Efficiently Reasoning about Qualitative Constraints through Variable Elimination

Michael Sioutis
Université Lille-Nord de France, Artois
CRIL-CNRS UMR 8188
Lens, France
sioutis@cril.fr

Zhiguo Long
University of Technology Sydney
QCIS, FEIT
Sydney, Australia
zhiguo.long@student.uts.edu.au

Sanjiang Li
University of Technology Sydney
QCIS, FEIT
Sydney, Australia
sanjiang.li@uts.edu.au

ABSTRACT

We introduce, study, and evaluate a novel algorithm in the context of qualitative constraint-based spatial and temporal reasoning, that is based on the idea of variable elimination, a simple and general exact inference approach in probabilistic graphical models. Given a qualitative constraint network $\mathcal{N}$, our algorithm enforces a particular directional local consistency on $\mathcal{N}$, which we denote by $\overleftarrow{\diamond}$-consistency. Our discussion is restricted to *distributive* subclasses of relations, i.e., sets of relations closed under converse, intersection, and weak composition and for which weak composition *distributes* over non-empty intersections for all of their relations. We demonstrate that enforcing $\overleftarrow{\diamond}$-consistency on a given qualitative constraint network defined over a distributive subclass of relations allows us to decide its satisfiability. The experimentation that we have conducted with random and real-world qualitative constraint networks defined over a distributive subclass of relations of the Region Connection Calculus, shows that our approach exhibits unparalleled performance against competing state-of-the-art approaches for checking the satisfiability of such constraint networks.

CCS Concepts

•**Computing methodologies** → **Spatial and physical reasoning; Temporal reasoning;** •**Theory of computation** → *Constraint and logic programming;* Algorithm design techniques;

Keywords

Spatial/temporal reasoning, qualitative constraint, distributive class of relations, variable elimination, exact inference

1. INTRODUCTION

Spatial and temporal reasoning is a major field of study in Artificial Intelligence. This field has received a lot of attention over the past decades, as it extends to a plethora of areas and domains that include, but are not limited to, ambient intelligence, dynamic GIS, cognitive robotics, and spatiotemporal design. In this context, an emphasis has been made on qualitative constraint-based spatial and temporal reasoning, which abstracts from numerical quantities of space and time by using qualitative descriptions instead (e.g., *precedes*, *contains*, *is left of*). The conciseness of the constraint language used in the qualitative approach provides a promising framework that further boosts research and applications in spatial and temporal reasoning.

SETN '16, May 18-20, 2016, Thessaloniki, Greece

DOI: http://dx.doi.org/10.1145/2903220.2903226

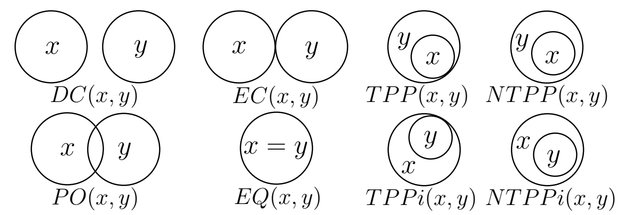


Figure 1: The relations of RCC-8

A subset of the Region Connection Calculus (RCC), denoted by RCC-8, is the dominant constraint language in Artificial Intelligence for representing and reasoning about qualitative spatial information [27]. In particular, RCC-8 makes use of the topological relations *disconnected* (DC), *externally connected* (EC), *equal* (EQ), *partially overlapping* (PO), *tangential proper part* (TPP), *tangential proper part inverse* ($TPPi$), *non-tangential proper part* ($NTPP$), and *non-tangential proper part inverse* ($NTPPi$) to encode knowledge about the spatial relations between regions in some topological space, as depicted in Figure 1. Other notable and well known constraint languages that can be used for representing and reasoning about qualitative spatial or temporal information include Point Algebra [39], Cardinal Direction Calculus [12, 24], Interval Algebra [1], and Block Algebra [3]. The problem of representing and reasoning about qualitative information can be modeled as a qualitative constraint network (QCN), i.e., a network of constraints corresponding to qualitative spatial or temporal relations between spatial or temporal variables respectively.

Given a QCN $\mathcal{N}$, we are particularly interested in its *satisfiability problem*, which is the problem of deciding if there exists a spatial or temporal interpretation of the variables of $\mathcal{N}$ that satisfies its constraints, such an interpretation being called a *solution* of $\mathcal{N}$. We focus on the recently studied notion of *distributive* subclasses of relations [22, 25], i.e., sets of relations closed under converse, intersection, and weak composition and for which weak composition *distributes* over non-empty intersections for all of their relations, and exploit such subclasses of relations to efficiently reason about qualitative constraints. We draw motivation for our work from the fact that real-world QCNs that have been used in the literature are defined over distributive subclasses of relations and call for efficient reasoning methods that can scale to millions of spatial or temporal variables [22, 31, 33]. Specifically, we make the following contributions: (i) we consider a particular directional local consistency, denoted by $\overleftarrow{\diamond}$-consistency, and demonstrate that $\overleftarrow{\diamond}$-consistent and not trivially inconsistent QCNs defined over a distributive subclass of relations are satisfiable, (ii) we introduce and study a novel algorithm in the context of qualitative constraint-based spatial and temporal reasoning, that efficiently enforces $\overleftarrow{\diamond}$-consistency on a given QCN and that is based on the idea of variable elimination, a simple and general exact inference approach in probabilistic graphical models, such as Bayesian networks and Markov random fields [40], and, finally, (iii) we evaluate our approach with random and real-world QCNs defined over a distributive subclass of relations of RCC-8, and show that it exhibits unparalleled performance against competing state-of-the-art approaches for checking the satisfiability of such QCNs.

The paper is organized as follows. In Section 2 we give some preliminary notions about qualitative constraint languages, spatial and temporal QCNs, distributive subclasses of relations, and related constraint properties of QCNs. In Section 3 we present our approach for efficiently checking the satisfiability of a given QCN defined over a distributive subclass of relations, and also show how a solution of that QCN can actually be extracted; the latter contribution involves performing a generic backtrack-free procedure for refining a $\overleftarrow{\diamond}$-consistent and not trivially inconsistent QCN $\mathcal{N}$ defined over a distributive subclass of relations to a scenario of $\mathcal{N}$, i.e., an atomic satisfiable sub-QCN of $\mathcal{N}$, and then using some known method from the literature for valuating the variables of $\mathcal{N}$ in order to satisfy its constraints. In Section 4 we evaluate our approach with random and real-world QCNs defined over a distributive subclass of relations of RCC-8, against competing state-of-the-art approaches for checking the satisfiability of such QCNs. Finally, in Section 5 we conclude and give some perspectives for future work.

2. PRELIMINARIES

A (binary) qualitative spatial or temporal constraint language is based on a finite set B of *jointly exhaustive and pairwise disjoint* (JEPD) relations defined on a domain D [19], called the set of *base* relations. The base relations of the set B of a particular qualitative constraint language can be used to represent the definite knowledge between any two entities with respect to the given level of granularity. B contains the *identity* relation Id, and is closed under the *converse* operation ($^{-1}$). Indefinite knowledge can be specified by a union of possible base relations, and is represented by the set containing them. Hence, 2^{B} represents the total set of relations. 2^{B} is equipped with the usual set-theoretic operations (union and intersection), the converse operation, and the *weak composition* operation denoted by $\diamond$ [29]. For every $r \in 2^{\mathsf{B}}$, we have that $r^{-1} = \{b^{-1} \mid b \in r\}$. The weak composition ($\diamond$) of two base relations $b, b' \in \mathsf{B}$ is defined as the strongest relation $r \in 2^{\mathsf{B}}$ that contains $b \circ b'$, or, formally, $b \diamond b' = \{b'' \in \mathsf{B} \mid b'' \cap (b \circ b') \neq \emptyset\}$, where $b \circ b' = \{(x, y) \in \mathsf{D} \times \mathsf{D} \mid \exists z \in \mathsf{D} \text{ s.t. } (x, z) \in b \land (z, y) \in b'\}$ is the relational composition of b and b'. For every $r, r' \in 2^{\mathsf{B}}$, we have that $r \diamond r' = \{b \diamond b' \mid b \in r, b' \in r'\}$.

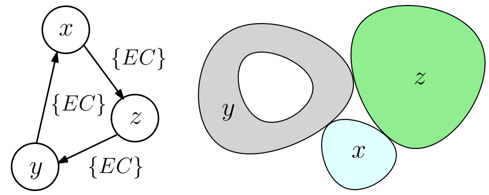


Figure 2: A QCN of RCC-8 along with a solution

The Region Connection Calculus (RCC) is a first-order theory for representing and reasoning about mereotopological information [27]. The domain D of RCC comprises all possible non-empty regular subsets of some topological space. These subsets serve as regions in RCC. Further, they do not have to be internally connected and do not have a particular dimension, but they are usually required to be *closed* [28]. A subset X of some topological space is regular closed, if X equals the closure of its interior. The base relations of RCC are the following: *disconnected* (DC), *externally connected* (EC), *equal* (EQ), *partially overlapping* (PO), *tangential proper part* (TPP), *tangential proper part inverse* ($TPPi$), *non-tangential proper part* ($NTPP$), and *non-tangential proper part inverse* ($NTPPi$). These eight base relations form the RCC-8 constraint language and are depicted in Figure 1 (using the Euclidean plane). Relation EQ is the identity relation Id of RCC-8. Other notable and well known qualitative spatial and temporal constraint languages include Point Algebra [39], Cardinal Direction Calculus [12, 24], Interval Algebra [1], and Block Algebra [3].

The weak composition operation $\diamond$ along with the converse operation $^{-1}$, and the total set of relations 2^{B} along with the identity relation Id of a qualitative constraint language, form an algebraic structure $(2^{\mathsf{B}}, \mathsf{Id}, \diamond, ^{-1})$ that can correspond to a *relation algebra* in the sense of Tarski [36]. This topic has been extensively discussed in [11]. In fact, [11] summarizes findings on the relationship between relation algebras and qualitative constraint languages into the following result:

Proposition 1 ([11]). *Each one of the qualitative constraint languages of Point Algebra, Cardinal Direction Calculus, Interval Algebra, Block Algebra, and* RCC-*8 is a relation algebra with the algebraic structure* $(2^{\mathsf{B}}, \mathsf{Id}, \diamond, ^{-1})$.

In what follows, for a qualitative constraint language that is a relation algebra with the algebraic structure $(2^{\mathsf{B}}, \mathsf{Id}, \diamond, ^{-1})$, we will simply say that it is a relation algebra, as the algebraic structure will always be of the same format.

The problem of representing and reasoning about qualitative information can be modeled as a *qualitative constraint network* (QCN), defined in the following manner:

Definition 1. *A* QCN *is a tuple* (V, C) *where:*

- $V = \{v_1, \ldots, v_n\}$ *is a non-empty finite set of spatial or temporal variables, where each such variable represents a spatial or temporal entity respectively;*
- C *is a mapping* $C : V \times V \to 2^{\mathsf{B}}$ *such that* $C(v, v) = \{\mathsf{Id}\}$ *for every* $v \in V$ *and* $C(v, v') = (C(v', v))^{-1}$ *for every* $v, v' \in V$.

An example of a QCN of RCC-8 is shown in Figure 2. In particular, the QCN comprises the set of variables $\{x, y, z\}$ and the constraints $C(x, y) = C(y, z) = C(z, x) = \{EC\}$; for simplicity, converse relations as well as Id loops are not mentioned or shown in the figure.

Note that we always regard a QCN as a complete network. In what follows, given a QCN $\mathcal{N} = (V, C)$ and $v, v' \in V$, relation $C(v, v')$ will also be denoted by $\mathcal{N}[v, v']$. Further, $\mathcal{N}\downarrow_{V'}$, with $V' \subseteq V$, will denote the QCN $\mathcal{N}$ restricted to V'. Finally, all considered graphs will be undirected.

Definition 2. *Let $\mathcal{N} = (V, C)$ be a* QCN*, then:*

- *a* solution *of $\mathcal{N}$ is a mapping $\sigma : V \rightarrow \mathsf{D}$, such that for each pair of variables $(u, v) \in V \times V$, we have that $(\sigma(u), \sigma(v))$ satisfies $C(u, v)$, i.e., there exists a base relation $b \in C(u, v)$ such that $(\sigma(u), \sigma(v)) \in b$;*
- *$\mathcal{N}$ is* satisfiable *iff it admits a solution;*
- *a* QCN *$\mathcal{N}'$ is* equivalent *to $\mathcal{N}$ if and only if it admits the same set of solutions as $\mathcal{N}$;*
- *a* sub-QCN *(refinement) $\mathcal{N}'$ of $\mathcal{N}$, denoted by $\mathcal{N}' \subseteq \mathcal{N}$, is a* QCN *$(V, C')$ s.t. $C'(v, v') \subseteq C(v, v')\ \forall v, v' \in V$;*
- *$\mathcal{N}$ is* atomic *iff $\forall v, v' \in V$, $C(v, v')$ is a* singleton relation*, i.e., a relation $\{b\}$ with $b \in \mathsf{B}$;*
- *a* partial scenario *of $\mathcal{N}$ on $V' \subseteq V$ is an atomic satisfiable sub-*QCN *$\mathcal{S}$ of $\mathcal{N}\downarrow_{V'}$;*
- *a* scenario *of $\mathcal{N}$ is a partial scenario of $\mathcal{N}$ on V;*
- *$\mathcal{N}$ is* weakly globally consistent *iff, for any $V' \subset V$, every partial scenario of $\mathcal{N}$ on V' can be extended to a partial scenario on $V' \cup \{v\} \subseteq V$, for any $v \in V \setminus V'$;*
- *a base relation $b \in C(v, v')$, with $v, v' \in V$, is a* feasible *base relation of $\mathcal{N}$ iff there exists a scenario $\mathcal{S} = (V, C')$ of $\mathcal{N}$ such that $C'(v, v') = \{b\}$;*
- *$\mathcal{N}$ is* minimal *iff $\forall v, v' \in V$ and $\forall b \in C(v, v')$, b is a feasible base relation of $\mathcal{N}$;*
- *the* constraint graph *of $\mathcal{N}$ is the graph (V, E), denoted by $\mathsf{G}(\mathcal{N})$, for which we have that $\{v, v'\} \in E$ iff $C(v, v') \neq \mathsf{B}$ and $v \neq v'$;*
- *$\mathcal{N}$ is* trivially inconsistent *iff $\exists u, v' \in V$ w/ $C(u, v) = \emptyset$.*

Note that the constraint graph of a QCN does not contain the *universal* relation B, as B is the non-restrictive relation that contains all base relations, thus, it does not really pose a constraint. Given two QCNs $\mathcal{N} = (V, C)$ and $\mathcal{N}' = (V', C')$, we have that $\mathcal{N} \cup \mathcal{N}'$ yields the QCN $\mathcal{N}'' = (V'', C'')$, where $V'' = V \cup V'$, $C''(u, v) = C''(v, u) = \mathsf{B}$ for all $(u, v) \in (V \setminus V') \times (V' \setminus V)$, $C''(u, v) = C(u, v) \cap C'(u, v)$ for every $u, v \in V \cap V'$, $C''(u, v) = C(u, v)$ for all $(u, v) \in (V \times V) \setminus (V' \times V')$, and $C''(u, v) = C'(u, v)$ for all $(u, v) \in (V' \times V') \setminus (V \times V)$.

The method of *algebraic closure* [29] (or *closure under weak composition*) applies the following iterative procedure on a given QCN $\mathcal{N} = (V, C)$ until a fixed state is reached:

$$\forall v_i, v_k, v_j \in V, C(v_i, v_j) \leftarrow C(v_i, v_j) \cap (C(v_i, v_k) \diamond C(v_k, v_j))$$

Due to the definition of the weak composition operation denoted by symbol $\diamond$, the algebraic closure method is sound for checking the satisfiability of a QCN; it only removes base relations that do not participate in any solution of that QCN. If a QCN becomes trivially inconsistent after the application of the algebraic closure method, then it is unsatisfiable.

Definition 3. *A* QCN *$\mathcal{N} = (V, C)$ is* $\diamond$-consistent *iff $\forall v_i, v_k, v_j \in V$ we have that $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$.*

Proposition 2 ([11]). *Let $\mathcal{N} = (V, C)$ be an atomic* QCN *of Point Algebra, Cardinal Direction Calculus, Interval Algebra, Block Algebra, or* RCC*-8. Then, $\mathcal{N}$ is satisfiable if and only if it is $\diamond$-consistent.*

Applying (or enforcing) $\diamond$-consistency on a given QCN $\mathcal{N}$, i.e., making $\mathcal{N}$ $\diamond$-consistent, requires the implementation of the algebraic closure method through an algorithm. Such an algorithm requires $O(n^3|\mathsf{B}|)$ runtime for a given QCN over n variables [39]. The $\diamond$-consistent QCN obtained after the application of the algebraic closure method on a QCN $\mathcal{N}$ is unique and equivalent to $\mathcal{N}$. Further, it is called the *$\diamond$-consistent* QCN *of $\mathcal{N}$* and it is denoted by $\diamond(\mathcal{N})$. Network $\diamond(\mathcal{N})$ corresponds to the largest (with respect to $\subseteq$) $\diamond$-consistent sub-QCN of $\mathcal{N}$. By definition, $\diamond$-consistency involves the complete underlying graph of a given QCN $\mathcal{N}$, as it considers all possible triples of variables of $\mathcal{N}$. We can obtain a weaker notion of $\diamond$-consistency, by considering only the pairs of variables of $\mathcal{N}$ that correspond to a subset of the set of edges of its complete underlying graph.

Definition 4. *Given a* QCN *$\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, $\mathcal{N}$ is* $\mathring{\diamond}_G$-consistent *iff $\forall \{v_i, v_j\}, \{v_i, v_k\}, \{v_k, v_j\} \in E$ we have that $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$.*

Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, we have that $\mathring{\diamond}_G$-consistency can be applied on $\mathcal{N}$ in $O(\delta|E||\mathsf{B}|)$ time, where δ is the maximum degree of G [6]. The *$\mathring{\diamond}_G$-consistent* QCN *of $\mathcal{N}$* is denoted by $\mathring{\diamond}_G(\mathcal{N})$. Clearly, if G is a complete graph, $\mathring{\diamond}_G$-consistency is equivalent to $\diamond$-consistency. In the general case, it should be clear that $\diamond(\mathcal{N}) \subseteq \mathring{\diamond}_G(\mathcal{N})$.

Definition 5. *A* subclass *of relations is a subset $\mathcal{A} \subseteq 2^{\mathsf{B}}$ that contains the singleton relations of 2^{B} and is closed under converse, intersection, and weak composition.*

Given three relations r, r', $r'' \in 2^{\mathsf{B}}$, we say that weak composition distributes over intersection if we have that $r \diamond (r' \cap r'') = (r \diamond r') \cap (r \diamond r'')$ and $(r' \cap r'') \diamond r = (r' \diamond r) \cap (r'' \diamond r)$.

Definition 6. *A subclass $\mathcal{A} \subseteq 2^{\mathsf{B}}$ is* distributive *iff weak composition distributes over non-empty intersections for all relations r, r', $r'' \in \mathcal{A}$. A distributive subclass $\mathcal{A} \subseteq 2^{\mathsf{B}}$ is* maximal *iff there exists no distributive subclass $\mathcal{A}' \subseteq 2^{\mathsf{B}}$ such that $\mathcal{A}'$ properly contains $\mathcal{A}$.*

We list the following two properties for a given qualitative constraint language $\mathcal{L}$.

$$\mathcal{L} \text{ is a relation algebra.} \tag{1}$$

$$\text{Every atomic } \diamond\text{-consistent QCN of } \mathcal{L} \text{ is satisfiable.} \tag{2}$$

With respect to distributive subclasses of relations, we have the following result:

Theorem 1 ([25]). *Let $\mathcal{N} = (V, C)$ be a* QCN *defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2. If $\mathcal{N}$ is $\diamond$-consistent and not trivially inconsistent, then $\mathcal{N}$ is minimal and weakly globally consistent.*

We recall the definition of a *Helly* subclass of relations.

Definition 7. *A subclass $\mathcal{A} \subseteq 2^{\mathsf{B}}$ is* Helly *[8] if and only if for any n relations r_1, r_2, $\ldots$ $r_n \in \mathcal{A}$ we have that:*

$$\bigcap_{i=1}^{n} r_i \neq \emptyset \text{ iff } \forall (1 \leq i, j \leq n)\ r_i \cap r_j \neq \emptyset$$

Then, we have the following result by Long and Li in [25]:

Theorem 2 ([25]). *A subclass $\mathcal{A} \subseteq 2^{\mathsf{B}}$ of a qualitative constraint language that satisfies property 1 is distributive if and only if it is Helly.*

Let us recall the definition of the *patchwork* property [26], which we tailor to the context of $\diamond$-consistency.

Definition 8. *A subclass* $\mathcal{A} \subseteq 2^{\mathsf{B}}$ *has* patchwork *iff for any two* $\diamond$*-consistent and not trivially inconsistent* QCN*s* $\mathcal{N} = (V, C)$ *and* $\mathcal{N}' = (V', C')$ *defined over* $\mathcal{A}$ *such that* $C(u, v) = C'(u, v)$ *for every* $u, v \in V \cap V'$*, we have that the* QCN $\mathcal{N} \cup \mathcal{N}'$ *is satisfiable.*

Intuitively, patchwork ensures that "patching" together $\diamond$-consistent and not trivially inconsistent QCNs satisfying some conditions, yields a unified QCN that is satisfiable.

Given two graphs $G = (V, E)$ and $G' = (V', E')$, G is a *subgraph* of G' (and G' is a *supergraph* of G), denoted by $G \subseteq G'$, iff $V \subseteq V'$ and $E \subseteq E'$, and G and G' are *equal*, denoted by $G = G'$, iff $V = V'$ and $E = E'$. A graph G is said to be *chordal* (or triangulated) if every cycle of length at least 4 has a chord, which is an edge connecting two non-adjacent nodes of the cycle [10]. We have the following result with respect to chordal graphs and $\diamond_G$-consistency:

Proposition 3 ([2, 32]). *Let* $\mathcal{N} = (V, C)$ *be a* QCN *defined over a subclass of relations that has patchwork, and* $G = (V, E)$ *a graph such that* $\mathsf{G}(\mathcal{N}) \subseteq G$*. If* G *is chordal, and* $\mathcal{N}$ *is* $\diamond_G$*-consistent and not trivially inconsistent, then* $\mathcal{N}$ *is satisfiable.*

Due to Propositions 1 and 2, and the notion of weak global consistency in Theorem 1, which guarantees patchwork for a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2, it is clear that Proposition 3 applies to QCNs defined over a distributive subclass of relations of Point Algebra, Interval Algebra, Cardinal Direction Calculus, Block Algebra, or RCC-8.[1]

3. APPROACH

We propose a novel approach in the context of qualitative constraint-based spatial and temporal reasoning for solving a given QCN defined over a distributive subclass of relations, that is based on the idea of *variable elimination*, a simple and general exact inference algorithm in probabilistic graphical models, such as Bayesian networks and Markov random fields [40]. Notably, a variant of the variable elimination algorithm has also been introduced for finite-domain constraint satisfaction problems (CSPs) defined over *connected row convex* constraints [9] by Zhang et al. in [41]

3.1 Satisfiability Checking

Given a QCN, we are particularly interested in its *satisfiability problem*, i.e., the problem of checking (or deciding) if there exists a valuation of the variables of the QCN such that all of its constraints are satisfied by that valuation, such a valuation being called a *solution* of the QCN (as defined in Section 2). Here, we show how we can efficiently decide the satisfiability of a given QCN defined over a distributive subclass of relations. To this end, the notion of $\overleftarrow{\diamond}$*-consistency* will be essential to us, defined in the following manner:

[1]Actually, the result also applies to QCNs defined over larger subclasses of relations (so-called *tractable*) that properly contain the distributive subclasses of relations for the considered qualitative constraint languages [2], as patchwork holds for those larger subclasses of relations as well [17]. However, we will not be mentioning those subclasses of relations in our work here, as they are not of our interest for the techniques that we will present in the sequel.

Definition 9. *A* QCN $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ *is* $\overleftarrow{\diamond}$-consistent *iff for all* $v_i, v_k, v_j \in V$ *with* $i, j < k$ *we have that* $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$*.*

Given a QCN $\mathcal{N} = (V, C)$ with some ordering of its variables, the $\overleftarrow{\diamond}$*-consistent* QCN *of* $\mathcal{N}$ with respect to that ordering is denoted by $\overleftarrow{\diamond}(\mathcal{N})$. It is clear that $\diamond(\mathcal{N}) \subseteq \overleftarrow{\diamond}(\mathcal{N})$ for any possible ordering of the variables of $\mathcal{N}$. For simplicity, in what follows, when we state that a QCN is $\overleftarrow{\diamond}$-consistent, the ordering of the variables of that QCN for the use case at hand will be implicitly considered.

Theorem 3 ([25]). *Let* $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ *be a* QCN *defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2. If we have that* $C(v_i, v_j) \subseteq C(v_i, v_{n-1}) \diamond C(v_{n-1}, v_j)$ *for all* $v_i, v_j \in V$ *with* $i, j < n - 1$*, then* $\mathcal{N}{\downarrow}_{V \setminus \{v_{n-1}\}}$ *is satisfiable only if* $\mathcal{N}$ *is satisfiable.*

Using Theorem 3, we prove the following proposition:

Proposition 4. *Let* $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ *be a* QCN *defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2. If* $\mathcal{N}$ *is* $\overleftarrow{\diamond}$*-consistent and not trivially inconsistent, then* $\mathcal{N}$ *is satisfiable.*

Proof. Let V_l with $0 \leq l < n$ denote the set of variables $\{v_0, v_1, \ldots, v_l\}$, i.e., the set of variables of V from v_0 up to (and including) v_l. Then, due to $\mathcal{N}$ being $\overleftarrow{\diamond}$-consistent, for each $v_k \in V$ with $0 < k < n$ we have that $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$ for all $v_i, v_j \in V$ with $i, j < k$. As such, by Theorem 3, for each k with $0 < k < n$ we have that $\mathcal{N}{\downarrow}_{V_{k-1}}$ is unsatisfiable if $\mathcal{N}{\downarrow}_{V_k}$ is unsatisfiable. By the "if...then" transitive property, we have that $\mathcal{N}{\downarrow}_{V_1}$ is unsatisfiable if $\mathcal{N}{\downarrow}_{V} = \mathcal{N}$ is unsatisfiable. Let us assume that $\mathcal{N}$ is indeed unsatisfiable. Then, as $\mathcal{N}{\downarrow}_{V_1}[v_0, v_0] = \mathcal{N}{\downarrow}_{V_1}[v_1, v_1] = \{\mathsf{Id}\}$ by definition of a QCN, this can only mean that $\mathcal{N}{\downarrow}_{V_1}[v_0, v_1] = \emptyset$, or, equivalently, that $\mathcal{N}{\downarrow}_{V_1}[v_1, v_0] = \emptyset$, as $\mathcal{N}{\downarrow}_{V_1}[v_1, v_0] = (\mathcal{N}{\downarrow}_{V_1}[v_0, v_1])^{-1}$ by definition of a QCN. This is a contradiction, as $\mathcal{N}$ is not trivially inconsistent. We can conclude that if $\mathcal{N}$ is $\overleftarrow{\diamond}$-consistent and not trivially inconsistent, then $\mathcal{N}$ is satisfiable. $\square$

Next, we present an algorithm that uses the notion of $\overleftarrow{\diamond}$-consistency to decide the satisfiability of a given QCN defined over a distributive subclass of relations. The algorithm is given in Algorithm 1 and is called VarElimination as it is built on the idea of the variable elimination algorithm [40] that we mentioned earlier. However, we note that variables are not really eliminated in the process, they are just ignored, at some point, from further consideration. Given a QCN $\mathcal{N} = (V, C)$, algorithm VarElimination attempts to apply $\overleftarrow{\diamond}$-consistency on $\mathcal{N}$, and terminates with a value of (True, _[2]) if it succeeds in doing so without producing a trivial inconsistency. Algorithm VarElimination has a runtime of $O(\delta^2|V|)$, where δ is the maximum degree of the graph G returned through its output, as it iterates $O(|V|)$ variables (lines 3–4) and performs $O(\delta^2)$ constant operations for the variable v at hand in each iteration (lines 5–15). The relations of a qualitative constraint language are implemented

[2]The _ symbol denotes an anonymous variable, which we use when we are not interested in its instantiation for the use case at hand. We borrowed that syntax from Prolog [7].

Algorithm 1: VarElimination($\mathcal{N}$,α)

in/out : A QCN $\mathcal{N} = (V, C)$ over n variables.
in : A bijection α of V onto $\{0, 1, \dots, n-1\}$.
out : True or False, and a graph G.

1 **begin**
2 $\quad G \leftarrow (V, E = E(\mathsf{G}(\mathcal{N})))$;
3 $\quad$ **for** x **from** $n-1$ **to** 1 **do**
4 $\quad\quad v \leftarrow \alpha^{-1}(x)$;
5 $\quad\quad \mathsf{adj} \leftarrow \{v' \mid \{v', v\} \in E \wedge \alpha(v') < \alpha(v)\}$;
6 $\quad\quad$ **foreach** $v', v'' \in \mathsf{adj}$ **do**
7 $\quad\quad\quad$ **if** $\alpha(v') < \alpha(v'')$ **then**
8 $\quad\quad\quad\quad$ **if** $\{v', v''\} \notin E$ **then**
9 $\quad\quad\quad\quad\quad E \leftarrow E \cup \{\{v', v''\}\}$;
10 $\quad\quad\quad\quad \mathsf{temp} \leftarrow C(v', v'') \cap (C(v', v) \diamond C(v, v''))$;
11 $\quad\quad\quad\quad$ **if** $\mathsf{temp} \subset C(v', v'')$ **then**
12 $\quad\quad\quad\quad\quad C(v', v'') \leftarrow \mathsf{temp}$;
13 $\quad\quad\quad\quad\quad C(v'', v') \leftarrow \mathsf{temp}^{-1}$;
14 $\quad\quad\quad\quad$ **if** $C(v', v'') = \emptyset$ **then**
15 $\quad\quad\quad\quad\quad$ **return** (False,G);
16 $\quad$ **return** (True,G);

as bit vectors [20,38]. In this way, the fast bitwise AND operation can be used to perform the intersection operation. Note that the size of a bit vector is $|\mathsf{B}|$, with each of its bits corresponding to a base relation. Further, the converse of a relation of 2^{B} is obtained by a lookup in a converse table that stores the converse relation r^{-1} for each relation $r \in 2^{\mathsf{B}}$. In a similar manner, the weak composition of two relations of 2^{B} is obtained by a lookup in a weak composition table. However, in this case, a 2-dimensional array is required to store the weak compositions between all the relations, which has a $O(2^{2|\mathsf{B}|})$ memory footprint and can pose a challenge even for today's modern computers (e.g., consider Block Algebra with 13^p base relations for some integer $p \geq 1$). In the case where storing a $2^{|\mathsf{B}|} \times 2^{|\mathsf{B}|}$ 2-dimensional array is impractical, we can consider Hogge's method [16], which uses four small tables instead of a single large one, each one containing at most $2^{|\mathsf{B}|+1}$ entries. The result of a weak composition can then be obtained by the union of four table lookups plus three shift operations and some logical ANDs [16].

THEOREM 4. *Given a* QCN $\mathcal{N} = (V, C)$*, with* $|V| = n$*, defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2, and a bijection* $\alpha : V \rightarrow \{0, 1, \dots, n-1\}$*, we have that algorithm* VarElimination *terminates and returns* (True,_) *if and only if* $\mathcal{N}$ *is satisfiable.*

PROOF. We rewrite each variable of V as follows. For each $u \in V$, u is rewritten as $v_{\alpha(u)}$. As such, the set of variables V will be the set $\{v_0, v_1, \dots, v_{n-1}\}$. Let $\mathcal{N}'$ denote the refined QCN of $\mathcal{N}$ that results from the application of algorithm VarElimination on $\mathcal{N}$ until its termination with an output of either (True,_) or (False,_) (the refinement results due to the consistency operations in lines 12–13). It is clear that in general $\diamond(\mathcal{N}) \subseteq \mathcal{N}'$, as algorithm VarElimination performs less consistency operations than a $\diamond$-consistency enforcing algorithm. We address the "only if" part first. If VarElimination terminates and returns (True,_), for each $v_k \in V$ with $0 < k < n$ we will have that $\mathcal{N}'[v_i, v_j] \subseteq \mathcal{N}'[v_i, v_k] \diamond \mathcal{N}'[v_k, v_j]$ for all $v_i, v_j \in V$ with $i, j < k$, and for all $v_i, v_j \in V$ we will have that $\mathcal{N}'[v_i, v_j] \neq \emptyset$. Thus, by definition of $\overleftarrow{\diamond}$-consistency, we have that $\mathcal{N}'$ is $\overleftarrow{\diamond}$-consistent, or, more precisely, that there exists a QCN $\overleftarrow{\diamond}(\mathcal{N})$ such that $\mathcal{N}' = \overleftarrow{\diamond}(\mathcal{N})$, viz., the $\overleftarrow{\diamond}$-consistent QCN of $\mathcal{N}$ with respect to the considered ordering of the variables of V. Further, as $\mathcal{N}'[v_i, v_j] \neq \emptyset$ for all $v_i, v_j \in V$, we have that $\mathcal{N}'$ is not trivially inconsistent. By Proposition 4 we have that $\mathcal{N}'$ is satisfiable. As $\mathcal{N}'$ is equivalent to $\mathcal{N}$ (because $\diamond(\mathcal{N}) \subseteq \mathcal{N}' \subseteq \mathcal{N}$ and $\diamond(\mathcal{N})$ is equivalent to $\mathcal{N}$), we have that $\mathcal{N}$ is satisfiable. Next, we address the "if" part. If $\mathcal{N}$ is satisfiable, then it yields a unique $\diamond$-consistent and not trivially inconsistent QCN of $\mathcal{N}$, viz., $\diamond(\mathcal{N})$. Let us assume that given $\mathcal{N}$ and some bijection $\alpha : V \rightarrow \{0, 1, \dots, n-1\}$, algorithm VarElimination terminates and returns (False,_). This would mean that the QCN $\mathcal{N}'$, i.e., the refined QCN of $\mathcal{N}$ that would result from the application of algorithm VarElimination on $\mathcal{N}$ until its termination with the output of (False,_), would be trivially inconsistent (lines 14–15). As we already established that $\diamond(\mathcal{N}) \subseteq \mathcal{N}'$, we would have that $\diamond(\mathcal{N})$ is trivially inconsistent. This is a contradiction, as $\diamond(\mathcal{N})$ cannot be trivially inconsistent when $\mathcal{N}$ is satisfiable ($\diamond(\mathcal{N})$ is equivalent to $\mathcal{N}$). We can conclude that algorithm VarElimination terminates and returns (True,_) if and only if $\mathcal{N}$ is satisfiable. □

Due to Propositions 1 and 2, and Theorem 4, it is clear that algorithm VarElimination is sound and complete for deciding the satisfiability of a QCN defined over a distributive subclass of relations of Point Algebra, Interval Algebra, Cardinal Direction Calculus, Block Algebra, or RCC-8.

Given a QCN $\mathcal{N}$, we note that algorithm VarElimination may consider pairs of variables that do not exist in $E(\mathsf{G}(\mathcal{N}))$ (lines 2–9), which are nevertheless involved in consistency operations with respect to the constraints that are associated with them (lines 10–15). It would be interesting to explore if there exists a condition such that no pair of variables that does not exist in $E(\mathsf{G}(\mathcal{N}))$ needs to be considered, as this would allow us to perform less consistency operations in general. To this end, the bijection α that is given as input to algorithm VarElimination plays an important role, as it defines the ordering in which the variables of V are eliminated. Let us show when the ordering defined by α guarantees that no pair of variables that does not exist in $E(\mathsf{G}(\mathcal{N}))$ will be considered by algorithm VarElimination.

First, we recall some graph theoretic concepts. Given a graph $G = (V, E)$, with $|V| = n$, and a vertex $v \in V$, $N(v)$ denotes the set of neighbors of v in G, i.e., $N(v) = \{u \mid \{u, v\} \in E\}$. A vertex $v \in V$ is said to be a *simplicial* vertex of G if the subgraph of G induced by $N(v)$ is complete. Further, let $\alpha : V \rightarrow \{0, 1, \dots, n-1\}$ be a bijection of V onto $\{0, 1, \dots, n-1\}$, and let G_i denote the subgraph of G induced by $V_i = \{\alpha^{-1}(0), \alpha^{-1}(1), \dots, \alpha^{-1}(i)\}$, with $0 \leq i < n$. (Note that $G_{n-1} = G$.) The ordering $(\alpha^{-1}(n-1), \alpha^{-1}(n-2), \dots, \alpha^{-1}(0))$ of the vertices of V is said to be a *perfect elimination ordering* of G, if for every $n > i > 0$, vertex $\alpha^{-1}(i)$ is a simplicial vertex of graph G_i. Then, we have the following theorem:

THEOREM 5 ([13]). *A graph* G *is chordal if and only if it admits a perfect elimination ordering.*

With respect to chordal graphs, algorithm VarElimination constructs a chordal graph as a byproduct. In particular we have the following result:

PROPOSITION 5. *Given a* QCN $\mathcal{N} = (V, C)$*, with* $|V| = n$*, and a bijection* $\alpha : V \to \{0, 1, \ldots, n-1\}$*, we have that if algorithm* VarElimination *terminates and returns* (True,G)*, then* G *is a chordal graph such that* $\mathsf{G}(\mathcal{N}) \subseteq G$.

PROOF. Since algorithm VarElimination terminates and returns (True,G), we can ignore lines 10–15 of its operation. Graph G is initialized to $(V, E = E(\mathsf{G}(\mathcal{N})))$ (line 2), where V is also the set of vertices of $\mathsf{G}(\mathcal{N})$. Let us show that $\mathsf{G}(\mathcal{N}) \subseteq G$ and G is also chordal after the termination of algorithm VarElimination. As defined earlier, G_i is the subgraph of G induced by $\{\alpha^{-1}(0), \alpha^{-1}(1), \ldots, \alpha^{-1}(i)\}$, with $0 \leq i < n$. For each x from $n-1$ to 1, line 5 provides us with the set of neighbors of vertex $\alpha^{-1}(x)$ in G_x, denoted by $N_x(\alpha^{-1}(x))$. Then, in lines 6–9 we add edges to E (if not existing in E and, thus, neither in $E(\mathsf{G}(\mathcal{N}))$) such that every two vertices in $N_x(\alpha^{-1}(x))$ become connected by an edge in G_x. As such, vertex $\alpha^{-1}(x)$ becomes a simplicial vertex of G_x, since the subgraph of G_x induced by $N_x(\alpha^{-1}(x))$ becomes complete. (Note that the check $\alpha(v') < \alpha(v'')$ in line 7 for vertices $v', v'' \in V$ does not cause a problem, as we deal with edges that are not ordered pairs of vertices, but rather doubletons of vertices; this is also apparent from our notation.) After processing the set of vertices $\{\alpha^{-1}(1), \alpha^{-1}(2), \ldots, \alpha^{-1}(n-1)\}$ of V, we will have admitted a perfect elimination ordering $(\alpha^{-1}(n-1), \alpha^{-1}(n-2), \ldots, \alpha^{-1}(0))$ of G through the addition of new edges to the set of edges E when necessary. As such, we have that the edge-augmented graph $G = (V, E)$ of $\mathsf{G}(\mathcal{N})$ is a chordal graph by Theorem 5. We can conclude that if algorithm VarElimination terminates and returns (True,G), then G is a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$. □

As shown in the proof of Proposition 5, given a satisfiable QCN $\mathcal{N} = (V, C)$, with $|V| = n$, and a bijection $\alpha : V \to \{0, 1, \ldots, n-1\}$, algorithm VarElimination treats the ordering $(\alpha^{-1}(n-1), \alpha^{-1}(n-2), \ldots, \alpha^{-1}(0))$ as a perfect elimination ordering of $\mathsf{G}(\mathcal{N})$ and consequently constructs a chordal supergraph G of $\mathsf{G}(\mathcal{N})$. We can assert the following result:

PROPOSITION 6. *Given a* QCN $\mathcal{N} = (V, C)$*, with* $|V| = n$*, and a bijection* $\alpha : V \to \{0, 1, \ldots, n-1\}$*, we have that algorithm* VarElimination *terminates and returns* (_,G)*, with* $G = \mathsf{G}(\mathcal{N})$*, if* $\mathsf{G}(\mathcal{N})$ *is a chordal graph and* $(\alpha^{-1}(n-1), \alpha^{-1}(n-2), \ldots, \alpha^{-1}(0))$ *a perfect elimination ordering of it.*

Proposition 6 ensures that if its specified condition holds, then no pair of variables that does not exist in $E(\mathsf{G}(\mathcal{N}))$ will be considered by algorithm VarElimination. In light of this result, the question arises whether a perfect elimination ordering of a given chordal graph is easily obtainable. In fact, given a chordal graph $G = (V, E)$, with $|V| = n$, we can obtain a perfect elimination ordering of G in $O(|V| + |E|)$ time using the *maximum cardinality search* (MCS) algorithm [35]. In particular, MCS visits the vertices of a graph in an order such that, at any point, a vertex is visited that has the largest number of visited neighbors. Consequently, MCS produces a bijection $\alpha : V \to \{0, 1, \ldots, n-1\}$ such that $(\alpha^{-1}(n-1), \alpha^{-1}(n-2), \ldots, \alpha^{-1}(0))$ is a perfect elimination ordering of G. Given a QCN $\mathcal{N} = (V, C)$, if $\mathsf{G}(\mathcal{N})$ is not chordal, MCS will define an elimination ordering of the variables of $\mathcal{N}$, which, although not perfect, in general will allow less pairs of variables that do not exist in $E(\mathsf{G}(\mathcal{N}))$ to be considered by algorithm VarElimination, than a randomly chosen elimination ordering. An alternative would be to use some special greedy heuristic instead of the MCS algorithm to obtain an elimination ordering, the simplest and fastest of which being the *approximate minimum degree* heuristic [15]. This heuristic has a runtime of $O(|V||E|)$ for a given graph $G = (V, E)$ [15]. Thus, its use can still be an overkill for large QCNs, which are of our particular interest in the evaluation that takes place in Section 4. Another choice is the *minimum fill-in* heuristic with a runtime of $O(|V|^3)$ [18, 30], which again makes its use prohibitive for large QCNs.

3.2 Extracting a Solution

In the previous section we showed how we can efficiently decide the satisfiability of a given QCN defined over a distributive subclass of relations. Here, we show how a solution of that QCN can actually be extracted. The contribution involves performing a generic backtrack-free procedure for refining a $\overleftarrow{\diamond}$-consistent and not trivially inconsistent QCN $\mathcal{N}$ defined over a distributive subclass of relations to a scenario of $\mathcal{N}$, i.e., an atomic satisfiable sub-QCN of $\mathcal{N}$, and then using some known method from the literature for valuating the variables of $\mathcal{N}$ in order to satisfy it; such methods in general dictate that a given QCN should be atomic and satisfiable in order for a valuation of its variables to take place (e.g., as it is required in the case of RCC-8 in [5]).

PROPOSITION 7. *Let* $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ *be a* $\overleftarrow{\diamond}$*-consistent and not trivially inconsistent* QCN *defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2. Then,* $\mathcal{N}$ *can be refined to a scenario* $\mathcal{S}$ *of* $\mathcal{N}$ *as follows. For each* k *from* 1 *to* $n-1$*, and for each* $i \in \{0, \ldots, k-1\}$*, do:*

- $C(v_k, v_i) \leftarrow \bigcap_{j=0}^{k-1} C(v_k, v_j) \diamond C(v_j, v_i)$;
- $C(v_k, v_i) \leftarrow \{b\}$ *for some* $b \in C(v_k, v_i)$;
- $C(v_i, v_k) \leftarrow (C(v_k, v_i))^{-1}$.

PROOF. Let V_l with $0 \leq l < n$ denote the set of variables $\{v_0, v_1, \ldots, v_l\}$, i.e., the set of variables of V from v_0 up to (and including) v_l. By Proposition 4 we have that $\mathcal{N}$ is satisfiable. As such, $\mathcal{N}{\downarrow_{V_{l'}}}$ is satisfiable for some $0 \leq l' < n-1$ (as with any other restriction of $\mathcal{N}$ to a proper subset of its variables). Since $\mathcal{N}{\downarrow_{V_{l'}}}$ is satisfiable, we can refine it to a scenario $\mathcal{S}{\downarrow_{V_{l'}}}$ of $\mathcal{N}{\downarrow_{V_{l'}}}$, and have that $\mathcal{N}[v_j, v_{j'}] = \mathcal{S}{\downarrow_{V_{l'}}}[v_j, v_{j'}]$ for every $j, j' \in \{0, \ldots, l'\}$. We will show that we can extend that scenario to a scenario of $\mathcal{N}{\downarrow_{V_{l'+1}}}$ with our proposed construction that is specified by the three successive operations listed in our proposition. It is clear that $\mathcal{N}{\downarrow_{V_{l'+1}}}$ is $\overleftarrow{\diamond}$-consistent with respect to the variable ordering $\{v_0, v_1, \ldots, v_{l'+1}\}$ and not trivially inconsistent. Let $\mathcal{N}_i$ with $i \in \{0, \ldots, l'\}$ denote $\bigcap_{j=0}^{l'} \mathcal{N}[v_{l'+1}, v_j] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_j, v_i]$. First, we need to show that $\mathcal{N}_i$ is not empty. By Theorem 2 it suffices to show that $\mathcal{N}[v_{l'+1}, v_j] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_j, v_i] \cap \mathcal{N}[v_{l'+1}, v_{j'}] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_{j'}, v_i] \neq \emptyset$ for all $j, j' \in \{0, \ldots, l'\}$. As $\mathcal{N}$ is defined on a qualitative constraint language that satisfies property 1 (i.e., that is a relation algebra), due to the Peircean law that holds for relation algebras (cf. [11, Chapt. 3]), we have that $\mathcal{N}[v_{l'+1}, v_j] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_j, v_i] \cap \mathcal{N}[v_{l'+1}, v_{j'}] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_{j'}, v_i] \neq \emptyset$ iff $\mathcal{N}[v_{l'+1}, v_{j'}] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_{j'}, v_i] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_i, v_j] \cap \mathcal{N}[v_{l'+1}, v_j] \neq \emptyset$ iff $\mathcal{N}[v_{j'}, v_{l'+1}] \diamond \mathcal{N}[v_{l'+1}, v_j] \cap \mathcal{S}{\downarrow_{V_{l'}}}[v_{j'}, v_i] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_i, v_j] \neq \emptyset$. As $\mathcal{S}{\downarrow_{V_{l'}}}[v_{j'}, v_j] \subseteq \mathcal{S}{\downarrow_{V_{l'}}}[v_{j'}, v_i] \diamond \mathcal{S}{\downarrow_{V_{l'}}}[v_i, v_j]$ and $\mathcal{S}{\downarrow_{V_{l'}}}[v_{j'}, v_j] \subseteq \mathcal{N}[v_{j'}, v_{l'+1}] \diamond \mathcal{N}[v_{l'+1}, v_j]$, we have that $\mathcal{N}_i$ is not empty. As such, there exists a base relation $b \in \mathsf{B}$ such that $b \in \mathcal{N}_i$. Let us assign the base relation b to $\mathcal{N}[v_{l'+1}, v_i]$, and have that $\mathcal{N}[v_{l'+1}, v_i] = \{b\}$ and $\mathcal{N}[v_i, v_{l'+1}] = \{b^{-1}\}$. We will now show that the refined $\mathcal{N}{\downarrow}_{V_{l'+1}}$ remains $\overleftarrow{\diamond}$-consistent with respect to the variable ordering $\{v_0,v_1,\ldots,v_{l'+1}\}$ and not trivially inconsistent. As $b \neq \emptyset$, $\mathcal{N}{\downarrow}_{V_{l'+1}}$ remains not trivially inconsistent. To show that $\mathcal{N}{\downarrow}_{V_{l'+1}}$ also remains $\overleftarrow{\diamond}$-consistent with respect to the considered variable ordering, we need to show that $\mathcal{N}[v_j,v_i] \subseteq \mathcal{N}[v_j,v_{l'+1}] \diamond \mathcal{N}[v_{l'+1},v_i]$, i.e., $\mathcal{N}[v_j,v_i] \subseteq \mathcal{N}[v_j,v_{l'+1}] \diamond \{b\}$, for every $j \in \{0,\ldots,l'\}$. As $b \in \mathcal{N}_i$, we have that $\{b\} \subseteq \mathcal{N}[v_{l'+1},v_j] \diamond \mathcal{N}[v_j,v_i]$. (Note that $\mathcal{N}[v_j,v_i] = \mathcal{S}{\downarrow}_{V_{l'}}[v_j,v_i]$.) Thus, $\{b\} \cap \mathcal{N}[v_{l'+1},v_j] \diamond \mathcal{N}[v_j,v_i] \neq \emptyset$. Then, due to the Peircean law, we have that $\mathcal{N}[v_j,v_{l'+1}] \diamond \{b\} \cap \mathcal{N}[v_j,v_i] \neq \emptyset$. As $\mathcal{N}[v_j,v_i]$ is a singleton relation, we can only have that $\mathcal{N}[v_j,v_i] \subseteq \mathcal{N}[v_j,v_{l'+1}] \diamond \{b\}$. Up to this point, we have shown that given any scenario $\mathcal{S}{\downarrow}_{V_{l'}}$ of $\mathcal{N}{\downarrow}_{V_{l'}}$ for some $0 \leq l' < n-1$, and as long as $\mathcal{N}{\downarrow}_{V_{l'+1}}$ remains $\overleftarrow{\diamond}$-consistent and not trivially inconsistent, then, for any $i \in \{0,\ldots,l'\}$, every base relation $b \in \mathcal{N}_i$ is a feasible base relation of $\mathcal{N}{\downarrow}_{V_{l'+1}}$ with regard to constraint $\mathcal{N}[v_{l'+1},v_i]$. Further, we showed that after assigning to $\mathcal{N}[v_{l'+1},v_i]$ a base relation of $\mathcal{N}_i$, the refined $\mathcal{N}{\downarrow}_{V_{l'+1}}$ will remain $\overleftarrow{\diamond}$-consistent and not trivially inconsistent. This property allows us to extend the scenario $\mathcal{S}{\downarrow}_{V_{l'}}$ to a scenario of $\mathcal{N}{\downarrow}_{V_{l'+1}}$ by (i) choosing some not already considered constraint $\mathcal{N}[v_{l'+1},v_i]$ of $\mathcal{N}{\downarrow}_{V_{l'+1}}$, for some $i \in \{0,\ldots,l'\}$, and calculating $\mathcal{N}_i$ with respect to the refined $\mathcal{N}{\downarrow}_{V_{l'+1}}$ that has resulted from the assignment of a base relation to each one of its already considered constraints (if any), (ii) soundly assigning any base relation of $\mathcal{N}_i$ to $\mathcal{N}[v_{l'+1},v_i]$ and, thus, further refining $\mathcal{N}{\downarrow}_{V_{l'+1}}$, and (iii) repeating the procedure for all not already considered constraints of $\mathcal{N}{\downarrow}_{V_{l'+1}}$ by going back to (i). It is important to stress that in each loop of the aforementioned procedure, a relation $\mathcal{N}_i$, for some $i \in \{0,\ldots,l'\}$, is by definition calculated with respect to the refined $\mathcal{N}{\downarrow}_{V_{l'+1}}$ that has resulted from the assignment of a base relation to each one of its already considered constraints; in technical terms, in each loop, $\mathcal{N}{\downarrow}_{V_{l'+1}}$ is mutated (i.e., *refined* in our context) in place, and a calculation of some $\mathcal{N}_i$ takes into account the latest refinement of $\mathcal{N}{\downarrow}_{V_{l'+1}}$. The procedure terminates when all unassigned constraints of $\mathcal{N}{\downarrow}_{V_{l'+1}}$ have been considered, and $\mathcal{N}{\downarrow}_{V_{l'+1}}$ has become an atomic and satisfiable (by Proposition 4) QCN. Using this procedure, and with respect to k and i as they appear in our proposition, we can refine $\mathcal{N}$ to a scenario $\mathcal{S}$ of $\mathcal{N}$, by strictly following the ordering $(1,\ldots,n-1)$ for k, and any permutation of the ordering $(0,\ldots,k-1)$ for i. In particular, we start by assigning to $\mathcal{N}[v_0,v_1]$ any base relation b such that $b \in \mathcal{N}[v_0,v_1]$, as $\mathcal{N}{\downarrow}_{V_1}$ being a QCN of two variables is minimal and weakly globally consistent, and acquire the rest of the scenario up to $\mathcal{N}$ incrementally, at which point $\mathcal{N}$ will have been refined to an atomic satisfiable QCN. □

Next, we present an algorithm for extracting a solution of a given satisfiable QCN $\mathcal{N} = (V,C)$ defined over a distributive subclass of relations. The algorithm is given in Algorithm 2 and is called ExtractSolution. First, algorithm ExtractSolution uses algorithm VarElimination to make $\mathcal{N}$ $\overleftarrow{\diamond}$-consistent, then, it applies the procedure specified in Proposition 7 to refine $\mathcal{N}$ to a scenario of $\mathcal{N}$, and, finally, it uses some known method from the literature to extract a solution of that scenario. Algorithm ExtractSolution has a runtime of $\max(\{O(|V|^3),\omega\})$, where ω is the runtime of the method that valuates the variables of $\mathcal{N}$ in order to satisfy it (line 12), and $O(|V|^3)$ includes the runtime of algorithm VarElimination (line 2), and the time needed to iterate $O(|V|)$ variables (lines 5–6) and realize $O(|V|^2)$ constant operations for the variable v at hand in each iteration (lines 7–11). Once a scenario of $\mathcal{N}$ is obtained, a solution of $\mathcal{N}$ can be constructed using some *canonical model*, i.e., a structure that allows to model any satisfiable sentence of the qualitative constraint language at hand. In light of this information, and with regard to runtime ω, of particular interest is the case of RCC-8, for which several canonical models alongside a valuation method have been defined in order to obtain interesting solutions. For instance, the literature offers a domain of regular closed subsets of the set of real numbers with a valuation method that runs in $O(|V|^3)$ time [5], a domain of countably many homeomorphic disjoint components forming a topological space with a valuation method that, again, runs in $O(|V|^3)$ time [21,23], and the usual domain of regions corresponding to regular closed subsets of some topological space that do not have to be internally connected and do not have a particular dimension with a valuation method that runs in $O(|V|^4)$ time [28]. The canonical model of Renz [28], allows for a simple representation of regions with respect to a set of RCC-8 constraints, and, further, enables one to generate realizations in any dimension $d \geq 1$.

Algorithm 2: ExtractSolution($\mathcal{N}$,α)

```
   in/out : A QCN N = (V, C) over n variables.
   in     : A bijection α of V onto {0, 1, ..., n − 1}.
   out    : True or False, and a mapping μ.
1  begin
2      (decision, _) ← VarElimination(N, α);
3      if decision = False then
4          return (False, Null);
5      for x from 1 to n − 1 do
6          v ← α^{-1}(x);
7          foreach v' ∈ V | α(v') < α(v) do
8              adj ← {v'' ∈ V | α(v'') < α(v)};
9              C(v, v') ← ⋂_{v''∈adj} C(v, v'') ⋄ C(v'', v');
10             C(v, v') ← {b} for some b ∈ C(v, v');
11             C(v', v) ← (C(v, v'))^{-1};
12     μ ← (f : V → D) s.t. μ satisfies N;
13     return (True, μ);
```

THEOREM 6. *Given a* QCN $\mathcal{N} = (V,C)$, *with* $|V| = n$, *defined over a distributive subclass of relations of Point Algebra, Interval Algebra, Cardinal Direction Calculus, Block Algebra, or* RCC-*8, and a bijection* $\alpha : V \rightarrow \{0,1,\ldots,n-1\}$, *we have that algorithm* ExtractSolution *terminates and returns* (True,μ)*, only if* μ *is a solution of* $\mathcal{N}$.

PROOF. If ExtractSolution terminates and returns (True,μ), we will show that the mapping μ is created in such a way as to correspond to a solution of $\mathcal{N}$. By Theorem 4 we have that algorithm VarElimination is sound and complete for deciding the satisfiability of $\mathcal{N}$, so lines 2–4 ensure that $\mathcal{N}$ is in fact satisfiable. As such, we focus solely on extracting a solution of $\mathcal{N}$. Algorithm VarElimination refines $\mathcal{N}$ in place, rendering it $\overleftarrow{\diamond}$-consistent with respect to the ordering $(\alpha^{-1}(0),\alpha^{-1}(1),\ldots,\alpha^{-1}(n-1))$ of V that is defined by the bijection α (see the proof of Theorem 4). $\mathcal{N}$ is then further refined to a scenario of $\mathcal{N}$ using the procedure specified in Proposition 7. This procedure is implemented exactly in lines 5–11 of the algorithm. A solution μ of $\mathcal{N}$ is then obtained in line 12, using some known method from the literature for valuating the variables of $\mathcal{N}$ in order to satisfy it. In particular, a valuation method for an atomic satisfiable QCN of Block Algebra is presented in [3], valuation methods for atomic satisfiable QCNs of Point Algebra and Interval Algebra are presented in [37], a valuation method for an atomic satisfiable QCN of Cardinal Direction Calculus is presented in [24], and several valuation methods for an atomic satisfiable QCN of RCC-8 are presented in [5, 21, 28]. We can conclude that algorithm ExtractSolution terminates and returns (True,μ), only if μ is a solution of $\mathcal{N}$. □

4. EXPERIMENTATION

We evaluate the performance of our implementation of the $\overleftarrow{\diamond}$-consistency enforcing VarElimination algorithm, against state-of-the-art implementations of $\overset{\diamond}{_G}$-consistency and $\diamond$-consistency enforcing algorithms, for checking the satisfiability of a given QCN defined over a distributive subclass of relations. Algorithm VarElimination is implemented under the hood of a novel reasoner called Pyrrhus. A state-of-the-art $\overset{\diamond}{_G}$-consistency enforcing algorithm implementation is provided by reasoner Sarissa and a state-of-the-art $\diamond$-consistency enforcing algorithm implementation is provided by reasoner Phalanx, both of which are in-house implementations and are presented in [31]. Pyrrhus, like any of our other aforementioned reasoners, is a generic and open source qualitative constraint-based spatial and temporal reasoner written in pure Python[3] and can be found online in the following address: http://www.cril.fr/~sioutis/work.php.

Technical Specifications.

The experimentation was carried out on a computer with an Intel Core i7-2820QM processor with a 2.30 GHz frequency per CPU core, 8 GB of RAM, and the Trusty Tahr x86_64 OS (Ubuntu Linux). Pyrrhus, Sarissa, and Phalanx were run with PyPy 2.2.1[4], which implements Python 2.7. Only one of the CPU cores was used.

Dataset and Measures.

We considered random RCC-8 networks generated by the BA(n, m) model [4], the use of which in qualitative constraint-based reasoning is well motivated in [31], and real-world RCC-8 datasets that have been recently used in [22, 33]

In particular, we used the BA(n, m) model to create random scale-free graphs of order n with a preferential attachment value m; each such graph was then treated as the constraint graph of a given QCN of RCC-8, by labeling the constraints of the QCN corresponding to edges of the graph with relations from the maximal distributive subclass $\mathcal{D}_8^{64}$ of RCC-8 [22], and the rest of the constraints of the QCN strictly with the universal relation, viz., B. We considered 10 satisfiable and 10 unsatisfiable RCC-8 network instances of BA(n, m) for each order $1000 \leq n \leq 10000$ of their constraint graphs with a 1000-vertex step and a preferential attachment value of $m = 2$. Both satisfiable and unsatisfiable network instances were randomly filtered out of a large number of 1 000 network instances to ensure validity of the results. Regarding real-world RCC-8 datasets, we employed the ones recently used in [22,33], described as follows (in the description by constraints we mean non-universal relations).

- `nuts`: an RCC-8 network of a nomenclature of territorial units with 2 235/3 176 variables/constraints.[5]
- `adm1`: an RCC-8 network of the administrative geography of Great Britain with 11 762/44 832 variables/constraints [14].
- `gadm1`: an RCC-8 network of the German administrative units with 42 749/159 600 variables/constraints.[5]
- `gadm2`: an RCC-8 network of the world's administrative areas with 276 729/589 573 variables/constraints.[6]
- `adm2`: an RCC-8 network of the administrative geography of Greece with 1 732 999/5 236 270 variables/constraints.[5]
- `footprints`: an RCC-8 network of geographic "footprints" in the Southampton area of the UK with 3 470/446 847 variables/constraints [22].
- `statareas`: an RCC-8 network of statistical areas in Tasmania with 1 562/10 101 variables/constraints [22].

The aforementioned datasets are satisfiable. Further, the relations of each dataset are contained in one of the maximal distributive subclasses $\mathcal{D}_8^{41}$ and $\mathcal{D}_8^{64}$ of RCC-8 [22].

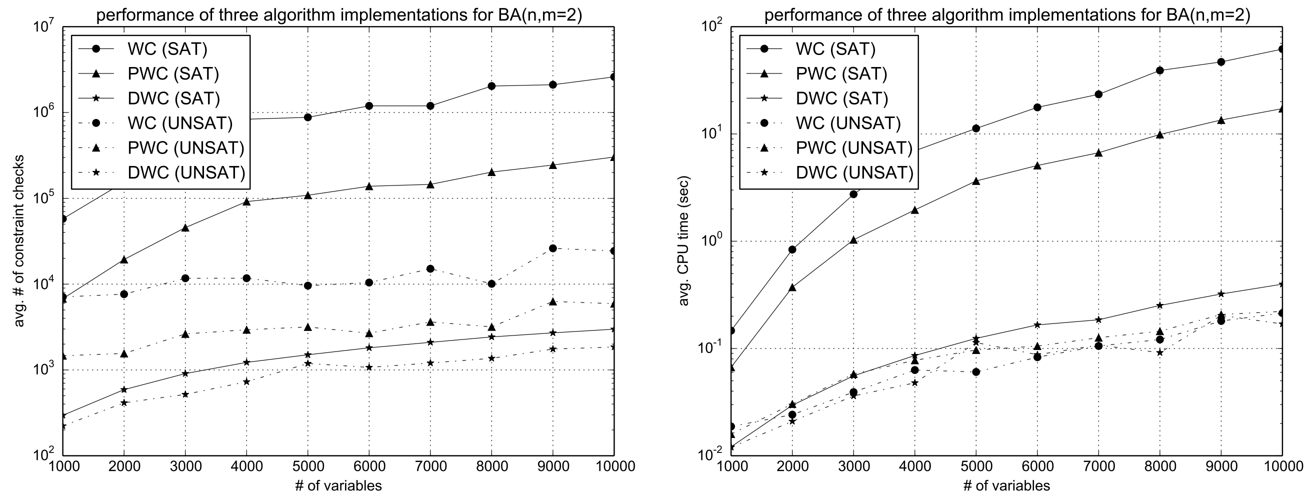
Our experimentation involves two measures, which we describe as follows. The first measure considers the number of *constraint checks* performed by a local consistency enforcing algorithm implementation. Given a QCN $\mathcal{N} = (V, C)$ and $v_i, v_k, v_j \in V$, a constraint check is performed when we compute relation $r = C(v_i, v_j) \cap (C(v_i, v_k) \diamond C(v_k, v_j))$ and check if $r \subset C(v_i, v_j)$, so that we can propagate its constrainedness. (Weak compositions that yield relation B are disregarded.) The second measure concerns the CPU time and is strongly correlated with the first one, as the runtime of local consistency enforcing algorithm implementations relies heavily on the number of constraint checks performed.

Results.

In what follows, WC (for $\diamond$-consistency or closure under weak composition) will denote the $\diamond$-consistency enforcing algorithm implementation of Phalanx, PWC (for $\overset{\diamond}{_G}$-consistency or partial closure under weak composition) will denote the $\overset{\diamond}{_G}$-consistency enforcing algorithm implementation of Sarissa, and DWC (for $\overleftarrow{\diamond}$-consistency or directional closure under weak composition) will denote the $\overleftarrow{\diamond}$-consistency enforcing algorithm implementation of Pyrrhus, viz., the implementation of algorithm VarElimination. As a final note, the maximum cardinality search algorithm was used to obtain a variable elimination ordering for DWC, and a triangulation of the constraint graph of a given QCN for PWC as described in [31] (to be able to make sound use of Proposition 3).

Regarding random scale-free RCC-8 networks, the experimental results are shown in Figure 3. DWC performs significantly less constraint checks than PWC and WC for both satisfiable and unsatisfiable network instances, as shown in Figure 3a. In particular, across all network instances of different size, DWC performs on average 98.2% and 99.8% less constraint checks than PWC and WC respectively for satisfiable network instances, and 70.1% and 92.2% less constraint checks than PWC and WC respectively for unsatisfiable network instances. This also reflects on the CPU time, as shown in Figure 3b. In particular, across all network instances of different size, DWC is on average 94.7% and 98.0% faster than PWC and WC respectively for satisfiable network in-

[3] https://www.python.org/
[4] http://pypy.org/
[5] Retrieved from: http://www.linkedopendata.gr/
[6] http://gadm.geovocab.org/



(a) Comparison on # of constraint checks

(b) Comparison on CPU time

Figure 3: Performance comparison for random scale-free RCC-8 networks

stances, and 20.8% and 1.8% faster than PWC and WC respectively for unsatisfiable network instances; we note that for unsatisfiable network instances all approaches are in a virtual tie, as they unveil the inconsistencies in centiseconds and any difference in performance is thus marginal.

Table 1: Evaluation with real-world RCC-8 datasets

network	WC	PWC	DWC
nuts	$\frac{0.12s}{9\,632}$	$\frac{0.09s}{5\,808}$	$\frac{0.08s}{1\,180}$
adm1	$\frac{13\,783.39s}{1\,287\,288\,879}$	$\frac{83.36s}{18\,498\,096}$	$\frac{0.31s}{92\,266}$
gadm1	$\frac{25\,587.76s}{1\,927\,158\,080}$	$\frac{105.07s}{34\,140\,998}$	$\frac{0.82s}{339\,611}$
gadm2	$\frac{6.72s}{1\,891\,032}$	$\frac{1.61s}{1\,885\,100}$	$\frac{0.56s}{483\,377}$
adm2	∞	$\frac{399.60s}{118\,799\,994}$	$\frac{2.25s}{5\,471\,745}$
footprints	$\frac{345.69s}{689\,888\,647}$	$\frac{167.73s}{119\,117\,222}$	$\frac{2.99s}{14\,251\,630}$
statareas	$\frac{29.48s}{29\,009\,108}$	$\frac{0.25s}{485\,082}$	$\frac{0.06s}{32\,408}$

Regarding real-world RCC-8 datasets, the experimental results are summarized in Table 1, where a fraction $\frac{x}{y}$ denotes that an approach required x seconds of CPU time and performed y constraint checks to decide the satisfiability of a given network instance. Symbol ∞ denotes that an implementation hit the memory limit. We also note that we decomposed the large network instances using the simple decomposition approach proposed in [34]; this pre-processing step took negligible time to realize. Again, we can see that DWC significantly outperforms PWC and WC with regard to both the CPU time required and the number of constraint checks performed for deciding the satisfiability of a network instance. It should suffice to mention that for the largest of the network instances, viz., adm2, DWC decides its satisfiability in 2.25 sec, when PWC requires 399.60 sec for the same task, and WC does not even complete that task as it hits the memory limit after several hours of reasoning. The same trend holds for the number of constraint checks performed by the different algorithm implementations.

5. CONCLUSION AND FUTURE WORK

We introduced, studied, and evaluated a novel algorithm in the context of qualitative constraint-based spatial and temporal reasoning, that is based on the idea of variable elimination, a simple and general exact inference approach in probabilistic graphical models. Given a qualitative constraint network $\mathcal{N}$, our algorithm enforces a particular directional local consistency on $\mathcal{N}$, which we denote by $\overleftarrow{\diamond}$-consistency. We focused on *distributive* subclasses of relations, i.e., sets of relations closed under converse, intersection, and weak composition and for which weak composition *distributes* over non-empty intersections for all of their relations. We demonstrated that enforcing $\overleftarrow{\diamond}$-consistency on a given qualitative constraint network defined over a distributive subclass of relations allows us to decide its satisfiability. The experimentation that we have conducted with random and real-world qualitative constraint networks defined over a distributive subclass of relations of the Region Connection Calculus, shows that our approach exhibits unparalleled performance against competing state-of-the-art approaches for checking the satisfiability of such constraint networks.

Future work consists of exploring whether $\overleftarrow{\diamond}$-consistency can be efficiently used as the backbone of a backtracking algorithm for checking the satisfiability of arbitrary qualitative constraint networks, i.e., networks defined over any of the relations of a qualitative constraint language. Our experimentation suggests that we should be able to have better performance in general, as any such backtracking algorithm defined in the literature largely utilizes its core local consistency enforcing algorithm. We would also like to explore the implication of $\overleftarrow{\diamond}$-consistency in the *minimality problem* [2] and the *redundancy problem* [22, 33]; these problems exhibit functions that build on the local consistency enforcing algorithms used for checking the satisfiability of a given qualitative constraint network.

Acknowledgment.

The work was funded by a PhD grant from Université d'Artois and region Nord–Pas-de-Calais. We would like to thank David A. Randell, Katia Papakonstantinopoulou, and Panagiotis Liakos for proof-reading this presentation.

6. REFERENCES

[1] Allen, J. F. Maintaining Knowledge about Temporal Intervals. *Commun. ACM 26* (1983), 832–843.

[2] Amaneddine, N., Condotta, J.-F., and Sioutis, M. Efficient Approach to Solve the Minimal Labeling Problem of Temporal and Spatial Qualitative Constraints. In *IJCAI* (2013).

[3] Balbiani, P., Condotta, J.-F., and del Cerro, L. F. Tractability Results in the Block Algebra. *J. Log. Comput. 12* (2002), 885–909.

[4] Barabasi, A.-L., and Albert, R. Emergence of scaling in random networks. *Science 286* (1999), 509–512.

[5] Challita, K. A semi-dynamical approach for solving qualitative spatial constraint satisfaction problems. *Theor. Comput. Sci. 440-441* (2012), 29–38.

[6] Chmeiss, A., and Condotta, J.-F. Consistency of Triangulated Temporal Qualitative Constraint Networks. In *ICTAI* (2011).

[7] Clocksin, W., and Mellish, C. *Programming in Prolog*. Springer Berlin Heidelberg, 2003.

[8] Danzer, L., Grünbaum, B., and Klee, V. Helly's Theorem and Its Relatives. In *Convexity* (1963), vol. 7 of *Proceedings of Symposia in Pure Mathematics*, American Mathematical Society.

[9] Deville, Y., Barette, O., and van Hentenryck, P. Constraint Satisfaction over Connected Row Convex Constraints. *Artif. Intell. 109* (1999), 243–271.

[10] Diestel, R. *Graph Theory, 4th Edition*, vol. 173 of *Graduate Texts in Mathematics*. Springer, 2012.

[11] Dylla, F., Mossakowski, T., Schneider, T., and Wolter, D. Algebraic Properties of Qualitative Spatio-Temporal Calculi. In *COSIT* (2013).

[12] Frank, A. U. Qualitative Spatial Reasoning with Cardinal Directions. In *ÖGAI* (1991).

[13] Fulkerson, D. R., and Gross, O. A. Incidence matrices and interval graphs. *Pacific J. Math. 15* (1965), 835–855.

[14] Goodwin, J., Dolbear, C., and Hart, G. Geographical Linked Data: The Administrative Geography of Great Britain on the Semantic Web. *T. GIS 12*, Issue Supplement s1 (2008), 19–30.

[15] Heggernes, P., Eisenstat, S. C., Kumfert, G., and Pothen, A. The computational complexity of the minimum degree algorithm. Tech. rep., ICASE, NASA Langley Research Center, 2001.

[16] Hogge, J. C. TPLAN: a temporal interval-based planner with novel extensions. Tech. rep., University of Illinois at Urbana-Champaign. Department of Computer Science UIUCDCS-R-87-1367., 1987.

[17] Huang, J. Compactness and its implications for qualitative spatial and temporal reasoning. In *KR* (2012).

[18] Jégou, P., and Terrioux, C. Tree-Decompositions with Connected Clusters for Solving Constraint Networks. In *CP* (2014).

[19] Ladkin, P. B., and Maddux, R. D. On binary constraint problems. *J. ACM 41* (1994), 435–469.

[20] Ladkin, P. B., and Reinefeld, A. Fast Algebraic Methods for Interval Constraint Problems. *Ann. Math. Artif. Intell. 19* (1997), 383–411.

[21] Li, S. On Topological Consistency and Realization. *Constraints 11* (2006), 31–51.

[22] Li, S., Long, Z., Liu, W., Duckham, M., and Both, A. On redundant topological constraints. *Artif. Intell. 225* (2015), 51–76.

[23] Li, S., and Wang, H. RCC8 binary constraint network can be consistently extended. *Artif. Intell. 170* (2006), 1–18.

[24] Ligozat, G. Reasoning about cardinal directions. *J. Vis. Lang. Comput. 9* (1998), 23–44.

[25] Long, Z., and Li, S. On Distributive Subalgebras of Qualitative Spatial and Temporal Calculi. In *COSIT* (2015).

[26] Lutz, C., and Milicic, M. A Tableau Algorithm for DLs with Concrete Domains and GCIs. *J. Autom. Reasoning 38* (2007), 227–259.

[27] Randell, D. A., Cui, Z., and Cohn, A. A Spatial Logic Based on Regions & Connection. In *KR* (1992).

[28] Renz, J. A Canonical Model of the Region Connection Calculus. *J. Appl. Non-Classical Logics 12* (2002), 469–494.

[29] Renz, J., and Ligozat, G. Weak Composition for Qualitative Spatial and Temporal Reasoning. In *CP* (2005).

[30] Rose, D. J. A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. In *Graph Theory and Computing*. Academic Press, 1972, pp. 183–217.

[31] Sioutis, M., Condotta, J.-F., and Koubarakis, M. An Efficient Approach for Tackling Large Real World Qualitative Spatial Networks. *Int. J. Artif. Intell. Tools* (2015). In press.

[32] Sioutis, M., and Koubarakis, M. Consistency of Chordal RCC-8 Networks. In *ICTAI* (2012).

[33] Sioutis, M., Li, S., and Condotta, J.-F. Efficiently Characterizing Non-Redundant Constraints in Large Real World Qualitative Spatial Networks. In *IJCAI* (2015).

[34] Sioutis, M., Salhi, Y., and Condotta, J.-F. A Simple Decomposition Scheme For Large Real World Qualitative Constraint Networks. In *FLAIRS* (2015).

[35] Tarjan, R. E., and Yannakakis, M. Simple Linear-Time Algorithms to Test Chordality of Graphs, Test Acyclicity of Hypergraphs, and Selectively Reduce Acyclic Hypergraphs. *SIAM J. Comput. 13* (1984), 566–579.

[36] Tarski, A. On the calculus of relations. *J. Symb. Log. 6* (1941), 73–89.

[37] van Beek, P. Reasoning About Qualitative Temporal Information. *Artif. Intell. 58* (1992), 297–326.

[38] van Beek, P., and Manchak, D. W. The design and experimental analysis of algorithms for temporal reasoning. *J. Artif. Intell. Res. 4* (1996), 1–18.

[39] Vilain, M. B., and Kautz, H. A. Constraint Propagation Algorithms for Temporal Reasoning. In *AAAI* (1986).

[40] Zhang, N., and Poole, D. A simple approach to Bayesian network computations. In *AI* (1994).

[41] Zhang, Y., and Marisetti, S. Solving connected row convex constraints by variable elimination. *Artif. Intell. 173* (2009), 1204–1219.