \circledast \top) = \top$

[3] From identity, it follows that $\forall a \in E, (a \circledast \bot) = a$, then for any $a, \bot \prec a \prec \top$, strict monotonicity implies that $(a \circledast a) \succ a$ and idempotency implies that $(a \circledast a) = a$.

[4] This result is well known for t-conorms. From monotonicity and idempotency, we have $\forall b \preccurlyeq a, (a \circledast \bot) = a \preccurlyeq (a \circledast b) \preccurlyeq a = (a \circledast a)$ and therefore $a \circledast b = a$.

[5] The finer approach which associates a valuation to each tuple of a relation allows the expression of gradual violation of a constraint. However, since $\bot$ is the identity, and since domains are finite, such a gradual relation may be simply expressed as a conjunction of annotated constraints and the restriction to annotated constraints is made without any loss of generality.

The valuation of the top of the relaxation lattice, CSP (X, D, C), is obviously $\perp$. The valuations of the other relaxations can be understood as a distance to this ideal problem. The best assignments of X are the solutions of the closest consistent problems of the lattice. The monotonicity of $\circledast$ ensures that the order on problems defined by this valuation distribution is consistent with the inclusion order on relaxations.

Definition 6 *Given a VCSP $\mathcal{P} = (X, D, C, S, \varphi)$, and (X, D, C'), (X, D, C''), two relaxations of $\mathcal{P}$:*

$$C' \subsetneq C'' \Rightarrow \mathcal{V}_{\mathcal{P}}(X, D, C') \succcurlyeq \mathcal{V}_{\mathcal{P}}(X, D, C'')$$

If $\circledast$ is strictly monotonic, the inequality becomes strict if the valuation of $\mathcal{P}$ is not $\top$.

This last result shows that strict monotonicity is indeed a highly desirable property since it guarantees that the order induced by the valuation distribution will respect the *strict* inclusion order on relaxations (if the VCSP valuation is not equal to $\top$). In this case, optimal consistent relaxations are always selected among inclusion-maximal consistent relaxations, which seems quite *rational*.

Since idempotency and strict monotonicity are incompatible as soon as E has more than two elements, idempotency can be seen as an undesirable property, at least from the rationality point of view. Using an idempotent operator, it is possible for a consistent non inclusion-maximal relaxation to get an optimal valuation.

There is an immediate relation between optimal assignments and optimal consistent relaxations. Indeed, to any assignment A of X, we can associate the classical *consistent* CSP $[A]_{\mathcal{P}}$ obtained by excluding the constraints violated by A.

Definition 7 *Given a VCSP $\mathcal{P} = (X, D, C, S, \varphi)$ and an assignment A of the variables of X, we note $[A]_{\mathcal{P}}$ the classical consistent CSP (X, D, C') where $C' = \{c \in C, A$ satisfies $c\}$. $[A]_{\mathcal{P}}$ is called the consistent relaxation of $\mathcal{P}$ associated to A.*

Obviously, $\mathcal{V}_{\mathcal{P}}(A) = \mathcal{V}_{\mathcal{P}}([A]_{\mathcal{P}})$ and $[A]_{\mathcal{P}}$ is among the optimal problems that A satisfies. It is equivalent to look for an optimal assignment or for an optimal consistent relaxation.

3 Classes of VCSP and their relationships

We consider some extensions of the CSP framework as VCSP:

$\wedge$-VCSP or classical CSP correspond to the trivial boolean lattice $E = \{t, f\}$, $t = \perp \prec f = \top$, $\circledast = \wedge$ (or $\max$), all constraints being annotated with $\top$. The operation $\wedge$ is both idempotent and strictly monotonic (this is the only case where both properties may exist simultaneously).

Max-VCSP or possibilistic CSP [Schiex, 1992] correspond to the operation $\circledast = \max$. Traditionally, $E = [0, 1]$, $0 = \perp$, $1 = \top$. The annotation of a constraint is interpreted as a priority degree. A preferred assignment minimizes the priority of the most important violated constraint. The idempotency of $\max$ lead to the so-called "drowning-effect": if a constraint with priority α has to be necessarily violated then any constraint with a priority lower then α is simply ignored and can be rejected from any consistent relaxation without changing its valuation.

Obviously, a classical CSP is simply a specific possibilistic CSP where only one valuation other than $\perp$ is used. Note that finite fuzzy CSP [Dubois *et al.*, 1993] can easily be cast as possibilistic CSP and vice-versa.

Σ-VCSP or penalty CSP, correspond to the operation $\circledast = +$ in $\mathbb{N} \cup \{+\infty\}$, using the usual ordering $<$. First considered in [Shapiro and Haralick, 1981], penalty CSP have been considered as *Partial CSP* in [Freuder and Wallace, 1992], all constraint valuations being equal to 1.

Π-VCSP or probabilistic CSP correspond to the operation $x \circledast y = 1 - (1 - x)(1 - y)$ in $E = [0, 1]$. Π-VCSP have been defined in [Fargier and Lang, 1993] to enable the user to represent ill-known problems, where the existence of constraints in the real problem is uncertain. Each constraint c is annotated with its probability of existence, all supposed to be independent. The probability that an assignment that violates 2 constraints c_1 and c_2 will not be a solution of the real problem is therefore $1 - (1 - \varphi(c_1))(1 - \varphi(c_2))$.

Lex-VCSP (or lexicographic CSP) offer a combination of penalty and possibilistic CSP and suppress the "drowning effect" of the latter [Fargier *et al.*, 1993]. A valuation is either a designated element $\top$ or a multiset (elements may be repeated) of elements of $[0, 1[$ (or any other totally ordered set, defining priorities).

The operation $\circledast$ is simply multi-set union, extended to treat $\top$ as an absorbing element (the empty multi-set is the identity $\perp$). The order $\succ$ is the lexicographic (or alphabetic) total order induced by the order $>$ on multisets and extended to give $\top$ its role of maximum element: let v and v' be two multisets and α and α' be the largest elements in v and v', $v \succ v'$ iff either $\alpha > \alpha'$ or ($\alpha = \alpha'$ and $v - \{\alpha\} \succ v' - \{\alpha'\}$). The recursion ends on $\varnothing$, the minimum multi-set.

3.1 Properties and relations

In order to compare the previous VCSP classes, that relies on different valuation structures, we introduce the following notion:

Definition 8 *A VCSP $\mathcal{P} = (X, D, C, S, \varphi)$ is a* refinement *of the VCSP $\mathcal{P}' = (X, D, C', S', \varphi')$ if for any pair of assignments A, A' of X such that $\mathcal{V}_{\mathcal{P}'}(A) \succ \mathcal{V}_{\mathcal{P}'}(A')$ in S' then $\mathcal{V}_{\mathcal{P}}(A) \succ \mathcal{V}_{\mathcal{P}}(A')$ in S. $\mathcal{P}$ is a* strong refinement *of $\mathcal{P}'$ if the property holds when A, A' are assignments of subsets of X.*

This main point is that if $\mathcal{P}$ is a *refinement* of $\mathcal{P}'$, then the set of optimal assignments of $\mathcal{P}$ is included in the set of optimal assignments of $\mathcal{P}'$; the problem of finding an optimal assignment of $\mathcal{P}$ can be reduced to the same problem in $\mathcal{P}'$.

Definition 9 *Two VCSP $\mathcal{P} = (X, D, C, S, \varphi)$ and $\mathcal{P}' = (X, D, C', S', \varphi')$ will be said* equivalent *iff each one is a refinement of the other. They will be said* strongly equivalent *if each one is a strong refinement of the other.*

Equivalent VCSP define the same ordering on assignments of X and have the same set of optimal assignments: the problem of finding an optimal assignment is equivalent in both VCSP.

Considering all previous VCSP classes, we may partition them according to the idempotency of the operator: $\wedge$-VCSP and Max-VCSP on one side and Σ-VCSP, Π-VCSP and Lex-VCSP on the other. Interestingly, this partition is in agreement with polynomial transformations that exist between instances of different classes for the problem of finding an optimal assignment (or the corresponding decision problem of existence of an assignment with a valuation lower than a given $v \in E$).

Idempotent classes: a $\wedge$-VCSP is nothing but a specific Max-VCSP that uses only the valuation $\top$ and the transformation from $\wedge$-VCSP to Max-VCSP is simply the identity.

Conversely, the problem of the existence of an assignment of valuation strictly lower than v in a Max-VCSP (X, D, C, S, φ) can easily be reduced to the existence of a solution for the classical CSP (X, D, C'') such that $C' = \{c \in C \mid \varphi(c) \geq v\}$: if such a constraint is violated, the assignment valuation is larger than v and conversely. Thus, using binary search, the optimal assignment in a Max-VCSP can be found in a logarithmic number of resolution of a classical CSP. Indeed, all the traditional polynomial classes and problems (k-consistency enforcing…) of classical CSP can be extended to Max-VCSP in this way [Fargier, 1994].

Stricly monotonic classes: in this section, we put Π-VCSP aside because Π-VCSP combination operator relies on multiplication of *real* numbers. However, note that if we also relax the integrity constraint on penalties in Σ-VCSP (penalties are allowed to take values in $\mathbb{R}$ instead of $\mathbb{N}$), then these frameworks are related by a simple isomorphism: a constraint with a probability $\varphi(c)$ of existence can be transformed in a constraint with a penalty of $-\log(1-\varphi(c))$ (and conversely using the transformation $1-e^{-\varphi(c)}$). The two VCSP obtained in this way are obviously *strongly equivalent*.

A penalty CSP can easily be polynomially transformed in a lexicographic CSP: the valuation $k \in \mathbb{N}$ is transformed in a multiset containing a given element $\alpha \neq 0$ repeated k times (noted $\{(\alpha, k)\}$), where α is a fixed priority and the valuation $+\infty$ is transformed to $\top$. The lexicographic VCSP obtained (in polynomial time) is obviously *strongly equivalent* to the original penalty VCSP.

Interestingly, a lexicographic CSP may also be transformed into a *strongly equivalent* penalty CSP. Let $\alpha_1, \ldots, \alpha_k$ be the elements of $]0, 1[$ that appear in all the Lex-VCSP annotations, sorted in increasing order. Let n_i be the number of occurrences of α_i in all the annotations of the VCSP. The lowest priority α_1 is associated to the penalty $f(\alpha_1) = 1$, and inductively α_i is associated to $f(\alpha_i) = f(\alpha_{i-1}) \times (n_{i-1}+1)$ (this way, the penalty $f(\alpha_i)$ associated to priority i is strictly larger than the largest possible sum of $f(\alpha_j), j < i$. This is immediately satisfied for α_1 and inductively verified for α_i). An initial lexicographic valuation is converted in the sum of the penalties $f(\alpha_i)$ for each α_i in the valuation. The valuation $\top$ is converted to $+\infty$. All the operations involved, sum and multiplication, are polynomial and the sizes of the operands remain polynomial: if k is the number of priorities used in the VCSP and ℓ the maximum number of occurrences of a priority, then the largest penalty $f(\alpha_k)$ is in $O(\ell^k)$, with a length in $O(k.\log(\ell))$ while the original annotations used at least space $O(k + \log(\ell))$. Therefore, the transformation is polynomial.

An isthm between idempotent and strictly monotonic VCSP is provided by Lex-VCSP: a Max-VCSP can be transformed in a Lex-VCSP by annotating each constraint with a multi-set containing one occurrence of the original (possibilistic) annotation if it is not equal to 1, or by $\top$ else. In this case, an optimal assignment of the Lex-VCSP not only minimizes the priority of the most important constraint violated, but also, the number of constraint violated successively at each level of priority, from the highest first to the lowest. The Lex-VCSP is therefore a *strong refinement* of the original possibilistic VCSP, obtained in polynomial time and the problem of finding *one* optimal assignment for the Max-VCSP may be reduced to the problem of finding one optimal assignment of the corresponding Lex-VCSP.

The partition between idempotent and strictly monotonic VCSP classes is also made clear at the level of polynomial classes: the existence of an assignment with a valuation lower than v in a strictly monotonic binary VCSP *with domains of cardinality two* is obviously NP-hard by restriction to MAX2SAT [Garey *et al.*, 1976]. One of the few polynomial classes which seems to extend to all classes of VCSP is the class of CSP structured in hyper-tree (see [Dechter *et al.*, 1990; Shafer, 1991]).

4 Extending traditional algorithms

4.1 Local consistency

In classical binary CSP (all constraints are supposed to involve two variables only), satisfiability defines an NP-complete problem. k-consistency properties and algorithms [Freuder, 1982] offer a range of polynomial time weaker properties: enforcing strong k-consistency in a consistent CSP will never lead to an empty CSP.

From the VCSP point of view, strong k-consistency enforcing defines a kind of lower bound of the CSP valuation: if strong k-consistency enforcing yields an empty CSP, then we know that the CSP valuation is greater than $\top$ and therefore equal to $\top$, else it is simply greater than $\perp$, which is always true.

Arc-consistency (strong 2-consistency) is certainly the most prominent level of local consistency and has been extended to Max-VCSP years ago [Rosenfeld *et al.*, 1976]. In Max-VCSP, arc-consistency can be defined as follows:

Definition 10 *A VCSP $\mathcal{P}$ is said to be arc-consistent iff (1) there exists, for each variable, a value that defines an assignment with a valuation strictly lower than $\top$ and (2) any assignment A of one variable can be extended to an assignment A' on two variables with the same valuation ($\mathcal{V}_{\mathcal{P}}(A) = \mathcal{V}_{\mathcal{P}}(A')$).*

Polynomial worst-case time algorithms that enforce this property on Max-CSP are defined in [Rosenfeld *et al.*, 1976; Snow and Freuder, 1990; Schiex, 1992]. These algorithms yield an arc-consistent Max-VCSP with the same valuation distribution on complete assignments, and a lower bound on the VCSP valuation can easily be derived from it.

Obviously, this definition could also be used in non idempotent VCSP. But it is useless if we can not define the corresponding arc-consistency enforcing algorithms that should compute, in polynomial time, a VCSP $\mathcal{P}'$ which is both arc-consistent and in some sense "*equivalent*" to the original VCSP $\mathcal{P}$. The strongest level of equivalence one could achieve (stronger than our strong equivalence notion, def. 9) is the equality of the valuations in both VCSP for all complete assignments.

But the generalization of AC enforcing algorithms that consists in using $\min$ and $\circledast$ respectively for projection and combination of constraints fails for non idempotent monotonic VCSP as it has been shown in a similar framework (see [Bistarelli *et al.*, 1995], in these proceedings). The distribution of valuations may be modified and the algorithm may fail to terminate. However, it is still an open question whether more drastic modifications of the algorithms/properties, or a weakening of the "equivalence" notion (as def. 9) would allow us to recover something related to arc-consistency.

4.2 Tree search

Following the works from [Shapiro and Haralick, 1981; Schiex, 1992; Freuder and Wallace, 1992; Dubois *et al.*, 1993], we try to extend some traditional CSP algorithms to the *binary* VCSP framework to solve the problem of finding a provenly optimal assignment. The class of algorithms which we are interested in are hybrid algorithms that combine backtrack tree-search with some level of local consistency enforcing at each node. These algorithms have been called look-ahead, prospective or prophylactic algorithms. Some possible instances have been considered in [Nadel, 1989]: Backtrack, Forward-Checking , Really Full Look Ahead. We consider here that such algorithms are described by the type of local consistency enforcing maintained at each node: check-backward, check forward, arc-consistency or more...

In prospective algorithms, an assignment is extended until either a complete assignment (a solution) is found, or the given local consistency property is not verified on the current assignment: backtrack occurs. The extension of such algorithms to the VCSP framework, where the problem is now an optimization problem, relies on a transformation of the *Backtrack* tree search schema to a *Depth First Branch and Bound* algorithm. DFBB is a simple depth first tree search algorithm, which, like *Backtrack*, extend an assignment until either (1) a complete assignment is reached: a new "better" solution is found or (2) a given lower bound on the valuation of the best assignment that can be found by extending the current assignment exceeds the valuation of the current best solution found: backtrack occurs. The lower bound used defines the algorithm. Our aim is to derive a lower bound from any given local consistency property.

In classical CSP, seen as $\wedge$-VCSP, the actual local consistency property used gives the "lower bound": for example, in *Really Full Look Ahead*, the inexistence of an arc-consistent closure of the CSP guarantees that the valuation of any extension of the current assignment will be greater than $\top$ and therefore equal to $\top$. However, as we pointed out earlier, no arc-consistency enforcing algorithm is available for strictly monotonic VCSP. We will therefore use classical local consistency notions plus the notion of relaxation of a VCSP (which defines classical CSP) to define our class of bounds:

Property 1 *Given a classical local consistency property L, a lower bound on the valuation of a given VCSP $\mathcal{P}$ is defined by the valuation of an optimal relaxation of $\mathcal{P}$ among those that satisfy the "local consistency" property L used (consistency of the current assignment, absence of domain wipe-out after check-forward or arc-consistency enforcing...).*

This valuation is a lower bound of the valuation of an optimal assignment since the valuation of an optimal assignment is also the valuation of an optimal *consistent* relaxation and all the relaxations where the "local consistency" property L is not verified are non consistent.

These lower bounds verify two interesting properties:

- they guarantee that the extended algorithm will behave as the original "classical" algorithm when applied to a classical CSP seen as a $\wedge$-VCSP (a classical CSP seen as a $\wedge$-VCSP has only one relaxation with a valuation lower than $\top$: itself);
- a stronger local consistency property will define a better lower bound, leading to a tree search with less nodes but possibly more computation at each node.

4.3 Extending Backtrack

Backtrack uses the local inconsistency of the current partial assignment as the condition for backtracking. Therefore, the lower bound derived is the valuation of an optimal relaxation in which the current assignment is consistent. This is simply the relaxation which precisely rejects the constraints violated by the current assignment (these constraints have to be rejected or else local inconsistency will occur; rejecting these constraint suffices to restore the consistency of the current assignment in the relaxation). The lower bound is therefore simply defined by:

$$\underset{\substack{c \in C \\ A \text{ violates } c}}{\circledast} \varphi(c)$$

and is obviously computable in polynomial time.

The lower bound can easily be computed incrementally when a new variable x_i is assigned: the lower bound associated to the father of the current node is aggregated with the valuations of all the constraints violated by x_i using $\circledast$..

The generic VCSP algorithm defined encompass all the "Branch and Bound" algorithms defined for Max-VCSP or Σ-VCSP in [Schiex, 1992; Freuder and Wallace, 1992; Fargier, 1994; Ruttkay, 1994]. Note that for Max-VCSP, thanks to idempotency, it is useless to test whether constraints whose valuation is lower than the lower bound associated to the father node have to be rejected since their rejection cannot influence the bound.

4.4 Extending Forward Checking

Forward-checking uses an extremely limited form of arc-consistency: backtracking occurs as soon as all the possible extensions of the current assignment A on any uninstantiated variable are locally inconsistent: the assignment is said non forward-checkable. Therefore, the lower bound used is the minimum valuation among the valuations of all the relaxations that makes the current assignment forward-checkable.

A relaxation in which A is forward-checkable (1) should necessarily reject all the constraints violated by A itself and (2) for each uninstantiated variable x_i it should reject one of the sets $C(x_i, \nu)$ of constraints that are violated if x_i is instantiated with value ν of its domain. Since $\circledast$ is monotonic, the minimum valuation is reached by taking into account, *for each variable*, the valuation of the set $C(x_i, \nu)$ of minimum valuation. The bound is again computable in polynomial time since it is the aggregation of (1) the valuations all the constraints violated by A itself (*i.e.*, the bound used in the extension of the backtrack algorithm, see 4.3) and (2) the valuations of the constraint in all the $C(x_i, \nu)$. This computation needs less than $(e.n.d)$ constraint checks and $\circledast$ operations (e is the number of constraints); all the minimum valuation can be computed with less than $(d.n)$ comparisons and aggregated with less than n $\circledast$ operations. Note that the lower bound derived includes the bound used in the backtrack extension plus an extra component and will always be better than the "*Backtrack*" bound.

The lower bound may be incrementally computed by maintaining during tree search, and for each value ν of every unassigned variable x_i the aggregated valuation $B(\nu, x_i)$ of all the constraints that will be violated if ν is assigned to x_i given the current assignment. Initially, all $B(\nu, x_i)$ are equal to $\perp$. When the assignment A is extended to $A' = A \cup \{x_j = \mu\}$, the B may be updated as follows:

$$B(\nu, x_i) \leftarrow B(\nu, x_i) \circledast \Big(\underset{\substack{c \in C, X_c = \{x_i, x_j\} \\ A' \cup \{x_i = \nu\} \text{ violates } c}}{\circledast} [\varphi(c)] \Big)$$

that takes into account all the constraints between x_i and x_j that are necessarily violated if μ is assigned to x_j. Upon backtrack, the B have to be restored to their previous values, as domains in classical *Forward-checking*. Note that the B offer a default value heuristic: choose the value with a minimum B.

The lower bound is simply obtained by aggregating, using $\circledast$, the valuations of all the constraints violated by the assignment and all the minimum $B(\nu, x_i)$ for each unassigned variable. The aggregated valuation $v(A')$, $A' = A \cup \{x_j = \mu\}$), of all the constraints violated by the assignment A' is easily computed by taking the valuation $v(A)$ computed on the father node $\circledast$'ed with $B(\mu, x_j)$.

Additional sophistications include deleting values ν of the domains of non instantiated variables if the aggregated valuation of $v(A')$ and $B(\nu, x_i)$ exceeds the upper bound (see [Freuder and Wallace, 1992]). The generic VCSP algorithm defined encompass the forward-checking based algorithm for Max-VCSP described in [Schiex, 1992] or the *Partial Forward-checking* algorithm defined for Σ-VCSP in [Freuder and Wallace, 1992]. Note that for Max-VCSP, and thanks to idempotency, the updating of B can ignore constraints whose valuation is less than the B updated or than the current lower-bound.

4.5 Trying to extend Really Full Look Ahead

Really Full Look Ahead maintains arc consistency during tree search and backtracks as soon as the current assignment induces a domain wipe-out: the CSP has no arc-consistent closure. For a VCSP, the bound which can be derived from arc-consistency will be the minimum valuation among the valuations of all the relaxations such that the current assignment does not induces a domain wipe-out.

Let us consider any class $\circledast$-VCSP of the VCSP framework such that $\circledast$ is strictly monotonic and for any $a, b \in E$, $a, b \prec \top$, $(a \circledast b) \prec \top$. Let ℓ be any valuation different from $\top$ and $\perp$. The decision problem corresponding to the computation of the lower bound in this class can be formulated as:

Problem 1 (MAX-AC-CSP) *Given such a $\circledast$-VCSP and a valuation v, is there a set $C' \subset C$ such that the relaxation (X, D, C') has a non empty arc-consistent closure and a valuation lower than v?*

Theorem 1 MAX-AC-CSP *is strongly NP-complete.*

Sketch of proof: The problem belongs to NP since computing the arc-consistent closure of a CSP can be done in polynomial time and we have supposed that $\circledast$ and $\succ$ are polynomial in the size of their arguments.

We give the polynomial transformation from MAX2SAT [Garey *et al.*, 1976] to MAX-AC-CSP. An instance of MAX2SAT is defined by a set Φ of e 2-clauses and a positive integer k, the problem being the existence of a consistent subset of Φ of cardinality larger than k. Let Υ be the set of n propositional variables occurring in Φ. We consider the binary CSP (X, D, C) which is composed of $n + 2n.(e+1)$ variables: (1) the first n variables $x_1, \ldots, x_n$ corresponds to the n variables of Υ and have a domain of cardinality two corresponding to the boolean values t and f; (2) the next $2n.(e+1)$ variables will have a domain of size 1, containing the only value $\bigstar$. This set of variables is composed of n sets of $2e+2$ variables $\{t_{i,1}, \ldots, t_{i,e+1}, f_{i,1}, \ldots, f_{i,e+1}\}$ associated to the original variable x_i.

The constraints that appear in C are composed of 3 sets: (1) a set of e constraint corresponding to the e clauses of Φ on the variables $x_1, \ldots, x_n$; (2) for each variable x_i and its associated $e+1$ variables $t_{i,j}$, a set C_t^i of $e+1$ constraints which connect x_i with $t_{i,j}$ and allow only the tuple $(t, \bigstar)$; (3) for each variable x_i and its associated e+1 variables $f_{i,j}$, a set C_f^i of $e+1$ constraints which connect x_i with $f_{i,j}$ and allow only the tuple $(f, \bigstar)$. All the constraints have the valuation ℓ.

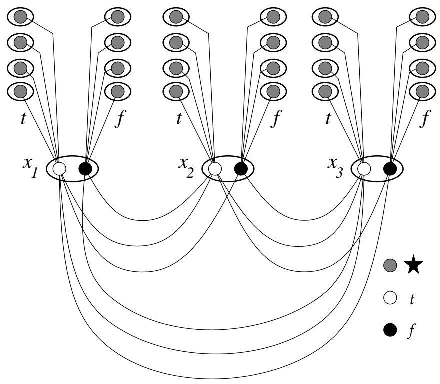

Figure 1: The micro-structure of the CSP

For example, Figure 1 illustrates the micro-structure of the CSP built from the set $\Phi = \{x_1 \vee x_2, x_2 \vee x_3, x_1 \vee x_3\}$. The transformation is clearly polynomial. Furthermore, one may prove that the existence of a truth assignment that satisfies at least k clauses of Φ is equivalent to the existence of a relaxation with a non-empty arc-consistent closure and a valuation lower than $\ell \circledast \cdots \circledast \ell$ with $n.(e+1) + (n-k)$ occurrences of ℓ. This shows that MAX2SAT $\propto$ MAX-AC-CSP. $\square$

Therefore, extending *Really Full Look Ahead* seems difficult since computing the lower bound itself is NP-complete. For idempotent VCSP, the bound may be computed using polynomial time algorithms for enforcing arc-consistency [Rosenfeld *et al.*, 1976; Snow and Freuder, 1990].

4.6 Experimentations

The *Forward-Checking* algorithm has been coded and applied to random VCSP generated as follows: a classical random CSP with 16 variables, domains of size 9 is generated as in [Hubbe and Freuder, 1992]. A first possibilistic VCSP is obtained by randomly assigning a valuation $\frac{1}{4}, \frac{1}{2}, \frac{3}{4}$ or 1 to each constraint. A lexicographic VCSP is then built simply by using the transformation from possibilistic to lexicographic CSP described in section 3. This VCSP is a strong refinement of the original possibilistic CSP.

Because of limited space, we only report mean number of constraint checks performed to find an optimal assignment and prove optimality for a slice of the random CSP space (see Figure 2): constraint satisfiability is fixed to 60% and the constraint graph goes from tree structured CSP to a complete graph. At each point 50 classical, possibilistic and corresponding lexicographic CSP are solved with the following heuristics: the variable which minimizes the ratio domain/degree is chosen, the value that minimizes B is chosen. A first conclusion is that solving consistent CSP as VCSP,

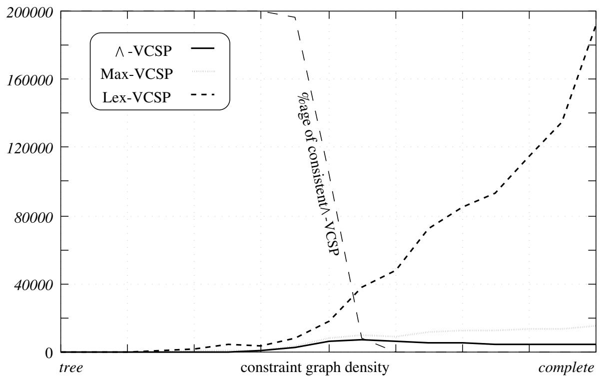

Figure 2: Number of cc for $\wedge$, $\max$ and lex. VCSP

i.e., uselessly trying to anticipate a possible inconsistency, is relatively inexpensive, even for Lex-VCSP. On inconsistent CSP, possibilistic CSP are not much harder than classical CSP, but the transition phase is apparently extended to the left. Last, but not least, lexicographic CSP are incredibly more difficult which again shows the computational complexity of strictly monotonic $\circledast$: rationality seems expensive. Stronger argument could probably be obtained using recent developments in complexity theory, the transformations of Section 3.1 defining *metric reductions* between optimization problems [Krentel, 1988].

5 Conclusion

The VCSP framework enables the expression of a large number of real constraint satisfaction/optimization problems. If idempotent VCSP have already received a lot of attention and most classical CSP algorithms/properties have been extended to this setting [Fargier, 1994], the case of non idempotent operators, a desirable property as it has been shown, seems much harder to tackle and few CSP algorithms have been extended to this case [Freuder and Wallace, 1992].

Since local consistency enforcing algorithms are unavailable in this case, we have considered a general class of bounds, that could be used in a depth first branch and bound algorithm and which have been derived from *classical* local consistency properties. It appears that at the level of arc-consistency, the problem of computing the bound is as difficult as solving a VCSP itself and other types of bounds have to be considered.

References

[Bistarelli *et al.*, 1995] S. Bistarelli, U. Montanari, and F. Rossi. Constraint solving over semirings. In *Proc. of the 14^{th} IJCAI*, Montreal, Canada, August 1995.

[Borning *et al.*, 1989] A. Borning, M. Mahert, A. Martindale, and M. Wilson. Constraint hierarchies and logic programming. In *Int. conf. on logic programming*, pages 149–164, 1989.

[Dechter *et al.*, 1990] R. Dechter, A. Dechter, and J. Pearl. Optimization in constraint networks. In R.M OIiver and J.Q. Smith, editors, *Influence Diagrams, Belief Nets and Decision Analysis*, chapter 18, pages 411–425. John Wiley & Sons Ltd., 1990.

[Dubois and Prade, 1982] D. Dubois and H. Prade. A class of fuzzy measures based on triangular norms. a general framework for the combination of uncertain information. *Int. Journal of Intelligent Systems*, 8(1):43–61, 1982.

[Dubois *et al.*, 1993] D. Dubois, H. Fargier, and H. Prade. The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction. In *Proc. 2^{nd} IEEE Conference on Fuzzy Sets*, San Francisco, CA, March 1993.

[Fargier and Lang, 1993] H. Fargier and J. Lang. Uncertainty in constraint satisfaction problems: a probabilistic approach. In *Proc. of ECSQARU '93, LNCS 747*, pages 97–104, 1993.

[Fargier *et al.*, 1993] H. Fargier, J. Lang, and T. Schiex. Selecting preferred solutions in Fuzzy Constraint Satisfaction Problems. In *Proc. of the 1^{st} European Congress on Fuzzy and Intelligent Technologies*, 1993.

[Fargier, 1994] Hélène Fargier. *Problèmes de satisfaction de contraintes flexibles et application à l'ordonnancement de production*. Phd thesis (in french), Institut de Recherche en Informatique de Toulouse, France, June 1994.

[Freuder and Wallace, 1992] E.C. Freuder and R.J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58:21–70, December 1992.

[Freuder, 1982] Eugene C. Freuder. A sufficient condition for backtrack-free search. *Journal of the ACM*, 29(1):24–32, 1982.

[Garey *et al.*, 1976] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[Hubbe and Freuder, 1992] Paul D. Hubbe and Eugene C. Freuder. An efficient cross-product representation of the constraint satisfaction problem search space. In *Proc. of AAAI-92*, pages 421–427, San Jose, CA, 1992.

[Krentel, 1988] M. Krentel. The complexity of optimization problems. *Journal of Computer and System Sciences*, 36:490–509, 1988.

[Martin-Clouaire, 1992] R. Martin-Clouaire. Dealing with soft constraints in a constraint satisfaction problem. In *Proc. of the Int. Conf. on Information Processing of Uncertainty in Knowledge based Systems*, pages 37–40, Palma de Mallorca, July 1992.

[Minoux, 1976] M. Minoux. Structures algébriques généralisées des problèmes de cheminement dans les graphes. *Recherche opérationnelle*, 10(6):33–62, 1976.

[Moulin, 1988] H. Moulin. *Axioms of cooperative decision making*. Cambridge University Press, 1988.

[Nadel, 1989] Bernard A. Nadel. Constraint satisfaction algorithms. *Comput. Intell.*, 5(4):188–224, November 1989.

[Rosenfeld *et al.*, 1976] A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *IEEE Trans. on Systems, Man, and Cybernetics*, 6(6):173–184, 1976.

[Ruttkay, 1994] Zsofia Ruttkay. Fuzzy constraint satisfaction. In *Proc. FUZZ-IEEE'94*, Orlando, Florida, 1994.

[Schiex, 1992] T. Schiex. Possibilistic constraint satisfaction problems or "How to handle soft constraints ?". In *Proc. of the 8^{th} Int. Conf. on Uncertainty in AI*, Stanford, CA, July 1992.

[Shafer, 1991] G. Shafer. An axiomatic study of computation in hypertrees. Working paper 232, University of Kansas, 1991.

[Shapiro and Haralick, 1981] L. Shapiro and R. Haralick. Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3:504–519, 1981.

[Snow and Freuder, 1990] P. Snow and E.C. Freuder. Improved relaxation and search methods for approximate constraint satisfaction with a maximin criterion. In *Proc. of the 8^{th} biennal conf. of the canadian society for comput. studies of intelligence*, pages 227–230, May 1990.