Relaxation of Qualitative Constraint Networks

Dominique D'Almeida, Jean-François Condotta,
Christophe Lecoutre, and Lakhdar Saïs

CRIL-CNRS, Université d'Artois, rue de l'Université, 62307 Lens, France
{dalmeida,condotta,lecoutre,sais}@cril.univ-artois.fr

Abstract. In this paper, we propose to study the interest of relaxing qualitative constraints networks by using the formalism of discrete Constraint Satisfaction Problem (CSP). This allows us to avoid the introduction of new definitions and properties in the domain of qualitative reasoning. We first propose a general (but incomplete) approach to show the unsatisfiability of qualitative networks, by using a relaxation on any set of relations. Interestingly enough, for some qualitative calculi, the proposed scheme can be extended to determine the satisfiability of any qualitative network, leading to an original, simple and complete method. However, as the efficiency of our approach depends on the chosen relaxation, total relations should be preferred due to their connections with the hardness of constraint networks. We then present some preliminary experimental results, with respect to unsatisfiability, which show some promising improvements on some classes of random qualitative networks.

1 Introduction

The need for reasoning about time and space arises in many areas of Artificial Intelligence, including computer vision, natural language understanding, geographic information systems (GIS), scheduling, planning, diagnosis and genetics. Numerous formalisms for representing and reasoning about time and space in a qualitative way have been proposed in the past two decades [1,28,22,3,27,19,4].

Those formalisms involve a finite set of basic relations denoting qualitative relationships between temporal or spatial entities. Intersection, overlapping, containment, precedence are examples of such qualitative relationships. For instance, in the field of qualitative reasoning about temporal data, there is a well-known formalism called Allen's calculus [1]. It is based on intervals of the rational line for representing temporal entities and thirteen basic relations between such intervals are used to represent the qualitative situations between temporal entities: an interval can follow another one, meet another one, and so on.

Typically, Qualitative Constraint Networks (QCNs) are used to express information on a spatial or temporal situation. Each constraint of a QCN represents a set of acceptable qualitative configurations between some temporal or spatial entities and is defined by a set of basic relations. The total relation, which is the set of all basic relations, is the term used to describe a total uncertainty in the configurations. The density of such relations in a qualitative networks can take part to the difficulty to solve those problems.

I. Miguel and W. Ruml (Eds.): SARA 2007, LNAI 4612, pp. 93–108, 2007.

The aim of this paper is to demonstrate that a relaxation of some relations can lead, in some cases, to a significant computational speed up of the satisfiability checking task. To avoid the introduction of new particular definitions (and properties) in qualitative reasoning, when relaxation is applied, a discrete encoding preserving the chosen relations of qualitative constraint networks is used. Using a relaxation on any set of relations, it is then possible to show the unsatisfiability of some networks. Interestingly enough, for some qualitative calculi, this basic relaxation scheme is extended to determine the satisfiability of any qualitative constraint network leading to an original, simple and complete method. Nevertheless, even if the approach can be used in the general case, we focus our attention on relaxing total relations since these relations are intuitively related to computational efficiency.

Relaxation based approaches are widely used in many domains ranging from constraint satisfaction to linear programming and knowledge representation. For example, in constraint satisfaction, relaxations are used to solve dynamic [20], over-constrained [21] and distributed [31] constraint networks. Some other works are based on concepts of abstract interpretation [14,23], theory of abstraction [16] or theory approximation [8,29]. Whereas in [9], abstract interpretation [14,23] is exploited to improve constraint solving in an object-oriented context, the concept of Galois insertion (at the heart of abstract interpretation) has also been used to deal with flexible constraints [5,6]. Generally speaking, many abstractions (e.g. [15,10,30,11]) proposed in the literature can be seen as a kind of value or variable clustering. The framework introduced in [24] allows to deal with general clustering which means that an element (a value or a variable) can belong to several clusters.

The paper is organized as follows. In the next section, some background on qualitative and discrete formalisms is provided. Then we present the general relaxation scheme that we propose. Before concluding, some experimental results on some classes of random qualitative constraint networks are presented.

2 Technical Background

In this section, we provide the technical background useful for the reading of this paper. First, we present the concept of qualitative calculus before introducing qualitative constraint networks. Then, we introduce discrete constraint networks and describe an encoding of qualitative constraint networks into discrete ones.

2.1 Qualitative Calculus

A qualitative calculus involves a finite set B of binary[1] relations, called basic relations, defined on a domain D. The elements of D represent temporal or spatial entities. Each basic relation of B corresponds to a particular possible configuration between two temporal or spatial entities. The relations of B

[1] In this paper, we focus on binary relations but this work can be extended to non-binary ones.

are jointly exhaustive and pairwise disjoint, which means that any pair of elements of D belongs to exactly one basic relation in B. Moreover, for each basic relation $B \in \mathsf{B}$ there exists another basic relation of B, denoted by $B^{\sim}$, corresponding to the transposition of B. In addition, we suppose that a particular relation of B, denoted by Id, is the identity relation on D. The set A is defined as the set of relations corresponding to all possible unions of the basic relations: $\mathsf{A} = \{\bigcup E : E \subseteq \mathsf{B}\}$. It is customary to represent an element $B_1 \cup \ldots \cup B_m$ (with $B_i \in \mathsf{B}$ for each i such that $1 \leq i \leq m$) of A by the set $\{B_1, \ldots, B_m\}$ belonging to 2^{B}. Hence, we make no distinction between A and 2^{B} in the rest of this paper.

As an example, consider the well-known temporal qualitative formalism called Allen's calculus [2]. It uses intervals of the rational line for representing temporal entities. Hence, D is the set $\{(x^-, x^+) \in \mathbb{Q} \times \mathbb{Q} : x^- < x^+\}$. The set of basic relations consists of a set of thirteen binary relations $\mathsf{B} = \{eq, b, bi, m, mi, o, oi, s, si, d, di, f, fi\}$ corresponding to all possible configurations between two intervals. These basic relations are depicted in Figure 1. We have $\mathsf{Id} = eq$.

Relation	Symbol	Inverse	Meaning
precedes	b	bi	
meets	m	mi	
overlaps	o	oi	
starts	s	si	
during	d	di	
finishes	f	fi	
equals	eq	eq	

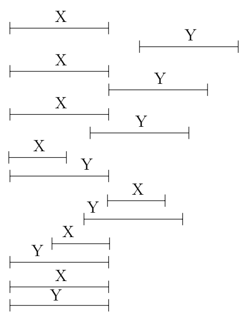

Fig. 1. The basic relations of Allen's calculus

As a set of subsets, A is equipped with the usual set-theoretic operations including intersection ($\cap$) and union ($\cup$). As a set of binary relations, it is also equipped with the operation of converse ($\sim$) and an operation of composition ($\circ$) sometimes called weak composition or qualitative composition. The converse of a relation R in A is the union of the transpositions of the basic relations contained in R. The composition $A \circ B$ of two basic relations A and B is the relation $R = \{C \in \mathsf{B} \mid \exists x, y, z \in \mathsf{D}^3, x\ A\ y, y\ B\ z \text{ and } x\ C\ z\}$. The composition $R \circ S$ of $R, S \in \mathsf{A}$ is the relation $T = \bigcup_{A \in R, B \in S}\{A \circ B\}$. Computing the results of these various operations for relations of 2^{B} can be done efficiently by using tables giving the results of these operations for the basic relations of B. For instance, consider the relations $R = \{eq, b, o, si\}$ and $S = \{d, f, s\}$ of Allen's calculus, we have $R^{\sim} = \{eq, bi, oi, s\}$. The relation $R \circ S$ is $\{d, f, s, b, o, m, eq, si, oi\}$.

2.2 Qualitative Constraint Networks

A qualitative constraint network (QCN) is a pair composed of a set of variables and a set of constraints. Each variable represents a spatial or temporal entity of the problem that is represented, and each constraint consists of a set of acceptable basic relations (the possible configurations) between two variables. More formally, a QCN is defined in the following way:

Definition 1. *A* QCN $\mathcal{N}$ *is a pair* (V, Q) *where* $V = \{v_1, \ldots, v_n\}$ *is a finite set of* n *variables and* Q *is a map that assigns to each pair* (v_i, v_j) *of* $V \times V$ *a set* $Q(v_i, v_j) \in 2^{\mathsf{B}}$ *of basic relations.* $Q(v_i, v_j)$ *will also be denoted by* Q_{ij}*.* Q *is such that* $Q_{ii} \subseteq \{\mathsf{Id}\}$ *and* $Q_{ij} = Q_{ji}^{\sim}$ *for all* $v_i, v_j \in V$*.*

A *solution* of a QCN $\mathcal{N}$ is a map σ from V to D such that $(\sigma(v_i), \sigma(v_j))$ satisfies Q_{ij} for all $v_i, v_j \in V$. $\mathcal{N}$ is consistent iff it admits a solution. A QCN $\mathcal{N}' = (V', Q')$ is a *sub*-QCN of $\mathcal{N}$ (denoted by $\mathcal{N}' \subseteq \mathcal{N}$) if and only if $V = V'$ and $Q'_{ij} \subseteq Q_{ij}$ for all $v_i, v_j \in V$. A QCN $\mathcal{N}' = (V', Q')$ is *equivalent* to $\mathcal{N}$ if and only if $V = V'$ and both networks have the same solutions. $\mathcal{N}$ is *atomic* iff each constraint of $\mathcal{N}$ contains exactly one basic relation. A *scenario* of $\mathcal{N}$ is an atomic sub-QCN of $\mathcal{N}$. We will denote by $scen(\mathcal{N})$ and $sol(\mathcal{N})$ a scenario and a solution of $\mathcal{N}$, respectively.

Given a QCN $\mathcal{N}$, the main issue to be addressed is the consistency problem: to decide whether or not $\mathcal{N}$ admits (at least) a solution. Most of the algorithms used for solving this problem are based on a method that we call the $\circ$-closure method, also called weak composition closure and denoted WC. The $\circ$-closure method is a constraint propagation method allowing to enforce the $(0, 3)$-consistency of $\mathcal{N}$, which means that all restrictions of $\mathcal{N}$ to 3-variables are consistent. The $\circ$-closure method involves iteratively performing the following operation: $Q_{ij} := Q_{ij} \cap (Q_{ik} \circ Q_{kj})$, for all v_i, v_j, v_k of V, until a fix-point is reached. This method yields a sub-QCN $\mathcal{N}' = (V, Q')$ of $\mathcal{N}$ which is equivalent to it, and such that $Q'_{ij} \subseteq Q'_{ik} \circ Q'_{kj}$, for all v_i, v_j, v_k of V. This last condition is expressed by saying that the sub-network is $\circ$-closed (to simplify, we will assume that a $\circ$-closed QCN does not contain the empty relation associated with a constraint).

2.3 Discrete Constraints Networks

Definition 2. *A Discrete Constraint Network (*DCN*)* $\mathcal{P}$ *is a pair* (X, C) *where* X *is a finite set of variables and* C *a finite set of constraints. Each variable* $x \in X$ *has an associated domain, denoted* $dom^{\mathcal{P}}(x)$*, which represents the set of values allowed for* x*. Each constraint* $c \in C$ *involves a subset of variables of* X*, called scope and denoted* $scp(c)$*, and has an associated relation denoted* $rel^{\mathcal{P}}(c)$*, which represents the set of tuples allowed for the variables of its scope.*

When possible, we will write $dom(x)$ and $rel(c)$ instead of $dom^{\mathcal{P}}(x)$ and $rel^{\mathcal{P}}(c)$. If $\mathcal{P}$ and $\mathcal{P}'$ are two DCNs defined on the same sets of variables X and constraints C, then we will write $\mathcal{P} \preceq \mathcal{P}'$ (and we will say that $\mathcal{P}$ is a subnetwork of $\mathcal{P}'$) iff $\forall x \in X$, $dom^{\mathcal{P}}(x) \subseteq dom^{\mathcal{P}'}(x)$. A solution to a discrete constraint network is an assignment of values to all the variables such that all the constraints are satisfied. A constraint network is said to be satisfiable or consistent iff it admits at least one solution. The Constraint Satisfaction Problem (CSP) is the NP-complete task of determining whether a given constraint network is satisfiable. A CSP instance is then defined by a constraint network, and solving it involves either finding one (or more) solution or determining its unsatisfiability.

To solve a CSP instance, one can apply inference or search methods. Usually, domains of variables are reduced by removing inconsistent values, i.e. values that can not occur in any solution. Indeed, it is possible to filter domains by considering some properties of constraint networks. Generalized Arc Consistency (GAC) remains the central property of constraint networks and establishing GAC on a given network $\mathcal{P}$ involves removing all values that are not generalized arc-consistent. Remark that for binary constraint networks, GAC is simply referred as AC (Arc Consistency).

Definition 3. *Let $\mathcal{P} = (X, C)$ be a* DCN*. A pair (x, a), with $x \in X$ and $a \in dom(x)$, is generalized arc-consistent (GAC) iff $\forall c \in C \mid x \in scp(c)$, there exists a support of (x, a) in c, i.e. a tuple $t \in rel(c)$ such that $t[x] = a$ and $t[y] \in dom(y)$ $\forall y \in scp(c)$[2]. $\mathcal{P}$ is GAC iff $\forall x \in X$, $dom(x) \neq \emptyset$ and $\forall a \in dom(x)$, (x, a) is GAC.*

We will denote by $GAC(\mathcal{P})$ the constraint network obtained after enforcing GAC on $\mathcal{P}$. Inconsistency proved when applying GAC is denoted by $GAC(\mathcal{P}) = \bot$.

2.4 From Qualitative to Discrete Constraints Networks

In this paper, we propose to use an encoding to map a qualitative constraint network $\mathcal{N}$ into a discrete one $\mathcal{P}$. Each constraint of $\mathcal{N}$ is mapped to a variable of $\mathcal{P}$ whose domain corresponds to the atomic relations of the constraint (and, as a consequence, a subset of B), and each triple of constraints of $\mathcal{N}$ is mapped to a ternary constraint of $\mathcal{P}$ such that the associated relation contains all valid 3-tuples satisfying the weak composition. More formally, we obtain:

Definition 4. *Let $\mathcal{N} = (V, Q)$ be a* QCN*. $\mathsf{T}_{\mathsf{DCN}}(\mathcal{N})$ is the* DCN *$\mathcal{P} = (X, C)$ defined as follows:*

- *for each pair of variables $v_i, v_j \in V$ with $1 \leq i \leq j \leq n$, X contains a variable x_{ij} such that $dom(x_{ij}) = Q_{ij}$;*
- *for each triple of variables $v_i, v_j, v_k \in V$ with $1 \leq i < k < j \leq n$, C contains a ternary constraint c_{ijk} such that $scp(c_{ijk}) = \{x_{ij}, x_{ik}, x_{kj}\}$ and $rel(c_{ijk}) = \{(a, b, c) \in \mathsf{B}^3 : a \in b \circ c\}$.*

The idea of mapping qualitative networks into discrete ones is quite natural, and has been formalized in [26,12]. Interestingly, there are some relationships between the two frameworks. For example, if a QCN $\mathcal{N}$ is consistent, then $\mathsf{T}_{\mathsf{DCN}}(\mathcal{N})$ is consistent [12]. Unfortunately, the encoding is not complete for some qualitative calculi (e.g. the cyclic interval algebra [18,4]): the qualitative network $\mathcal{N}$ can be

[2] $t[x]$ denotes the value assigned to x in t.

inconsistent whereas the discrete network $\mathsf{T_{DCN}}(\mathcal{N})$ is consistent. Nevertheless, we have the following weaker property: if $\mathsf{T_{DCN}}(\mathcal{N})$ is consistent then $\mathcal{N}$ admits a $\circ$-closed scenario. Besides, introducing the concept of so-called *nice* qualitative calculus, i.e. a calculus for which a scenario is consistent if and only if it is $\circ$-closed, we can establish that: a QCN $\mathcal{N}$ defined in a *nice* qualitative calculus is consistent iff $\mathsf{T_{DCN}}(\mathcal{N})$ is consistent. It is important to remark that many qualitative calculi are nice, and in particular the well-known Allen's calculus.

It is possible to obtain a qualitative network from a discrete network using the following operator T_{QCN}.

Definition 5. *Let $\mathcal{N} = (V, Q)$ be a* QCN *and $\mathcal{P} = (X, C)$ be a* DCN *such that $\mathcal{P} \preceq T_{DCN}(\mathcal{N})$. $T_{QCN}(\mathcal{P})$ is the* QCN *(V, Q') defined by $Q'_{ij} = dom(x_{ij})$ and $Q'_{ji} = (Q'_{ij})^{\smile}$ for all $1 \leq i \leq j \leq n$.*

This operator is useful to show the connections existing between qualitative and discrete local consistencies. Indeed, we can prove [12] that if $\mathsf{T_{DCN}}(\mathcal{N})$ is GAC then $\mathcal{N}$ is $\circ$-closed. As a consequence, a way to obtain the $\circ$-closure of a QCN is to transform it into a DCN (via $\mathsf{T_{DCN}}$), apply a GAC algorithm and get back the result (via T_{QCN}) under the form of a DCN. This is illustrated in Figure 2.

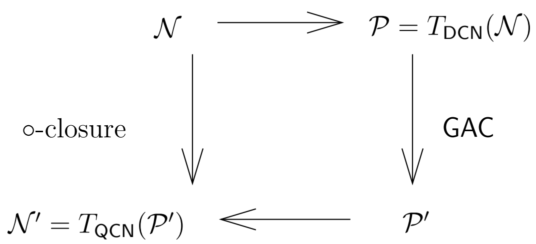

Fig. 2. Relationship between $\circ$-closure and GAC

The interest of encoding qualitative networks into discrete ones is two-fold. First, we can benefit from some state-of-the-art generic CSP solvers that are freely available. Second, the formalism classically used for qualitative algebra is based on networks whose macro-structure corresponds to complete graphs. Here, introducing relaxation in qualitative networks would require extending current qualitative definitions and properties. The major part of QCN solvers works with matrices as data structures to represent qualitative constraints. For example, QAT (a qualitative algebra toolkit [13]) uses such a data structure. With these solvers, the representation of a relaxed QCN is not a trivial task and implies heavy changes concerning the methods used for reasoning.

3 Relaxing Qualitative Constraints Networks

There is a particular relation in any qualitative algebra: the total relation that we will denote by ψ. This relation, which is such that $\psi = \mathsf{B}$, represents the fact that we have no information about the configuration of any two variables. When solving a QCN, ψ relations may represent an overhead for the resolution since they must be taken into account while not participating (at least, initially) to filter the search space. This is why we are going to propose to relax qualitative constraint networks by simply discarding all ψ relations.

In order to make our approach quite general, we introduce relaxation and restriction with respect to a subset of relations $\mathsf{R} \subseteq 2^{\mathsf{B}}$ (even if we will choose $\mathsf{R} = \psi$ for our experimentation). From now on, we will consider given the qualitative calculus as well as R (assumed to be closed for the converse operation). Then, we present a general scheme, as well as an algorithm, that can be followed when the qualitative calculus respects some conditions.

3.1 Theoretical Results

To perform our relaxation, we propose a generalization of the mapping operator introduced in Definition 4. The relaxation involves only taking into account variables (of the discrete network) whose domain does not belong to R. More formally, the definition of the new operator, denoted by $T_{DCN}^{-\mathsf{R}}$, is given by:

Definition 6. *Let $\mathcal{N} = (V, Q)$ be a QCN. The discrete relaxation $T_{DCN}^{-\mathsf{R}}(\mathcal{N})$ of N is the DCN $\mathcal{P} = (X, C)$ defined as follows:*

- *for each pair of variables $v_i, v_j \in V$ with $1 \leq i \leq j \leq n$ such as $Q_{ij} \notin \mathsf{R}$, X contains a variable x_{ij} such that $dom(x_{ij}) = Q_{ij}$;*
- *for each triple of variables $v_i, v_j, v_k \in V$ with $1 \leq i < k < j \leq n$, C contains a ternary constraint C_{ijk}, such that $scp(C_{ijk}) = \{x_{ij}, x_{ik}, x_{kj}\}$ and $rel(c_{ijk}) = \{(a, b, c) \in \mathsf{B}^3 : a \in b \circ c\}$, iff x_{ij}, x_{ik} and x_{kj} belong to X.*

It is immediate to see that $T_{DCN}^{-\mathsf{R}}$ is equivalent to T_{DCN} when $\mathsf{R} = \emptyset$. Also, $T_{DCN}^{-\mathsf{R}}(\mathcal{N})$ is clearly a sub-network of $T_{DCN}(\mathcal{N})$.

The following proposition shows that it is possible to exploit discrete relaxations to prove the unsatisfiability of a qualitative network. The proof is immediate since $T_{DCN}^{-\mathsf{R}}(\mathcal{N})$ is a sub-network of $T_{DCN}(\mathcal{N})$ which is equivalent to $\mathcal{N}$ with respect to satisfiability.

Proposition 1. *Let $\mathcal{N}$ be a QCN. If $T_{DCN}^{-\mathsf{R}}(\mathcal{N})$ is unsatisfiable then $\mathcal{N}$ is unsatisfiable.*

Figure 3 shows an illustration of discrete relaxations of a qualitative constraint network (from Allen's calculus). On the left, we have the discrete network that corresponds to the direct mapping (according to Definition 4) of the QCN presented at the top of the figure: each constraint becomes a variable and each triple of variables becomes a ternary constraint. On the right, we have the discrete network that corresponds to the discrete relaxation based on the total relation. The variable $x_{2,3}$ has been discarded since it corresponds to the total relation $Q_{2,3}$. Consequently, ternary constraints that may involve $x_{2,3}$ have been discarded too. Trivially, both networks are unsatisfiable. Indeed, we have v_1 *before* v_4 and v_1 *after* v_4. Hence, in the relaxed discrete network, $dom(x_{1,4})$ is restricted to

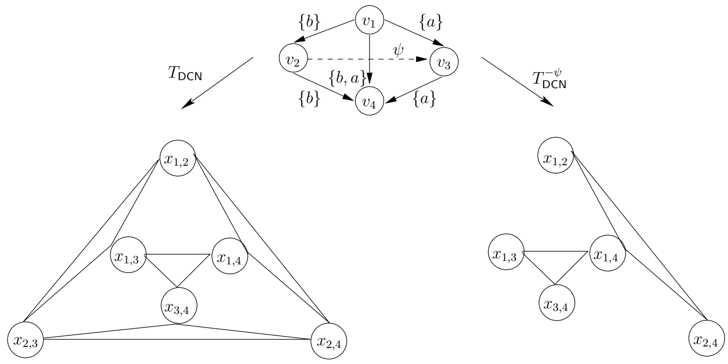

Fig. 3. Illustration of discrete encodings (with and without relaxation)

the *before* value (using the ternary constraint $c_{1,2,4}$) and also to the *after* value (using the ternary constraint $c_{1,3,4}$).

Once a discrete network using $T^{-\mathsf{R}}_{DCN}$ has been generated, it is possible to use any CSP solver to find solutions, if any. As mentioned above, if no solution is found, it means that the initial qualitative network is unsatisfiable. However, for any found solution I (i.e. consistent instantiation), it may be interesting to exploit it in the qualitative network. Of course, in the general case, each solution does not correspond to a piece of a consistent scenario of the qualitative network, but by considering each of them in turn, it is possible to render the approach complete. Indeed, if no consistent scenario can be built from all solutions of the discrete relaxation, the qualitative network is proved to be unsatisfiable.

To do this, we need to propose a generalization of the mapping operator introduced in Definition 5.

Definition 7. *Let $\mathcal{N} = (V, Q)$ be a* QCN*,* R *be a subset of* 2^{B} *and* $\mathcal{P} = (X, C)$ *be a* DCN *such that* $\mathcal{P} \preceq T^{-\mathsf{R}}_{DCN}(\mathcal{N})$*. The qualitative restriction* $T^{+\mathsf{R}}_{QCN}(\mathcal{P})$ *of* $\mathcal{P}$ *is the* QCN $\mathcal{N}' = (V, Q')$ *defined by:*

- $\forall\ 1 \leq i \leq n$, $Q'_{ii} = \{\mathsf{Id}\}$;
- $\forall\ 1 \leq i < k < j \leq n$, $Q'_{ij} = dom(x_{ij})$ *and* $Q'_{ji} = (Q'_{ij})^{\sim}$ *if* $x_{ij} \in X$*, and* $Q'_{ij} = Q_{ij}$ *and* $Q'_{ji} = Q_{ji}$*, otherwise.*

We can apply this operator to any solution found in $\mathcal{P}$. Indeed, by considering I as a DCN (the domain of each variable being reduced to a single value), we can build the qualitative restriction of I and then obtain a qualitative network $\mathcal{N}'$. It is interesting to note that in this case, the constraints of $\mathcal{N}'$ are basically composed of relations of R as well as atomic relations. This can contribute to facilitate our task (i.e. determining the satisfiability/unsatisfiability of $\mathcal{N}$). Indeed, there exists a class $\mathcal{H} \subset 2^{\mathsf{B}}$ of relations, called ORD-Horn relations [25], such that, for some qualitative calculi, weak composition closure is complete with respect to satisfiability. QCN only composed of ORD-Horn relations are called ORD-Horn QCN. We have the following proposition.

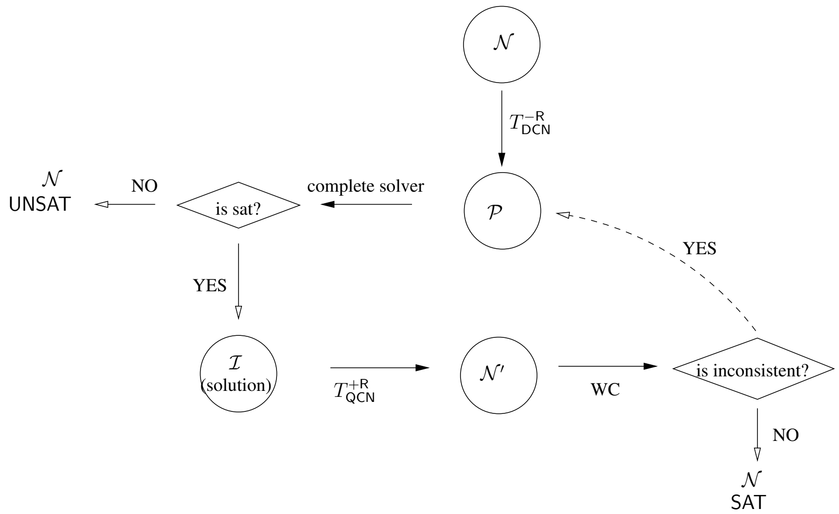

Fig. 4. A general scheme to determine the satisfiability of a QCN $\mathcal{N}$

Proposition 2. *Let $\mathcal{N}$ be a* QCN *and I be a solution of $T^{-R}_{DCN}(\mathcal{N})$. If* R *is a subset of ORD-Horn relations and $T^{+R}_{QCN}(I)$ is closed by weak composition, then $\mathcal{N}$ is satisfiable.*

The proof can be derived from the elements introduced above. This proposition can be directly exploited by some qualitative calculi, e.g. the Allen's one. Indeed, for the Allen's calculus, the set of ORD-Horn relations is closed with respect to composition, intersection and converse. It implies that, applying a weak composition closure algorithm on an ORD-Horn network yields another ORD-Horn network. In other words, for the Allen's algebra, Propositions 1 and 2 provide us with an original way to determine the satisfiability of a qualitative network by using two simple mapping operators, a CSP solver and a weak composition closure algorithm.

3.2 General Scheme

A general scheme about the exploitation of discrete networks to deal with the relaxation of qualitative networks is given in Figure 4. This scheme holds if the qualitative calculus is such that all atomic relations are ORD-Horn relations and the weak composition closure is complete with respect to satisfiability. Remark that the closure of the set of ORD-Horn relations for composition and intersection may improve the applicability of this scheme.

The aim of the scheme is to determine the satisfiability of a given qualitative network $\mathcal{N}$. Using the operator T^{-R}_{DCN} (see Definition 6), we first obtain a discrete network $\mathcal{P}$. Then, we use a complete solver to search for a solution. If no solution is found, then $\mathcal{N}$ is proved to be unsatisfiable (see Proposition 1). Otherwise, a solution I is returned by the solver. Using the operator T^{+R}_{QCN} (see Definition

7), we can then obtain a new qualitative network $\mathcal{N}'$. If this network can be enforced to be $\circ$-closed, it means that the satisfiability of $\mathcal{N}$ has been proved (see Proposition 2). If this is not the case, we just ask the complete solver to search for the next solution.

Algorithm 1. checkSatisfiability

```
Data: QCN 𝒩
Result: the satisfiability of 𝒩
begin
    𝒫 = T_DCN^{-ψ}(𝒩)
    /* we assume that solve(𝒫), called iteratively, returns the
       solutions of 𝒫, one by one                                */
    solution ← solve(𝒫)
    while solution ≠ null do
        if GAC(solution ∩ T_DCN(𝒩)) ≠ ⊥ then
            return SAT
        solution ← solve(𝒫)
    return UNSAT
end
```

We will instantiate this general scheme by using the total relation as relaxation (i.e. by choosing $\mathsf{R} = \psi$) and considering the Allen's calculus whose atomic and total relations belong to the set of ORD-Horn relations, which consequently satisfies the previous requirements. More precisely, we will use the function *checkSatisfiability* depicted by Algorithm 1. The main difference between this algorithm and the scheme presented above is the fact that, instead of using the operator $T_{QCN}^{+\mathsf{R}}$ and the $\circ$-closure operation, we use a GAC algorithm. Indeed, if we consider the network obtained by restricting $T_{DCN}(N)$ with respect to the last found solution, and if we apply GAC, then we obtain the same result as the one obtained by applying the $\circ$-closure on $T_{QCN}^{+\mathsf{R}}(solution)$ (see Figure 2).

4 Experiments

To study the practical interest of our proposed relaxation framework, some preliminary experiments on unsatisfiability detection have been conducted using total relations relaxation. Some qualitative constraint networks have been randomly generated and converted to discrete constraint networks using the qualitative algebra toolkit QAT [13].We have limited our attention to random networks because of the absence of structured qualitative instances. Then, Abscon, a state-of-the-art generic CSP solver (see `http://www.cril.univ-artois.fr/CPAI06`) has been run on the obtained discrete instances. Note that the satisfiability detection part of our general framework is currently under development.

In our experiments, as path consistency (which corresponds to weak composition) on qualitative constraint networks can lead to the elimination of some total relations (replaced by implied ones), two different relaxation modes have been considered. For the first one, relaxation corresponds to the elimination of all total relations: this will be called strong relaxation. For the second one, relaxation (elimination of total relations) is done after achieving weak composition: this will be called weak relaxation. Of course, the strong relaxation can eliminate more total relations than the weak one. Using these two relaxation modes, an experimental comparison has been conducted against a direct resolution (no relaxation).

To generate random QCN instances, we have used three parameters : the number of variables (N), the density of non-total relations (D) and the average number of basic relations for those constraints (L). The generated QCN instances are composed of 50 variables built from the well-known Allen's calculus (which is composed of 13 basic relations). For each instance, the number of atomic relations in each constraint has been fixed to $L_1 = 3.25$ and $L_2 = 6.5$ while the density of non-total relations has been varied from 0.01 to 0.99 with a step of 0.01. For each parameter setting, 100 networks have been generated.

Comparison is done according to the percentage of detected unsatisfiable instances and average cpu time (in ms) needed to solve such instances. The first measure is used to show the precision of our relaxation framework i.e. the highest this percentage is, the better our approximation is.

Figure 4 indicates that for L_1 (figure on top) and L_2 (figure on bottom), the strong relaxation is far less efficient, in term of precision (i.e. detected unsatisfiable instances), than the weak one. Note that for L_1, 100% of unsatisfiable instances have been detected by the weak relaxation, and that for L_2, the gap between weak and strong relaxations is increasing. It then appears that applying weak composition before relaxation improves the precision of our unsatisfiability detection method. Note that these experiments are only presented around the threshold which is the most interesting area. Of course, beyond the highest density mentioned in Figure 4, the accuracy of both relaxation methods always reaches 100%.

It is interesting to see that in term of cpu time, the two relaxation modes present totally different behaviours. This can be observed for both strong and weak relaxations in Figures 6 and 7, respectively. One could imagine that strong relaxation is faster than weak relaxation on detected unsatisfiable instances, but here, we have to be aware that the average computation is not done on the same basis. Indeed, we make our computation by only keeping the instances that were detected unsatisfiable by, on the one hand (Figure 6), both the strong relaxation and the direct resolution, and on the other hand (Figure 7), both the weak relaxation and the direct resolution. Note that, as the density increases, the two relaxations tend to be very close to the behaviour of the direct resolution. Also, remark that in Figure 7 (on bottom), at a density set to 0.18, we obtain a significant gain in term of cpu time when using the weak relaxation.

Finally, to make our experimental protocol more significant, we have conducted another series of experiments on a sample of hard QCN instances above the threshold. To this end, a number of atomic relations per constraint ($L_3 = 9.75$) and a non-total relations density ($D = 0.65$) have been chosen such that the generated networks are difficult to solve. As we can observe in

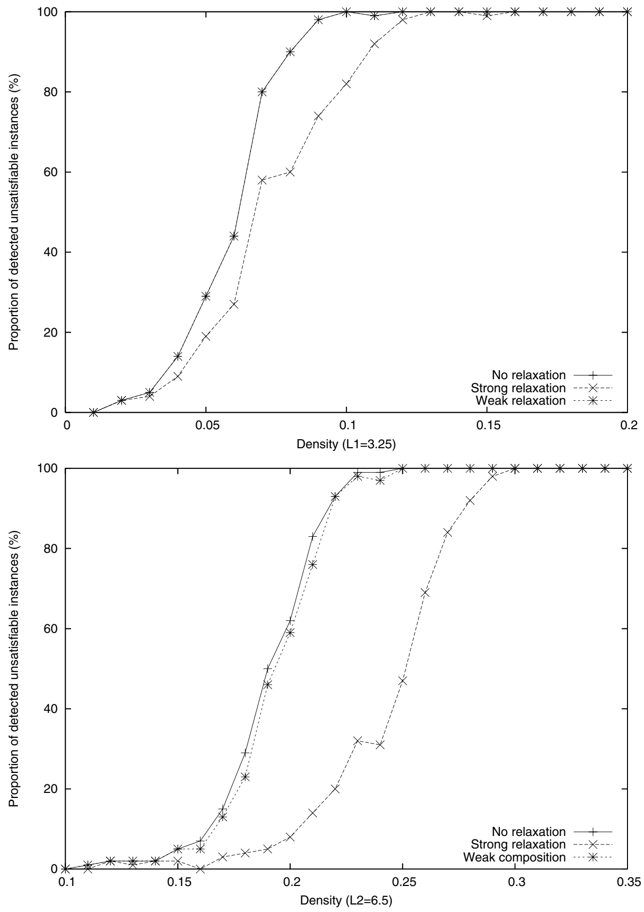

Fig. 5. Proportion of instances detected as unsatisfiable

Table 1. Results on a sample of difficult instances

Method	No relaxation	Strong relaxation	Weak relaxation
# instances	65	65	65
# Time Out (TO)	32	16	16
# Sat	0	0	0
# Unsat	33	49	49
# Time Out / # instances	49,23%	24,62%	24,62%
Total Time (33 unsat) in s	14,572	3,765	4,016
Avg Time (33 unsat) in s	441.5	114.1	121.7
Ratio	0,00%	74,16%	72,43%

Table 1, our approach is very efficient for detecting the unsatisfiability of these hard instances. Indeed, without relaxation, 49.23% of instances are not solved within the allowed cpu time (1200s), whereas on the same networks only 24.62%

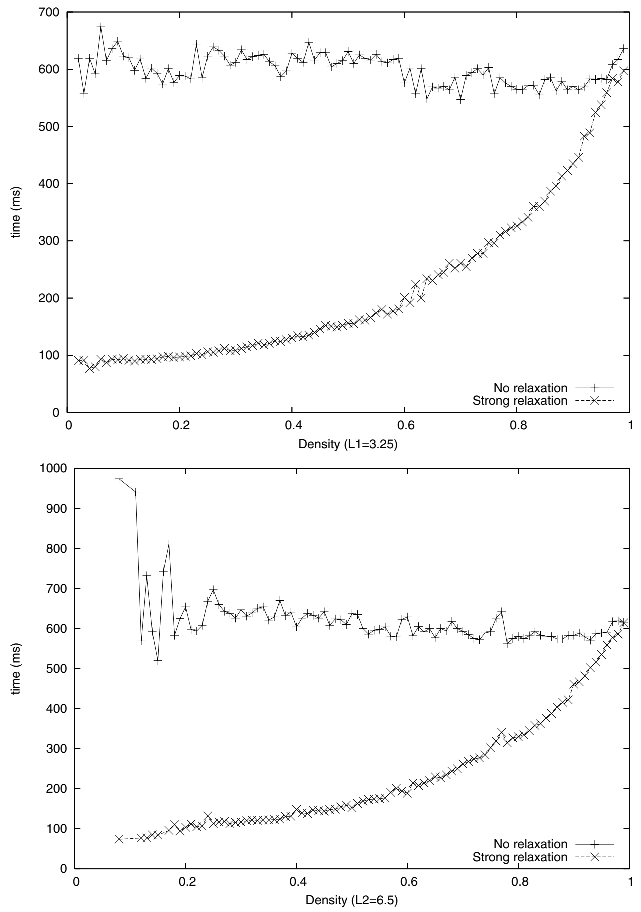

Fig. 6. Average cpu time to detect unsatisfiability (strong relaxation)

of instances are not detected to be unsatisfiable by our approach. Interestingly enough, on such detected instances a huge gain (up to a four-fold improvement) is obtained with respect to cpu time. On these hard instances, the weak and strong relaxations present very close performances.

5 Future Works and Conclusions

In this paper, a general and complete relaxation framework for qualitative reasoning is proposed. It allows the user to consider any kind of relaxation and any type of relations. Its originality comes from the fact that we exploit a mapping towards discrete constraint networks. Considering total relations for relaxation, promising preliminary results have been obtained with respect to unsatisfiability.

The generality and flexibility behind our approach offer some rooms for further improvements. It can be strengthened by integrating results from constraint

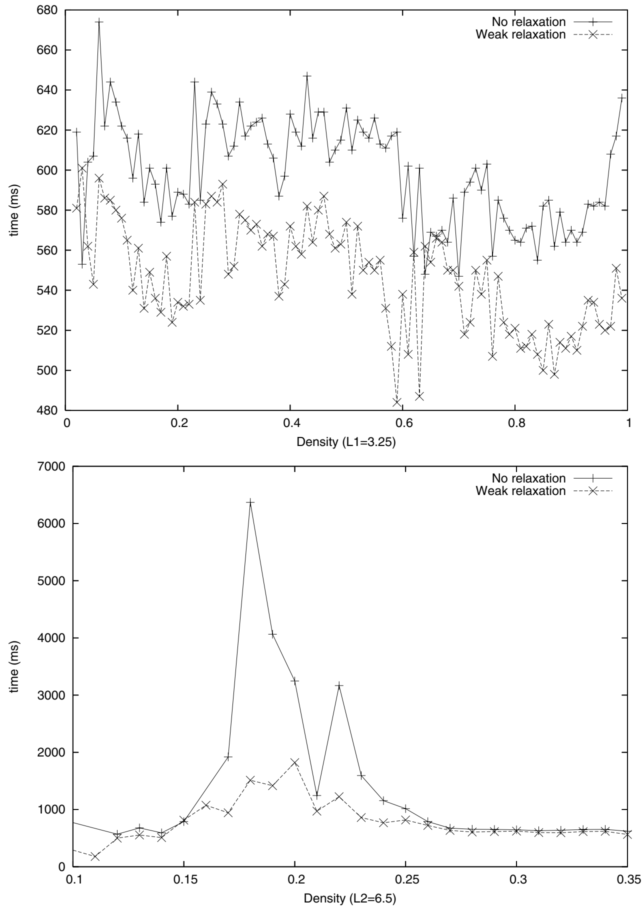

Fig. 7. Average cpu time to detect unsatisfiability (strong relaxation)

satisfaction and satisfiability problems. For example, the completeness can be efficiently addressed by exploiting advanced methods in satisfiability problems (e.g. randomisation and restarts [17]). We can also imagine the integration of different forms of restriction in order to reintroduce some of the previous relaxed relations. This could be done, for example, by using constraint weighting [7] to select the best relations that might be added.

References

1. Allen, J.F.: An interval-based representation of temporal knowledge. In: Proceedings of IJCAI'81, pp. 221–226 (1981)
2. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Communications of the ACM 26(11), 832–843 (1983)

3. Balbiani, P., Condotta, J.F., Fariñas del Cerro, L.: A model for reasoning about bidimensional temporal relations. In: Proceedings of KR'98, pp. 124–130 (1998)
4. Balbiani, P., Osmani, A.: A model for reasoning about topologic relations between cyclic intervals. In: Proceedings of KR'00, pp. 378–385 (2000)
5. Bistarelli, S., Codognet, P., Rossi, F.: An abstraction framework for soft constraints and its relationship with constraint propagation. In: Choueiry, B.Y., Walsh, T. (eds.) SARA 2000. LNCS (LNAI), vol. 1864, pp. 71–86. Springer, Heidelberg (2000)
6. Bistarelli, S., Codognet, P., Rossi, F.: Abstracting soft constraints: framework, properties, examples. Artificial Intelligence 139, 175–211 (2002)
7. Boussemart, F., Hemery, F., Lecoutre, C., Sais, L.: Boosting systematic search by weighting constraints. In: Proceedings of ECAI'04, pp. 146–150 (2004)
8. Cadoli, M.: Tractable Reasoning in Aritificial Intelligence.LNCS, vol. 941, Springer, Heidelberg (1995)
9. Caseau, Y.: Abstract interpretation of constraints on order-sorted domains. In: Proceedings of ISLP'91, pp. 435–452 (1991)
10. Choueiry, B., Faltings, B., Weigel, R.: Abstraction by interchangeability in resource allocation. In: Proceedings of IJCAI'95, pp. 1694–1710 (1995)
11. Choueiry, B., Noubir, G.: On the computation of local interchangeability in discrete constraint satisfaction problems. In: Proceedings of AAAI'98, pp. 326–333 (1998)
12. Condotta, J.F., Dalmeida, D., Lecoutre, C., Sais, L.: From qualitative to discrete constraint networks. In: Freksa, C., Kohlhase, M., Schill, K. (eds.) KI 2006. LNCS (LNAI), vol. 4314, pp. 54–64. Springer, Heidelberg (2007)
13. Condotta, J.F., Ligozat, G., Saade, M.: The QAT: A Qualitative Algebra Toolkit. In: Proceedings of TIME'2006, pp. 69–77 (2006)
14. Cousot, P., Cousot, R.: Abstract interpretation frameworks. Logic and Computation 2(4), 447–511 (1992)
15. Freuder, E., Sabin, D.: Interchangeability supports abstraction and reformulation for constraint satisfaction. In: Proceedings of SARA'95 (1995)
16. Giunchiglia, F., Walsh, T.: A theory of abstraction. Artificial Intelligence 56(2-3), 323–390 (1992)
17. Gomes, C.P., Selman, B., Crato, N., Kautz, H.: Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. Journal of Automated Reasoning 24, 67–100 (2000)
18. Hornsby, K., Egenhofer, M.J., Hayes, P.J.: Modeling cyclic change. In: Proceedings of REIS'99, pp. 98–109 (2006)
19. Isli, A., Cohn, A.G.: A new approach to cyclic ordering of 2D orientations using ternary relation algebras. Artificial Intelligence 122(1–2), 137–187 (2000)
20. Jussien, N.: Relaxation de Contraintes pour les problémes dynamiques. PhD thesis, Universitè de Rennes I (1997)
21. Jussien, N., Boizumault, P.: Implementing constraint relaxation over finite domains using ATMS. In: Jampel, M., Maher, M.J., Freuder, E.C. (eds.) Over-Constrained Systems. LNCS, vol. 1106, Springer, Heidelberg (1996)
22. Ligozat, G.: Reasoning about cardinal directions. Journal of Visual Languages and Computing 1(9), 23–44 (1998)
23. Marriott, K.: Frameworks for abstract interpretation. Acta Informatica 30, 103–129 (1993)
24. Merchez, S., Lecoutre, C., Boussemart, F.: Abstraction de réseaux de contraintes. Revue d'Intelligence Artificielle 20(1), 31–62 (2006)
25. Nebel, B., Bürckert, H.J.: Reasoning About Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra. Journal of the ACM 42(1), 43–66 (1995)

26. Pham, D.N., Thornton, J., Sattar, A.: Modelling and solving temporal reasoning as propositional satisfiablitly. In: van Beek, P. (ed.) CP 2005. LNCS, vol. 3709, pp. 117–131. Springer, Heidelberg (2005)
27. Pujari, A.K., Kumari, G.V., Sattar, A.: Indu: An interval and duration network. In: Proceedings of the Australian Joint Conference on Artificial Intelligence, pp. 291–303 (1999)
28. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Proceedings of KR'92, pp. 165–176 (1992)
29. Selman, B., Kautz, H.: Knowledge compilation and theory approximation. Journal of the ACM 43(2), 193–224 (1996)
30. Shrag, R., Miranker, D.: Abstraction and the csp phase transition boundary. In: Proceedings of the 4^{th} International Symposium on Artificial Intelligence and Mathematics, pp.138–141, 1996.
31. Yokoo, M.: Constraint relaxation in distributed constraint satisfaction problems. In: Proceedings of ICTAI'03, pp. 56–63 (1993)