International Journal on Artificial Intelligence Tools


# Leveraging Variable Elimination for Efficiently Reasoning about Qualitative Constraints

Michael Sioutis
*Örebro University, MPI@AASS, Örebro, Sweden*
michael.sioutis@oru.se

Zhiguo Long
*Southwest Jiaotong University, Chengdu, China*
zhiguolong@swjtu.edu.cn

Sanjiang Li
*University of Technology Sydney, QSI, Sydney, Australia*
sanjiang.li@uts.edu.au

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 utilizes a particular directional local consistency, which we denote by $\overleftarrow{\diamond_G}$-consistency, in order to efficiently decide the satisfiability of $\mathcal{N}$. 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 $\overleftarrow{\diamond_G}$-consistency in a given qualitative constraint network defined over a distributive subclass of relations allows us to decide its satisfiability, and obtain similar useful results for the problems of minimal labelling and redundancy. Further, we present a generic method that allows extracting a scenario from a satisfiable network, i.e., an atomic satisfiable subnetwork of that network, in a very simple and effective manner. 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 and the Interval Algebra, shows that our approach exhibits unparalleled performance against state-of-the-art approaches for checking the satisfiability of such constraint networks.

## 1. Introduction

Qualitative Spatial and Temporal Reasoning (QSTR) is a major field of study in Artificial Intelligence, and in particular in Knowledge Representation & Reasoning. This field has received a lot of attention over the past decades, as it extends to a plethora of areas and domains that include ambient intelligence, dynamic GIS, cognitive robotics, spatio-temporal design, and qualitative model generation from video [5, 20, 22, 26, 50, 51, 61, 69, 75]. QSTR abstracts from numerical quantities of space and time by using qualitative descriptions instead (e.g., *precedes*, *contains*,

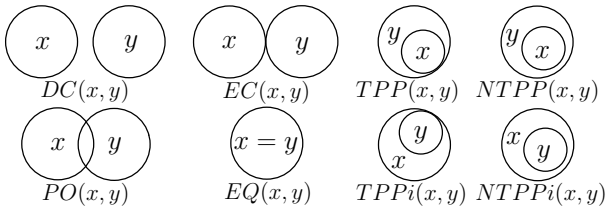


Figure 1. The base relations of RCC-8.

*is left of*), thus providing a concise framework that allows for rather inexpensive reasoning about entities located in space and time and further boosts research and applications in spatial and temporal reasoning.

A fragment 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 [54]. 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 [74], Cardinal Direction Calculus [23,43], Interval Algebra [1], Block Algebra [3], and Cardinal Direction Calculus for extended objects [29, 45, 68]. In general, qualitative constraint languages involve constraints defined over a finite set of binary relations, called *base relations* (or *atoms*) [44]. Thus, qualitative information can be modelled 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 $\mathcal{N}$. Other fundamental reasoning problems include the *minimal labelling* (or *deductive closure*) problem and the *redundancy* problem [59]. The minimal labelling problem is the problem of finding the strongest implied constraints of $\mathcal{N}$, and the redundancy problem is the problem of determining if a given constraint in $\mathcal{N}$ is entailed by the rest of $\mathcal{N}$ (that constraint being called redundant, as its removal does not change the solution set of the QCN). In general, for most qualitative constraint languages the satisfiability checking problem is NP-complete. Further, the redundancy problem, the minimal labelling problem, and the satisfiability checking problem are equivalent under polynomial Turing reductions [27].

In this paper, we focus on the recently studied notion of *distributive* subclasses of relations [39, 47], 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. Distributive subclasses of relations present the nice property that checking the satisfiability of any qualitative constraint network that is defined over such a subclass of relations becomes a tractable task. Intuitively, this is due to the fact that relations from such a subclass are *convex* in Helly's sense [13] and, hence, allow any partial scenario of a $\diamond$-*consistent*[a] [57] QCN to be extended to a complete scenario (cf. [52]). 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 [39, 63, 65]. In particular, and to the best of our knowledge, the current state-of-the-art techniques that have been developed for tackling qualitative constraint networks are mostly geared towards networks whose satisfiability problem is NP-hard in general and/or whose constraint graphs are more or less dense [2, 39, 53, 58, 65]. Furthermore, practical methods for extracting a scenario from a satisfiable qualitative constraint network typically require that network to be already $\diamond$-consistent [12, 55].[b] In addition, such methods are either based on the semantics of the qualitative constraint language at hand and allow for a single scenario to be retrieved efficiently, e.g., one can use a table lookup scheme that defines a quadratic procedure in the number of variables of certain $\diamond$-consistent QCNs of RCC-8 [55], or are generic and built on the incremental functionality of $\diamond$-consistency [25], yielding a procedure that is cubic in the number of variables of certain $\diamond$-consistent QCNs (cf. [15]). Scenario extraction is a very important task, as it allows for introducing preferences among different solutions of a QCN and obtaining crisp qualitative descriptions from a set of qualitative relations. Here, we fill this research gap by tapping into the properties that distributive subclasses of relations have to offer and addressing the issue of efficiently reasoning about qualitative constraints in a generic and algebraic manner.

Specifically, we make the following contributions:

(1) we consider a particular directional local consistency, which we denote by $\overleftarrow{\diamond}_G$-consistency, formally study it and establish its well-behaveness [11], and demonstrate its relation with the satisfiability checking problem, the minimal labelling problem, and the redundancy problem of QCNs defined over a distributive subclass of relations;
(2) we introduce and study a novel algorithm in the context of qualitative

[a]The notion of $\diamond$-consistency (formally defined in Definition 2.3) is very similar to the notion of path consistency in traditional constraint programming; in particular, $\diamond$-consistency can be seen as path consistency with weak composition. These concepts will become clear in Section 2.

[b]To the best of our knowledge, there exists only one significant exception, namely, that of van Beek in [72] for Point Algebra, where a scenario can be extracted from any satisfiable QCN $\mathcal{N} = (V, C)$ in $O(|V|^2)$ time, without having to close that QCN under $\diamond$-consistency. We provide a detailed review of state-of-the-art scenario extraction techniques in Section 4.2.

constraint-based spatial and temporal reasoning that efficiently utilizes $\overleftarrow{\diamond}_G$-consistency for 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 [76];

(3) we define an efficient and generic method for extracting a scenario from a satisfiable QCN defined over a distributive subclass of relations that builds on the aforementioned algorithm and, further, exploits the involved distributivity property to iteratively collect feasible base relations from each of the constraints of the QCN in a backtrack-free manner;

(4) we evaluate our approach with random QCNs of RCC-8 and Interval Algebra and real-world QCNs of RCC-8 defined over a distributive subclass of relations, and show that it exhibits unparalleled performance against 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 consider a particular directional local consistency, which we denote by $\overleftarrow{\diamond}_G$-consistency and formally study it. In Section 4 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 scenario and, in turn, a solution of that QCN can actually be extracted. In Section 5 we evaluate our approach with random QCNs of RCC-8 and Interval Algebra and real-world QCNs of RCC-8, against competing state-of-the-art approaches for checking the satisfiability of such QCNs. In Section 6 we discuss some related work and, finally, in Section 7 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 $\mathsf{B}$ of *jointly exhaustive and pairwise disjoint* relations defined over an infinite domain $\mathsf{D}$, which is called the set of *base relations* [44]. The base relations of a particular qualitative constraint language can be used to represent the definite knowledge between any two of its entities with respect to the considered level of granularity. The set $\mathsf{B}$ contains the identity relation $\mathsf{Id}$, 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, $2^{\mathsf{B}}$ represents the total set of relations. The set $2^{\mathsf{B}}$ is equipped with the usual set-theoretic operations of union and intersection, the converse operation, and the *weak composition* operation denoted by the symbol $\diamond$ [44]. For all $r \in 2^{\mathsf{B}}$, we have that $r^{-1} = \bigcup\{b^{-1} \mid b \in r\}$. The weak composition ($\diamond$) of two base relations $b, b' \in \mathsf{B}$ is defined as the smallest (i.e., strongest) relation $r \in 2^{\mathsf{B}}$ that includes $b \circ b'$, or, formally, $b \diamond b' = \{b'' \in \mathsf{B} \mid b'' \cap (b \circ b') \neq \emptyset\}$, where $b \circ b' = \{(x, y) \in \mathsf{D} \times \mathsf{D} \mid \exists z \in \mathsf{D}$ such that $(x, z) \in b \wedge (z, y) \in b'\}$ is the

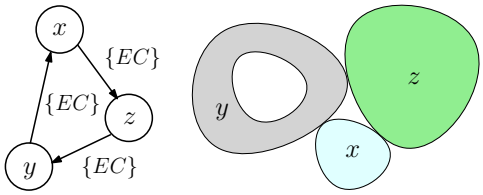


Figure 2. A QCN of RCC-8 along with a solution.

(true) composition of $b$ and $b'$. Further, for all $r, r' \in 2^{\mathsf{B}}$, we have that $r \diamond r' = \bigcup\{b \diamond b' \mid b \in r, b' \in r'\}$. Finally, we assume that $r \diamond \mathsf{B} = \mathsf{B}$ for every $r \in 2^{\mathsf{B}}$.

The Region Connection Calculus (RCC) is a first-order theory for representing and reasoning about mereotopological information [54]. 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* [56]. A subset $X$ of some topological space is regular closed, if $X$ equals the closure of its interior. As mentioned in the introduction, a fragment of the Region Connection Calculus (RCC), denoted by RCC-8, is the dominant qualitative spatial constraint language in Artificial Intelligence for representing and reasoning about qualitative spatial information [54]. The eight base relations of the RCC-8 constraint language are depicted in Figure 1 (using the Euclidean plane). Relation $EQ$ is the identity relation Id of RCC-8.

The weak composition operation $\diamond$, the converse operation $^{-1}$, the union operation $\cup$, the complement operation $^{\mathsf{C}}$, and the total set of relations $2^{\mathsf{B}}$ along with the identity relation Id of a qualitative constraint language, form an algebraic structure $(2^{\mathsf{B}}, \mathsf{Id}, \diamond, ^{-1}, ^{\mathsf{C}}, \cup)$ that can correspond to a *relation algebra* in the sense of Tarski [71]. This topic has been extensively discussed in [21]. In fact, [21] summarizes findings on the relationship between relation algebras and qualitative constraint languages into the following result:

**Proposition 2.1.** *[21] Each one of the qualitative constraint languages of Point Algebra, RCC-8, Cardinal Direction Calculus, Interval Algebra, and Block Algebra is a relation algebra with the algebraic structure* $(2^{\mathsf{B}}, \mathsf{Id}, \diamond, ^{-1}, ^{\mathsf{C}}, \cup)$.

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

Qualitative spatial or temporal information can be modelled as a *qualitative constraint network* (QCN), defined in the following manner:

**Definition 2.1.** 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 respec-

tively;
- $C$ is a mapping $C : V \times V \rightarrow 2^{\mathsf{B}}$ 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 two variables $v, v' \in V$, 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 2.2.** 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 $\mathcal{N}' = (V, C')$ is *equivalent* to $\mathcal{N}$ if and only if $\mathcal{N}'$ 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 $(V, C')$ such that $C'(v, v') \subseteq C(v, v')\ \forall v, v' \in V$;
- $\mathcal{N}$ 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 $\mathcal{N}$;
- a constraint $C(v, v')$, with $v, v' \in V$, is redundant in $\mathcal{N}$ iff $\nexists b \in \mathsf{B} \setminus C(v, v')$ such that $b$ is a feasible base relation of $\mathcal{N}' = (V, C')$, where $C'(v, v') = C'(v', v) = \mathsf{B}$ and $C'(u, u') = C(u, u')\ \forall (u, u)' \in (V \times V) \setminus \{(v, v'), (v', v)\}$;
- 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 $\exists v, v' \in V$ with $C(v, v') = \emptyset$.

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* [57] (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 certainly unsatisfiable.

**Definition 2.3.** A QCN $\mathcal{N} = (V, C)$ is $\diamond$-*consistent* iff $\forall v_i, v_k, v_j \in V$ we have that $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$.

Applying $\diamond$-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 [74]. 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 $\mathcal{N}$. We recall the following result regarding $\diamond$-consistency:

**Proposition 2.2.** *[21] 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.*

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 2.4.** A QCN $\mathcal{N} = (V, C)$ is $\diamond_G$-*consistent* with respect to a graph $G = (V, E)$ 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 $G = (V, E)$, $\diamond_G$-consistency can be applied on $\mathcal{N}$ in $O(\delta|E||\mathsf{B}|)$ time, where $\delta$ is the maximum degree of $G$ [9]. The $\diamond_G$-*consistent* QCN *of* $\mathcal{N}$ is denoted by $\diamond_G(\mathcal{N})$. Clearly, if $G$ is a complete graph, $\diamond_G$-consistency

becomes identical to $\diamond$-consistency. In the general case, it should be clear that $\diamond(\mathcal{N}) \subseteq \overset{\diamond}{}_G(\mathcal{N})$.

**Definition 2.5.** A *subclass* of relations is a subset $\mathcal{A} \subseteq 2^{\mathsf{B}}$ that contains the singleton relations of $2^{\mathsf{B}}$ 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 2.6.** 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 $\mathcal{L}$:

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

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

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

**Theorem 2.1.** [47] Let $\mathcal{N} = (V, C)$ be a $\mathsf{QCN}$ 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 2.7.** A subclass $\mathcal{A} \subseteq 2^{\mathsf{B}}$ is *Helly* [13] 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 [47]:

**Theorem 2.2.** [47] 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 [49], which we tailor to the context of $\diamond$-consistency.

**Definition 2.8.** A subclass $\mathcal{A} \subseteq 2^{\mathsf{B}}$ of a qualitative constraint language has *patchwork* iff for any two $\diamond$-consistent and not trivially inconsistent $\mathsf{QCN}$s $\mathcal{N} = (V, C)$ and $\mathcal{N}' = (V', C')$ defined over $\mathcal{A}$ where $C(u, v) = C'(u, v)$ for every $u, v \in V \cap V'$, we have that the $\mathsf{QCN}$ $\mathcal{N} \cup \mathcal{N}'$ 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 [19]. Given a graph $G = (V, E)$, with $|V| = n$, and a vertex $v \in V$, $N(v)$ denotes the set of neighbours 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 $\alpha : V \rightarrow \{0, 1, \ldots, n-1\}$ be a bijection of $V$ onto $\{0, 1, \ldots, n-1\}$, and let $G_i$ denote the subgraph of $G$ induced by $V_i = \{\alpha^{-1}(0), \alpha^{-1}(1), \ldots, \alpha^{-1}(i)\}$, with $0 \leq i < n$. (Note that $G_{n-1} = G$.) The bijection $\alpha$, and specifically the derived ordering $(\alpha^{-1}(n-1), \alpha^{-1}(n-2), \ldots, \alpha^{-1}(0))$ 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 2.3.** [24] A graph $G$ is chordal if and only if it admits a perfect elimination ordering.

Finally, we have the following result with respect to chordal graphs and $\overset{\diamond}{}_G$-consistency:

**Proposition 2.3.** *[2, 64] Let $\mathcal{N} = (V, C)$ be a* QCN *defined over a subclass of relations that has patchwork for $\diamond$-consistent and not trivially inconsistent* QCN*s defined over that subclass, and $G = (V, E)$ a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$. If $\mathcal{N}$ is $\overset{\diamond}{}_G$-consistent and not trivially inconsistent, then $\mathcal{N}$ is satisfiable.*

Due to Propositions 2.1 and 2.2, and the notion of weak global consistency in Theorem 2.1, 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 2.3 applies to QCNs defined over a distributive subclass of relations of Point Algebra, Interval Algebra, Cardinal Direction Calculus, Block Algebra, or RCC-8.[c]

## 3. $\overset{\leftarrow}{\overset{\diamond}{}_G}$-Consistency: A New Consistency for QCNs

In this section we propose a new local consistency in the context of qualitative constraint-based reasoning that serves as the counterpart of *directional path con-*

[c]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 [33]. 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.

*sistency* in traditional constraint programming [17] or quantitative temporal reasoning [16], and is mainly distinguished by the fact that the involved consistency notions are tailored to handle infinite domains and qualitative relations. We call this novel local consistency *directional partial closure under weak composition* and we denote it by $\overset{\leftarrow}{\underset{G}{\diamond}}$*-consistency*. In particular, $\overset{\leftarrow}{\underset{G}{\diamond}}$-consistency entails consistency for all *ordered* triples of variables of a QCN that correspond to triangles of a given graph $G$. This ordering can be specified by a bijection between the set of the variables of a QCN and a set of integers, and can be chosen randomly, via an algorithm, or through some heuristic search as we will discuss later on in Section 4.

**Definition 3.1.** A QCN $\mathcal{N} = (V, C)$ is $\overset{\leftarrow}{\underset{G}{\diamond}}$*-consistent* with respect to a graph $G = (V, E)$ and an ordering $(\alpha^{-1}(0),\alpha^{-1}(1),\ldots,\alpha^{-1}(n-1))$ defined by a bijection $\alpha : V \rightarrow \{0, 1, \ldots, n-1\}$ iff for all $v_i, v_j, v_k \in V$ such that $\{v_k, v_i\}, \{v_k, v_j\}, \{v_i, v_j\} \in E$ and $\alpha(v_i), \alpha(v_j) < \alpha(v_k)$ we have that $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$.

We note that given a QCN $\mathcal{N} = (V, C)$ and the complete graph $K_V$ on the set of variables $V$, $\overset{\leftarrow}{\underset{K_V}{\diamond}}$-consistency will be denoted by $\overset{\leftarrow}{\diamond}$-consistency.

For simplicity, in what follows, when we state that a QCN is $\overset{\leftarrow}{\underset{G}{\diamond}}$-consistent, the ordering of the variables of that QCN for the use case at hand will be implicitly considered. That ordering, if not mentioned or provided separately, will be specified by the subscripts of the variables that will be used. For instance, a set of variables $\{v_0, v_1, \ldots, v_{n-1}\}$ specifies the ordering $(v_0, v_1, \ldots, v_{n-1})$.

Figure 3 illustrates a QCN $\mathcal{N}$ of Point Algebra that is $\overset{\leftarrow}{\underset{G}{\diamond}}$-consistent w.r.t. the complete graph $G$ on $\{x, y, z\}$ and the ordering $(z, y, x)$, and hence also $\overset{\leftarrow}{\diamond}$-consistent w.r.t. that ordering, but that is not $\overset{\leftarrow}{\diamond}$-consistent w.r.t. the ordering $(x, y, z)$.

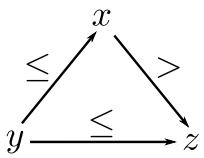


Figure 3. Illustration of $\overset{\leftarrow}{\underset{G}{\diamond}}$-consistency: $\mathcal{N}$ is $\overset{\leftarrow}{\diamond}$-consistent w.r.t. the ordering $(z, y, x)$, while $\mathcal{N}$ is not $\overset{\leftarrow}{\diamond}$-consistent w.r.t. the ordering $(x, y, z)$.

In order to compare the pruning (or inference) capability of different consistency operations that are based on the same graph structure, we introduce a preorder. Let $\overset{\phi}{\underset{G}{}}$ and $\overset{\psi}{\underset{G}{}}$ be two consistencies defined by some operations $\phi$ and $\psi$ respectively and a graph $G$. Then, $\overset{\phi}{\underset{G}{}}$ is *stronger* than $\overset{\psi}{\underset{G}{}}$ iff whenever $\overset{\phi}{\underset{G}{}}$ holds on a QCN $\mathcal{N}$ with respect to a graph $G$, $\overset{\psi}{\underset{G}{}}$ also holds on $\mathcal{N}$ with respect to $G$, and $\overset{\phi}{\underset{G}{}}$ is *strictly stronger* than $\overset{\psi}{\underset{G}{}}$ iff $\overset{\phi}{\underset{G}{}}$ is stronger than $\overset{\psi}{\underset{G}{}}$ and there exists at least one QCN $\mathcal{N}$ and a graph $G$ such that $\overset{\psi}{\underset{G}{}}$ holds on $\mathcal{N}$ with respect to $G$ but $\overset{\phi}{\underset{G}{}}$ does not.

We can obtain the following result:

**Proposition 3.1.** $\overset{\diamond}{\underset{G}{}}$*-consistency is strictly stronger than* $\overset{\leftarrow}{\underset{G}{\diamond}}$*-consistency.*

**Proof.** By definition of ${}^{\diamond}_{G}$-consistency and ${}^{\overleftarrow{\diamond}}_{G}$-consistency, it is easy to see that if a QCN is ${}^{\diamond}_{G}$-consistent then it is also ${}^{\overleftarrow{\diamond}}_{G}$-consistent w.r.t. any ordering. Therefore, we have that ${}^{\diamond}_{G}$-consistency is stronger than ${}^{\overleftarrow{\diamond}}_{G}$-consistency. Now, consider the QCN of Point Algebra shown in Figure 3. It is ${}^{\overleftarrow{\diamond}}$-consistent w.r.t. the ordering $(z, y, x)$. However, $C(y, x) = \{<, =\} \not\subseteq C(y, z) \diamond C(z, x) = \{<, =\} \diamond \{<\} = \{<\}$, that is, the QCN is not $\diamond$-consistent. Thus, we can conclude that ${}^{\diamond}_{G}$-consistency is strictly stronger than ${}^{\overleftarrow{\diamond}}_{G}$-consistency. □

Let ${}^{\phi}_{G}$ be a consistency defined by some operation $\phi$ and a graph $G$. We now recall the definition of *well-behaveness* for such a consistency, which characterizes certain essential properties, such as the existence of a unique closure under the consistency operations [11].

**Definition 3.2.** A consistency ${}^{\phi}_{G}$ is *well-behaved* iff for any QCN $\mathcal{N} = (V, E)$ and any graph $G = (V, E)$ the following properties hold:

- ${}^{\phi}_{G}(\mathcal{N}) \subseteq \mathcal{N}$ (viz., the ${}^{\phi}_{G}$-closure of $\mathcal{N}$ w.r.t. $G$) is the largest (w.r.t. $\subseteq$) ${}^{\phi}_{G}$-consistent sub-QCN of $\mathcal{N}$ *(Dominance)*;
- ${}^{\phi}_{G}(\mathcal{N})$ is equivalent to $\mathcal{N}$ *(Equivalence)*;
- ${}^{\phi}_{G}({}^{\phi}_{G}(\mathcal{N})) = {}^{\phi}_{G}(\mathcal{N})$ *(Idempotence)*;
- if $\mathcal{N}' \subseteq \mathcal{N}$ then ${}^{\phi}_{G}(\mathcal{N}') \subseteq {}^{\phi}_{G}(\mathcal{N})$ *(Monotonicity)*.

Next, we arrive at one of our main results in this work.

**Theorem 3.1.** ${}^{\overleftarrow{\diamond}}_{G}$-consistency is well-behaved.

**Proof.** Let $\mathcal{N} = (V, C)$ be a QCN of a qualitative constraint language, $G = (V, E)$ a graph, and $(\alpha^{-1}(0), \alpha^{-1}(1), \ldots, \alpha^{-1}(n-1))$ some ordering of variables in $V$ that is defined by a bijection $\alpha : V \to \{0, 1, \ldots, n-1\}$. We prove the properties of well-behaveness as follows.

- (Dominance) Let $\mathcal{N}_1 = (V, C_1), \mathcal{N}_2 = (V, C_2), \ldots, \mathcal{N}_k = (V, C_k)$ be all the $k$ different ${}^{\overleftarrow{\diamond}}_{G}$-consistent and not trivially inconsistent sub-QCNs of $\mathcal{N}$, where $k$ is some positive integer. We need to show that $\mathcal{N}' = (V, C') = \bigcup_{i=1}^{k} \mathcal{N}_i$ is ${}^{\overleftarrow{\diamond}}_{G}$-consistent. Let us consider three variables $v, v', v'' \in V$ such that $\{v, v'\}, \{v, v''\}, \{v', v''\} \in E$ and $\alpha(v), \alpha(v') < \alpha(v'')$ and a base relation $b$ such that $b \in C'(v, v')$. Then, we have that $b \in C_i(v, v')$ for some $i \in \{1, 2, \ldots, k\}$. Since $\mathcal{N}_i$ is ${}^{\overleftarrow{\diamond}}_{G}$-consistent, we have that $C_i(v, v') \subseteq C_i(v, v'') \diamond C_i(v'', v')$ and, as it is not trivially inconsistent, there exist base relations $b' \in C_i(v, v'')$ and $b'' \in C_i(v'', v')$ such that $b \in b' \diamond b''$. Therefore, we have that $b' \in C'(v, v'')$ and $b'' \in C'(v'', v')$. It follows that $b \in C'(v, v'') \diamond C'(v'', v')$ and that $\mathcal{N}'$ is ${}^{\overleftarrow{\diamond}}_{G}$-consistent. Clearly, by its construction $\mathcal{N}'$ is unique and the largest ${}^{\overleftarrow{\diamond}}_{G}$-consistent sub-QCN of $\mathcal{N}$. If ${}^{\overleftarrow{\diamond}}_{G}(\mathcal{N})$ is trivially inconsistent to begin with, then ${}^{\overleftarrow{\diamond}}_{G}(\mathcal{N})$ is clearly the unique largest ${}^{\overleftarrow{\diamond}}_{G}$-consistent sub-QCN of $\mathcal{N}$.

- (Equivalence) Let $\mathcal{N}' = (V, C')$ be the QCN where $\forall v, v' \in V$ and $\forall b \in \mathsf{B}$ we have that $b \in C'(v, v')$ iff there exists a solution $\sigma$ of $\mathcal{N}$ such that $(\sigma(v), \sigma(v')) \in b$ (note that $\mathcal{N}'$ is minimal). Clearly, $\mathcal{N}'$ is a sub-QCN of $\mathcal{N}$ and it is necessarily $\diamond$-consistent (as a *disjunctively non-redundant* QCN [58]) and, hence, it is also $\overleftarrow{\underset{G}{\diamond}}$-consistent. It follows that $\mathcal{N}' \subseteq \overleftarrow{\underset{G}{\diamond}}(\mathcal{N}) \subseteq \mathcal{N}$ and, hence, $\overleftarrow{\underset{G}{\diamond}}(\mathcal{N})$ and $\mathcal{N}$ share the same set of solutions.
- (Idempotence) $\overleftarrow{\underset{G}{\diamond}}(\mathcal{N})$ is $\overleftarrow{\underset{G}{\diamond}}$-consistent and, hence, we have that by dominance of $\overleftarrow{\underset{G}{\diamond}}$-consistency the largest $\overleftarrow{\underset{G}{\diamond}}$-consistent sub-QCN of $\overleftarrow{\underset{G}{\diamond}}(\mathcal{N})$ is itself. Thus, $\overleftarrow{\underset{G}{\diamond}}(\overleftarrow{\underset{G}{\diamond}}(\mathcal{N})) = \overleftarrow{\underset{G}{\diamond}}(\mathcal{N})$.
- (Monotonicity) As $\mathcal{N}' \subseteq \mathcal{N}$, we have that $\overleftarrow{\underset{G}{\diamond}}(\mathcal{N}')$ is a $\overleftarrow{\underset{G}{\diamond}}$-consistent sub-QCN of $\mathcal{N}$. In addition, by dominance of $\overleftarrow{\underset{G}{\diamond}}$-consistency $\overleftarrow{\underset{G}{\diamond}}(\mathcal{N})$ is the largest $\overleftarrow{\underset{G}{\diamond}}$-consistent sub-QCN of $\mathcal{N}$ and, hence, $\overleftarrow{\underset{G}{\diamond}}(\mathcal{N}') \subseteq \overleftarrow{\underset{G}{\diamond}}(\mathcal{N})$.

We can conclude that $\overleftarrow{\underset{G}{\diamond}}$-consistency is well-behaved. □

We recall the following result that suggests that, given any QCN over a distributive subclass of relations, one can extend any scenario of that QCN to a scenario of the QCN that is obtained by introducing a new variable in the network, under certain conditions:

**Theorem 3.2.** [47] 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 3.2, we can prove the following proposition:

**Proposition 3.2.** *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 $i, j < k$. Thus, by Theorem 3.2, 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. □

In the following proposition, we specify the conditions under which one can opt to apply $\overleftarrow{\diamond}_G$-consistency in order to achieve the same end result as $\overleftarrow{\diamond}$-consistency. This result allows for optimizing algorithms that utilize the properties of $\overleftarrow{\diamond}$-consistency in order to achieve their objective, as enforcing $\overleftarrow{\diamond}_G$-consistency instead in a given QCN can cut down on the number of consistency operations to be performed and, hence, lead to faster execution. We will also use this result in our own algorithm for checking the satisfiability of a QCN in Section 4.

**Proposition 3.3.** *Let $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ be a QCN of a qualitative constraint language, and $G = (V, E)$ a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$ and $(v_{n-1}, \ldots, v_1, v_0)$ is a perfect elimination ordering of $G$. If $\mathcal{N}$ is not trivially inconsistent, then $\mathcal{N}$ is $\overleftarrow{\diamond}_G$-consistent if and only if $\mathcal{N}$ is $\overleftarrow{\diamond}$-consistent.*

**Proof.** Clearly, if $\mathcal{N}$ is $\overleftarrow{\diamond}$-consistent then it is also $\overleftarrow{\diamond}_G$-consistent by definition of $\overleftarrow{\diamond}_G$-consistency. Let us assume that $\mathcal{N}$ is $\overleftarrow{\diamond}_G$-consistent but not $\overleftarrow{\diamond}$-consistent. Then, there exists some $k \in \{2, 3, \ldots, n-1\}$ and some edge $\{v_i, v_j\} \notin G$ with $i, j < k$, such that $\mathcal{N}[v_i, v_j] \not\subseteq \mathcal{N}[v_i, v_k] \diamond \mathcal{N}[v_k, v_j]$. This can only be possible if $\{v_i, v_k\}, \{v_j, v_k\} \in \mathsf{G}(\mathcal{N})$. If $\{v_i, v_k\}, \{v_j, v_k\} \in \mathsf{G}(\mathcal{N})$, then $\{v_i, v_k\}, \{v_j, v_k\} \in G$, as $\mathsf{G}(\mathcal{N}) \subseteq G$. Further, as $G$ is a chordal graph and $(v_{n-1}, \ldots, v_1, v_0)$ is a perfect elimination ordering of $G$, we have that $v_k$ is a simplicial vertex of the subgraph $G_k$ induced by $(v_0, v_1, \ldots, v_k)$, i.e., the subgraph of $G_k$ induced by the neighbourhood of $v_k$ in $G_k$ is complete. Thus, we have that $\{v_i, v_j\} \in G_k$ and, hence, that $\{v_i, v_j\} \in G$. This proves our result by contradiction. □

By Propositions 3.2 and 3.3 we can obtain the following corollary:

**Corollary 3.1.** *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, and $G = (V, E)$ a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$ and $(v_{n-1}, v_1, \ldots, v_0)$ is a perfect elimination ordering of $G$. If $\mathcal{N}$ is $\overleftarrow{\diamond}_G$-consistent and not trivially inconsistent, then $\mathcal{N}$ is satisfiable.*

In Corollary 3.1 we established how $\overleftarrow{\diamond}_G$-consistency can be used to decide satisfiability of a QCN. Next, we show that $\overleftarrow{\diamond}_G$-consistency can be used to characterize minimal and redundant constraints on demand. In particular, we show that $\overleftarrow{\diamond}_G$-consistency can allow determining the minimality or the redundancy of a constraint in an incremental (or targeted) manner, without having to compute the full closure of the network at hand under $\diamond_G$-consistency [2, 65]. This result can be particularly useful in dynamic and time-critical applications where spatial and/or temporal information is updated every so often or obtained gradually and not all at once, as in spatio-temporal stream reasoning for example [14, 31]. We will first need to prove the following result:

**Proposition 3.4.** *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*

*property 1. Then, we have that* $\overleftarrow{\diamond}(\mathcal{N})[v_0, v_1] = \diamond(\mathcal{N})[v_0, v_1]$.

**Proof.** Let $\mathcal{M} = \overleftarrow{\diamond}(\mathcal{N})$. In the case where $\mathcal{M}[v_0, v_1] = \emptyset$, by Proposition 3.1 it trivially follows that $\diamond(\mathcal{M})[v_0, v_1] = \emptyset$. Therefore, we consider the case where $\mathcal{M}[v_0, v_1] \neq \emptyset$. This also suggests that $\mathcal{M}$ is not trivially inconsistent by the very definition of $\overleftarrow{\diamond}$-consistency (which, as a reminder, considers complete graphs). 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) $v_l$. Let $\mathcal{M}_i$ with $i \in \{0, \ldots, l'\}$ and $0 \leq l' < n-1$ denote $\bigcap_{j=0}^{l'} \mathcal{M}[v_{l'+1}, v_j] \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_j, v_i])$. For the moment we assume that $\diamond(\mathcal{M}{\downarrow_{V_{l'}}})$ is not trivially inconsistent; later on we will show that this assumption is valid. We first show that $\mathcal{M}_i$ is not empty. Note that by Theorem 2.2 we have that the subclass of relations under consideration is Helly, as it satisfies property 1 and it is distributive as well. Thus, it suffices to show that

$$\mathcal{M}[v_{l'+1}, v_j] \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_j, v_i]) \cap \mathcal{M}[v_{l'+1}, v_{j'}] \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_{j'}, v_i]) \neq \emptyset$$

for all $j, j' \in \{0, \ldots, l'\}$. As $\mathcal{M}$ 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 (see [21, Chapt. 3]), we have that

$$\begin{aligned}
& \mathcal{M}[v_{l'+1}, v_j] \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_j, v_i]) \cap \mathcal{M}[v_{l'+1}, v_{j'}] \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_{j'}, v_i]) \neq \emptyset \\
\text{iff} \quad & \mathcal{M}[v_{l'+1}, v_j] \cap \mathcal{M}[v_{l'+1}, v_{j'}] \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_{j'}, v_i]) \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_i, v_j]) \neq \emptyset \\
\text{iff} \quad & \mathcal{M}[v_{j'}, v_{l'+1}] \diamond \mathcal{M}[v_{l'+1}, v_j] \cap (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_{j'}, v_i]) \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_i, v_j]) \neq \emptyset.
\end{aligned}$$

Since

$$\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_{j'}, v_j] \subseteq \diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_{j'}, v_i] \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_i, v_j])$$

and

$$\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_{j'}, v_j] \subseteq \mathcal{M}[v_{j'}, v_{l'+1}] \diamond \mathcal{M}[v_{l'+1}, v_j],$$

we have that $\mathcal{M}_i$ is not empty. At this point, we can show that $\mathcal{M}_{i'} \subseteq \mathcal{M}_i \diamond (\diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_i, v_{i'}])$ for every $i' \in \{0, \ldots, l'\}$ by using the fact that weak composition distributes over non-empty intersection. We have that

$$\begin{aligned}
\mathcal{M}_i \diamond \diamond(\mathcal{M}{\downarrow_{V_{l'}}}[v_i, v_{i'}]) &= \Big(\bigcap_{j=0}^{l'} \mathcal{M}[v_{l'+1}, v_j] \diamond \diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_j, v_i]\Big) \diamond \diamond(\mathcal{M}{\downarrow_{V_l}})[v_i, v_{i'}] \\
&= \bigcap_{j=0}^{l'} (\mathcal{M}[v_{l'+1}, v_j] \diamond \diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_j, v_i] \diamond \diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_i, v_{i'}]) \\
&\supseteq \bigcap_{j=0}^{l'} (\mathcal{M}[v_{l'+1}, v_j] \diamond \diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_j, v_{i'}]) \\
&= \mathcal{M}_{i'}.
\end{aligned}$$

This result suggests that the $\mathsf{QCN}$ $\mathcal{M}' = (V_{l'+1}, C')$, where $C'(v_{l'+1}, v_i) = \mathcal{M}_i$ and $C'(v_i, v_{l'+1}) = (\mathcal{M}_i)^{-1}$ for each $i \in \{0, \ldots, l'\}$, $C'(v_j, v_{j'}) = \diamond(\mathcal{M}{\downarrow_{V_{l'}}})[v_j, v_{j'}]$ for every $j, j' \in \{0, \ldots, l'\}$, and $C'(v_{l'+1}, v_{l'+1}) = \{\mathsf{Id}\}$, is the unique largest $\diamond$-consistent and not trivially inconsistent sub-$\mathsf{QCN}$ of $\mathcal{M}{\downarrow_{V_{l'+1}}}$, viz., $\diamond(\mathcal{M}{\downarrow_{V_{l'+1}}})$. Up until this point we have shown that, for any $0 \leq l' < n-1$, we can extend $\diamond(\mathcal{M}{\downarrow_{V_{l'}}})$ to the not trivially inconsistent $\mathsf{QCN}$ $\diamond(\mathcal{M}{\downarrow_{V_{l'+1}}})$ without revising any of the constraints of $\diamond(\mathcal{M}{\downarrow_{V_{l'}}})$ in the process; however, we remind the reader that this result holds under the assumption that $\diamond(\mathcal{M}{\downarrow_{V_{l'}}})$ is not trivially inconsistent. As we dealt with the case where $\mathcal{M}[v_0, v_1] \neq \emptyset$ and as for $l' = 1$ it is clear that we have that $\diamond(\mathcal{M}{\downarrow_{V_1}})[v_0, v_1] = \mathcal{M}[v_0, v_1]$, we can assert that $\diamond(\mathcal{M}{\downarrow_{V_1}})$ is not trivially inconsistent and, hence, the general result follows inductively. By dominance of $\diamond$-consistency and Proposition 3.1 it follows that $\diamond(\mathcal{N}) = \diamond(\mathcal{M})$. In particular, we have that $\diamond(\mathcal{N}) \subseteq \mathcal{M} \subseteq \mathcal{N}$ and since $\diamond(\mathcal{N})$ is the largest $\diamond$-consistent sub-$\mathsf{QCN}$ of $\mathcal{N}$, it is also the the largest $\diamond$-consistent sub-$\mathsf{QCN}$ of $\mathcal{M}$. Therefore, we can conclude that $\overleftarrow{\diamond}(\mathcal{N})[v_0, v_1] = \diamond(\mathcal{N})[v_0, v_1]$. □

The following result follows directly from Theorem 2.1 and Propositions 3.3 and 3.4 and concerns the minimal labelling problem:

**Corollary 3.2.** *Let $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ be a $\mathsf{QCN}$ defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2, and $G = (V, E)$ a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$ and $(v_{n-1}, \ldots, v_1, v_0)$ is a perfect elimination ordering of $G$. If $\mathcal{N}$ is $\overleftarrow{\diamond}_G$-consistent and not trivially inconsistent, then every base relation $b \in \mathcal{N}[v_0, v_1]$ is a feasible base relation of $\mathcal{N}$.*

The following result follows from Corollary 3.2, the definition of a redundant constraint (see Definition 2.2), and equivalence of $\overleftarrow{\diamond}_G$-consistency (in that it does not remove feasible base relations) and concerns the redundancy problem:

**Corollary 3.3.** *Let $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ be a satisfiable $\mathsf{QCN}$ defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2, $G = (V, E)$ a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$ and $(v_{n-1}, \ldots, v_1, v_0)$ is a perfect elimination ordering of $G$, and $\mathcal{N}' = (V, C')$ the $\mathsf{QCN}$ where $C'(v_0, v_1) = C'(v_1, v_0) = \mathsf{B}$ and $C'(u, u') = C(u, u')$ $\forall (u, u)' \in (V \times V) \setminus \{(v_0, v_1), (v_1, v_0)\}$. Then, we have that the constraint $\mathcal{N}[v_0, v_1]$ is redundant in $\mathcal{N}$ if and only if $\overleftarrow{\diamond}_G(\mathcal{N}')[v_0, v_1] \subseteq \mathcal{N}[v_0, v_1]$.*

**Proof.** By definition of a redundant constraint, it is clear that $\mathcal{N}[v_0, v_1]$ is redundant in $\mathcal{N}$ iff every solution of $\mathcal{N}'$ satisfies $\mathcal{N}[v_0, v_1]$. The result then follows directly from Corollary 3.2 and equivalence of $\overleftarrow{\diamond}_G$-consistency. In particular, we have that $\overleftarrow{\diamond}_G(\mathcal{N}')[v_0, v_1]$ contains *all* the feasible base relations of $\mathcal{N}'$ with respect to the constraint $\mathcal{N}'[v_0, v_1]$. Therefore, by checking if $\overleftarrow{\diamond}_G(\mathcal{N}')[v_0, v_1] \subseteq \mathcal{N}[v_0, v_1]$, we check if every solution of $\mathcal{N}'$ satisfies $\mathcal{N}[v_0, v_1]$. □

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

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

```
1  begin
2      G ← (V, E = E(G(N)));
3      for x from n − 1 to 1 do
4          v ← α^{-1}(x);
5          adj ← {v' | {v', v} ∈ E ∧ α(v') < α(v)};
6          foreach v', v'' ∈ adj do
7              if α(v') < α(v'') then
8                  if {v', v''} ∉ E then
9                      E ← E ∪ {{v', v''}};
10                 r ← C(v', v'') ∩ (C(v', v) ⋄ C(v, v''));
11                 if r ⊂ C(v', v'') then
12                     C(v', v'') ← r;
13                     C(v'', v') ← r^{-1};
14                 if C(v', v'') = ∅ then
15                     return (False,G);
16     return (True,G);
```

## 4. Efficient Approach for Solving a QCN

In this section we propose a new approach in the context of qualitative constraint-based 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 [76]. 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 [18] by Zhang et al. in [77].

### 4.1. *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.

Next, we describe algorithm VarElimination, presented formally in Algorithm 1,

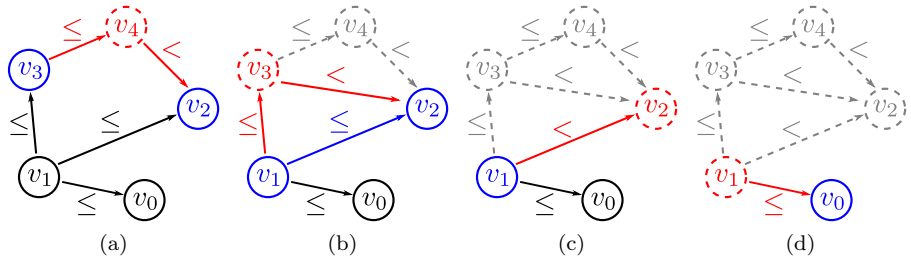


Figure 4. Illustration of the execution of VarElimination.

which is built on the idea of the variable elimination algorithm [76] that we mentioned earlier and uses the notion of $\overleftarrow{\diamond}_G$-consistency to decide the satisfiability of a given QCN defined over a distributive subclass of relations. 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 $\overleftarrow{\diamond}_G$-consistency on $\mathcal{N}$, and terminates with a value of (True, _[d]) if it succeeds in doing so without producing a trivial inconsistency. Specifically, the algorithm iterates over the variables of the QCN with respect to a given ordering (lines 3–4) and, for each variable, it retrieves all the variables that correspond to its neighbouring vertices in the given graph $G$ (when the variable at hand itself is viewed as a vertex) and that precede it in the ordering (line 5). Then, in each such iteration, it directionally closes all the resulting triples of variables under weak composition with respect to that ordering (lines 6–15).

**Example 4.1.** Consider the QCN $\mathcal{N} = (V, C)$ of Point Algebra, where $V = \{v_0, \ldots, v_4\}$; the non-universal relations between the variables of $\mathcal{N}$ are shown in Figure 4(a). Let $\alpha : V \to \{0, 1, \ldots, 4\}$ with $\alpha(v_0) = 0$, $\alpha(v_1) = 1$, $\alpha(v_2) = 2$, $\alpha(v_3) = 3$, and $\alpha(v_4) = 4$ be the bijection that is given as input to algorithm VarElimination along with the QCN $\mathcal{N}$. Figure 4 illustrates the execution of VarElimination. In particular, Figure 4(a) shows the state of the execution when considering $x = 4$ and $v = v_4$ in the *for* loop of the algorithm. In this state, we have adj $= \{v_3, v_2\}$. Note that $\{v_3, v_2\} \notin E$. Therefore, we add $\{v_3, v_2\}$ in $E$ and $C(v_3, v_2)$ becomes $\{<, =\} \diamond \{<\}$, i.e., $\{<\}$, as shown in Figure 4(b), which shows the state of the execution when considering $x = 3$ and $v = v_3$. In that state, we have adj $= \{v_1, v_2\}$ and, hence, $C(v_1, v_2)$ is refined to $\{<, =\} \cap (\{<, =\} \diamond \{<\})$, i.e., $\{<\}$. The result of the refinement is shown in Figure 4(c), where in turn we have $x = 2$, $v = v_2$, and adj $= \{v_1\}$. As there is only one element in adj, the process defined in lines 8–15 of algorithm VarElimination is not executed this time. Next, we consider $x = 1$ and $v = v_1$, as illustrated in Figure 4(d). Note that, again, in this case adj $= \{v_0\}$ has only one element. Therefore, the relation $C(v_1, v_0)$ is not refined and the end of the

[d]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 [10].

execution is reached.

Algorithm VarElimination has a runtime of $O(\Delta^2|V|)$, where $\Delta$ is the maximum vertex degree of the graph $G$ returned through its output, as it iterates over $O(|V|)$ variables (lines 3–4) and performs $O(\Delta^2)$ operations for the variable $v$ at hand in each iteration (lines 5–15). The relations of a qualitative constraint language are implemented as bit vectors [37,73]. In this way, the fast bitwise AND operation can be used to perform the intersection operation. We 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 $2^{\mathsf{B}}$ 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 $2^{\mathsf{B}}$ 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 [32], 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 [32].

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 $\alpha$ 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. In what follows, we specify the conditions under which the ordering defined by $\alpha$ guarantees that no pair of variables that does not exist in $E(\mathsf{G}(\mathcal{N}))$ will be considered by algorithm VarElimination.

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

**Proposition 4.1.** *Given a* QCN $\mathcal{N} = (V, C)$*, with* $|V| = n$*, and a bijection* $\alpha : V \rightarrow \{0, 1, \ldots, n-1\}$*, we have that if algorithm* VarElimination *terminates and returns* (True,$G$)*, then* $G$ *is a chordal graph such that* $\mathsf{G}(\mathcal{N}) \subseteq G$ *and* ($\alpha^{-1}(n-1)$,$\alpha^{-1}(n-2)$,...,$\alpha^{-1}(0)$) *is a perfect elimination ordering of* $G$*.*

**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 that $G$ is also chordal after the termination of algorithm VarElimination. As

defined earlier in Section 2, $G_i$ 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 neighbours 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 $G_x$. Thus, 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. Thus, we have that the edge-augmented graph $G = (V, E)$ of $\mathsf{G}(\mathcal{N})$ is a chordal graph by Theorem 2.3. 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 4.1, 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 4.2.** *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* $(\_,\mathsf{G}(\mathcal{N}))$ *if* $\mathsf{G}(\mathcal{N})$ *is a chordal graph and* $(\alpha^{-1}(n-1),\alpha^{-1}(n-2),\ldots,\alpha^{-1}(0))$ *is a perfect elimination ordering of* $\mathsf{G}(\mathcal{N})$*.*

Proposition 4.2 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 [70]. 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 neighbours. 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 is the *approximate minimum degree* heuristic [30]. This heuristic has a runtime of $O(|V||E|)$ for a given graph $G = (V, E)$ [30]. 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 5. Another choice is the *minimum fill-in* heuristic with a runtime of $O(|V|^3)$ [34,60], which again makes its use prohibitive for large QCNs. An overview of other heuristic algorithms for triangulating graphs is provided in [7].

Finally, we establish the correctness of our algorithm:

**Theorem 4.1.** 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 $\alpha : V \to \{0, 1, \ldots, n-1\}$, we have that algorithm VarElimination terminates and returns (True,$G$) 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 $v_{\alpha(u)}$. Thus, the set of variables $V$ will be the set $\{v_0, v_1, \ldots, v_{n-1}\}$. Let $\mathcal{N}'$ denote the refined QCN of $\mathcal{N}$ that results from the execution of algorithm VarElimination 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,$G$), 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 $\{v_i, v_k\}, \{v_j, v_k\} \in E(G)$ and $i, j < k$, and for all $v_i, v_j \in V$ with $\{v_i, v_j\} \in E(G)$ we will have that $\mathcal{N}'[v_i, v_j] \neq \emptyset$. Thus, by definition of $\overset{\leftarrow}{\diamond}_G$-consistency, we will have that $\mathcal{N}'$ is $\overset{\leftarrow}{\diamond}_G$-consistent, and, more precisely, that $\mathcal{N}' = \overset{\leftarrow}{\diamond}_G(\mathcal{N})$. Further, as $\mathcal{N}'[v_i, v_j] \neq \emptyset$ for all $v_i, v_j \in V$, we will have that $\mathcal{N}'$ is not trivially inconsistent. Note also that by Proposition 4.1 we will have that $G$ is a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$ and $(\alpha^{-1}(n-1), \alpha^{-1}(n-2), \ldots, \alpha^{-1}(0))$ (or, equivalently, $(v_{n-1}, \ldots, v_1, v_0)$) is a perfect elimination ordering of $G$. Then, by Corollary 3.1 we will 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 will have that $\mathcal{N}$ is satisfiable. Next, we address the "if" part. If $\mathcal{N}$ is satisfiable, then it yields the not trivially inconsistent QCN $\diamond(\mathcal{N})$. Let us assume that given $\mathcal{N}$ and some bijection $\alpha : V \to \{0, 1, \ldots, n-1\}$, 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 execution of algorithm VarElimination 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 and, hence, that $\mathcal{N}$ is unsatisfiable, which concludes this part of our proof by contraposition. Thus, we can conclude that algorithm VarElimination terminates and returns (True,_) if and only if $\mathcal{N}$ is satisfiable. □

Due to Propositions 2.1 and 2.2, and Theorem 4.1, 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.

### 4.2. *Scenario Extraction*

In Section 4.1 we showed how we can efficiently decide the satisfiability of a given QCN defined over a distributive subclass of relations. In this section, given a QCN $\mathcal{N}$, we show how a scenario of $\mathcal{N}$, i.e., an atomic satisfiable sub-QCN of $\mathcal{N}$, can actually be extracted. The contribution involves performing a generic backtrack-free procedure for refining a $\overleftarrow{\diamond}_G$-consistent and not trivially inconsistent QCN $\mathcal{N}$ defined over a distributive subclass of relations to an atomic $\overleftarrow{\diamond}_G$-consistent sub-QCN of $\mathcal{N}$. After having refined $\mathcal{N}$ to a scenario, some known method from the literature for valuating the variables of $\mathcal{N}$ in order to satisfy it can be used; 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 [8]). We will give an overview of such methods in a later section.

Before presenting our backtrack-free procedure for efficiently extracting a scenario from a satisfiable QCN defined over a distributive subclass of relations, we will review some state-of-the-art techniques that accomplish the same task for this particular case of QCNs. As noted in the introduction, these techniques are either semantics-based or generic.

**State-of-the-art Semantics-based Scenario Extraction** To the best of our knowledge, the most notable state-of-the-art semantics-based scenario extraction techniques for QCNs defined over a distributive subclass of relations (or even a larger subclass than that in some cases) have been designed for RCC-8, Interval Algebra, and Point Algebra.

With respect to RCC-8, in [55] Renz specifies a particular procedure that, given a $\diamond$-consistent and not trivially inconsistent QCN $\mathcal{N} = (V, C)$ of RCC-8 defined over a certain subclass of relations, refines that QCN to a scenario via a table lookup scheme in $O(|V|^2)$ time. In particular, this procedure uses a lookup table that maps every relation of the certain subclass of relations considered to a base relation (see Lemma 20 in [55]). As a reminder, in order to close a given QCN over $n$ variables under $\diamond$-consistency a time of $O(n^3|\mathsf{B}|)$ is required.

With respect to Interval Algebra, in [42] Ligozat presents a particular method that, given a $\diamond$-consistent and not trivially inconsistent QCN $\mathcal{N} = (V, C)$ of Interval Algebra defined over a certain (so called *pre-convex* [41]) subclass of relations, refines that QCN to a scenario through a successive operation. In particular, this operation calls for assigning an interval to each variable, in any order, such that relevant constraints are respected and the endpoints of previously used intervals are avoided whenever possible (see Theorem 2 in [42]). Such intervals are called intervals of *maximal dimension* in [42]. Note again that, as is the case with Renz's approach, a given QCN has to be closed under $\diamond$-consistency first.

With respect to Point Algebra, in [72] van Beek takes an algorithmic approach to compute a scenario of a given QCN of Point Algebra that does not require that QCN to be closed under $\diamond$-consistency. This approach is based on topological sort [35].

Topological sort can be used to find a consistent instantiation of a QCN of Point Algebra if the temporal information is a strict partial order, i.e., if the allowed relations are restricted to $\{<\}$, $\{>\}$, and B. Topological sort is $O(n^2)$ for a given QCN of Point Algebra over $n$ variables. Of course, topological sort alone does not work, as the labels of a QCN of Point Algebra may be any one of the eight different relations, viz., $\emptyset$, $\{<\}$, $\{<,=\}$, $\{<,>\}$, $\{=\}$, $\{=,>\}$, $\{>\}$, and B, and hence may have less information about the relation between two points than is required. However, van Beek managed to appropriately handle the "problematic" relations and was able to present a sound and complete algorithm for extracting a scenario from a QCN of Point Algebra within the $O(n^2)$ time bound (see Chapter 3 in [72]). Notably, this algorithm is sound and complete also for a restricted subclass of Interval Algebra networks that can be translated without loss of information into Point Algebra networks. These special Interval Algebra networks are called SA networks [74]. In particular, in SA networks, the allowed relations between two intervals are only those relations of Interval Algebra that can be translated, using the relations $\emptyset$, $\{<\}$, $\{<,=\}$, $\{<,>\}$, $\{=\}$, $\{=,>\}$, $\{>\}$, and B of Point Algebra, into conjunctions of such relations of Point Algebra between the end points of the intervals. Interestingly, the class of SA networks gives rise to the smaller one of the two distributive subclasses of relations that have been defined for Interval Algebra in [47].

**State-of-the-art Generic Scenario Extraction** To the best of our knowledge, there exists a single state-of-the-art generic scenario extraction technique for QCNs defined over a distributive subclass of relations, which consists in closing a given QCN under $\diamond$-consistency, choosing some base relation from an uninstantiated constraint, instantiating that constraint with the chosen base relation, and repeating the process until a scenario is obtained (cf. [15]). This method, the correctness of which is a direct consequence of Theorem 2.1, is described in the following proposition:

**Proposition 4.3.** *cf. [15] 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 $\diamond$-consistent and not trivially inconsistent, then $\mathcal{N}$ can be refined to a scenario $\mathcal{S}$ of $\mathcal{N}$ as follows. For every $i, j \in \{0, \ldots, n-1\}$ such that $i < j$, do:*

- $\mathcal{N}[v_i, v_j] \leftarrow \{b\}$ *for some* $b \in \mathcal{N}[v_i, v_j]$*;*
- $\mathcal{N}[v_j, v_i] \leftarrow (\mathcal{N}[v_i, v_j])^{-1}$*;*
- $\mathcal{N} \leftarrow \diamond(\mathcal{N})$.

A recursive algorithmic instantiation of the aforementioned proposition can be found in [12]. Note that the result of Theorem 2.1 guarantees that the execution of the algorithm in [12] will be backtrack-free for a satisfiable QCN defined over a distributive subclass of relations. Given a satisfiable QCN $\mathcal{N} = (V, C)$ defined over a distributive subclass of relations, the scenario extraction method described in

Proposition 4.3 runs in $O(|V|^3|\mathsf{B}|)$ time by exploiting the incremental functionality of $\diamond$-consistency [25, Section 3]. In particular, each time a constraint in a $\diamond$-consistent $\mathsf{QCN}$ is refined to a singleton relation, the closure under $\diamond$-consistency is applied by first checking only the constraints that can be directly affected by that refinement. Provided that a constraint is indeed affected and is, in turn, refined to a smaller (i.e., stronger) relation, other constraints that can be directly affected by that refinement will be checked, and so on. However, note that in any case a constraint cannot be refined to a smaller relation more than $|\mathsf{B}|$ times. Thus, the $O(|V|^3|\mathsf{B}|)$ time bound is achieved.

Now, we are ready to present our own generic scenario extraction technique for a $\mathsf{QCN}$ defined over a distributive subclass of relations. This technique consists in first closing the $\mathsf{QCN}$ under $\overleftarrow{\diamond}_G$-consistency with respect to some ordering of its variables, and then performing exact refinements for its constraints by following the reverse ordering of the one we used to enforce $\overleftarrow{\diamond}_G$-consistency. The following result describes the steps of our scenario extraction technique:

**Proposition 4.4.** *Let $\mathcal{N} = (V = \{v_0, v_1, \ldots, v_{n-1}\}, C)$ be a $\mathsf{QCN}$ defined over a distributive subclass of relations of a qualitative constraint language that satisfies properties 1 and 2, and $G = (V, E)$ a chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$ and $(v_{n-1}, v_1, \ldots, v_0)$ is a perfect elimination ordering of $G$. If $\mathcal{N}$ is $\overleftarrow{\diamond}_G$-consistent and not trivially inconsistent, 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:*

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

**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) $v_l$. We know that $\mathcal{N}$ is not trivially inconsistent and by Proposition 3.3 we have that $\mathcal{N}$ is also $\overleftarrow{\diamond}$-consistent. Therefore, by Proposition 3.2 we have that $\mathcal{N}$ is satisfiable and, hence, that $\mathcal{N}{\downarrow_{V_{l'}}}$ is satisfiable for any $0 \leq l' < n-1$. Since $\mathcal{N}{\downarrow_{V_{l'}}}$ is satisfiable for some $0 \leq l' < n-1$, 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 $j, j' \in \{0, \ldots, l'\}$. See Figure 5 for illustration.

We will show that we can extend that scenario to a scenario of $\mathcal{N}{\downarrow_{V_{l'+1}}}$ with our proposed construction scheme. 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. By definition of $\overleftarrow{\diamond}$-consistency it follows that the updated $\mathcal{N}$ remains $\overleftarrow{\diamond}$-consistent and not trivially inconsistent. By Proposition 3.4 we have that $\diamond(\mathcal{N})$ is not trivially inconsistent, as we would have that $\mathcal{N}[v_0, v_1] = \emptyset$ otherwise, and that would be a contradiction. From this we can deduce that $\diamond(\mathcal{N})[v_{l'+1}, v_i] \subseteq \bigcap_{j=0}^{l'} \mathcal{N}[v_{l'+1}, v_j] \diamond \mathcal{N}[v_j, v_i]$ for all $i \in \{0, \ldots, l'\}$. Thus, there exists a base relation $b \in \mathsf{B}$ such that $b \in \bigcap_{j=0}^{l'} \mathcal{N}[v_{l'+1}, v_j] \diamond \mathcal{N}[v_j, v_i]$ for any $i \in \{0, \ldots, l'\}$. Let us assign the base relation $b$ to

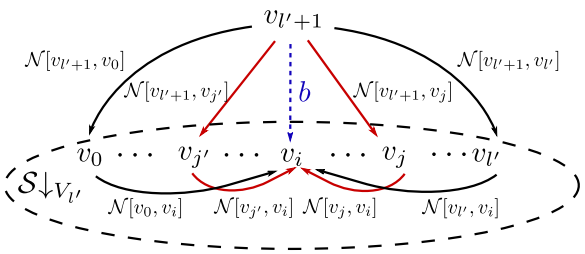


Figure 5. Illustration of the proof of Proposition 4.4, updating relation $\mathcal{N}[v_{l'+1}, v_i]$ with $b$.

$\mathcal{N}[v_{l'+1}, v_i]$ for some $i \in \{0, \ldots, l'\}$, 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 not trivially inconsistent $\mathcal{N}\downarrow_{V_{l'+1}}$ remains $\overleftarrow{\diamond}$-consistent with respect to the variable ordering $(v_0, v_1, \ldots, v_{l'+1})$. 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 $j \in \{0, \ldots, l'\}$. By the way $b$ has been acquired, we have that $\{b\} \subseteq \mathcal{N}[v_{l'+1}, v_j] \diamond \mathcal{N}[v_j, v_i]$. 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 that holds for relation algebras (see [21, Chapt. 3]), 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 $\mathcal{N}[v_j, v_i] \subseteq \mathcal{N}[v_j, v_{l'+1}] \diamond \{b\}$. By repeating this process for all the remaining $i \in \{0, \ldots, l'\}$, we can extend $\mathcal{S}\downarrow_{V_{l'}}$ to a scenario of $\mathcal{N}\downarrow_{V_{l'+1}}$. The general result follows inductively. □

By comparing the state-of-the-art generic scenario extraction method (see Proposition 4.3) with our own procedure, one can clearly see the advantage of our approach. In particular, the main difference lies in the fact that we do not require a QCN to be $\diamond$-consistent, but $\overleftarrow{\diamond}_G$-consistent. As we will demonstrate in the experimental evaluation in Section 5, this alone makes for a huge performance boost. Further, consider the third step of the method described in Proposition 4.3. By definition of $\diamond$-consistency, when a constraint is revised, one typically has to *at least* compute the weak compositions of its neighbouring constraints and perform an intersection of the results. This suggests that the step under discussion will perform *at least* the same number of operations as the first step of our own approach. By comparing the state-of-the-art semantics-based scenario extraction methods with our own procedure, with the exception of Point Algebra and a restricted subclass of Interval Algebra networks, we can again see that, as opposed to those methods, we do not require a QCN to be $\diamond$-consistent. In addition, our approach allows for extracting any scenario of a given QCN (note that the choice of a base relation in the second step of our approach can be completely random), and not just a scenario whose extraction is force-guided by the semantics of the qualitative constraint language at hand (e.g., check the notion of *maximal dimension* intervals in [42]).

**Example 4.2.** Consider the $\overleftarrow{\diamond}_G$-consistent and not trivially inconsistent QCN obtained in Example 4.1, whose constraint graph $G$ is shown in Figure 4(d). Note

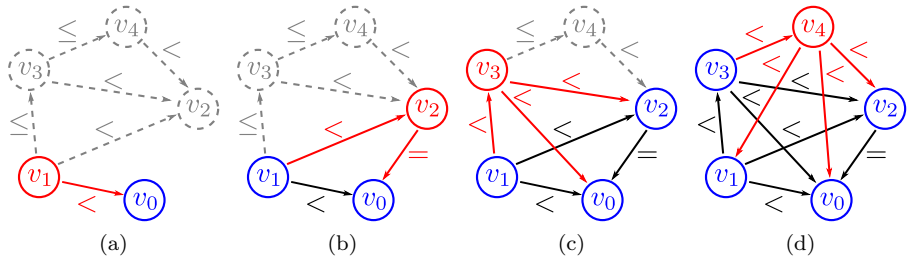


Figure 6. Illustration of scenario extraction.

that $G$ is a chordal graph and $(v_4, \ldots, v_0)$ is a perfect elimination ordering of $G$. For $k = 1$, according to the update rules given in Proposition 4.4, we assign $\mathcal{N}[v_1, v_0] \diamond \mathcal{N}[v_0, v_0] = \{<, =\}$ to $\mathcal{N}[v_1, v_0]$. Then, we (randomly) choose a base relation $b$ in $\mathcal{N}[v_1, v_0]$, say $b = \{<\}$, and assign it to $\mathcal{N}[v_1, v_0]$. Now we have that $\mathcal{N}[v_1, v_0] = \{<\}$ and $\mathcal{N}[v_0, v_1] = \{>\}$, and the corresponding constraint graph is shown in Figure 6(a). For $k = 2$, again according to the update rules, we first update $\mathcal{N}[v_2, v_0]$ with $\bigcap_{j=0}^{1} \mathcal{N}[v_2, v_j] \diamond \mathcal{N}[v_j, v_0] = \{<, =, >\}$, and then we choose the base relation $\{=\}$ in $\mathcal{N}[v_2, v_0]$ and set $\mathcal{N}[v_2, v_0] = \{=\} = \mathcal{N}[v_0, v_2]$. Similarly, we end up assigning $\{>\}$ to $\mathcal{N}[v_2, v_1]$ and $\{<\}$ to $\mathcal{N}[v_1, v_2]$. The corresponding constraint graph at this point is shown in Figure 6(b). For $k = 3$, we update $\mathcal{N}[v_3, v_0]$ with $\bigcap_{j=0}^{2} \mathcal{N}[v_3, v_j] \diamond \mathcal{N}[v_j, v_0] = \{<\}$ and assign $\{<\}$ to $\mathcal{N}[v_3, v_0]$ and $\{>\}$ to $\mathcal{N}[v_0, v_3]$. Next, we update $\mathcal{N}[v_3, v_1]$ with $\{>, =\}$ and assign $\{>\}$ to $\mathcal{N}[v_3, v_1]$ and $\{<\}$ to $\mathcal{N}[v_1, v_3]$. Further, we update $\mathcal{N}[v_3, v_2]$ with $\{<\}$ and assign $\{<\}$ to $\mathcal{N}[v_3, v_2]$ and $\{>\}$ to $\mathcal{N}[v_2, v_3]$. The current corresponding constraint graph is shown in Figure 6(c). Finally, for $k = 4$, we update $\mathcal{N}[v_4, v_0]$ with $\bigcap_{j=0}^{3} \mathcal{N}[v_4, v_j] \diamond \mathcal{N}[v_j, v_0] = \{<\}$ and assign $\{<\}$ to $\mathcal{N}[v_4, v_0]$ and $\{>\}$ to $\mathcal{N}[v_0, v_4]$. Next, we update $\mathcal{N}[v_4, v_1]$ with $\{>, =\}$ and assign $\{>\}$ to $\mathcal{N}[v_4, v_1]$ and $\{<\}$ to $\mathcal{N}[v_1, v_4]$. Further, we update $\mathcal{N}[v_4, v_2]$ with $\{<\}$ and assign $\{<\}$ to $\mathcal{N}[v_4, v_2]$ and $\{>\}$ to $\mathcal{N}[v_2, v_4]$. As a last step, we update $\mathcal{N}[v_4, v_3]$ with $\{<\}$ and assign $\{<\}$ to $\mathcal{N}[v_4, v_3]$ and $\{>\}$ to $\mathcal{N}[v_3, v_4]$. The finalized corresponding constraint graph is shown in Figure 6(d).

Regarding the time complexity of our scenario extraction approach, it is easy to see that it will require $O(|V|^3)$ time for a given $\overleftarrow{\diamond}_G$-consistent QCN $\mathcal{N} = (V, C)$. Specifically, our procedure iterates over $O(|V|)$ variables and realizes $O(|V|^2)$ operations (weak compositions and intersections) in total for the variable $v$ at hand in each iteration.

### 4.3. *Solution Generation*

In this section, we present an algorithm for generating 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 GenerateSolution. First, algorithm GenerateSolution uses algorithm VarElimination to make $\mathcal{N}$ $\overleftarrow{\diamond}_G$-consistent, then, it

**Algorithm 2:** GenerateSolution($\mathcal{N}$,$\alpha$)

```
   in/out : A QCN N = (V, C) over n variables.
   in     : A bijection α : V → {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) such that μ satisfies N;
13     return (True, μ);
```

applies the procedure specified in Proposition 4.4 to refine $\mathcal{N}$ to a scenario of $\mathcal{N}$, and, finally, it uses some known method from the literature to generate a solution of that scenario. Algorithm GenerateSolution has a runtime of $\max(\{O(|V|^3), \omega\})$, where $\omega$ 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 extract a scenario from $\mathcal{N}$ (lines 5–11), provided that $\mathcal{N}$ is satisfiable. 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 $\omega$, 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 [8], a domain of countably many homeomorphic disjoint components forming a topological space with a valuation method that, again, runs in $O(|V|^3)$ time [38, 40], 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 [56]. The canonical model of Renz [56] 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 $d \geq 1$. It is interesting to note that the approach of Li in [40] gives a simpler and more direct algorithm for generating

realizations of atomic $\diamond$-consistent RCC-8 networks than that of Renz, by adopting, in principle, the same route as that of Renz: firstly, a simple canonical model is identified, then the model is embedded in Euclidean spaces, and, lastly, the realization is completed by modifying some constraints involving proper part relations.

Regarding the qualitative constraint languages that are of our particular interest in this work (see Proposition 2.1), we can prove the following result:

**Theorem 4.2.** 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 $\alpha : V \to \{0, 1, \ldots, n-1\}$, we have that algorithm GenerateSolution terminates and returns (True,$\mu$), only if $\mu$ is a solution of $\mathcal{N}$.

**Proof.** If GenerateSolution terminates and returns (True,$\mu$), we will show that the mapping $\mu$ is created in such a way as to correspond to a solution of $\mathcal{N}$. By Theorem 4.1 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. Thus, we focus solely on generating a solution of $\mathcal{N}$. Algorithm VarElimination refines $\mathcal{N}$ in place, rendering it $\overleftarrow{\diamond}_G$-consistent with respect to the ordering ($\alpha^{-1}(0)$,$\alpha^{-1}(1)$,...,$\alpha^{-1}(n-1)$) of $V$ that is defined by the bijection $\alpha$ (see the proof of Theorem 4.1). $\mathcal{N}$ is then further refined to a scenario of $\mathcal{N}$ using the procedure specified in Proposition 4.4. This procedure is implemented exactly in lines 5–11 of the algorithm. A solution $\mu$ 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 [72], a valuation method for an atomic satisfiable QCN of Cardinal Direction Calculus is presented in [43], and several valuation methods for an atomic satisfiable QCN of RCC-8 are presented in [8,38,56]. We can conclude that algorithm GenerateSolution terminates and returns (True,$\mu$), only if $\mu$ is a solution of $\mathcal{N}$. □

## 5. Experimentation

In this section, we evaluate the performance of our implementation of the VarElimination algorithm for enforcing $\overleftarrow{\diamond}_G$-consistency, against state-of-the-art implementations of the algorithms for enforcing $\diamond_G$-consistency and $\diamond$-consistency, 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 algorithm implementation for enforcing $\diamond_G$-consistency is provided by reasoner Sarissa and a state-of-the-art algorithm implementation for enforcing $\diamond$-consistency is provided by reasoner Phalanx, both of which are in-house implementations and are detailed in [63]. Pyrrhus, like our other aforementioned reasoners, is a generic and open source qualitative constraint-based

spatial and temporal reasoner written in pure Python[e] 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[f], which implements Python 2.7. Only one of the CPU cores was used.

**Dataset and Measures** We considered random RCC-8 and Interval Algebra networks generated by the BA(n, m) model [4], the use of which in qualitative constraint-based reasoning is well motivated in [63], and real-world RCC-8 datasets that have been recently used in [39, 65]

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 by labelling its edges with relations from a maximal distributive subclass. In short, a scale-free graph is a graph whose degree distribution follows a power law. We considered 10 satisfiable and 10 unsatisfiable RCC-8 and Interval Algebra 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 [39, 65], described as follows (in the description by *constraints* we mean *non-universal relations*).

- `nuts`: an RCC-8 network that defines a topological nomenclature of territorial units in Greece with 2 235/3 176 variables/constraints.[g]
- `adm1`: an RCC-8 network of the administrative geography of Great Britain with 11 762/44 832 variables/constraints [28].
- `gadm1`: an RCC-8 network that captures the topological relations among German administrative units with 42 749/159 600 variables/constraints.[g]
- `gadm2`: an RCC-8 network of the topology of the world's administrative areas with 276 729/589 573 variables/constraints.[h]
- `adm2`: an RCC-8 network of the administrative geography of Greece with 1 732 999/5 236 270 variables/constraints.[g]
- `footprints`: an RCC-8 network of geographic "footprints" in the area of Southampton of the UK with 3 470/446 847 variables/constraints [39].
- `statareas`: an RCC-8 network that captures the topology of statistical areas

[e] `https://www.python.org/`
[f] `http://pypy.org/`
[g] Retrieved from: `http://www.linkedopendata.gr/`
[h] `http://gadm.geovocab.org/`

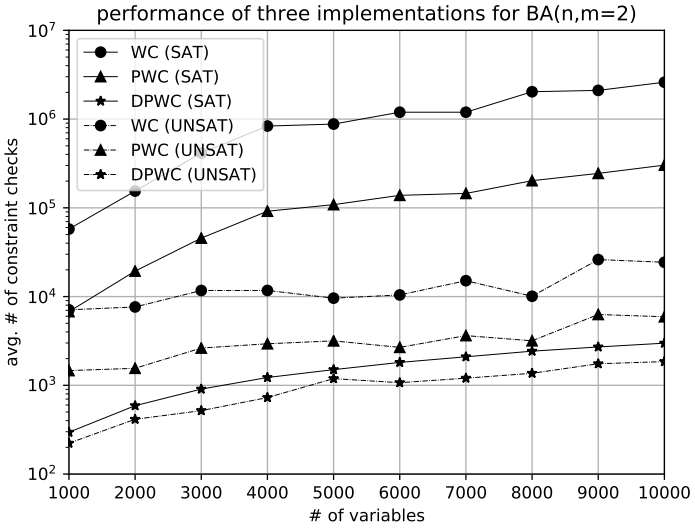


Figure 7. Comparison on # of constraint checks.

in Tasmania with 1 562/10 101 variables/constraints [39].

The aforementioned datasets are satisfiable. Further, the relations of each dataset are contained in either of the maximal distributive subclasses $\mathcal{D}_8^{41}$ and $\mathcal{D}_8^{64}$ of RCC-8 [39].

The maximum cardinality search algorithm [70] was used to obtain a variable elimination ordering $\alpha$ for the implementation of algorithm VarElimination, and a triangulation of the constraint graph of a given QCN based on $\alpha$ for the implementation of the algorithm for enforcing $\overset{\diamond}{}_G$-consistency as described in [63]. We note that in our case any variable elimination ordering would be adequate for the evaluation to follow, as it would affect all involved algorithms proportionally and would not qualitatively distort the obtained results.

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, because it is impossible for relation B to be properly contained in any relation. The second measure concerns the CPU time and it is strongly correlated with the first one, as the runtime of any well-structured implementation of an algorithm for enforcing a local consistency should rely mainly on the number of constraint checks performed.

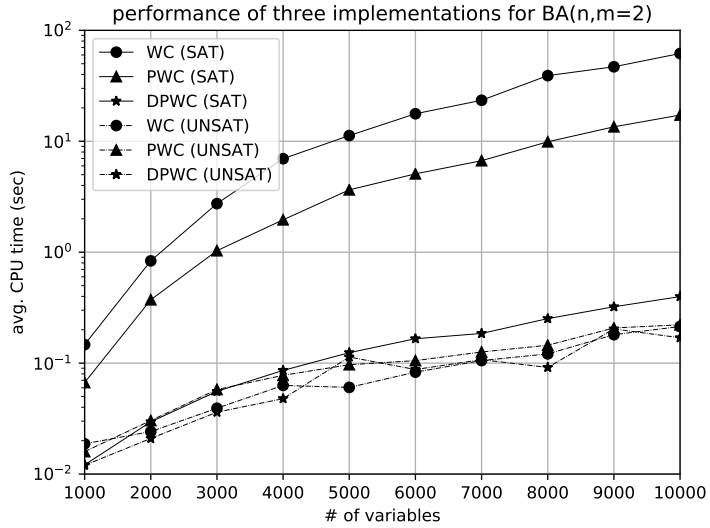


Figure 8. Comparison on CPU time.

**Results** In what follows, WC (which stands for $\diamond$-consistency or closure under weak composition) will denote the implementation of the algorithm for enforcing $\diamond$-consistency of Phalanx, PWC (which stands for $\overset{\diamond}{}_G$-consistency or partial closure under weak composition) will denote the implementation of the algorithm for enforcing $\overset{\diamond}{}_G$-consistency of Sarissa, and DPWC (which stands for $\overset{\leftarrow}{\overset{\diamond}{}}_G$-consistency or directional partial closure under weak composition) will denote the implementation of the algorithm for enforcing $\overset{\leftarrow}{\overset{\diamond}{}}_G$-consistency of Pyrrhus, i.e., the implementation of algorithm VarElimination. As mentioned earlier, the maximum cardinality search algorithm was used to obtain a variable elimination ordering $\alpha$ for DPWC, and a triangulation of the constraint graph of a given QCN based on $\alpha$ for PWC. A triangulation of the constraint graph of a QCN is necessary for PWC in order to be able to make sound use of Proposition 2.3 and decide the satisfiability of that QCN.

Regarding random scale-free RCC-8 networks, the experimental results are shown in Figures 7 and 8. DPWC performs significantly less constraint checks than PWC and WC for both satisfiable and unsatisfiable network instances, as shown in Figure 7. In particular, across all network instances of different size, DPWC 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 8. In particular, across all network instances of different size, DPWC is on average 94.7% and 98.0% faster than PWC and WC respectively for satisfiable network instances, and 20.8% and 1.8% faster than PWC and WC respectively for unsatisfiable network instances; we note that for unsat-

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

| network | WC | PWC | DPWC |
|---|---|---|---|
| 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 | $\infty$ | $\frac{399.60s}{118\,799\,994}$ | $\frac{2.25s}{5\,471\,745}$ |
| footprints | $\frac{345.69s}{689\,888\,647}$ | $\frac{167.73s}{119\,117\,222}$ | $\frac{2.99s}{14\,251\,630}$ |
| statareas | $\frac{29.48s}{29\,009\,108}$ | $\frac{0.25s}{485\,082}$ | $\frac{0.06s}{32\,408}$ |

isfiable network instances all approaches are in a virtual tie, as they unveil the inconsistencies in centiseconds and any difference in performance is thus marginal.

Regarding random scale-free Interval Algebra networks, the results are qualitatively identical to the ones obtained in the case of RCC-8 and, hence, are not reported in detail. It should suffice to mention that DPWC exhibits the same dominant behaviour against PWC and WC in this case as well. The reader is also kindly asked to refer to the work of Long in [46, Section 3.6.3] if he wishes to view a detailed experimental evaluation with Interval Algebra networks of an algorithm that uses DPWC as its backbone (called DPC+ in [46]), which goes beyond the scope of this paper, but is nevertheless interesting.

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 $\infty$ 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 [67]; this pre-processing step took negligible time to realize. Again, we can see that DPWC 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, DPWC 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.

### *Synopsis*

In this section we showed that algorithm VarElimination *significantly* outperforms the state-of-the-art techniques for checking the satisfiability of a QCN defined over

a distributive subclass of relations. Taking into account the fact that algorithm VarElimination performs a single pass through the variables of a QCN in the worst-case scenario, while the rest of the algorithms evaluated here typically revisit a variable multiple times, the results are not surprising and were, somewhat, expected. However, the exact advantage of our approach over competing methods for checking the satisfiability of a QCN defined over a distributive subclass of relations had to be demonstrated.

## 6. Related Work

In this section we overview the related work that is directly linked to the contributions of this paper in that certain results from the literature have been revised and/or extended here and, analogously, certain results from this paper have inspired novel contributions to qualitative constraint-based spatial and temporal reasoning in the recent literature. Other related work has been referenced throughout the paper where appropriate.

A preliminary version of this work has appeared in the *Proceedings of the* $9^{th}$ *Hellenic Conference on Artificial Intelligence (SETN* 2016*)* [66]. In this work, we have extended the results of [66] as follows. We formally introduced a generalized and novel local consistency in the context of qualitative constraint-based spatial and temporal reasoning, denoted by $\overleftarrow{\diamond}_G$-consistency, that combines $\overleftarrow{\diamond}$-consistency, introduced in [66], with $\diamond_G$-consistency, introduced in [9]. In that sense, $\overleftarrow{\diamond}_G$-consistency can be seen as $\diamond_G$-consistency restricted (directionally) to a variable ordering of a considered QCN. With respect to $\overleftarrow{\diamond}_G$-consistency we proved several useful properties, such as the existence of a unique largest closure of any QCN under this consistency and the equivalence of $\overleftarrow{\diamond}_G$-consistency and $\overleftarrow{\diamond}$-consistency under certain conditions (as a reminder, $\overleftarrow{\diamond}$-consistency is $\overleftarrow{\diamond}_G$-consistency where $G$ is a complete graph). Further, we obtained certain theoretical results regarding the use of $\overleftarrow{\diamond}_G$-consistency for solving the fundamental reasoning problems of minimal labelling and redundancy. The aforementioned contributions took place in Section 3 of this paper. In addition, we simplified the proof of correctness of the generic scenario extraction method that appears in [66] and demonstrated its specifics with an illustration. What is more, we contrasted that method to state-of-the-art semantics-based and generic approaches for extracting a scenario from a satisfiable QCN, and we detailed the ways in which our own procedure is superior to those state-of-the-art techniques. Finally, we considered both RCC-8 and Interval Algebra networks in our experimental evaluation, which confirmed the significant advantage of our approach over state-of-the-art techniques for checking the satisfiability of a QCN defined over a distributive subclass of relations.

The foundations that were laid in this work gave rise to the contributions of Long et al. in [48], where the authors demonstrate how $\overleftarrow{\diamond}_G$-consistency can be used in order to efficiently achieve $\diamond_G$-consistency (and $\diamond$-consistency) for QCNs defined over a distributive subclass of relations. Based on the work of Long et al. in [48],

Sioutis and Condotta in [62] further show how $\overset{\diamond}{}_G$-consistency can be achieved for arbitrary QCNs, by utilizing a simple abstraction scheme that relaxes the constraints of a given QCN in such a way that they are defined by relations of a distributive subclass of relations. On another interesting note, the idea of utilizing $\overset{\leftarrow}{\overset{\diamond}{}}_G$-consistency for solving QCNs defined over a distributive subclass of relations influenced the work of Kong et al. in [36], where directional path consistency is employed in a novel algorithm that decides CSPs of a constraint language that is closed under a *majority operation* [6].

## 7. 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 utilizes a particular directional local consistency on $\mathcal{N}$, which we denote by $\overset{\leftarrow}{\overset{\diamond}{}}_G$-consistency, in order to efficiently decide the satisfiability of $\mathcal{N}$ In this paper, we restricted the discussion 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 demonstrated that enforcing $\overset{\leftarrow}{\overset{\diamond}{}}_G$-consistency in a given qualitative constraint network defined over a distributive subclass of relations allows us to decide its satisfiability, and obtained similar useful results for the problems of minimal labelling and redundancy, which are fundamental reasoning problems in the context of qualitative constraint-based spatial and temporal reasoning. Further, we presented a generic method that allows extracting a scenario from a satisfiable network, i.e., an atomic satisfiable subnetwork of that network, in a very simple and effective manner, and contrasted with related state-of-the-art techniques in the literature. 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 and the Interval Algebra, 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 $\overset{\leftarrow}{\overset{\diamond}{}}_G$-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. To this end, we would have to define an efficient incremental variant of the algorithm for enforcing $\overset{\leftarrow}{\overset{\diamond}{}}_G$-consistency that we presented here. We would also like to further explore the implication of $\overset{\leftarrow}{\overset{\diamond}{}}_G$-consistency in the problems of minimal labelling and redundancy. Finally, we would like to investigate a singleton-based variant of

$\overleftarrow{\diamond}_G$-consistency that will involve closing each base relation of each of the constraints of a given qualitative constraint network under $\overleftarrow{\diamond}_G$-consistency. Specifically, this variant will ensure that each such base relation of the qualitative constraint network can define a singleton relation in the corresponding closure of that network under $\overleftarrow{\diamond}_G$-consistency. This stronger local consistency may help to extend the tractability results of $\overleftarrow{\diamond}_G$-consistency to a greater subclass of relations that does not have to be necessarily distributive.

## Acknowledgements

We would like to thank Dimitris Vrakas, a general chair, and Nick Bassiliades and Antonis Bikakis, the program chairs of the 9$^{\text{th}}$ Hellenic Conference on Artificial Intelligence for selecting our work that was published in the proceedings of that conference as a candidate for inclusion in this special issue of the International Journal on AI Tools (IJAIT). We would also like to thank the reviewers for their thoughtful comments and efforts towards improving our manuscript. Michael Sioutis wishes to acknowledge support from the MoveCare project (ID: 732158), which is funded by the European Commission under the H2020 framework programme for research and innovation. Finally, parts of this work were conducted during the doctoral studies of Michael Sioutis in the University of Artois and Zhiguo Long in the University of Technology Sydney.

## References

1. James F. Allen. Maintaining Knowledge about Temporal Intervals. *Commun. ACM*, 26:832–843, 1983.
2. Nouhad Amaneddine, Jean-François Condotta, and Michael Sioutis. Efficient Approach to Solve the Minimal Labeling Problem of Temporal and Spatial Qualitative Constraints. In *IJCAI*, 2013.
3. Philippe Balbiani, Jean-François Condotta, and Luis Fariñas del Cerro. Tractability Results in the Block Algebra. *J. Log. Comput.*, 12:885–909, 2002.
4. A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
5. Mehul Bhatt, Hans Guesgen, Stefan Wölfl, and Shyamanta Hazarika. Qualitative Spatial and Temporal Reasoning: Emerging Applications, Trends, and Directions. *Spatial Cognition & Computation*, 11:1–14, 2011.
6. Andrei A. Bulatov and Peter Jeavons. An Algebraic Approach to Multi-sorted Constraints. In *CP*, 2003.
7. Andrés Cano and Serafín Moral. Heuristic Algorithms for the Triangulation of Graphs. In *IPMU*, 1994.
8. Khalil Challita. A semi-dynamical approach for solving qualitative spatial constraint satisfaction problems. *Theor. Comput. Sci.*, 440-441:29–38, 2012.
9. Assef Chmeiss and Jean-Francois Condotta. Consistency of Triangulated Temporal Qualitative Constraint Networks. In *ICTAI*, 2011.
10. W. Clocksin and C.S. Mellish. *Programming in Prolog*. Springer Berlin Heidelberg, 2003.

11. Jean-François Condotta and Christophe Lecoutre. A Class of df-Consistencies for Qualitative Constraint Networks. In *KR*, 2010.
12. Jean-François Condotta, Ali Mensi, Issam Nouaouri, Michael Sioutis, and Lamjed Ben Said. A Practical Approach for Maximizing Satisfiability in Qualitative Spatial and Temporal Constraint Networks. In *ICTAI*, 2015.
13. Ludwig Danzer, Branko Grünbaum, and Victor Klee. Helly's Theorem and Its Relatives. In *Convexity*, volume 7 of *Proceedings of Symposia in Pure Mathematics*. American Mathematical Society, 1963.
14. Daniel de Leng and Fredrik Heintz. Qualitative Spatio-Temporal Stream Reasoning with Unobservable Intertemporal Spatial Relations Using Landmarks. In *AAAI*, 2016.
15. Rina Dechter. From Local to Global Consistency. *Artif. Intell.*, 55:87–108, 1992.
16. Rina Dechter, Itay Meiri, and Judea Pearl. Temporal Constraint Networks. *Artif. Intell.*, 49:61–95, 1991.
17. Rina Dechter and Judea Pearl. Network-Based Heuristics for Constraint-Satisfaction Problems. *Artif. Intell.*, 34:1–38, 1987.
18. Yves Deville, Olivier Barette, and Pascal van Hentenryck. Constraint Satisfaction over Connected Row Convex Constraints. *Artif. Intell.*, 109:243–271, 1999.
19. Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate Texts in Mathematics*. Springer, 2012.
20. Krishna Sandeep Reddy Dubba, Anthony G. Cohn, David C. Hogg, Mehul Bhatt, and Frank Dylla. Learning Relational Event Models from Video. *J. Artif. Intell. Res.*, 53:41–90, 2015.
21. Frank Dylla, Till Mossakowski, Thomas Schneider, and Diedrich Wolter. Algebraic Properties of Qualitative Spatio-Temporal Calculi. In *COSIT*, 2013.
22. Baher A. El-Geresy, Alia I. Abdelmoty, and Christopher B. Jones. Episodes in Space: Qualitative Representation and Reasoning Over Spatio-Temporal Objects. *Int. J. Artif. Intell. Tools*, 9:131–152, 2000.
23. Andrew U. Frank. Qualitative Spatial Reasoning with Cardinal Directions. In *ÖGAI*, 1991.
24. D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, 15:835–855, 1965.
25. Alfonso Gerevini. Incremental qualitative temporal reasoning: Algorithms for the point algebra and the ord-horn class. *Artif. Intell.*, 166(1-2):37–80, 2005.
26. Alfonso Gerevini, Lenhart K. Schubert, and Stephanie Schaeffer. The Temporal Reasoning Tools TimeGraph I-II. *Int. J. Artif. Intell. Tools*, 4:281–300, 1995.
27. Martin Charles Golumbic and Ron Shamir. Complexity and Algorithms for Reasoning about Time: A Graph-Theoretic Approach. *J. ACM*, 40:1108–1133, 1993.
28. John Goodwin, Catherine Dolbear, and Glen Hart. Geographical Linked Data: The Administrative Geography of Great Britain on the Semantic Web. *T. GIS*, 12(Issue Supplement s1):19–30, 2008.
29. Roop K. Goyal and Max J. Egenhofer. Consistent Queries over Cardinal Directions across Different Levels of Detail. In *DEXA Workshop*, 2000.
30. P. Heggernes, Stanley C. Eisenstat, Gary Kumfert, and Alex Pothen. The computational complexity of the minimum degree algorithm. Technical report, ICASE, NASA Langley Research Center, 2001.
31. Fredrik Heintz and Daniel de Leng. Spatio-Temporal Stream Reasoning with Incomplete Spatial Information. In *ECAI*, 2014.
32. John C. Hogge. TPLAN: a temporal interval-based planner with novel extensions. Technical report, University of Illinois at Urbana-Champaign. Department of Computer Science UIUCDCS-R-87-1367., 1987.

33. Jinbo Huang. Compactness and its implications for qualitative spatial and temporal reasoning. In *KR*, 2012.
34. Philippe Jégou and Cyril Terrioux. Tree-Decompositions with Connected Clusters for Solving Constraint Networks. In *CP*, 2014.
35. Donald E. Knuth. *The Art of Computer Programming, Volume I: Fundamental Algorithms, 2nd Edition*. Addison-Wesley, 1973.
36. Shufeng Kong, Sanjiang Li, and Michael Sioutis. Exploring Directional Path-Consistency for Solving Constraint Networks. *The Computer Journal*, 2017.
37. Peter B. Ladkin and Alexander Reinefeld. Fast Algebraic Methods for Interval Constraint Problems. *Ann. Math. Artif. Intell.*, 19:383–411, 1997.
38. Sanjiang Li. On Topological Consistency and Realization. *Constraints*, 11:31–51, 2006.
39. Sanjiang Li, Zhiguo Long, Weiming Liu, Matt Duckham, and Alan Both. On redundant topological constraints. *Artif. Intell.*, 225:51–76, 2015.
40. Sanjiang Li and Huaiqing Wang. RCC8 binary constraint network can be consistently extended. *Artif. Intell.*, 170:1–18, 2006.
41. Gérard Ligozat. Tractable relations in temporal reasoning: pre-convex relations. In *ECAI Workshop on Spatial and Temporal Reasoning*, 1994.
42. Gerard Ligozat. A New Proof of Tractability for ORD-Horn Relations. In *AAAI/IAAI*, 1996.
43. Gerard Ligozat. Reasoning about cardinal directions. *J. Vis. Lang. Comput.*, 9:23–44, 1998.
44. Gérard Ligozat and Jochen Renz. What Is a Qualitative Calculus? A General Framework. In *PRICAI*, 2004.
45. Weiming Liu, Xiaotong Zhang, Sanjiang Li, and Mingsheng Ying. Reasoning about cardinal directions between extended objects. *Artif. Intell.*, 174:951–983, 2010.
46. Zhiguo Long. *Qualitative Spatial and Temporal Representation and Reasoning: Efficiency in Time and Space*. PhD thesis, University of Technology Sydney, 2017.
47. Zhiguo Long and Sanjiang Li. On Distributive Subalgebras of Qualitative Spatial and Temporal Calculi. In *COSIT*, 2015.
48. Zhiguo Long, Michael Sioutis, and Sanjiang Li. Efficient Path Consistency Algorithm for Large Qualitative Constraint Networks. In *IJCAI*, 2016.
49. C. Lutz and M. Milicic. A Tableau Algorithm for DLs with Concrete Domains and GCIs. *J. Autom. Reasoning*, 38:227–259, 2007.
50. Masoumeh Mansouri and Federico Pecora. More knowledge on the table: Planning with space, time and resources for robots. In *ICRA*, 2014.
51. Masoumeh Mansouri and Federico Pecora. A robot sets a table: a case for hybrid reasoning with different types of knowledge. *J. Exp. Theor. Artif. Intell.*, 28:801–821, 2016.
52. Ugo Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Inf. Sci.*, 7:95–132, 1974.
53. Bernhard Nebel. Solving Hard Qualitative Temporal Reasoning Problems: Evaluating the Efficiency of Using the ORD-Horn Class. *Constraints*, 1:175–190, 1997.
54. David A. Randell, Zhan Cui, and Anthony Cohn. A Spatial Logic Based on Regions & Connection. In *KR*, 1992.
55. Jochen Renz. Maximal Tractable Fragments of the Region Connection Calculus: A Complete Analysis. In *IJCAI*, 1999.
56. Jochen Renz. A Canonical Model of the Region Connection Calculus. *J. Appl. Non-Classical Logics*, 12:469–494, 2002.
57. Jochen Renz and Gérard Ligozat. Weak Composition for Qualitative Spatial and Temporal Reasoning. In *CP*, 2005.

58. Jochen Renz and Bernhard Nebel. Efficient Methods for Qualitative Spatial Reasoning. *J. Artif. Intell. Res.*, 15:289–318, 2001.
59. Jochen Renz and Bernhard Nebel. Qualitative Spatial Reasoning Using Constraint Calculi. In *Handbook of Spatial Logics*, pages 161–215. 2007.
60. D. J. Rose. A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. In *Graph Theory and Computing*, pages 183–217. Academic Press, 1972.
61. Michael Sioutis, Marjan Alirezaie, Jennifer Renoux, and Amy Loutfi. Towards a Synergy of Qualitative Spatio-Temporal Reasoning and Smart Environments for Assisting the Elderly at Home. In *IJCAI Workshop on Qualitative Reasoning*, 2017.
62. Michael Sioutis and Jean-François Condotta. Efficiently Enforcing Path Consistency on Qualitative Constraint Networks by Use of Abstraction. In *IJCAI*, 2017.
63. Michael Sioutis, Jean-François Condotta, and Manolis Koubarakis. An Efficient Approach for Tackling Large Real World Qualitative Spatial Networks. *Int. J. Artif. Intell. Tools*, 25:1–33, 2016.
64. Michael Sioutis and Manolis Koubarakis. Consistency of Chordal RCC-8 Networks. In *ICTAI*, 2012.
65. Michael Sioutis, Sanjiang Li, and Jean-François Condotta. Efficiently Characterizing Non-Redundant Constraints in Large Real World Qualitative Spatial Networks. In *IJCAI*, 2015.
66. Michael Sioutis, Zhiguo Long, and Sanjiang Li. Efficiently Reasoning about Qualitative Constraints through Variable Elimination. In *SETN*, 2016.
67. Michael Sioutis, Yakoub Salhi, and Jean-François Condotta. A Simple Decomposition Scheme For Large Real World Qualitative Constraint Networks. In *FLAIRS*, 2015.
68. Spiros Skiadopoulos and Manolis Koubarakis. On the consistency of cardinal direction constraints. *Artif. Intell.*, 163:91–135, 2005.
69. P. A. Story and Michael F. Worboys. A Design Support Environment for Spatio-Temporal Database Applications. In *COSIT*, 1995.
70. Robert Endre Tarjan and Mihalis Yannakakis. Simple Linear-Time Algorithms to Test Chordality of Graphs, Test Acyclicity of Hypergraphs, and Selectively Reduce Acyclic Hypergraphs. *SIAM J. Comput.*, 13:566–579, 1984.
71. Alfred Tarski. On the calculus of relations. *J. Symb. Log.*, 6:73–89, 1941.
72. Peter van Beek. Reasoning About Qualitative Temporal Information. *Artif. Intell.*, 58:297–326, 1992.
73. Peter van Beek and Dennis W. Manchak. The design and experimental analysis of algorithms for temporal reasoning. *J. Artif. Intell. Res.*, 4:1–18, 1996.
74. Marc B. Vilain and Henry A. Kautz. Constraint Propagation Algorithms for Temporal Reasoning. In *AAAI*, 1986.
75. Matthias Westphal. *Qualitative Constraint-based Reasoning: Methods and Applications*. PhD thesis, Albert-Ludwigs-Universität Freiburg, 2014.
76. Nevin Zhang and David Poole. A simple approach to Bayesian network computations. In *Proceedings of the* $10^{th}$ *Canadian Conference on Artificial Intelligence*, 1994.
77. Yuanlin Zhang and Satyanarayana Marisetti. Solving connected row convex constraints by variable elimination. *Artif. Intell.*, 173:1204–1219, 2009.