Collective Singleton-based Consistency for Qualitative Constraint Networks: Theory and Practice

Michael Sioutis[a,*], Anastasia Paparrizou[b], Jean-François Condotta[b]

[a]*Aalto University, Department of Computer Science, Espoo, Finland*
[b]*Artois University, CRIL UMR 8188, Lens, France*

Abstract

Partial singleton weak path-consistency, or partial $\blacklozenge$-consistency for short, is essential for tackling challenging fundamental reasoning problems associated with qualitative constraints networks. Briefly put, partial $\blacklozenge$-consistency ensures that each base relation of each of the constraints of a qualitative constraint network can define a singleton relation in its corresponding partially weakly path-consistent, or partially $\diamond$-consistent for short, subnetwork. In this paper, we propose a stronger local consistency that couples $\blacklozenge$-consistency with the idea of collectively deleting certain unfeasible base relations by exploiting singleton checks. We then propose an algorithm for enforcing this new consistency and a lazy variant of that algorithm for approximating the new consistency that, given a qualitative constraint network, both outperform the respective algorithm for enforcing partial $\blacklozenge$-consistency in that network. With respect to the lazy algorithmic variant in particular, we show that it runs up to 5 times faster than our original exhaustive algorithm whilst exhibiting very similar pruning capability. We formally prove certain properties of our new local consistency and our algorithms, and motivate their usefulness through demonstrative examples and a thorough experimental evaluation with random qualitative constraint networks of the Interval Algebra and the Region Connection Calculus from the phase transition region of two different generation models. Finally, we provide

*Corresponding author

Email addresses: `michael.sioutis@aalto.fi` (Michael Sioutis), `paparrizou@cril.fr` (Anastasia Paparrizou), `condotta@cril.fr` (Jean-François Condotta)

Preprint submitted to Theoretical Computer Science *March 19, 2019*

evidence of the crucial role of the new consistency in tackling the minimal labeling problem of a qualitative constraint network, which is the problem of finding the strongest implied constraints of that network.

1. Introduction

Qualitative Spatial and Temporal Reasoning (QSTR) is a major field of study in Artificial Intelligence, and in particular in Knowledge Representation & Reasoning, that deals with the fundamental cognitive concepts of space and time in an abstract, qualitative, manner. In a sense, this approach is in line with the qualitative abstractions of spatial and temporal aspects of the common-sense background knowledge on which the human perspective of physical reality is based. For instance, in natural language one uses expressions such as *inside*, *before*, and *north of* to spatially or temporally relate one object with another object or oneself, without resorting to providing quantitative information about these entities. More formally, QSTR restricts the vocabulary of rich mathematical theories that deal with spatial and temporal entities to simple qualitative constraint languages. Thus, QSTR provides a concise framework that allows for rather inexpensive reasoning about entities located in space and time and, hence, further boosts research and applications to a plethora of areas and domains that include, but are not limited to, dynamic GIS [1], cognitive robotics [2], deep learning [3], spatio-temporal design [4], qualitative model generation from video [5], and ambient intelligence [6]. The interested reader may look into a more comprehensive review of the emerging applications, the trends, and the future directions of QSTR in [7]. In addition, a detailed survey of qualitative spatial and temporal calculi appears in [8].

As an illustration, the first constraint language to deal with time in a qualitative manner was proposed by Allen in [9], called Interval Algebra. Allen wanted to define a framework for reasoning about time in the context of natural language processing that would be reliable and efficient enough for reasoning about temporal information in a qualitative manner. In particular, Interval

Algebra uses intervals on the timeline to represent entities corresponding to actions, events, or tasks. Interval Algebra has become one of the most well-known qualitative constraint languages, due to its use for representing and reasoning about temporal information in various applications. Specifically, typical applications of Interval Algebra involve planning and scheduling [10, 11, 12, 13, 14], natural language processing [15, 16], temporal databases [17, 18], multimedia databases [19], molecular biology [20] (e.g., arrangement of DNA segments/intervals along a linear chain involves particular temporal-like problems [21]), and workflow [22].

As another illustration, inspired by the success of Interval Algebra, Randell et al. developed the Region Connection Calculus (RCC) in [23], which studies the different relations that can be defined between regions in some topological space; these relations are based on the primitive relation of connection. For example, the relation *disconnected* between two regions x and y suggests that none of the points of region x connects with a point of region y, and vice versa. Two fragments of RCC, namely, RCC8 and RCC5 (a sublanguage of RCC8 where no significance is attached to boundaries of regions), have been used in several real-life applications. In particular, Bouzy in [24] used RCC8 in programming the Go game, Lattner et al. in [25] used RCC5 to set up assistance systems in intelligent vehicles, Heintz et al. in [26] used RCC8 in the domain of autonomous *unmanned aircraft systems* (UAS), and Randell et al. in [27] used a particular discrete domain counterpart of RCC8 (called *discrete meterotopology*) to correct segmentation errors for images of hematoxylin and eosin (H&E)-stained human carcinoma cell line cultures. Other typical applications of RCC involve robot navigation [28, 29, 30], computer vision [31], and natural language processing [32, 33].

The problem of representing and reasoning about qualitative spatial or temporal information can be modeled as a qualitative constraint network (QCN) using a qualitative constraint language, such as one of those that we presented earlier, namely, RCC8 or Interval Algebra respectively. Specifically, a QCN is a network of constraints corresponding to qualitative spatial or temporal relations between spatial or temporal variables respectively, and the qualitative constraint language of choice is used to ground those constraints on a finite set of binary relations, called *base relations* (or *atoms*) [34]. In the case of Interval Algebra (IA) for example, which considers time intervals (as its temporal entities), each of its base relations represents an ordering of the four endpoints of two intervals in the timeline (e.g., *during*). Likewise, in the case of RCC8 each base relation corresponds to a spatial configuration that can hold between two regions in some topological space (e.g., *partially overlapping*). More details of the aforementioned notions are provided later on in Section 2.

The fundamental reasoning problems associated with a given QCN $\mathcal{N}$ are the problems of *satisfiability checking*, *minimal labeling* (or *deductive closure*), and *redundancy* (or *entailment*) [35]. In particular, the satisfiability checking problem is the problem of deciding if there exists a spatial or temporal valuation of the variables of $\mathcal{N}$ that satisfies its constraints, such a valuation being called a *solution* of $\mathcal{N}$, the minimal labeling 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 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 well-known qualitative constraint languages the satisfiability checking problem is NP-hard [36]. Further, the redundancy problem, the minimal labeling problem, and the satisfiability checking problem are equivalent under polynomial Turing reductions [20].

Many of the published works that study the aforementioned reasoning problems, use *partial $\diamond$-consistency*[1] as a means to define practical algorithms for efficiently tackling them [38, 39, 40, 41, 42, 43, 44]. Given a QCN $\mathcal{N}$ and a graph G, partial $\diamond$-consistency with respect to G, denoted by $\overset{\diamond}{_G}$-consistency, entails (weak) consistency for all triples of variables in $\mathcal{N}$ that correspond to

[1] Note that $\diamond$-consistency can be interpreted as *weak* path-consistency, i.e., path-consistency where the (true) composition of relations is replaced by *weak composition* [37] (a notion that will be formally defined in Section 2).

three-vertex cycles (triangles) in G. We note that if G is complete, $\underset{G}{\diamond}$-consistency becomes identical to $\diamond$-*consistency* [37]. Hence, $\diamond$-consistency is a special case of $\underset{G}{\diamond}$-consistency. In fact, earlier works have relied solely on $\diamond$-consistency; it was not until the introduction of chordal (or triangulated) graphs in QSTR, due to some generalized theoretical results of [45], that researchers started restricting $\diamond$-consistency to a triangulation (or chordal completion) of the constraint graph of an input QCN and benefiting from better complexity properties in more recent works. The importance of $\underset{G}{\diamond}$-consistency, and any other local consistency that uses $\underset{G}{\diamond}$-consistency as its basis, lies in the fact that it can be directly utilized to decide the satisfiability of QCNs that are defined over particular subclasses of qualitative spatial or temporal relations, called *tractable* classes of relations, it can be applied on a QCN as a preprocessing step to remove impossible, unfeasible, base relations and simplify the problem, and it can also be incorporated in a look-ahead subprocedure in backtracking algorithms whenever the use of such algorithms is appropriate (e.g., in the general case where a QCN is not defined over a tractable class of relations).

Motivation and Contributions

Adding to what has already been written, and with respect to the satisfiability checking problem in particular, the literature suggests that $\underset{G}{\diamond}$-consistency alone is sufficient in most cases to guarantee that a solution for a given QCN can be efficiently obtained, provided that the QCN is satisfiable (see also [46]). However, for the more challenging problems of minimal labeling and redundancy, a stronger local consistency is typically employed that builds upon $\underset{G}{\diamond}$-consistency, called singleton $\underset{G}{\diamond}$-consistency and denoted by $\underset{G}{\blacklozenge}$-consistency, as solving either of these problems benefits from concise representations of QCNs throughout the execution of the related algorithms [38, 39]. The need and usefulness of this consistency is especially apparent in the work of [38] for the minimal labeling problem, where $\underset{G}{\blacklozenge}$-consistency is repetitively utilized in the execution of the algorithm that is presented there in order to remove as many unfeasible (i.e., not corresponding to a solution) base relations as possible and, hence, drastically refine the QCN at hand. Simply put, given a QCN $\mathcal{N}$ and a graph G, $\blacklozenge_G$-consistency holds on $\mathcal{N}$ if and only if each base relation of each of the constraints of $\mathcal{N}$ is closed under $\lozenge_G$-consistency, i.e., after instantiating any constraint of that network with one of its base relations b and closing the network under $\lozenge_G$-consistency, the corresponding constraint in the $\lozenge_G$-consistent subnetwork will continue being defined by b.

It is then natural to ask whether we can have an even stronger local consistency than $\blacklozenge_G$-consistency (and $\lozenge_G$-consistency) for QCNs that can also be enforced more efficiently than $\blacklozenge_G$-consistency, as a positive answer to that question would suggest an immediate improvement for any algorithm that currently employs $\blacklozenge_G$-consistency. In this paper, we make the following contributions towards obtaining a positive answer to that question.

- We enrich the family of consistencies for QCNs by proposing a new singleton style consistency inspired by k-partitioning consistency for constraint satisfaction problems (CSPs) [47]. This filtering technique is based on domain partitioning combined with a local consistency, typically *arc consistency* [48], and allows for improved pruning capability over *singleton arc consistency* [49]. Similarly to k-partitioning consistency, our new consistency, denoted by $\blacklozenge_G^{\cup}$-consistency, combines singleton checks and $\lozenge_G$-consistency to present itself as a better alternative to $\blacklozenge_G$-consistency.

- With respect to our new consistency, we also propose an algorithm for applying it on a given QCN, which turns out being more efficient than the respective algorithm for applying $\blacklozenge_G$-consistency on that same QCN. As a brief intuitive explanation of this, $\blacklozenge_G^{\cup}$-consistency allows for proactively eliminating base relations anywhere in a given QCN and not only in the set of base relations of the constraint at hand that is singleton checked.

- Further, we obtain several theoretical results regarding our new local consistency and our novel algorithm for enforcing it, and show, among other things, that $\blacklozenge_G^{\cup}$-consistency is *strictly stronger* than $\blacklozenge_G$-consistency and, hence, than $\lozenge_G$-consistency.

- Moreover, based on our new algorithm, we present and thoroughly study a lazy algorithmic variant for approximating $\blacklozenge_G^{\cup}$-consistency that restricts singleton checks to constraints that are likely to lead to the removal of a base relation upon their propagation in a given QCN. We show that this variant runs up to 5 times faster than our original exhaustive algorithm whilst exhibiting similar pruning capability for the involved datasets here.

- In addition, we perform a thorough experimental evaluation of the algorithms for enforcing $\blacklozenge_G^{\cup}$-consistency and $\blacklozenge_G$-consistency, and the lazy algorithmic variant for approximating $\blacklozenge_G^{\cup}$-consistency, using random QCNs of two different calculi, viz., the Interval Algebra and the Region Connection Calculus [23], from the phase transition region of two different generation models. The results support our argument that $\blacklozenge_G^{\cup}$-consistency can be enforced more efficiently than $\blacklozenge_G$-consistency for a given QCN and, as mentioned earlier, are very complimentary of our lazy algorithmic variant for approximating $\blacklozenge_G^{\cup}$-consistency (and $\blacklozenge_G$-consistency).

- Finally, we provide evidence of the utility of the new consistency in tackling the minimal labeling problem of a qualitative constraint network, which, as a reminder, is the problem of finding the strongest implied constraints of that network, and also explore how it fares against the more straightforward problem of satisfiability checking.

Related Work

This work is inspired from similar works that exist for the constraint satisfaction problem (CSP). From the early beginning since the last decade, the CP community has put a noticeable research effort in proposing strong consistencies as alternatives to the classic and standard arc consistency (AC). Strong consistencies though, despite their pruning capacity, and thus the space reduction, were considered prohibitive in practice due to their operational cost. In recent years, many of these consistencies have been revisited, and new data structures that allow fast memory access make them much more competitive than in the past.

The most representative examples are the singleton arc consistency (SAC) [49] and the 1-Partition-AC or partition one AC (POAC) [47]. Also, weaker forms of SAC and POAC, such as NSAC [50, 51], or approximations of them [52] have been developed. We believe that qualitative spatial and temporal reasoning can also benefit from similar consistencies. Such consistencies could allow us to solve existing problems faster or reveal new challenges for the field.

The consistency proposed here is inspired by the k-partitioning consistency family. The k-partitioning consistency family is defined by fixing k and the level of consistency. If arc consistency is the underlying local consistency then we have the k-Partition-AC. For $k = 1$ we obtain 1-Partition-AC (aka partition one AC or POAC, which was mentioned earlier), which is the most used consistency from this family. POAC splits a domain into singleton sub-domains. For each such singleton, AC is applied and the produced domains are recorded. After this step, a union operation on the domains allows the respective algorithm to remove values that do not appear in the union and, by consequence, not in any problem solution either. The last operation makes POAC strictly stronger than SAC. Here we have to mention that due to the non-bidirectionality of supports for SAC, there exist another consistency that benefits from this property of SAC to achieve extra pruning. Interestingly, it is also stronger than POAC and is called BiSAC (bidirectional SAC) [53].

The rest of the paper is organized as follows. In Section 2 we give some preliminaries on qualitative spatial and temporal reasoning, and in Section 3 we focus on $\diamond_G$-consistency and $\blacklozenge_G$-consistency and, in particular, recall some related results from the literature, but also provide some new results of our own. Then, in Section 4 we introduce, formally define, and thoroughly study our new local consistency, namely, $\blacklozenge_G^{\cup}$-consistency. In Section 5 we present an algorithm for efficiently applying $\blacklozenge_G^{\cup}$-consistency on a given QCN $\mathcal{N}$, and in Section 6 we present a lazy algorithmic variant of that algorithm for efficiently approximating $\blacklozenge_G^{\cup}$-consistency on $\mathcal{N}$. In Section 7 we evaluate the algorithms for enforcing $\blacklozenge_G^{\cup}$-consistency and $\blacklozenge_G$-consistency and the lazy algorithmic variant for approximating $\blacklozenge_G^{\cup}$-consistency on a given QCN, and, finally, in Section 8 we conclude the paper and give some directions for future work.

2. Preliminaries

A (binary) qualitative spatial or temporal constraint language, is based on a finite set B of *jointly exhaustive and pairwise disjoint* relations defined over an infinite domain D, which is called the set of *base relations* [34]. 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 level of granularity provided by the domain D. The set B contains the identity relation Id, and is closed under the *converse* operation ($^{-1}$). Indefinite knowledge can be specified by a union of possible base relations, and is represented by the set containing them. Hence, 2^{B} represents the total set of relations. The set 2^{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 $\diamond$ [34]. 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 (true) composition of b and b'. For all $r, r' \in 2^{\mathsf{B}}$, we have that $r \diamond r' = \bigcup\{b \diamond b' \mid b \in r, b' \in r'\}$.

As an illustration, consider the well-known qualitative temporal constraint language of Interval Algebra (IA) introduced by Allen [9]. IA considers time intervals (as its temporal entities) and makes use of the temporal relations *precedes* (p), *precedes inverse* (pi), *meets* (m), *meets inverse* (mi), *overlaps* (o), *overlaps inverse* (oi), *starts* (s), *starts inverse* (si), *during* (d), *during inverse* (di), *finishes* (f), *finishes inverse* (fi), and *equals* (eq) to encode knowledge about the temporal relations between intervals on the timeline. These temporal relations constitute the set of base relations $\mathsf{B} = \{eq, p, pi, m, mi, o, oi, s, si, d, di, f, fi\}$ of IA, where each base relation of IA represents a particular ordering

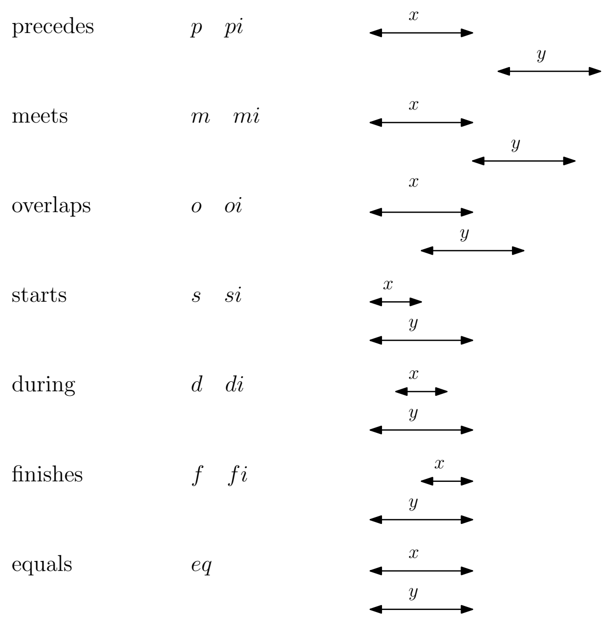


Figure 1: The base relations of IA, with $\cdot i$ denoting the converse of $\cdot$

of the four endpoints of two intervals on the timeline, and eq is the identity relation Id of IA. The base relations of IA are depicted in Figure 1.

As another illustration, the Region Connection Calculus (RCC) is a first-order theory for representing and reasoning about mereotopological information [23]. The domain D of RCC comprises all possible non-empty regular subsets of some topological space. These subsets do not have to be internally connected and do not have a particular dimension, but are commonly required to be regular *closed* [54]. In particular, a fragment of RCC, called RCC8, 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. These spatial relations constitute the set of base relations $\mathsf{B} = \{EQ, DC, EC, PO, TPP, TPPi, NTPP, NTPPi\}$ of RCC8, where each base relation of RCC8 represents a particular topological

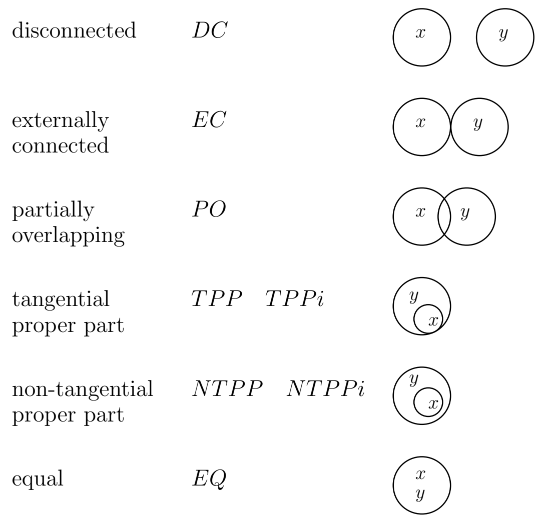


Figure 2: The base relations of RCC8, with $\cdot i$ denoting the converse of $\cdot$

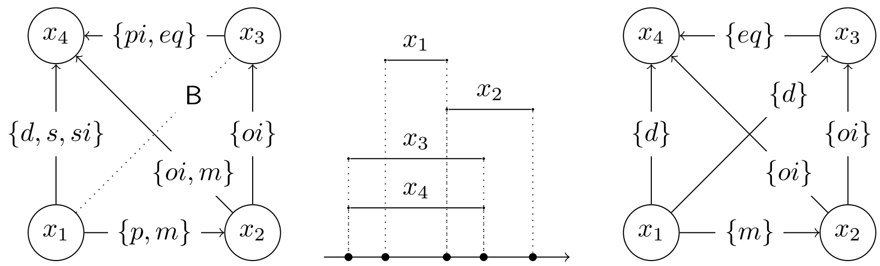
configuration of two regions in some topological space, and EQ is the identity relation Id of RCC8. The base relations of RCC8 are depicted in Figure 2.

Other notable and well-known qualitative spatial and temporal constraint languages include Point Algebra [55], Cardinal Direction Calculus [56, 57], and Block Algebra [58], and Cardinal Direction Calculus for extended objects [59, 60, 61].

The weak composition operation $\diamond$, the converse operation $^{-1}$, the union operation $\cup$, the complement operation $^{\complement}$, and the total set of relations 2^{B} along with the identity relation Id of a qualitative constraint language, form an algebraic structure $(2^{\mathsf{B}}, \mathsf{Id}, \diamond, ^{-1}, ^{\complement}, \cup)$ that can correspond to a *relation algebra* in the sense of Tarski [62].

Proposition 1 ([36])**.** *The languages of Point Algebra, Cardinal Direction Calculus, Interval Algebra, Block Algebra, and* RCC8 *are each a relation algebra with the algebraic structure* $(2^{\mathsf{B}}, \mathsf{Id}, \diamond, ^{-1}, ^{\complement}, \cup)$.

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



(a) A satisfiable QCN $\mathcal{N}$ (b) A solution σ of $\mathcal{N}$ (c) A scenario $\mathcal{S}$ of $\mathcal{N}$

Figure 3: Figurative examples of QCN terminology using IA

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

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

- $V = \{v_1, \ldots, v_n\}$ is a non-empty finite set of variables, each representing an entity of an infinite domain D;
- and C is a mapping $C : V \times V \to 2^{\mathsf{B}}$ such that $C(v, v) = \{\mathsf{Id}\}$ for all $v \in V$ and $C(v, v') = (C(v', v))^{-1}$ for all $v, v' \in V$, where $\bigcup \mathsf{B} = \mathsf{D} \times \mathsf{D}$.

An example of a QCN of IA is shown in Figure 3a; for simplicity, converse relations as well as Id loops are not mentioned or shown in the figure.

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

- a *solution* of $\mathcal{N}$ is a mapping $\sigma : V \to \mathsf{D}$ such that $\forall(u, v) \in V \times V$, $\exists b \in C(u, v)$ such that $(\sigma(u), \sigma(v)) \in b$ (see Figure 3b);
- $\mathcal{N}$ is *satisfiable* iff it admits a solution;
- a QCN is *equivalent* to $\mathcal{N}$ iff it admits the same set of solutions as $\mathcal{N}$;
- a *sub*-QCN[2] $\mathcal{N}'$ of $\mathcal{N}$, denoted by $\mathcal{N}' \subseteq \mathcal{N}$, is a QCN (V, C') such that

[2]This term can also be found by the name of *refined* QCN or simply *refinement* throughout the literature.

$C'(u,v) \subseteq C(u,v)\ \forall u,v \in V$; if in addition $\exists u,v \in V$ such that $C'(u,v) \subset C(u,v)$, then $\mathcal{N}' \subset \mathcal{N}$;

- $\mathcal{N}$ 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 *scenario* $\mathcal{S}$ of $\mathcal{N}$ is an atomic satisfiable sub-QCN of $\mathcal{N}$ (see Figure 3c);
- a base relation $b \in C(v,v')$ with $v,v' \in V$ is *feasible* (resp. *unfeasible*) in $\mathcal{N}$ iff there exists (resp. there does not exist) a scenario $\mathcal{S} = (V, C')$ of $\mathcal{N}$ such that $C'(v,v') = \{b\}$;
- $\mathcal{N}$ is *minimal* iff $\forall v,v' \in V$ and $\forall b \in C(v,v')$, b is a feasible base relation of $\mathcal{N}$;
- the *constraint graph* of $\mathcal{N}$, denoted by $\mathsf{G}(\mathcal{N})$, is the graph (V,E) where $\{u,v\} \in E$ iff $C(u,v) \neq \mathsf{B}$ and $u \neq v$;
- $\mathcal{N}$ is *trivially inconsistent* iff $\exists u,v \in V$ such that $C(u,v) = \emptyset$;
- $\mathcal{N}$ is the *empty* QCN on V, denoted by $\bot^V$, iff $C(u,v) = \emptyset$ for all $u,v \in V$.

Let us further introduce the following operations with respect to QCNs:

- given a QCN $\mathcal{N} = (V,C)$ and $v,v' \in V$, we define that $\mathcal{N}_{[v,v']/r}$ with $r \in 2^{\mathsf{B}}$ yields the QCN $\mathcal{N}' = (V,C')$ defined by $C'(v,v') = r$, $C'(v',v) = r^{-1}$ and $C'(y,w) = C(y,w)\ \forall (y,w) \in (V \times V) \setminus \{(v,v'),(v',v)\}$;
- given two QCNs $\mathcal{N} = (V,C)$ and $\mathcal{N}' = (V,C')$ on the same set of variables V, we define that $\mathcal{N} \cup \mathcal{N}'$ yields the QCN $\mathcal{N}'' = (V,C'')$, where $C''(v,v') = C(v,v') \cup C'(v,v')$ for all $v,v' \in V$.

We recall the following definition of $\diamond_G$-consistency, which, as noted in the introduction, is the basic local consistency used in the literature for solving fundamental reasoning problems of QCNs, such as the satisfiability checking problem.

Definition 3. Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, $\mathcal{N}$ is said to be $\diamond_G$*-consistent* iff $\forall \{v_i, v_j\}, \{v_i, v_k\}, \{v_k, v_j\} \in E$ we have that $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$.

We note again that if G is complete, $\diamond_G$-consistency becomes identical to $\diamond$-consistency [37], and, hence, $\diamond$-consistency is a special case of $\diamond_G$-consistency.

Given a graph $G = (V, E)$, a QCN $\mathcal{N} = (V, C)$ is $\blacklozenge_G$-consistent iff for every pair of variables $\{v, v'\} \in E$ and every base relation $b \in C(v, v')$, after instantiating $C(v, v')$ with $\{b\}$ and applying $\diamond_G$-consistency on $\mathcal{N}$, the revised constraint $C(v, v')$ is always supported by $\{b\}$. Formally, $\blacklozenge_G$-consistency of a QCN is defined as follows:

Definition 4. Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, $\mathcal{N}$ is said to be $\blacklozenge_G$*-consistent* iff $\forall \{v, v'\} \in E$ and $\forall b \in C(v, v')$ we have that $\{b\} = C'(v, v')$, where $(V, C') = \diamond_G(\mathcal{N}_{[v,v']/\{b\}})$.

An example of a $\blacklozenge_G$-consistent QCN is shown in Figure 5a later on. If G is a complete graph, i.e., $G = K_V$, we can easily verify that $\blacklozenge_G$-consistency corresponds to $\diamond_{\mathsf{B}}$-consistency of the family of $\diamond_f$-consistencies studied in [46]. Interestingly, $\blacklozenge_G$-consistency can also be seen as a counterpart of singleton arc consistency (SAC) [49] for QCNs. Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, for every $b \in \mathsf{B}$ and every $\{v, v'\} \in E$, we will say that b is $\blacklozenge_G$-consistent for $C(v, v')$ iff $\{b\} = C'(v, v')$, where $(V, C') = \diamond_G(\mathcal{N}_{[v,v']/\{b\}})$.

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

Next, we recall the definition of a ditributive subclass of relations.

Definition 6. Given three relations r, r', r'' of a subclass $\mathcal{A}$, 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 $(r' \cap r'') \diamond r = (r' \diamond r) \cap (r'' \diamond r)$. A subclass $\mathcal{A}$ is *distributive* iff weak composition distributes over non-empty intersection $\forall r, r', r'' \in \mathcal{A}$.

Distributive subclasses of relations are defined for all of the qualitative constraint languages mentioned in Proposition 1 [63]; in particular, for all those calculi the closures of their sets of base relations under weak composition, intersection, and converse (denoted by $\hat{\mathsf{B}}$) are distributive.

Finally, for the sake of simplicity in recalling and phrasing some results, in what follows we assume that all considered graphs are *biconnected* [64] (referred to as 2-connected in [64]). Specifically, given a QCN $\mathcal{N}$ and a graph G on a set of variables V, this assumption on G is useful for the sole purpose of allowing algorithms that apply ${}^{\diamond}_{G}$-consistency (and other local consistencies that use ${}^{\diamond}_{G}$-consistency as their basis) to propagate the assignment of the empty relation throughout the given graph G, whenever such an assignment occurs in the first place. This assumption will become relevant later on when we will present Propositions 3 and 4 in Section 3 and Proposition 10 in Section 4, at which points we will make explicit remarks.

3. A closer look at ${}^{\diamond}_{G}$-consistency and ${}^{\blacklozenge}_{G}$-consistency

Let us come back to ${}^{\diamond}_{G}$-consistency and ${}^{\blacklozenge}_{G}$-consistency and recall in this section some results from the literature that will be relevant in the rest of the paper, but also provide some new results of our own. Please note that we view a consistency ${}^{\phi}_{G}$, where ϕ is some operation and G a graph, as a predicate on QCNs, i.e., as a function that receives an input QCN and returns true or false depending on whether ${}^{\phi}_{G}$ holds on that QCN or not respectively.

In order to compare the pruning (or inference) capability of different consistencies, we introduce a preorder. Let ${}^{\phi}_{G}$ and ${}^{\psi}_{G}$ be two consistencies defined by some operations ϕ and ψ respectively and a graph G. Then, ${}^{\phi}_{G}$ is *stronger* than ${}^{\psi}_{G}$, denoted by ${}^{\phi}_{G} \trianglerighteq {}^{\psi}_{G}$, iff whenever ${}^{\phi}_{G}$ holds on a QCN $\mathcal{N}$ with respect to a graph G, ${}^{\psi}_{G}$ also holds on $\mathcal{N}$ with respect to G, and ${}^{\phi}_{G}$ is *strictly stronger* than ${}^{\psi}_{G}$, denoted by ${}^{\phi}_{G} \triangleright {}^{\psi}_{G}$, iff ${}^{\phi}_{G} \trianglerighteq {}^{\psi}_{G}$ and there exists at least one QCN $\mathcal{N}$ and a graph G such that ${}^{\psi}_{G}$ holds on $\mathcal{N}$ with respect to G but ${}^{\phi}_{G}$ does not hold on $\mathcal{N}$ with respect to G. Finally, ${}^{\phi}_{G}$ and ${}^{\psi}_{G}$ are *equivalent*, denoted by ${}^{\phi}_{G} \equiv {}^{\psi}_{G}$, iff we have both ${}^{\phi}_{G} \trianglerighteq {}^{\psi}_{G}$ and ${}^{\psi}_{G} \trianglerighteq {}^{\phi}_{G}$.[3]

We now recall the definition of a *well-behaved* consistency [46].

Definition 7. A consistency ${}^{\phi}_{G}$ is *well-behaved* iff for any QCN $\mathcal{N} = (V, C)$ and any graph $G = (V, E)$ the following properties hold:

- there exists a *unique* largest (w.r.t. $\subseteq$) ${}^{\phi}_{G}$-consistent sub-QCN of $\mathcal{N}$, denoted by ${}^{\phi}_{G}(\mathcal{N})$ and referred to as the ${}^{\phi}_{G}$-closure of $\mathcal{N}$ w.r.t. G *(Dominance)*;
- ${}^{\phi}_{G}(\mathcal{N})$ is equivalent to $\mathcal{N}$ *(Equivalence)*.[4]

We can obtain the following theorem:

Theorem 1. *If the property of dominance holds for a consistency ${}^{\phi}_{G}$, then for any* QCN *$\mathcal{N} = (V, C)$ and any graph $G = (V, E)$ the following properties hold:*

- ${}^{\phi}_{G}({}^{\phi}_{G}(\mathcal{N})) = {}^{\phi}_{G}(\mathcal{N})$ (Idempotence)*;*
- *if $\mathcal{N}' \subseteq \mathcal{N}$ then ${}^{\phi}_{G}(\mathcal{N}') \subseteq {}^{\phi}_{G}(\mathcal{N})$* (Monotonicity)*.*

Proof. (Idempotence) Let $\mathcal{N} = (V, C)$ be a QCN, and $G = (V, E)$ a graph. Then, ${}^{\phi}_{G}(\mathcal{N})$ is ${}^{\phi}_{G}$-consistent. Now, by dominance of ${}^{\phi}_{G}$-consistency the largest ${}^{\phi}_{G}$-consistent sub-QCN of ${}^{\phi}_{G}(\mathcal{N})$ is itself and, hence, ${}^{\phi}_{G}({}^{\phi}_{G}(\mathcal{N})) = {}^{\phi}_{G}(\mathcal{N})$. (Monotonicity) Let $\mathcal{N} = (V, C)$ and $\mathcal{N}' = (V, C')$ be two QCNs such that $\mathcal{N}' \subseteq \mathcal{N}$, and $G = (V, E)$ a graph. As $\mathcal{N}' \subseteq \mathcal{N}$, we have that ${}^{\phi}_{G}(\mathcal{N}')$ is a ${}^{\phi}_{G}$-consistent sub-QCN of $\mathcal{N}$. In addition, by dominance of ${}^{\phi}_{G}$-consistency we can assert that ${}^{\phi}_{G}(\mathcal{N})$ is the largest ${}^{\phi}_{G}$-consistent sub-QCN of $\mathcal{N}$. Thus, we have that ${}^{\phi}_{G}(\mathcal{N}') \subseteq {}^{\phi}_{G}(\mathcal{N})$. $\square$

It is routine to formally prove the following result for ${}^{\diamond}_{G}$-consistency:

Corollary 1 (cf. [46])**.** *We have that ${}^{\diamond}_{G}$-consistency is well-behaved.*

It is routine to formally prove the following result for ${}^{\blacklozenge}_{G}$-consistency as well:

[3] Note that this notion of equivalence concerning consistencies is not to be confused with the notion of equivalence concerning QCNs; this latter notion has been defined in Definition 2.

[4] Note that in [46] the property of equivalence is not explicitly considered in what the authors refer to as a well-behaved consistency (although all of their defined consistencies are characterized by that property), but we think that it is a reasonable property to include in the definition of a well-behaved consistency.

Corollary 2 (cf. [46])**.** *We have that* $\blacklozenge_G$*-consistency is well-behaved.*

The aforementioned two results are derived from respective results of [46] where complete graphs are used in all cases. The generalization to an arbitrary graph G is trivial.

We recall the following general result regarding the pruning capability of $\diamond_G$-consistency in comparison with that of $\blacklozenge_G$-consistency:

Proposition 2 ([46])**.** *We have that* $\blacklozenge_G$*-consistency* $\triangleright$ $\diamond_G$*-consistency.*

The next result shows the link between $\diamond_G$-consistency and minimal QCNs:

Proposition 3 ([65])**.** *Let* $\mathcal{A}$ *be a distributive subclass of relations of a relation algebra with the property that any atomic* QCN *over* $\mathcal{A}$ *that is* $\diamond$*-consistent is satisfiable. Then, for any* QCN $\mathcal{N} = (V, C)$ *over* $\mathcal{A}$ *and any chordal graph* $G = (V, E)$ *such that* $\mathsf{G}(\mathcal{N}) \subseteq G$*, we have that* $\forall\{u, v\} \in E$ *and* $\forall b \in C'(u, v)$*, where* $(V, C') = \diamond_G(\mathcal{N})$*, the base relation* b *is feasible.*

It should be noted that the aforementioned proposition is essentially obtained from the merge of Proposition 3 and Theorem 4 as they appear in [65], and is presented informally in [65] in the form of the following statement: *"The last two results (namely, Proposition* 3 *and Theorem* 4 *in [65]) show that given a* QCN *that is defined over a distributive subclass of relations, enforcing partial* $\diamond$*-consistency on it is able to make the relations corresponding to the edges of a triangulation of its constraint graph become minimal."*

The property described in Proposition 3 is satisfied by all of the qualitative constraint languages mentioned in Proposition 1 [36].

Finally, the following result shows the connection between $\blacklozenge_G$-consistency and minimal QCNs:

Proposition 4 (cf. [38])**.** *Let* $\mathcal{A}$ *be a subclass of relations of a relation algebra with the property that for any* QCN $\mathcal{N} = (V, C)$ *over* $\mathcal{A}$ *there exists a graph* $G = (V, E)$ *such that, if* $\diamond_G(\mathcal{N})$ *is not trivially inconsistent, then* $\mathcal{N}$ *is satisfiable. Then, for any such* $\mathcal{N}$ *and* G*, we have that* $\forall\{u, v\} \in E$ *and* $\forall b \in C'(u, v)$*, where* $(V, C') = \blacklozenge_G(\mathcal{N})$*, the base relation* b *is feasible.*

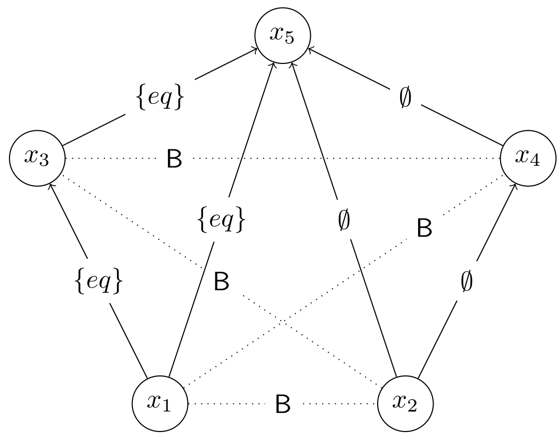


Figure 4: Given the unsatisfiable QCN $\mathcal{N} = (V, C)$ of IA above (which is defined over a distributive subclass of relations of IA) and the non-biconnected (separable) graph G with the edge set $\{\{x_1, x_3\}, \{x_1, x_5\}, \{x_3, x_5\}, \{x_2, x_4\}, \{x_2, x_5\}, \{x_4, x_5\}\}$, we have that $\mathcal{N}$ is $\overset{\diamond}{_G}$-consistent (and $\overset{\blacklozenge}{_G}$-consistent) and that the base relation eq in constraints $C(x_1, x_3)$, $C(x_1, x_5)$, and $C(x_3, x_5)$ is unfeasible in $\mathcal{N}$; this is an example of the pathological case described in Remark 1

It should be noted that the aforementioned proposition appears as point (2) under Proposition 4 in [38], and we have generalized its wording here to make it applicable to the case where $\overset{\diamond}{_G}$-consistency is complete for deciding the satisfiability of a given QCN (indeed, the property of $\overset{\diamond}{_G}$-consistency being complete for deciding satisfiability is what is used to prove Proposition 4 in [38]).

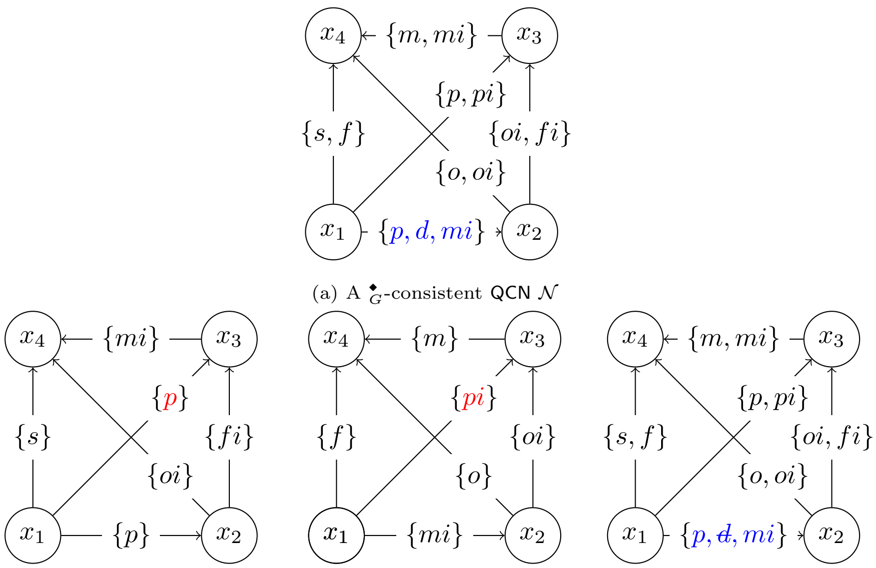
It is important to make the following remark:

Remark 1. *As noted in Section 2, all our graphs are assumed to be biconnected. If this assumption does not hold, Propositions 3 and 4 may suffer from the pathological case where the* QCN *at hand is unsatisfiable, and the constraints corresponding to a biconnected component in the given graph G get labeled with the empty relation (and hence have all their base relations removed as unfeasible), whilst the constraints corresponding to another biconnected component in the given graph G are not labeled with the empty relation (and hence may appear to hold feasible base relations). This case may indeed occur due to the inability of any suitable algorithm for applying $\overset{\diamond}{_G}$-consistency (and $\overset{\blacklozenge}{_G}$-consistency as a consequence) to propagate the empty relation from biconnected component to* *biconnected component (see Definition 3 and Figure 4). The reader may easily drop the assumption at the cost of taking care of this pathological case via more complicated wording of the presented result; in particular, it could be stated that if the assignment of the empty relation takes place, then all the base relations in the constraints corresponding to the edges of G are unfeasible in the* QCN.

As a note, an interesting case where the property described in Proposition 4 can be satisfied, is the case where the considered subclass of relations is obtained from a relation algebra that has *patchwork* [66] for $\overset{\diamond}{_G}$-consistent and not trivially inconsistent QCNs over that subclass, where $G = (V, E)$ is any chordal graph such that $\mathsf{G}(\mathcal{N}) \subseteq G$ for a given QCN $\mathcal{N} = (V, C)$. In that case, we will indeed have that $\mathcal{N}$ is satisfiable if $\overset{\diamond}{_G}(\mathcal{N})$ is not trivially inconsistent [38]. As a matter of fact, patchwork holds for all the qualitative constraint languages mentioned in Proposition 1 [45]. Of course, in general, the property may be satisfied in other cases as well; for instance, patchwork may not hold, but the overall property may hold for using complete graphs (and, hence, when $\diamond$-consistency is used) or when constraints in the structure of the constraint graphs of the QCNs are imposed (a trivial case being restricting the constraint graphs to being trees).

4. $\blacklozenge^{\cup}_{G}$-Consistency: a new local consistency for QCNs

We define a new local consistency for QCNs inspired by k-partitioning consistency for constraint satisfaction problems (CSPs), where arc consistency is used as the underlying local consistency of choice, or *k-Partition-*AC for short [47]. This technique splits a variable domain into disjoint domains, where each of them contains at most k elements. In the case of QCNs, these elements correspond to base relations. With respect to k-Partition-AC, the most common and preferred approach is splitting a domain into singleton sub-domains, which is the case where $k = 1$, otherwise many questions arise, such as what should the size of each sub-domain be, how should this size be fixed, and which elements should be considered for a given use case. Although having many questions to deal with is not necessarily bad in general, the most important aspect regarding



(a) A $\blacklozenge_G$-consistent QCN $\mathcal{N}$

(b) $\mathcal{N}_1 = \diamond_G(\mathcal{N}_{[x_1,x_3]/\{p\}})$

(c) $\mathcal{N}_2 = \diamond_G(\mathcal{N}_{[x_1,x_3]/\{pi\}})$

(d) $\mathcal{N}_1 \cup \mathcal{N}_2$ $(\subset \mathcal{N})$

Figure 5: A $\blacklozenge_G$-consistent QCN $\mathcal{N}$ of IA along with a demonstration of how enforcing $\blacklozenge_G^{\cup}$-consistency can further eliminate base relations; here G is the complete graph on the set of variables of $\mathcal{N}$

1-Partition-AC is that it offers the nice property that it is strictly stronger than singleton arc consistency (SAC) [49].

In this work, we adapt the aforementioned technique to qualitative constraint networks and use $\diamond_G$-consistency as our underlying local consistency of choice.[5] Given a QCN $\mathcal{N}$, enforcing this consistency for $k = 1$ will eliminate every base relation that is not $\blacklozenge_G$-consistent for some constraint in $\mathcal{N}$, but also every base relation that is not *supported* by some base relation in $\mathcal{N}$ through $\diamond_G$-consistency. We call this new local consistency $\blacklozenge_G^{\cup}$-consistency, and better explain it with a demonstrative example as follows. Consider the $\blacklozenge_G$-consistent QCN $\mathcal{N} = (V, C)$ of IA in Figure 5. We can see that the base relation d is

[5]The partitioning scheme can be combined with any local consistency or propagation technique. Here, the definition is restricted to $\diamond_G$-consistency as it is the most essential of local consistencies used for dealing with QCNs.

$\blacklozenge_G$-consistent for $C(x_1, x_2)$, but it is not supported by any of the base relations that define constraint $C(x_1, x_3)$, namely, p and pi, through $\diamond_G$-consistency. In particular, by instantiating $C(x_1, x_3)$ with p and pi respectively and closing the corresponding QCNs under $\diamond_G$-consistency, we obtain the atomic sub-QCNs presented in Figures 5b and 5c respectively. A unification of those two sub-QCNs results in the elimination of the base relation d in $C(x_1, x_2)$ of the unified QCN, as depicted in Figure 5d. After eliminating the base relation d in $C(x_1, x_2)$, the revised QCN $\mathcal{N}$ becomes $\blacklozenge_G^{\cup}$-consistent.

Now we can formally define this consistency.

Definition 8. Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, $\mathcal{N}$ is said to be $\blacklozenge_G^{\cup}$-*consistent* iff $\mathcal{N}$ is $\blacklozenge_G$-consistent and $\forall \{v, v'\} \in E$, $\forall b \in C(v, v')$, and $\forall \{u, u'\} \in E$ we have that $\exists b' \in C(u, u')$ such that $b \in C'(v, v')$, where $(V, C') = \diamond_G(\mathcal{N}_{[u,u']/\{b'\}})$.

We prove the following result to be used in the sequel, which suggets that $\blacklozenge_G^{\cup}$-consistency can only eliminate unfeasible base relations:

Proposition 5. *Let* $\mathcal{N} = (V, C)$ *be a* QCN, $G = (V, E)$ *a graph, and* $b \in C(v, v')$ *with* $v, v' \in V$ *a base relation. Then, if* $\exists \{u, u'\} \in E$ *such that* $b \notin C'(v, v')$, *where* $(V, C') = \bigcup \{\diamond_G(\mathcal{N}_{[u,u']/\{b'\}}) \mid b' \in C(u, u')\}$, *we have that* b *is an unfeasible base relation.*

Proof. Let us assume that b is a feasible base relation. Then, by definition of feasible base relations there exists a scenario $\mathcal{S} = (V, C')$ of $\mathcal{N}$ such that $C'(v, v') = \{b\}$. Further, by the equivalence property of $\diamond_G$-consistency it holds that $\diamond_G(\mathcal{S}) = \mathcal{S}$ (as $\mathcal{S}$, being a scenario, is an atomic and satisfiable QCN and, hence, none of its base relations can be removed by application of $\diamond_G$-consistency). Thus, it follows that $\forall \{u, u'\} \in E$ we have that $b \in C''(v, v')$, where $(V, C'') = \diamond_G(\mathcal{N}_{[u,u']/C'(u,u')})$, as $\mathcal{S} \subseteq \mathcal{N}_{[u,u']/C'(u,u')}$ and, hence, $\diamond_G(\mathcal{S}) \subseteq \diamond_G(\mathcal{N}_{[u,u']/C'(u,u')})$ by the monotonicity property of $\diamond_G$-consistency. As $\mathcal{S} \subseteq \mathcal{N}$, it follows that $\forall \{u, u'\} \in E$ we have that $C'(u, u') \subseteq C(u, u')$ and, hence, that $\exists b' \in C(u, u')$ such that $b \in C'''(v, v')$, where $(V, C''') = \diamond_G(\mathcal{N}_{[u,u']/\{b'\}})$, by simply considering the base relation $b' \in C(u, u')$ to be the one of the singleton relation $C'(u, u')$ of $\mathcal{S}$. Therefore, by definition of operation $\cup$ with respect to QCNs we can derive that $\forall\{u, u'\} \in E$ it holds that $b \in C^*(v, v')$, where $(V, C^*) = \bigcup\{\diamond_G(\mathcal{N}_{[u,u']/\{b'\}}) \mid b' \in C(u, u')\}$, which concludes our proof by contraposition. □

We recall the following result to be used in one of our proofs later on:

Proposition 6 ([38])**.** *For any* QCN*s $\mathcal{N}_1$ and $\mathcal{N}_2$ on a set of variables V and any graph $G = (V, E)$, if $\mathcal{N}_1$ and $\mathcal{N}_2$ are $\blacklozenge_G$-consistent, then $(\mathcal{N}_1 \cup \mathcal{N}_2)$ is $\blacklozenge_G$-consistent.*

We note that the aforementioned result describes a sufficient property for proving dominance for a new consistency, but that property might not be necessary in general and, hence, does not solely follow from the well-behaveness of the consistency at hand. We prove the same property for $\blacklozenge^{\cup}_G$-consistency, to be used in what follows.

Proposition 7. *For any* QCN*s $\mathcal{N}_1$ and $\mathcal{N}_2$ on a set of variables V and any graph $G = (V, E)$, if $\mathcal{N}_1$ and $\mathcal{N}_2$ are $\blacklozenge^{\cup}_G$-consistent, then $(\mathcal{N}_1 \cup \mathcal{N}_2)$ is $\blacklozenge^{\cup}_G$-consistent.*

Proof. Let $\mathcal{N}_1 = (V, C_1)$, $\mathcal{N}_2 = (V, C_2)$, $(\mathcal{N}_1 \cup \mathcal{N}_2) = (V, C)$, $v, v' \in V$ be two variables, and $b \in C(v, v')$ a base relation. We only need to consider the case where $b \in C_1(v, v')$, as the case where $b \in C_2(v, v')$ is symmetric. Note that as both $\mathcal{N}_1$ and $\mathcal{N}_2$ are on V, both $C_1(v, v')$ and $C_2(v, v')$ are defined (see Definition 1). Since $\mathcal{N}_1$ is $\blacklozenge^{\cup}_G$-consistent, we have that $\mathcal{N}_1$ is $\blacklozenge_G$-consistent and $\forall\{u, u'\} \in E$ there exists $b' \in C_1(u, u')$ such that $b \in C_1'(v, v')$, where $(V, C_1') = \diamond_G(\mathcal{N}_{1[u,u']/\{b'\}})$, by definition of $\blacklozenge^{\cup}_G$-consistency. In addition, we have that $(\mathcal{N}_1 \cup \mathcal{N}_2)$ is $\blacklozenge_G$-consistent by Proposition 6. As $\mathcal{N}_1 \subseteq (\mathcal{N}_1 \cup \mathcal{N}_2)$, we have that $\mathcal{N}_{1[u,u']/\{b'\}} \subseteq (\mathcal{N}_1 \cup \mathcal{N}_2)_{[u,u']/\{b'\}}$ $\forall\{u, u'\} \in E$ and $\forall b' \in C_1(u, u')$. Thus, we have that $\diamond_G(\mathcal{N}_{1[u,u']/\{b'\}}) \subseteq \diamond_G((\mathcal{N}_1 \cup \mathcal{N}_2)_{[u,u']/\{b'\}})$ $\forall\{u, u'\} \in E$ and $\forall b' \in C_1(u, u')$ by the monotonicity property of $\diamond_G$-consistency. From that we can deduce that $\forall\{u, u'\} \in E$ there exists $b' \in C(u, u')$ such that $b \in C'(v, v')$, where $(V, C') = \diamond_G((\mathcal{N}_1 \cup \mathcal{N}_2)_{[u,u']/\{b'\}})$. Hence, by the assumption that $\mathcal{N}_1$ and $\mathcal{N}_2$ are $\blacklozenge^{\cup}_G$-consistent, we have proved that $(\mathcal{N}_1 \cup \mathcal{N}_2)$ is $\blacklozenge^{\cup}_G$-consistent as well. □

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

Theorem 2. *We have that $\blacklozenge^{\cup}_{G}$-consistency is well-behaved.*

Proof. (Dominance) From Proposition 7 we can assert that, for any QCN $\mathcal{N} = (V, C)$ and any graph $G = (V, E)$, there exists a unique $\blacklozenge^{\cup}_{G}$-consistent QCN $\bigcup\{\mathcal{N}' \mid \mathcal{N}' \subseteq \mathcal{N}$ and $\mathcal{N}'$ is $\blacklozenge^{\cup}_{G}$-consistent$\}$, which by its definition is the largest (w.r.t. $\subseteq$) $\blacklozenge^{\cup}_{G}$-consistent sub-QCN of $\mathcal{N}$ and, hence, the closure of $\mathcal{N}$ under $\blacklozenge^{\cup}_{G}$-consistency. (Equivalence) Let $\mathcal{N} = (V, C)$ be a QCN, $G = (V, E)$ a graph, and $\mathcal{N}' = (V, C')$ 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 σ of $\mathcal{N}$ such that $(\sigma(v), \sigma(v')) \in b$. Clearly, $\mathcal{N}'$ is a sub-QCN of $\mathcal{N}$ and it is necessarily $\blacklozenge^{\cup}_{K_V}$-consistent (where K_V denotes the complete graph on the set of variables V of $\mathcal{N}$), as by Proposition 5 we have that the application of $\blacklozenge^{\cup}_{G}$-consistency on any QCN (V, C) w.r.t. any graph $G = (V, E)$ can only remove unfeasible base relations, and not feasible ones. It follows that $\mathcal{N}' \subseteq \blacklozenge^{\cup}_{G}(\mathcal{N}) \subseteq \mathcal{N}$ and, as such, $\blacklozenge^{\cup}_{G}(\mathcal{N})$ and $\mathcal{N}$ share the same set of solutions. □

We prove the following general result regarding the pruning capability of $\blacklozenge_{G}$-consistency in comparison with that of $\blacklozenge^{\cup}_{G}$-consistency:

Proposition 8. *We have that $\blacklozenge^{\cup}_{G}$-consistency $\triangleright$ $\blacklozenge_{G}$-consistency.*

Proof. We have that $\blacklozenge^{\cup}_{G}$-consistency $\trianglerighteq$ $\blacklozenge_{G}$-consistency by the very definition of $\blacklozenge^{\cup}_{G}$-consistency, since, for any graph $G = (V, E)$, any QCN (V, C) that is $\blacklozenge^{\cup}_{G}$-consistent is already $\blacklozenge_{G}$-consistent. To prove strictness we use an example as follows. Consider the QCN $\mathcal{N} = (V, C)$ of Figure 5. The reader can verify that $\mathcal{N}$ is $\blacklozenge_{G}$-consistent, as we have that b is $\blacklozenge_{G}$-consistent for $C(v, v')$ $\forall \{v, v'\} \in E$ and $\forall b \in C(v, v')$. However, we have that $d \notin C'(x_1, x_2)$, where $(V, C') = \bigcup\{\lozenge_{G}(\mathcal{N}_{[x_1,x_3]/\{b'\}}) \mid b' \in C(x_1, x_3)\}$, as demonstrated in the figure. In detail, $\lozenge_{G}(\mathcal{N}_{[x_1,x_3]/\{p\}}) \cup \lozenge_{G}(\mathcal{N}_{[x_1,x_3]/\{pi\}})$ is a QCN such that d is not among the base relations that define the constraint on variables x_1 and x_2. Thus, $\blacklozenge^{\cup}_{G}$-consistency does not hold in $\mathcal{N}$. □

The next result follows trivially:

Proposition 9. *We have that $\blacklozenge^{\cup}_{G}$-consistency $\triangleright$ $\lozenge_{G}$-consistency.*

Proof. A direct consequence of Propositions 2 and 8 and the transitivity of $\triangleright$. $\square$

Finally, we introduce the following result that identifies the case where $\blacklozenge_G$-consistency and $\blacklozenge_G^{\cup}$-consistency are equivalent:

Proposition 10. *Let $\mathcal{A}$ be a subclass of relations of a relation algebra with the property that for any* QCN *$\mathcal{N} = (V, C)$ over $\mathcal{A}$ there exists a graph $G = (V, E)$ such that, if $\diamond_G(\mathcal{N})$ is not trivially inconsistent, then $\mathcal{N}$ is satisfiable. Then, for any such $\mathcal{N}$ and G, we have that $\blacklozenge_G$-consistency $\equiv$ $\blacklozenge_G^{\cup}$-consistency.*

Proof. We first prove that, if $\mathcal{N}$ is $\blacklozenge_G$-consistent, then $\mathcal{N}$ is also $\blacklozenge_G^{\cup}$-consistent. By Proposition 4 we have that $\forall \{u, v\} \in E$ and $\forall b \in C(u, v)$ the base relation b is feasible. In addition, by the equivalence property of $\blacklozenge_G^{\cup}$-consistency we have that the application of $\blacklozenge_G^{\cup}$-consistency on $\mathcal{N}$ can only remove unfeasible base relations and, hence, that $\blacklozenge_G^{\cup}(\mathcal{N}) = \mathcal{N}$, as every base relation $b \in C(u, v)$ $\forall \{u, v\} \in E$ is feasible. The proof that, if $\mathcal{N}$ is $\blacklozenge_G^{\cup}$-consistent, then $\mathcal{N}$ is also $\blacklozenge_G$-consistent, follows directly from the definition of $\blacklozenge_G^{\cup}$-consistency. $\square$

It is important to make the following remark:

Remark 2. *The reader is kindly reminded of Remark 1 in Section 3, as well as the discussion in the end of Section 2, which focus on the assumption of biconnectedness for all of our graphs. If this assumption does not hold, the reader can verify that the* QCN *of Figure 4 is $\blacklozenge_G$-consistent but not $\blacklozenge_G^{\cup}$-consistent. The reader may drop the assumption and consider the pathological case described in Remark 1 as a special case in a revised wording of Proposition 10.*

A hasty reading of Proposition 10 might give the impression that one should always opt to apply $\blacklozenge_G$-consistency for the cases where the considered QCN and the graph G satisfy the prerequisites detailed in that proposition, as $\blacklozenge_G$-consistency, being weaker than $\blacklozenge_G^{\cup}$-consistency in general, should be "easier" to apply. However, as we will experimentally show in Section 7, $\blacklozenge_G^{\cup}$-consistency is faster to apply. To give an intuition, any well-structured algorithm for applying $\blacklozenge_G^{\cup}$-consistency on a QCN will inescapably make better use of the singleton checks than the

Algorithm 1: $\mathsf{PSWC}^{\cup}(\mathcal{N}, G)$

```
in   : A QCN 𝒩 = (V, C), and a graph G = (V, E).
out  : A sub-QCN of 𝒩.
begin
    𝒩 ← PWC(𝒩, G);
    Q ← list(E);
    while Q ≠ ∅ do
        {v, v′} ← Q.pop();
        (V, C′) ← ⊥^V;
        foreach b ∈ C(v, v′) do
            (V, C′) ← (V, C′) ∪ PWC(𝒩_{[v,v′]/{b}}, G, {{v, v′}});
        if (V, C′) ⊂ 𝒩 then
            flag ← False;
            foreach {u, u′} ∈ E do
                if C′(u, u′) ⊂ C(u, u′) then
                    C(u, u′) ← C′(u, u′);
                    C(u′, u) ← C′(u′, u);
                    flag ← True;
            if flag then Q ← list(E);
    return 𝒩;
```

respective algorithm for applying $\blacklozenge_G$-consistency, as it will exploit the checks (by the very definition of $\blacklozenge_G^{\cup}$-consistency) to *proactively* eliminate certain unfeasible (and, hence, possibly not $\blacklozenge_G$-consistent for their constraints) base relations.

5. $\mathsf{PSWC}^{\cup}$: an algorithm for achieving $\blacklozenge_G^{\cup}$-consistency

In this section, we propose an algorithm for efficiently applying $\blacklozenge_G^{\cup}$-consistency on a given QCN $\mathcal{N}$, called $\mathsf{PSWC}^{\cup}$ (which stands for $\cup$-*collective* partial singleton closure under weak composition) and presented in Algorithm 1. This algorithm builds upon the algorithm for efficiently achieving $\blacklozenge_G$-consistency, called PSWC

Algorithm 2: PSWC($\mathcal{N}, G$)

in : A QCN $\mathcal{N} = (V, C)$, and a graph $G = (V, E)$.

out : A sub-QCN of $\mathcal{N}$.

begin
- $\mathcal{N} \leftarrow \mathsf{PWC}(\mathcal{N}, G)$;
- $Q \leftarrow list(E)$;
- **while** $Q \neq \emptyset$ **do**
  - $\{v, v'\} \leftarrow Q.pop()$;
  - $(V, C') \leftarrow \bot^V$;
  - **foreach** $b \in C(v, v')$ **do**
    - $(V, C') \leftarrow (V, C') \cup \mathsf{PWC}(\mathcal{N}_{[v,v']/\{b\}}, G, \{\{v, v'\}\})$;
  - **if** $C'(v, v') \subset C(v, v')$ **then**
    - $C(v, v') \leftarrow C'(v, v')$;
    - $C(v', v) \leftarrow C'(v', v)$;
    - $Q \leftarrow list(E)$;
- **return** $\mathcal{N}$;

(which stands for partial singleton closure under weak composition) and presented in Algorithm 2, which in itself is an advancement[6] of the respective algorithm for enforcing $\blacklozenge_G$-consistency that is presented in [38]. For the sake of completeness, we also present the state-of-the-art algorithm for applying $\diamond_G$-consistency on a given QCN, called PWC (which stands for partial closure under weak composition),

[6]We use a queue in both of our algorithms that is initialized with all of the edges of a given graph G, which correspond to constraints of a given QCN, and that is also filled with all of the aforementioned edges during execution whenever any of those constraints is revised, i.e., whenever a base relation is removed. This technique is equivalent to executing a break statement in the algorithm of [38] whenever a singleton check fails and, hence, a constraint is revised, forcing the inner loop in that algorithm to stop and using the outer loop to initiate singleton checks in a fresh QCN. We have found this tactic to work much better in practice, cutting down on the number of constraint checks performed by $\sim 20\%$. Further, using a queue allows for prioritizing certain edges, a strategy which is in line with similar techniques used in the algorithm for enforcing $\diamond_G$-consistency [67, 42, 68].

Algorithm 3: $\mathsf{PWC}(\mathcal{N}, G, e \leftarrow \emptyset)$

in : A QCN $\mathcal{N} = (V, C)$, a graph $G = (V, E)$, and optionally a set e such that $e \subseteq E$.

out : A sub-QCN of $\mathcal{N}$.

begin
- $Q \leftarrow set(e$ **if** $e \neq \emptyset$ **else** $E)$;
- **while** $Q \neq \emptyset$ **do**
  - $\{v, v'\} \leftarrow Q.pop()$;
  - **foreach** $v'' \in V \mid \{v, v''\}, \{v', v''\} \in E$ **do**
    - $r \leftarrow C(v, v'') \cap (C(v, v') \diamond C(v', v''))$;
    - **if** $r \subset C(v, v'')$ **then**
      - $C(v, v'') \leftarrow r$;
      - $C(v'', v) \leftarrow r^{-1}$;
      - $Q.add(\{v, v''\})$;
    - $r \leftarrow C(v'', v') \cap (C(v'', v) \diamond C(v, v'))$;
    - **if** $r \subset C(v'', v')$ **then**
      - $C(v'', v') \leftarrow r$;
      - $C(v', v'') \leftarrow r^{-1}$;
      - $Q.add(\{v'', v'\})$;
- **return** $\mathcal{N}$;

which is utilized as a subroutine by both $\mathsf{PSWC}^{\cup}$ and PSWC (see Algorithm 3).

The difference between algorithms $\mathsf{PSWC}^{\cup}$ and PSWC lies solely in the way that they exploit singleton checks. In particular, note the difference between the conditions in line 9 of both algorithms; $\mathsf{PSWC}^{\cup}$ will bring up all edges in the queue for revising the entire QCN even when the constraint at hand was not revised, but another constraint somewhere in the QCN was, whereas PSWC will keep its focus solely on the constraint at hand. This is due to the fact that algorithm $\mathsf{PSWC}^{\cup}$ will use a single singleton check to eliminate base relations anywhere in the network, and not just in the constraint at hand as algorithm PSWC does. Henceforth, we will refer to the exploited singleton checks that are used to collectively eliminate certain unfeasible base relations as *collective singleton checks*, defined as follows. Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, a *collective singleton check* for a constraint $C(v, v')$ with $\{v, v'\} \in E$ consists of computing the QCN $\mathcal{N}' = \bigcup\{\diamond_G(\mathcal{N}_{[v,v']/\{b\}}) \mid b \in C(v, v')\}$ and checking if $\mathcal{N}' \subset \mathcal{N}$. Simply put, a collective singleton check involves successively instantiating a given constraint of a QCN with each of its base relations, computing and unifying the corresponding $\diamond_G$-consistent QCNs, and checking if there exist stronger constraints in the resulting QCN than the respective ones in the original QCN so that the latter can be updated accordingly. The operations involved in the singleton checks themselves are based on the use of an algorithm for enforcing $\diamond_G$-consistency, such as PWC presented in Algorithm 3, and are in line with Definition 4 of $\blacklozenge_G$-consistency.

Before proving the correctness of algorithm $\mathsf{PSWC}^{\cup}$, we recall the following result regarding the correctness of algorithm PSWC:

Proposition 11 (cf. [38, 46])**.** *Given a* QCN $\mathcal{N} = (V, C)$ *of a relation algebra and a graph* $G = (V, E)$*, we have that algorithm* PSWC *terminates and returns* $\blacklozenge_G(\mathcal{N})$*.*

Now, we show that algorithm $\mathsf{PSWC}^{\cup}$ is complete for applying $\blacklozenge^{\cup}_G$-consistency on a given $\mathcal{N} = (V, C)$ for a given graph $G = (V, E)$. As the algorithm builds upon PSWC, the result is straightforward; hence, an intuitive proof is provided, which however manages to explain the overall functionallity of algorithm $\mathsf{PSWC}^{\cup}$ in sufficient detail.

Theorem 3. *Given a* QCN $\mathcal{N} = (V, C)$ *of a relation algebra and a graph* $G = (V, E)$*, we have that algorithm* $\mathsf{PSWC}^{\cup}$ *terminates and returns* $\blacklozenge^{\cup}_G(\mathcal{N})$*.*

Proof. (Intuition) It is easy to see that lines 9–14 in Algorithm 1 perform a superset of the operations performed in lines 9–11 in Algorithm 2. Thus, by Proposition 11 we know that given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, algorithm $\mathsf{PSWC}^{\cup}$ applies the set of operations required to make $\mathcal{N}$ $\blacklozenge_G$-consistent. We need to show that the rest of the operations maintain $\blacklozenge_G$-consistency and further achieve $\blacklozenge_{G}^{\cup}$-consistency. With respect to that, it is again easy to see that algorithm $\mathsf{PSWC}^{\cup}$ enforces exactly the conditions specified in Proposition 5 and, hence, removes the (unfeasible) base relations required to make $\mathcal{N}$ $\blacklozenge_{G}^{\cup}$-consistent. Further, since the algorithm will only terminate when b is guaranteed to have become $\blacklozenge_{G}$-consistent for $C(u,v)$ $\forall\{u,v\} \in E$ and $\forall b \in C(u,v)$ and no constraint is further revised to additionally achieve $\blacklozenge_{G}^{\cup}$-consistency, we can conclude that algorithm $\mathsf{PSWC}^{\cup}$ correctly applies $\blacklozenge_{G}^{\cup}$-consistency on $\mathcal{N}$. $\square$

Time complexity analysis

Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, we have that algorithm $\mathsf{PSWC}^{\cup}$ applies $\blacklozenge_{G}^{\cup}$-consistency on $\mathcal{N}$ in $O(\Delta \cdot |E|^3 \cdot \mathsf{B}^3)$ time, where Δ is the maximum vertex degree of graph G. In particular, algorithm PWC is executed $O(|E| \cdot |B|)$ times every time a constraint is revised, and such a constraint revision can occur $O(|E| \cdot |B|)$ times. Further, we note that the unification operations that take place in line 8 of the algorithm are handled in $O(|E| \cdot |B|)$ time in total, as we keep track of the constraints that are revised by algorithm PWC and we can have a total of $O(|E| \cdot |B|)$ constraint revisions. The same argument holds for the operations that take place in lines 9–14 of the algorithm. (These details are not included in the algorithm to allow for a more compact representation.) Now, by taking into account the worst-case time complexity of algorithm PWC, which is $O(\Delta \cdot |E| \cdot \mathsf{B})$ [69], a worst-case time complexity of $O(\Delta \cdot |E|^3 \cdot \mathsf{B}^3)$ can be obtained for algorithm $\mathsf{PSWC}^{\cup}$; this is also the worst-case time complexity of algorithm PSWC [38]. It is important to note that we cannot utilize the incremental functionality of algorithm PWC (see Theorem 1 in [70, Section 3] and the surrounding text) to obtain a better bound for our algorithm, as the singleton checks are perfomed independently of one another; to be more precise, the unification operations that take place in line 8 of the algorithm do not provide the level of interdependency required to tap into the incrementality of PWC.

Algorithm 4: ℓPSWC$^{\cup}(\mathcal{N}, G)$

in : A QCN $\mathcal{N} = (V, C)$, and a graph $G = (V, E)$.

out : A sub-QCN of $\mathcal{N}$.

```
begin
    N ← PWC(N, G);
    Q ← list(E ∩ E(G(N)));
    while Q ≠ ∅ do
        {v, v'} ← Q.pop();
        (V, C') ← ⊥^V;
        foreach b ∈ C(v, v') do
            (V, C') ← (V, C') ∪ PWC(N_[v,v']/{b}, G, {{v, v'}});
        C(v, v') ← C'(v, v');
        if (V, C') ⊂ N then
            foreach {u, u'} ∈ E \ {v, v'} do
                if C'(u, u') ⊂ C(u, u') then
                    C(u, u') ← C'(u, u');
                    C(u', u) ← C'(u', u);
                    Q.push({u, u'});
    return N;
```

6. ℓPSWC$^{\cup}$: a lazy variant for approximating $\blacklozenge_G^{\cup}$-consistency

In this section we propose a lazy variant of algorithm PSWC$^{\cup}$ that aims to efficiently approximate $\blacklozenge_G^{\cup}$-consistency for a given QCN $\mathcal{N} = (V, C)$ with respect to a graph $G = (V, E)$, when $\blacklozenge_G^{\cup}$-consistency (or $\blacklozenge_G$-consistency) is too costly to enforce. This algorithmic variant is presented in Algorithm 4 and is called ℓPSWC$^{\cup}$ (which stands for *lazy* $\cup$*-collective* partial singleton closure under weak composition).

Like PSWC$^{\cup}$, a key feature of this algorithm is that it utilizes *collective singleton checks* and, hence, collectively eliminates certain unfeasible base relations by exploiting the singleton checks that are typically performed by an algorithm for enforcing ${\blacklozenge}_G$-consistency, such as the one presented in Algorithm 2 and called PSWC. Unike $\mathsf{PSWC}^{\cup}$, a unique feature of our algorithm is that during its execution it takes a *lazy* (non-exhaustive) approach and performs a collective singleton check only for a constraint that has been revised and put into the queue due to a previous collective singleton check for some other constraint. As we will see in what follows, this behavior leads to a non-unique closure being obtained in general for a given input QCN. Further, as opposed to PSWC, our algorithm initially takes into account only non-universal relations of ${\lozenge}_G(\mathcal{N})$ for a QCN $\mathcal{N}$ $= (V, C)$ and a graph $G = (V, E)$. In all other aspects, algorithm $\ell\mathsf{PSWC}^{\cup}$ can be viewed as being similar to the one for efficiently achieving ${\blacklozenge}^{\cup}_G$-consistency, namely, $\mathsf{PSWC}^{\cup}$.

We prove the following main result regarding algorithm $\ell\mathsf{PSWC}^{\cup}$, which captures its major theoretical properties:

Theorem 4. *Given a* QCN $\mathcal{N} = (V, C)$ *of a relation algebra and a graph* G $= (V, E)$*, algorithm* $\ell\mathsf{PSWC}^{\cup}$ *terminates and returns a sub-*QCN $\mathcal{N}'$ *of* $\mathcal{N}$ *such that:*

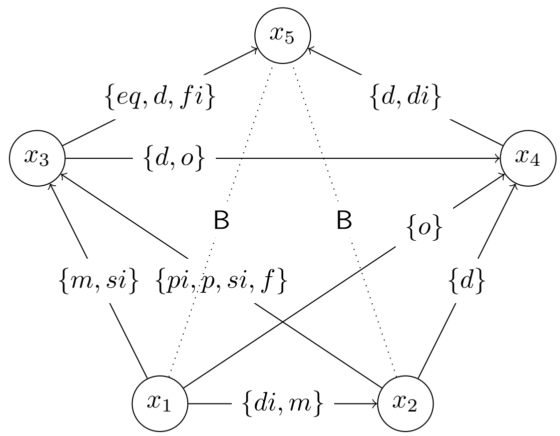
- $\mathcal{N}'$ *is* ${\lozenge}_G$*-consistent;*
- $\mathcal{N}'$ *is equivalent to* $\mathcal{N}$*;*
- $\mathcal{N}'$ *is non-unique in general;*
- $\mathcal{N}'$ *is incomparable to* ${\blacklozenge}_G(\mathcal{N})$ *in general;*
- $\mathcal{N}' \subseteq {\lozenge}_G(\mathcal{N})$*;*
- ${\blacklozenge}^{\cup}_G(\mathcal{N}) \subseteq \mathcal{N}'$*.*

Proof. First of all, and as it is detailed in the time complexity analysis that follows this proof, the algorithm terminates because it will only keep executing as long as a base relation has been removed from some constraint and the corresponding edge has been pushed into the queue. There is a finite number of base relations in any given QCN (by definition, the set of base relations is finite and a QCN involves a finite set of variables).

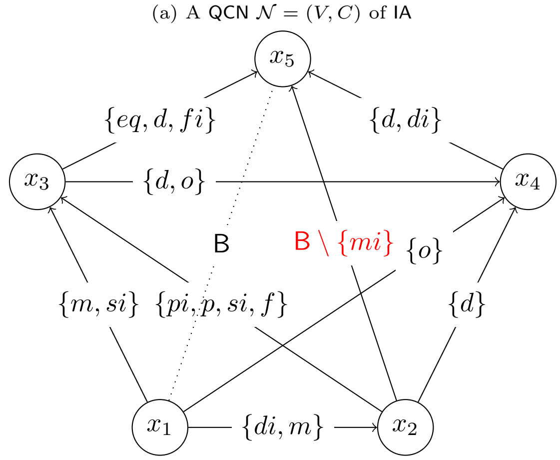
In line 2 of the algorithm, the original QCN $\mathcal{N}$ is made $\diamond_G$-consistent via a call to function PWC; let $\mathcal{N}' = (V, C') = \diamond_G(\mathcal{N})$. We need to show that the rest of the refinement operations in the algorithm entail $\diamond_G$-consistency as well. By utilizing the incremental functionality of algorithm PWC (see [70, Section 3]), in lines 7–8 of the algorithm, for a pair of variables $\{u, u'\} \in E$ a set of $\diamond_G$-consistent sub-QCNs of $\mathcal{N}'$ is created, namely, the set $S = \{\diamond_G(\mathcal{N}'_{[u,u']/\{b\}}) \mid b \in C'(u, u')\}$. Then, in those same lines, the operation $\bigcup S$ takes place. We show that $\bigcup S$ is $\diamond_G$-consistent. Let us assume that there exist k QCNs in S, with $k \leq |C'(u, u')|$, and hence let $\mathcal{N}_1 = (V, C_1)$, $\mathcal{N}_2 = (V, C_2)$, …, $\mathcal{N}_k = (V, C_k)$ be all the k different $\diamond_G$-consistent QCNs in S. We need to show that $\mathcal{N}^* = (V, C^*) = \bigcup_{i=1}^{k} \mathcal{N}_i$ is $\diamond_G$-consistent, which is a result that can serve as a proof that $\diamond_G$-consistency is closed under union. Let us consider three variables $v, v', v'' \in V$ such that $\{v, v'\}, \{v, v''\}, \{v', v''\} \in E$, 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 $\diamond_G$-consistent, we have that $C_i(v, v') \subseteq C_i(v, v'') \diamond C_i(v'', v')$ and, hence, there exist base relations $b' \in C_i(v, v'')$ and $b'' \in C_i(v'', v')$ such that $b \in b' \diamond b''$ by definition of $\diamond_G$-consistency. 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 $\diamond_G$-consistent. This proves that the algorithm terminates and returns a $\diamond_G$-consistent sub-QCN of $\mathcal{N}$.

Let $\mathcal{N}' = (V, C') = \diamond_G(\mathcal{N})$ (line 2 of the algorithm). By equivalence of $\diamond_G$-consistency $\mathcal{N}'$ is equivalent to $\mathcal{N}$. Further, let $b \in C'(u, u')$ with $\{u, u'\} \in E$ be a base relation. In lines 9–14 of the algorithm, the base relation b is eliminated only if $\exists \{v, v'\} \in E$ such that $b \notin C''(u, u')$, where $(V, C'') = \bigcup \{\diamond_G(\mathcal{N}'_{[v,v']/\{b'\}}) \mid b' \in C'(v, v')\}$. We need to show that b is an unfeasible base relation of $\mathcal{N}'$. This is something that follows directly from Proposition 5 and hence we have that the algorithm terminates and returns a sub-QCN of $\mathcal{N}$ that is equivalent to $\mathcal{N}$.

In what follows, we give an intuition of why the order in which the constraints are processed ultimately affects the output of the algorithm. The validity of



(a) A QCN $\mathcal{N} = (V, C)$ of IA



(b) The QCN $\blacklozenge_G(\mathcal{N})$

Figure 6: Given the QCN $\mathcal{N} = (V, C)$ of Figure 6a and the graph G that results by removing the edge $\{x_1, x_5\}$ from the complete graph on V, algorithm ℓPSWC$^{\cup}$ is unable to eliminate the base relation mi in $C(x_2, x_5)$ for any possible order in which the constraints are processed; however, mi is not $\blacklozenge_G$-consistent for $C(x_2, x_5)$, as shown in Figure 6b

the result itself is supported by a counterexample;[7] here, we only provide an argument about why the result is plausible. Let $\mathcal{N}' = (V, C') = \diamond_G(\mathcal{N})$ (line 2 of the algorithm). Further, consider two different pairs of variables $\{v, v'\}, \{u, u'\} \in E$, and let $\mathcal{N}^{uu'} = (V, C^{uu'}) = \bigcup\{\diamond_G(\mathcal{N}'_{[u,u']/\{b\}}) \mid b \in C'(u, u')\}$ and $\mathcal{N}^{vv'} =$

[7] `https://msioutis.gitlab.io/files/counterex.log`

$(V, C^{vv'}) = \bigcup\{\diamond_G(\mathcal{N}'_{[v,v']/\{b\}}) \mid b \in C'(v,v')\}$. Then, it is entirely possible that there exist two different pairs of variables $\{y,y'\},\{w,w'\} \in E \setminus \{\{v,v'\},\{u,u'\}\}$ such that $C^{uu'}(y,y') \subset C^{vv'}(y,y')$ and $C^{uu'}(w,w') \supset C^{vv'}(w,w')$. (In fact, such an example can be constructed by considering two copies of the QCN of Figure 5a inside a larger QCN.) It follows that $\mathcal{N}^{uu'} \not\subseteq \mathcal{N}^{vv'}$ and $\mathcal{N}^{vv'} \not\subseteq \mathcal{N}^{uu'}$. Since both $\mathcal{N}^{uu'}$ and $\mathcal{N}^{vv'}$ are sub-QCNs of $\mathcal{N}'$, but incomparable to each other even if we only take into account constraints between pairs of variables other than $\{v,v'\}$ and $\{u,u'\}$, this result suggests that different constraints may be revised and put into the queue of the algorithm depending on the order in which $\mathcal{N}^{uu'}$ and $\mathcal{N}^{vv'}$ are calculated. As the algorithm takes a *lazy* (non-exhaustive) approach during its execution and performs a collective singleton check (in lines 7–14) only for a constraint that has been revised and put into the queue due to a previous collective singleton check for some other constraint, the order in which these collective singleton checks are performed is important and can lead to different outputs for the same input QCN.

Consider the QCN of Figure 6a and let G be the graph that results by removing the edge $\{x_1, x_5\}$ from the complete graph on the set of variables of the QCN. Given that QCN and the graph G as input, algorithm ℓPSWC$^\cup$ is unable to eliminate any base relation in the QCN for any possible order in which the constraints are processed. However, the QCN is not $\blacklozenge_G$-consistent as shown in Figure 6b. Indeed, the base relation mi is not $\blacklozenge_G$-consistent for the constraint between variables x_2 and x_5. Therefore, $\blacklozenge_G$-consistency is able to eliminate more base relations than our algorithm in this case. Next, consider the $\blacklozenge_G$-consistent QCN of Figure 5a with respect to the complete graph G on the set of variables of the QCN. In this case, our algorithm is able to eliminate the base relation d for the constraint between variables x_1 and x_2 for any possible order in which the constraints are processed. This implies that given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, algorithm ℓPSWC$^\cup$ can produce an output sub-QCN $\mathcal{N}'$ of $\mathcal{N}$ such that $\blacklozenge_G(\mathcal{N}) \not\subseteq \mathcal{N}'$ and $\mathcal{N}' \not\subseteq \blacklozenge_G(\mathcal{N})$. This proves that the algorithm terminates and returns a sub-QCN of $\mathcal{N}$ that is, in general, incomparable to $\blacklozenge_G(\mathcal{N})$.

We have already established that algorithm ℓPSWC$^{\cup}$ terminates and returns a $\diamond_G$-consistent sub-QCN $\mathcal{N}'$ of $\mathcal{N}$ in the first part of this proof. By dominance of $\diamond_G$-consistency we have that $\diamond_G(\mathcal{N})$ is the largest (w.r.t. $\subseteq$) $\diamond_G$-consistent sub-QCN of $\mathcal{N}$. Therefore, it follows that $\mathcal{N}' \subseteq \diamond_G(\mathcal{N})$.

Finally, the fact that ℓPSWC$^{\cup}$ terminates and returns a sub-QCN $\mathcal{N}'$ of $\mathcal{N}$ such that $\blacklozenge^{\cup}_G(\mathcal{N}) \subseteq \mathcal{N}'$ follows directly from the structure of algorithm ℓPSWC$^{\cup}$, which considers only a subset of the set of collective singleton checks that is performed by PSWC$^{\cup}$. $\square$

Time complexity analysis

By its construction, given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, the worst-case time complexity of algorithm ℓPSWC$^{\cup}$ is essentially the same as that of algorithms PSWC$^{\cup}$ and PSWC, namely, $O(\Delta \cdot |E|^3 \cdot \mathsf{B}^3)$, where Δ is the maximum vertex degree of graph G, since it can be the case that the entire set of constraints is pushed into the queue each time a constraint revision occurs (see lines 10–15 of the algorithm). However, and as we aim to use ℓPSWC$^{\cup}$ to efficiently approximate $\blacklozenge^{\cup}_G$-consistency, in the next section we demonstrate that ℓPSWC$^{\cup}$ substantially outperforms PSWC$^{\cup}$ (and PSWC) in practice.

7. Experimental evaluation

To facilitate discussion and presentation of the results, we will first compare algorithm PSWC$^{\cup}$ to algorithm PSWC in Section 7.1, and then evaluate algorithm ℓPSWC$^{\cup}$ separately in Section 7.2. As the title of Section 7.2 suggests, viz., *"Evaluating ℓPSWC$^{\cup}$: where does it fit among* PSWC$^{\cup}$ *and* PSWC*?"*, this distinction is also done to allow us to focus a little more on the more interesting (we feel) behavior of ℓPSWC$^{\cup}$ and determine its place among PSWC$^{\cup}$ and PSWC.

Synopsis of key findings. Here we present a synopsis of the findings concerning a sample dataset of QCNs of IA and RCC8 (to be detailed in what follows) in order to give the reader an idea of what lies ahead. We have found PSWC$^{\cup}$ to outperform PSWC by 10% to 30% on average for the more time-consuming network instances (see for example the row for $\mathsf{m} = 4$ in Table 1b). On the other hand, $\mathsf{PSWC}^{\cup}$ is able to prune only slightly more base relations than PSWC on average; in particular, $\mathsf{PSWC}^{\cup}$ is able to prune around 3% more base relations than PSWC at best (see for example the row for $\mathsf{m} = 4$ in Table 2b). Regarding $\ell\mathsf{PSWC}^{\cup}$, we have found it to be up to 5 times faster than $\mathsf{PSWC}^{\cup}$ (and, hence, PSWC as well) on average for the more time-consuming network instances (see for example the row for $\mathsf{d} = 10$ in Table 8a). Further, the pruning capability of $\ell\mathsf{PSWC}^{\cup}$ is excellent for the entirety of the network instances of RCC8, but also for the majority of the network instances of the Interval Algebra (as the median value in Tables 6 and 7 suggests) Finally, $\mathsf{PSWC}^{\cup}$ (and $\ell\mathsf{PSWC}^{\cup}$) present a big overhead when used to check the satisfiability of network instances of Interval Algebra or RCC8 when put up against a state-of-the-art reasoner, but are ideal candidates for approximating and even determining in most cases the minimal labeling of those instances as a comparison with the state-of-the-art approach suggests (see Tables 11 and 12).

Let us now introduce the technical settings and details of the evaluation.

Technical specifications. The evaluation was carried out on a computer with an Intel Core i5-6200U processor (which has a max frequency of 2.7 GHz per CPU core under turbo mode[8]), 8 GB of RAM, and the Xenial Xerus x86_64 OS (Ubuntu Linux). All algorithms were coded in Python and run using the PyPy intepreter under version 5.1.2, which implements Python 2.7.10; the code is available upon request. Only one CPU core was used.

Dataset. We employed models $\mathsf{A}(\mathsf{n}, \mathsf{l}, \mathsf{d})$ [43] and $\mathsf{BA}(\mathsf{n}, \mathsf{m})$ [40] to generate random QCNs of IA and RCC8. In particular, $\mathsf{A}(\mathsf{n}, \mathsf{l}, \mathsf{d})$ can generate random QCNs of n variables with an average number l of base relations per non-universal constraint and an average degree d for the respective constraint graphs, and

[8]Turbo mode was maintained throughout the experimental evaluation by staying well within thermal design power (TDP) limit.

$\mathsf{BA(n,m)}$ can generate random QCNs of n variables with an average number $|\mathsf{B}|/2$ of base relations per non-universal constraint and by use of a *preferential attachment* [71] value m for the respective constraint graphs. Using model $\mathsf{A(n,l,d)}$, we generated 100 QCNs of IA and RCC8 of $\mathsf{n} = 70$ and $\mathsf{n} = 100$ variables respectively and with $\mathsf{l} = 6.5$ and $\mathsf{l} = 4.0$ base relations per non-universal constraint on average respectively, for all values of d ranging from 7 to 12 with a step of 1, as the *phase transition region* [72] for this model is observed for $8 \leq \mathsf{d} \leq 11$ for both of our calculi [43, 42]. Using model $\mathsf{BA(n,m)}$, we generated 100 QCNs of IA and RCC8 of $\mathsf{n} = 150$ and $\mathsf{n} = 200$ variables respectively for all values of m ranging from 2 to 5 with a step of 1, as the phase transition region for this model is observed for $\mathsf{m} \approx 3$ or 4 for both of our calculi [73]. Thus, we considered a total of 1 000 QCNs of IA and RCC8. Finally, regarding the graphs that were given as input to our algorithms, the *maximum cardinality search* algorithm [74] was used to obtain triangulations of the constraint graphs of our QCNs. The choice of such chordal graphs was reasonable given their extensive use in the recent literature on Qualitative Spatial and Temporal Reasoning, as reviewed in [75]; the use of those graphs itself was inspired by [44, 45, 76, 69, 77] among other works.

Measures. Our evaluation involved four measures, which we describe as follows. The first measure considers the number of *constraint checks per base relation removals* performed by an algorithm for meeting its objective. Given a QCN $\mathcal{N} = (V, C)$ and three variables $v_i, v_k, v_j \in V$, a constraint check occurs when we compute the 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 it if that condition is satisfied. The second measure concerns the *CPU time* and is naturally correlated with the first one, as the run-time of any proper implementation of an algorithm for enforcing a local consistency should, in principle, rely mainly on the number of constraint checks performed. The third measure compares the *pruning capability* between the evaluated algorithms, i.e., the number of base relation removals, and, finally, the fourth measure keeps track of the number of cases where the algorithms

Table 1: Evaluation of the computational effort of algorithms PSWC and $\text{PSWC}^{\cup}$ with random IA networks

(a) Evaluation with random IA networks of model $\mathsf{A}(\mathsf{n} = 70, \mathsf{l} = 6.5, \mathsf{d})$ [43]

	min		μ		max		σ	
d	PSWC	$\text{PSWC}^{\cup}$	PSWC	$\text{PSWC}^{\cup}$	PSWC	$\text{PSWC}^{\cup}$	PSWC	$\text{PSWC}^{\cup}$
7	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{2.53s}{10k}$	$\frac{2.22s}{9k}$	$\frac{7.15s}{21k}$	$\frac{5.01s}{16k}$	$\frac{1.14s}{4k}$	$\frac{0.92s}{3k}$
8	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{7.76s}{18k}$	$\frac{6.83s}{16k}$	$\frac{59.26s}{73k}$	$\frac{41.87s}{73k}$	$\frac{8.45s}{12k}$	$\frac{6.71s}{11k}$
9	$\frac{0.01s}{1}$	$\frac{0.04s}{1}$	$\frac{23.13s}{37k}$	$\frac{20.18s}{34k}$	$\frac{128.94s}{172k}$	$\frac{117.70s}{158k}$	$\frac{27.07s}{30k}$	$\frac{22.58s}{28k}$
10	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{42.58s}{36k}$	$\frac{35.69s}{33k}$	$\frac{302.00s}{205k}$	$\frac{256.44s}{171k}$	$\frac{63.16s}{52k}$	$\frac{53.73s}{47k}$
11	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{7.44s}{3k}$	$\frac{6.18s}{2k}$	$\frac{131.23s}{151k}$	$\frac{120.87s}{140k}$	$\frac{17.59s}{15k}$	$\frac{16.14s}{14k}$
12	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.26s}{121}$	$\frac{1.01s}{106}$	$\frac{10.29s}{2k}$	$\frac{8.91s}{846}$	$\frac{2.13s}{205}$	$\frac{1.72s}{177}$

(b) Evaluation with random IA networks of model $\mathsf{BA}(\mathsf{n} = 150, \mathsf{m})$ [40]

	min		μ		max		σ	
m	PSWC	$\text{PSWC}^{\cup}$	PSWC	$\text{PSWC}^{\cup}$	PSWC	$\text{PSWC}^{\cup}$	PSWC	$\text{PSWC}^{\cup}$
2	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.70s}{5k}$	$\frac{0.63s}{5k}$	$\frac{2.52s}{12k}$	$\frac{2.16s}{10k}$	$\frac{0.35s}{2k}$	$\frac{0.30s}{2k}$
3	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{11.51s}{19k}$	$\frac{9.63s}{16k}$	$\frac{74.09s}{81k}$	$\frac{52.93s}{58k}$	$\frac{11.15s}{11k}$	$\frac{7.95s}{8k}$
4	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{116.39s}{57k}$	$\frac{82.29s}{44k}$	$\frac{1438.72s}{552k}$	$\frac{881.27s}{348k}$	$\frac{208.09s}{97k}$	$\frac{144.64s}{74k}$
5	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.93s}{24}$	$\frac{0.78s}{22}$	$\frac{24.08s}{585}$	$\frac{24.74s}{569}$	$\frac{3.91s}{96}$	$\frac{3.33s}{91}$

yield *incomparable outputs*; this measure in particular is denoted by symbol $\#\|$ and we present it only whenever applicable (for instance, in Tables 7 and 10).

7.1. Comparing $\text{PSWC}^{\cup}$ *with* PSWC

With respect to IA, the results of our experimental evaluation are detailed in Tables 1 and 2, where a fraction $\frac{x}{y}$ denotes that an approach required x seconds of CPU time and performed y constraint checks per base relation removals on average per dataset of networks during its operation. Regarding computational

Table 2: Evaluation of the pruning capability of algorithm PSWC$^{\cup}$ compared to that of PSWC with the random IA networks of Table 1; a percentage of $x\%$ denotes that PSWC$^{\cup}$ removed $x\%$ more base relations with respect to the involved dataset

(a) Evaluation with the IA networks used in Table 1a

d	min	μ	max	med
7	0%	0%	0%	0%
8	0%	0.01%	0.40%	0%
9	0%	0.06%	1.28%	0%
10	0%	0.08%	1.76%	0%
11	0%	0%	0%	0%
12	0%	0%	0%	0%

(b) Evaluation with the IA networks used in Table 1b

m	min	μ	max	med
2	0%	0%	0%	0%
3	0%	$> 0\%$	0.19%	0%
4	0%	0.10%	3.02%	0%
5	0%	0%	0%	0%

effort, Table 1 shows that PSWC$^{\cup}$ outperformed PSWC in all cases by at least 10% on average and, in particular, that PSWC$^{\cup}$ was around 30% faster than PSWC on average for the more difficult instances (see the row for $\mathsf{m} = 4$ in Table 1b). Regarding pruning capability, Table 2, and the median value in particular, suggests that there is no difference between PSWC$^{\cup}$ and PSWC for the majority of the network instances. Of course, we note that this is not an issue of the implementation of the PSWC$^{\cup}$ algorithm itself, but of its underlying consistency, viz., $\blacklozenge^{\cup}_{G}$-consistency; thus, $\blacklozenge^{\cup}_{G}$-consistency and $\blacklozenge_{G}$-consistency are very close to one another in terms of the refinement that they achieve in a given QCN of IA. Nevertheless, due to the proactive nature of the collective singleton checks, PSWC$^{\cup}$ is able to refine a QCN more efficiently than PSWC and can sometimes achieve marginally improved pruning compared to that of PSWC. As an example, the reader may consider the row for $\mathsf{m} = 4$ in Table 2b and the row for $\mathsf{d} = 10$ in Table 2a.

Table 3: Evaluation of the computational effort of algorithms PSWC and PSWC$^{\cup}$ with random RCC8 networks

(a) Evaluation with random RCC8 networks of model $\mathsf{A}(\mathsf{n} = 100, \mathsf{l} = 4.0, \mathsf{d})$ [43]

	min		μ		max		σ	
d	PSWC	PSWC$^{\cup}$	PSWC	PSWC$^{\cup}$	PSWC	PSWC$^{\cup}$	PSWC	PSWC$^{\cup}$
7	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.57s}{3k}$	$\frac{1.42s}{2k}$	$\frac{3.53s}{5k}$	$\frac{3.34s}{4k}$	$\frac{1.05s}{2k}$	$\frac{0.96s}{2k}$
8	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{3.00s}{3k}$	$\frac{2.63s}{3k}$	$\frac{10.11s}{10k}$	$\frac{8.41s}{7k}$	$\frac{2.31s}{2k}$	$\frac{2.05s}{2k}$
9	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{4.65s}{3k}$	$\frac{4.15s}{3k}$	$\frac{19.75s}{12k}$	$\frac{16.90s}{9k}$	$\frac{4.68s}{3k}$	$\frac{4.15s}{3k}$
10	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{4.93s}{3k}$	$\frac{4.39s}{3k}$	$\frac{20.61s}{12k}$	$\frac{17.31s}{11k}$	$\frac{5.38s}{4k}$	$\frac{4.74s}{3k}$
11	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{5.01s}{3k}$	$\frac{4.38s}{3k}$	$\frac{44.25s}{21k}$	$\frac{29.91s}{14k}$	$\frac{8.39s}{4k}$	$\frac{6.99s}{4k}$
12	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.85s}{724}$	$\frac{1.69s}{657}$	$\frac{27.02s}{11k}$	$\frac{22.53s}{9k}$	$\frac{5.65s}{3k}$	$\frac{5.11s}{2k}$

(b) Evaluation with random RCC8 networks of model $\mathsf{BA}(\mathsf{n} = 200, \mathsf{m})$ [40]

	min		μ		max		σ	
m	PSWC	PSWC$^{\cup}$	PSWC	PSWC$^{\cup}$	PSWC	PSWC$^{\cup}$	PSWC	PSWC$^{\cup}$
2	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.40s}{2k}$	$\frac{0.37s}{2k}$	$\frac{1.06s}{4k}$	$\frac{0.93s}{3k}$	$\frac{0.26s}{724}$	$\frac{0.23s}{601}$
3	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{2.79s}{3k}$	$\frac{2.53s}{2k}$	$\frac{9.33s}{8k}$	$\frac{8.62s}{8k}$	$\frac{2.96s}{3k}$	$\frac{2.68s}{2k}$
4	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.85s}{554}$	$\frac{1.64s}{487}$	$\frac{29.28s}{9k}$	$\frac{28.56s}{8k}$	$\frac{5.82s}{2k}$	$\frac{5.16s}{2k}$
5	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.01s}{1}$	$\frac{0.02s}{1}$	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$

With respect to RCC8, the results of our experimental evaluation are detailed in Tables 3 and 4 and are qualitatively similar to those of Tables 1 and 2 for IA in most cases. The main difference lies in the fact that the pruning capability of PSWC$^{\cup}$ is exactly the same as that of PSWC for the network instances that were generated using model $\mathsf{BA}(\mathsf{n}, \mathsf{m})$ (see Table 4b). This came as a surprise, since in the case of IA this was the model for which PSWC$^{\cup}$ had the best pruning performance. However, as demonstrated in Table 3b, algorithm PSWC$^{\cup}$ was still able to outperform algorithm PSWC by around 10% on average for the more

Table 4: Evaluation of the pruning capability of algorithm $\mathsf{PSWC}^{\cup}$ compared to that of PSWC with the random $\mathsf{RCC8}$ networks of Table 3; a percentage of $x\%$ denotes that $\mathsf{PSWC}^{\cup}$ removed $x\%$ more base relations with respect to the involved dataset

(a) Evaluation with the RCC8 networks used in Table 3a

d	min	μ	max	med
7	0%	0%	0%	0%
8	0%	0%	0%	0%
9	0%	$> 0\%$	0.10%	0%
10	0%	0.01%	1.29%	0%
11	0%	$> 0\%$	0.04%	0%
12	0%	0%	0%	0%

(b) Evaluation with the RCC8 networks used in Table 3b

m	min	μ	max	med
2	0%	0%	0%	0%
3	0%	0%	0%	0%
4	0%	0%	0%	0%
5	0%	0%	0%	0%

time-consuming instances. Further, algorithm $\mathsf{PSWC}^{\cup}$ was able to outperform algorithm PSWC by even more than that on average for the more time-consuming instances that were generated using model $\mathsf{A}(\mathsf{n}, \mathsf{l}, \mathsf{d})$, as shown in Table 3a.

For all intents and purposes, and with respect to the involved datasets here, we can confidently say that between $\mathsf{PSWC}^{\cup}$ and PSWC, the $\mathsf{PSWC}^{\cup}$ algorithm is always the better choice. In particular, $\mathsf{PSWC}^{\cup}$ is always faster, and always removes at least as many base relations as PSWC does, since it enforces $\overset{\blacklozenge}{_G}{}^{\cup}$-consistency, a stricter consistency than the $\overset{\blacklozenge}{_G}$-consistency enforced by PSWC. Thus, $\mathsf{PSWC}^{\cup}$ represents a clear advancement of the state-of-the-art and can be seen as a better alternative to PSWC.

7.2. Evaluating $\ell\mathsf{PSWC}^{\cup}$: where does it fit among $\mathsf{PSWC}^{\cup}$ and PSWC?

In the previous section we showed that $\mathsf{PSWC}^{\cup}$ is a better alternative to PSWC with respect to both computational effort and pruning capability. In this section, and with respect to computational effort, it suffices to use algorithm $\mathsf{PSWC}^{\cup}$

Table 5: Evaluation of the computational effort of algorithms $\mathsf{PSWC}^{\cup}$ and $\ell\mathsf{PSWC}^{\cup}$ with random IA networks

(a) Evaluation with random IA networks of model $\mathsf{A}(\mathsf{n}=70, \mathsf{l}=6.5, \mathsf{d})$ [43]

	min		μ		max		σ	
d	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$
7	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{2.21s}{9k}$	$\frac{0.36s}{2k}$	$\frac{4.94s}{16k}$	$\frac{1.19s}{4k}$	$\frac{0.91s}{3k}$	$\frac{0.19s}{528}$
8	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{6.72s}{16k}$	$\frac{1.41s}{4k}$	$\frac{42.26s}{73k}$	$\frac{17.73s}{16k}$	$\frac{6.59s}{11k}$	$\frac{2.14s}{3k}$
9	$\frac{0.01s}{1}$	$\frac{0.04s}{1}$	$\frac{20.27s}{34k}$	$\frac{4.98s}{8k}$	$\frac{119.15s}{158k}$	$\frac{73.15s}{75k}$	$\frac{22.93s}{28k}$	$\frac{10.34s}{11k}$
10	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{35.64s}{33k}$	$\frac{8.92s}{9k}$	$\frac{254.70s}{171k}$	$\frac{106.09s}{90k}$	$\frac{53.40s}{47k}$	$\frac{19.04s}{16k}$
11	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{5.21s}{2k}$	$\frac{1.40s}{462}$	$\frac{95.65s}{140k}$	$\frac{15.57s}{9k}$	$\frac{13.30s}{14k}$	$\frac{2.82s}{2k}$
12	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.07s}{106}$	$\frac{0.36s}{39}$	$\frac{9.34s}{846}$	$\frac{5.39s}{623}$	$\frac{1.82s}{177}$	$\frac{0.80s}{89}$

(b) Evaluation with random IA networks of model $\mathsf{BA}(\mathsf{n}=150, \mathsf{m})$ [40]

	min		μ		max		σ	
m	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$
2	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.55s}{5k}$	$\frac{0.11s}{741}$	$\frac{1.89s}{10k}$	$\frac{0.49s}{3k}$	$\frac{0.26s}{2k}$	$\frac{0.06s}{266}$
3	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{9.64s}{16k}$	$\frac{2.32s}{4k}$	$\frac{52.56s}{58k}$	$\frac{18.74s}{23k}$	$\frac{7.89s}{8k}$	$\frac{3.02s}{4k}$
4	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{81.35s}{44k}$	$\frac{31.03s}{17k}$	$\frac{874.06s}{348k}$	$\frac{589.13s}{238k}$	$\frac{143.24s}{74k}$	$\frac{74.64s}{34k}$
5	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.77s}{22}$	$\frac{0.04s}{2}$	$\frac{24.75s}{569}$	$\frac{1.43s}{42}$	$\frac{3.31s}{91}$	$\frac{0.17s}{6}$

as a basis of comparison for evaluating algorithm $\ell\mathsf{PSWC}^{\cup}$. In particular, as we will show $\ell\mathsf{PSWC}^{\cup}$ to be substantially faster than $\mathsf{PSWC}^{\cup}$, the exact conclusions that will be drawn regarding computational effort will be the same as those that would have resulted from a comparison between $\ell\mathsf{PSWC}^{\cup}$ and PSWC. On the other hand, regarding pruning capability, $\ell\mathsf{PSWC}^{\cup}$, being a lazy algorithmic variant of $\mathsf{PSWC}^{\cup}$ (so, a weaker version of it), produces outputs that are directly comparable to those of $\mathsf{PSWC}^{\cup}$, but not directly comparable to those of PSWC. (See Theorem 4 again regarding the last statement.) For this reason, in this case,

Table 6: Evaluation of the pruning capability of algorithm ℓPSWC$^{\cup}$ compared to that of PSWC$^{\cup}$ with the random IA networks of Table 5; a percentage of $x\%$ denotes that ℓPSWC$^{\cup}$ removed $x\%$ more base relations with respect to the involved dataset

(a) Evaluation with the IA networks used in Table 5a

d	min	μ	max	med
7	−2.97%	−0.07%	0%	0%
8	−6.18%	−0.59%	0%	−0.20%
9	−37.09%	−3.64%	0%	−1.58%
10	−95.01%	−10.63%	0%	0%
11	−95.49%	−3.21%	0%	0%
12	0%	0%	0%	0%

(b) Evaluation with the IA networks used in Table 5b

m	min	μ	max	med
2	0%	0%	0%	0%
3	−3.94%	−0.22%	0%	0%
4	−74.43%	−1.45%	0%	0%
5	0%	0%	0%	0%

along with tables comparing the pruning capability between algorithms ℓPSWC$^{\cup}$ and PSWC$^{\cup}$, we will include tables comparing the pruning capability between algorithms ℓPSWC$^{\cup}$ and PSWC as well.

With respect to IA, the results of our experimental evaluation are detailed in Tables 5, 6, and 7. Regarding computational effort, Table 5 shows that ℓPSWC$^{\cup}$ substantially outperformed PSWC$^{\cup}$ in all cases and, in particular, that ℓPSWC$^{\cup}$ was around 4 times faster than PSWC$^{\cup}$ on average for the more difficult instances (see the row for d $= 10$ in Table 5a). Regarding pruning capability, Tables 6 and 7, and the median value in particular, suggest that there is only a slight difference between ℓPSWC$^{\cup}$ and PSWC$^{\cup}$ and PSWC respectively for the majority of the network instances. In particular, for the majority of instances, in the worst case (see the row for d $= 9$ in Tables 6a and 7a), the difference is less than -1.58%, i.e., ℓPSWC$^{\cup}$ removes just up to 1.58% less base relations than PSWC$^{\cup}$ (and PSWC) for the majority of instances, although it can remove up

Table 7: Evaluation of the pruning capability of algorithm $\ell\mathsf{PSWC}^{\cup}$ compared to that of PSWC with the random IA networks of Table 5; a percentage of $x\%$ denotes that $\ell\mathsf{PSWC}^{\cup}$ removed $x\%$ more base relations with respect to the involved dataset

(a) Evaluation with the IA networks used in Table 5a

d	min	μ	max	med	$\#\|$
7	-2.97%	-0.07%	0%	0%	0
8	-6.18%	-0.58%	0%	-0.09%	5
9	-36.99%	-3.59%	0.03%	-1.58%	14
10	-95.01%	-10.57%	0.18%	0%	10
11	-95.49%	-3.21%	0%	0%	0
12	0%	0%	0%	0%	0

(b) Evaluation with the IA networks used in Table 5b

m	min	μ	max	med	$\#\|$
2	0%	0%	0%	0%	0
3	-3.94%	-0.22%	0%	0%	4
4	-74.41%	-1.36%	0.03%	0%	23
5	0%	0%	0%	0%	0

to 10.63% less base relations than $\mathsf{PSWC}^{\cup}$ on average (see the row for $\mathsf{d} = 10$ in Table 6a). It is important to note that $\mathsf{PSWC}^{\cup}$ and PSWC unveiled 6 more inconsistencies than $\ell\mathsf{PSWC}^{\cup}$ in a total of 1 000 QCNs, in particular, 3 more for $\mathsf{A}(70, 6.5, 10)$ and $\mathsf{A}(70, 6.5, 11)$ respectively; this also explains the rather high minimum percentage values in Tables 6a and 7a for $\mathsf{d} \in \{10, 11\}$. Further, as demonstrated by measure $\#\|$ in Table 7, there were 56 cases of incomparable outputs between $\ell\mathsf{PSWC}^{\cup}$ and PSWC in a total of 1 000 QCNs. (We remind the reader that all outputs between $\ell\mathsf{PSWC}^{\cup}$ and $\mathsf{PSWC}^{\cup}$ are comparable, that is why this measure is not included in Table 6).

With respect to $\mathsf{RCC8}$, the results of our experimental evaluation are detailed in Tables 8, 9, and 10. Regarding computational effort, Table 8 shows that $\ell\mathsf{PSWC}^{\cup}$ substantially outperformed $\mathsf{PSWC}^{\cup}$ in all cases and, in particular, that $\ell\mathsf{PSWC}^{\cup}$ was around 5 times faster than $\mathsf{PSWC}^{\cup}$ on average for the more difficult instances (see the row for $\mathsf{d} = 10$ in Table 8a). Regarding pruning

Table 8: Evaluation of the computational effort of algorithms $\mathsf{PSWC}^{\cup}$ and $\ell\mathsf{PSWC}^{\cup}$ with random $\mathsf{RCC8}$ networks

(a) Evaluation with random $\mathsf{RCC8}$ networks of model $\mathsf{A}(\mathsf{n} = 100, \mathsf{l} = 4.0, \mathsf{d})$ [43]

	min		μ		max		σ	
d	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$
7	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.42s}{2k}$	$\frac{0.21s}{282}$	$\frac{3.36s}{4k}$	$\frac{0.50s}{679}$	$\frac{0.95s}{2k}$	$\frac{0.14s}{182}$
8	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{2.57s}{3k}$	$\frac{0.43s}{403}$	$\frac{8.33s}{7k}$	$\frac{2.07s}{3k}$	$\frac{2.01s}{2k}$	$\frac{0.39s}{343}$
9	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{3.44s}{3k}$	$\frac{0.67s}{492}$	$\frac{14.12s}{9k}$	$\frac{4.11s}{3k}$	$\frac{3.46s}{3k}$	$\frac{0.77s}{541}$
10	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{4.45s}{3k}$	$\frac{0.98s}{592}$	$\frac{17.32s}{11k}$	$\frac{6.09s}{4k}$	$\frac{4.80s}{3k}$	$\frac{1.17s}{695}$
11	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{4.47s}{3k}$	$\frac{1.12s}{531}$	$\frac{30.86s}{14k}$	$\frac{8.03s}{4k}$	$\frac{7.15s}{4k}$	$\frac{1.84s}{875}$
12	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.72s}{657}$	$\frac{0.47s}{182}$	$\frac{22.50s}{9k}$	$\frac{6.38s}{3k}$	$\frac{5.22s}{2k}$	$\frac{1.46s}{557}$

(b) Evaluation with random $\mathsf{RCC8}$ networks of model $\mathsf{BA}(\mathsf{n} = 200, \mathsf{m})$ [40]

	min		μ		max		σ	
m	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$	$\mathsf{PSWC}^{\cup}$	$\ell\mathsf{PSWC}^{\cup}$
2	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.48s}{2k}$	$\frac{0.08s}{187}$	$\frac{1.20s}{3k}$	$\frac{0.25s}{412}$	$\frac{0.30s}{601}$	$\frac{0.05s}{98}$
3	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{2.54s}{2k}$	$\frac{0.42s}{310}$	$\frac{8.60s}{8k}$	$\frac{1.74s}{2k}$	$\frac{2.70s}{2k}$	$\frac{0.48s}{345}$
4	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{1.65s}{487}$	$\frac{0.34s}{106}$	$\frac{28.50s}{8k}$	$\frac{5.53s}{2k}$	$\frac{5.18s}{2k}$	$\frac{1.09s}{336}$
5	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$	$\frac{0.01s}{1}$	$\frac{0.02s}{1}$	$\frac{0.00s}{1}$	$\frac{0.00s}{1}$

capability, Tables 9 and 10, and the median value in particular, suggest that there is no difference between $\ell\mathsf{PSWC}^{\cup}$ and $\mathsf{PSWC}^{\cup}$ and PSWC respectively for the majority of the network instances, i.e., $\ell\mathsf{PSWC}^{\cup}$ removes 0% less base relations than $\mathsf{PSWC}^{\cup}$ (and PSWC) for the majority of instances. In addition, the low minimum percentage values in Tables 9 and 10 suggest that this robust pruning performance applies to the entirety of the network instances of $\mathsf{RCC8}$; this shows that the pruning capability of $\ell\mathsf{PSWC}^{\cup}$ is excellent regarding $\mathsf{RCC8}$. It is important to note that both $\mathsf{PSWC}^{\cup}$ and PSWC did not unveil any more

Table 9: Evaluation of the pruning capability of algorithm ℓPSWC$^{\cup}$ compared to that of PSWC$^{\cup}$ with the random RCC8 networks of Table 8; a percentage of x% denotes that ℓPSWC$^{\cup}$ removed x% more base relations with respect to the involved dataset

(a) Evaluation with the RCC8 networks used in Table 8a

d	min	μ	max	med
7	-0.67%	-0.01%	0%	0%
8	-0.24%	-0.01%	0%	0%
9	-0.20%	-0.02%	0%	0%
10	-0.22%	-0.02%	0%	0%
11	-0.24%	-0.02%	0%	0%
12	-0.30%	-0.01%	0%	0%

(b) Evaluation with the RCC8 networks used in Table 8b

m	min	μ	max	med
2	0%	0%	0%	0%
3	-0.13%	< 0%	0%	0%
4	-0.06%	< 0%	0%	0%
5	0%	0%	0%	0%

inconsistencies than ℓPSWC$^{\cup}$. Further, as demonstrated by measure $\#\|$ in Table 10, there were just 3 cases of incomparable outputs between ℓPSWC$^{\cup}$ and PSWC in a total of 1 000 QCNs.

All in all, and with respect to the involved datasets here, we can safely state that the pruning capability of ℓPSWC$^{\cup}$ is excellent for the entirety of the network instances of RCC8, but also for the majority of the network instances of the Interval Algebra (as the median value in Tables 6 and 7 suggests). Further, regarding computational effort, PSWC$^{\cup}$ and PSWC are no match for ℓPSWC$^{\cup}$, as ℓPSWC$^{\cup}$ is up to 5 times faster than the aforementioned algorithms. However, choosing ℓPSWC$^{\cup}$ over PSWC$^{\cup}$ (or PSWC) cannot be directly advised, as ℓPSWC$^{\cup}$ produces non-unique outputs in general (see Theorem 4) and, therefore, cannot guarantee minimality of a given QCN for certain subclasses of relations in the way that PSWC$^{\cup}$ or PSWC do (see again Propositions 4 and 10 and the surrounding text). In the cases where certain properties (such as minimality)

Table 10: Evaluation of the pruning capability of algorithm $\ell\mathsf{PSWC}^{\cup}$ compared to that of PSWC with the random RCC8 networks of Table 8; a percentage of $x\%$ denotes that $\ell\mathsf{PSWC}^{\cup}$ removed $x\%$ more base relations with respect to the involved dataset

(a) Evaluation with the RCC8 networks used in Table 8a

d	min	μ	max	med	$\#\|$
7	-0.67%	-0.01%	0%	0%	0
8	-0.24%	-0.01%	0%	0%	0
9	-0.20%	-0.01%	0.10%	0%	0
10	-0.22%	-0.01%	1.18%	0%	1
11	-0.24%	-0.02%	0%	0%	2
12	-0.30%	-0.01%	0%	0%	0

(b) Evaluation with the RCC8 networks used in Table 8b

m	min	μ	max	med	$\#\|$
2	0%	0%	0%	0%	0
3	-0.13%	$< 0\%$	0%	0%	0
4	-0.06%	$< 0\%$	0%	0%	0
5	0%	0%	0%	0%	0

can be approximated rather than strictly enforced, $\ell\mathsf{PSWC}^{\cup}$ is the preferred choice.

7.3. *Evaluating the utility of* $\mathsf{PSWC}^{\cup}$ *(and* $\ell\mathsf{PSWC}^{\cup}$*) for* satisfiability checking *and* minimal labeling *of* QCN*s*

In this section we expirimentally investigate the usefulness (if any) of $\mathsf{PSWC}^{\cup}$ with respect to the fundamental reasoning problems of *satisfiability checking* and *minimal labeling*, which are typically associated with QCNs. In particular, we would like to see the efficiency of $\mathsf{PSWC}^{\cup}$ in determining the satisfiability of a given network instance and in discovering the unfeasible base relations of that instance (in regard to both CPU time and correctness of the respective decision). We focus on $\mathsf{PSWC}^{\cup}$ only, as the results for $\ell\mathsf{PSWC}^{\cup}$ and even PSWC can be drawn from the data that were presented in earlier sections. However, we will explicitly comment on how certain key findings carry over to $\ell\mathsf{PSWC}^{\cup}$ and PSWC where appropriate.

Table 11: Evaluation of the *satisfiability checking* and *minimal labeling* capacity of algorithm $\mathsf{PSWC}^{\cup}$ with random IA networks

(a) Evaluation with random IA networks of model $\mathsf{A}(\mathsf{n} = 70, \mathsf{l} = 6.5, \mathsf{d})$ [43]

d	Solver	Minimizer	$\mathsf{PSWC}^{\cup}$
7	$\frac{0.18s}{2}$	$\frac{12.58s}{3.84\%}$	$\frac{2.21s}{2\|3.84\%}$
8	$\frac{0.21s}{5}$	$\frac{26.27s}{8.75\%}$	$\frac{6.81s}{5\|8.72\%}$
9	$\frac{0.35s}{6}$	$\frac{301.42s}{13.67\%}$	$\frac{21.20s}{6\|12.31\%}$
10	$\frac{2.13s}{55}$	$\frac{1\,968.86s}{70.57\%}$	$\frac{37.99s}{54\|\frac{64.13\%}{(65.01\%)}}$
11	$\frac{5.77s}{100}$	$\frac{7.67s}{100\%}$	$\frac{6.23s}{99\|\frac{98.97\%}{(100\%)}}$
12	$\frac{0.30s}{100}$	$\frac{1.22s}{100\%}$	$\frac{1.04s}{100\|100\%}$

(b) Evaluation with random IA networks of model $\mathsf{BA}(\mathsf{n} = 150, \mathsf{m})$ [40]

m	Solver	Minimizer	$\mathsf{PSWC}^{\cup}$
2	$\frac{0.16s}{2}$	$\frac{8.18s}{3.14\%}$	$\frac{0.64s}{2\|3.14\%}$
3	$\frac{0.20s}{7}$	$\frac{44.49s}{9.42\%}$	$\frac{10.00s}{7\|9.42\%}$
4	$\frac{0.27s}{60}$	$\frac{130.00s}{66.89\%}$	$\frac{87.95s}{60\|66.83\%}$
5	$\frac{0.18s}{100}$	$\frac{0.91s}{100\%}$	$\frac{0.76s}{100\%}$

In this third phase of experimentation we utilize two additional software tools, which are presented as follows:

- Solver, the state-of-the-art reasoner for checking the satisfiability of QCNs of Interval Algebra and RCC8 that was introduced in [40] (and in particular the reasoner called Phalanx$\triangledown$ in that work);
- Minimizer, our own implementation for the sake of this experimental evaluation of the approach for solving the minimal labeling problem of QCNs of Interval Algebra and RCC8 that was first presented in [38].[9]

With respect to IA, the results of our experimental evaluation are detailed in Table 11, where a fraction $\frac{x}{y}$ for Solver denotes that it required x seconds of CPU time on average per dataset of networks during its operation and detected y instances as being unsatisfiable in total, a fraction $\frac{x}{z}$ for Minimizer denotes

[9] In particular, we ported the code to Python and included all recent advances that are associated with the components that comprise that approach, such as improvements in its underlying satisfiability checking module.

that it determined $z\%$ of base relations to be unfeasible in total, and a fraction $\frac{x}{y|z}$ for PSWC$^{\cup}$ denotes all the previous information combined together (from the viewpoint of PSWC$^{\cup}$). Regarding computational effort, Table 11 shows that Solver has no competition whatsoever. This was expected, as this type of reasoner is tailored to avoid "bad" branches in the search tree and reach a leaf (i.e., a solution) in the most efficient way possible. On the other hand, when the entire search tree needs to be taken into account, as is the case with Minimizer, the computational task is much more time-consuming; therefore, Minimizer has by far the worst performance among its competition (yet, substantially better performance than a naive approach for solving the minimal labeling problem [38]). Regarding PSWC$^{\cup}$, we can note that it of course presents an overhead with respect to Solver, but it is much faster in general than Minimizer. In addition, PSWC$^{\cup}$ fails to correctly determine the unsatisfiability of only 2 out of 1 000 network instances, and is able to discover all unfeasible base relations in most cases, i.e., it simulates the output of Minimizer in an almost exact manner. If we further aid PSWC$^{\cup}$ by using Solver to inform the algorithm about the satisfiability or unsatisfiability of an instance and, consequently, have PSWC$^{\cup}$ zero out those 2 out of 1 000 network instances, then the end result with respect to the percentage of unfeasible base relations discovered is even closer to that of Minimizer; these numbers are given in parentheses in the table. We remind the reader that PSWC$^{\cup}$ and PSWC detect the same number of unsatisfiable network instances for this dataset of Interval Algebra and that they unveil 6 more inconsistencies than ℓPSWC$^{\cup}$ in a total of 1 000 QCNs, in particular, 3 more for $\mathsf{A}(70, 6.5, 10)$ and $\mathsf{A}(70, 6.5, 11)$ respectively.

With respect to RCC8, the results of our experimental evaluation are detailed in Table 12, Regarding computational effort, we again have that Solver is by far the fastest tool, as it is only burdened with the ask of deciding whether a network instance is satisfiable or not. Further, Minimizer has again the worst performance among its competition by a large margin. Regarding PSWC$^{\cup}$, the most notable differences with respect to how it performed in the case of Interval Algebra network instances, are that in this case it correctly determines

Table 12: Evaluation of the *satisfiability checking* and *minimal labeling* capacity of algorithm $\mathsf{PSWC}^{\cup}$ with random RCC8 networks

(a) Evaluation with random RCC8 networks of model $\mathsf{A}(\mathsf{n} = 100, \mathsf{l} = 4.0, \mathsf{d})$ [43]

d	Solver	Minimizer	$\mathsf{PSWC}^{\cup}$
7	$\frac{0.01s}{25}$	$\frac{4.56s}{27.47\%}$	$\frac{1.44s}{25 \mid 27.47\%}$
8	$\frac{0.02s}{31}$	$\frac{7.64s}{34.22\%}$	$\frac{2.65s}{31 \mid 34.22\%}$
9	$\frac{0.02s}{45}$	$\frac{9.41s}{48.66\%}$	$\frac{4.16s}{45 \mid 48.66\%}$
10	$\frac{0.02s}{50}$	$\frac{140.27s}{54.65\%}$	$\frac{4.73s}{50 \mid 54.65\%}$
11	$\frac{0.02s}{69}$	$\frac{11.75s}{72.85\%}$	$\frac{4.52s}{69 \mid 72.85\%}$
12	$\frac{0.01s}{90}$	$\frac{3.39s}{91.50\%}$	$\frac{1.74s}{90 \mid 91.50\%}$

(b) Evaluation with random RCC8 networks of model $\mathsf{BA}(\mathsf{n} = 200, \mathsf{m})$ [40]

m	Solver	Minimizer	$\mathsf{PSWC}^{\cup}$
2	$\frac{0.01s}{15}$	$\frac{1.26s}{16.97\%}$	$\frac{0.38s}{15 \mid 16.97\%}$
3	$\frac{0.01s}{49}$	$\frac{6.56s}{50.92\%}$	$\frac{2.59s}{49 \mid 50.92\%}$
4	$\frac{0.01s}{90}$	$\frac{5.05s}{91.17\%}$	$\frac{1.65s}{90 \mid 91.17\%}$
5	$\frac{0.00s}{100}$	$\frac{0.01s}{100\%}$	$\frac{0.00s}{100\%}$

the unsatisfiability of *all* 1 000 network instances, and is able to discover all unfeasible base relations in *all* cases. Therefore, for this particular dataset of RCC8 network instances, $\mathsf{PSWC}^{\cup}$ can serve as a direct replacement for Minimizer. Taking into account the fact that $\mathsf{PSWC}^{\cup}$ can be 30 times faster than Minimizer (see for example the row for $\mathsf{d} = 10$ in Table 12a), $\mathsf{PSWC}^{\cup}$ appears to be an excellent choice for solving the minimal labeling problem of QCNs of RCC8. We remind the reader that $\mathsf{PSWC}^{\cup}$, PSWC, and $\ell\mathsf{PSWC}^{\cup}$ all detect the same number of unsatisfiable network instances for this dataset of RCC8.

In conclusion, and with respect to the involved datasets here, we can deduce that $\mathsf{PSWC}^{\cup}$ (and $\ell\mathsf{PSWC}^{\cup}$) are not good options for just checking the satisfiability of a network instance, as they present an overhead when compared to a state-of-the-art reasoner that is tailored to this specific task. However, we can also deduce that they are ideal candidates for efficiently approximating and even determining in most cases the minimal labeling of a network instance, especially when coupled with a satisfiability checker to deal with the rare cases where the singleton consistencies will fail to determine the unsatisfiability of a network instance.

8. Conclusion and future work

Partial singleton weak path-consistency, or partial ◆-consistency for short, is essential for tackling challenging fundamental reasoning problems associated with qualitative constraints networks. Briefly put, partial ◆-consistency ensures that each base relation of each of the constraints of a qualitative constraint network can define a singleton relation in its corresponding partially weakly path-consistent, or partially ◇-consistent for short, subnetwork. Further, partial ◆-consistency has been shown to play a crucial role in tackling the minimal labeling problem of a qualitative constraint network in particular, which is the problem of finding the strongest implied constraints of that network. In this paper, we proposed a stronger local consistency that couples ◆-consistency with the idea of collectively deleting certain unfeasible base relations by exploiting singleton checks. We then proposed an algorithm for enforcing this new consistency and a lazy variant of that algorithm for approximating the new consistency that, given a qualitative constraint network, both outperform the respective algorithm for enforcing partial ◆-consistency in that network. With respect to the lazy algorithmic variant in particular, we showed that it runs up to 5 times faster than our original exhaustive algorithm whilst exhibiting very similar pruning capability. We formally proved certain properties of our new local consistency and our algorithms, and motivated their usefulness through demonstrative examples and a thorough experimental evaluation with random qualitative constraint networks of the Interval Algebra and the Region Connection Calculus from the phase transition region of two different generation models.

There are several directions for future work. Regarding the algorithm that enforces our new consistency, we would like to explore queuing strategies such that the singleton checks are applied in a more fruitful manner. In particular, it would make sense to prioritize certain singleton checks that are more likely to eliminate base relations anywhere in the network at hand, because this could unveil certain inconsistencies faster, but also lead to fewer constraint checks overall. Such strategies have been used in the case of partial ◇-consistency [67, 42, 68]. Further, regarding the new local consistency itself, we would like to define a weaker variant of it that considers singleton checks in the *neighborhood* of the constraint in question, instead of the entire network. Early experiments in this direction have shown really promising results with respect to constraint satisfaction problems, which is due to the fact that constraint revisions tend to propagate themselves to just neighboring constraints [50].

Acknowledgements

We would like to thank Prof. Sven Schewe, Prof. Thomas Schneider, and Prof. Jef Wijsen, the program committee chairs of the 24^{th} International Symposium on Temporal Representation and Reasoning (TIME 2017) for selecting our work that was published in the proceedings of that conference (see [78]) as a candidate for inclusion in this special issue of the Elsevier Journal of Theoretical Computer Science (TCS). In addition, in this manuscript we have also revised preliminary material that appears in another conference paper (cf. [79]). Finally, we would also like to thank the reviewers for their thoughtful comments and efforts towards improving our manuscript.

References

[1] M. Bhatt, J. O. Wallgrün, Geospatial Narratives and Their Spatio-Temporal Dynamics: Commonsense Reasoning for High-Level Analyses in Geographic Information Systems, ISPRS Int. J. Geo-Information 3 (2014) 166–205.

[2] F. Dylla, J. O. Wallgrün, Qualitative Spatial Reasoning with Conceptual Neighborhoods for Agent Control, Journal of Intelligent and Robotic Systems 48 (2007) 55–78.

[3] N. Krishnaswamy, S. Friedman, J. Pustejovsky, Combining Deep Learning and Qualitative Spatial Reasoning to Learn Complex Structures from Sparse Examples with Noise, in: AAAI, 2019.

[4] P. A. Story, M. F. Worboys, A Design Support Environment for Spatio-Temporal Database Applications, in: COSIT, 1995.

[5] K. S. R. Dubba, A. G. Cohn, D. C. Hogg, M. Bhatt, F. Dylla, Learning Relational Event Models from Video, J. Artif. Intell. Res. 53 (2015) 41–90.

[6] M. Sioutis, M. Alirezaie, J. Renoux, A. 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.

[7] M. Bhatt, H. Guesgen, S. Wölfl, S. Hazarika, Qualitative Spatial and Temporal Reasoning: Emerging Applications, Trends, and Directions, Spatial Cognition & Computation 11 (2011) 1–14.

[8] F. Dylla, J. H. Lee, T. Mossakowski, T. Schneider, A. van Delden, J. van de Ven, D. Wolter, A Survey of Qualitative Spatial and Temporal Calculi: Algebraic and Computational Properties, ACM Comput. Surv. 50 (2017) 7:1–7:39.

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

[10] J. F. Allen, J. A. G. M. Koomen, Planning Using a Temporal World Model, in: IJCAI, 1983.

[11] J. F. Allen, Planning as Temporal Reasoning, in: KR, 1991.

[12] R. N. Pelavin, J. F. Allen, A Model for Concurrent Actions Having Temporal Extent, in: AAAI, 1987.

[13] J. Dorn, Dependable Reactive Event-Oriented Planning, Data Knowl. Eng. 16 (1995) 27–49.

[14] L. Mudrová, N. Hawes, Task scheduling for mobile robots using interval algebra, in: ICRA, 2015.

[15] F. Song, R. Cohen, The Interpretation of Temporal Relations in Narrative, in: IJCAI, 1988.

[16] P. Denis, P. Muller, Predicting Globally-Coherent Temporal Structures from Texts via Endpoint Inference and Graph Decomposition, in: IJCAI, 2011.

[17] R. T. Snodgrass, The Temporal Query Language TQuel, ACM Trans. Database Syst. 12 (1987) 247–298.

[18] C. X. Chen, C. Zaniolo, Universal Temporal Data Languages, in: DDLP, 1998.

[19] T. D. C. Little, A. Ghafoor, Interval-Based Conceptual Models for Time-Dependent Multimedia Data, IEEE Trans. Knowl. Data Eng. 5 (1993) 551–563.

[20] M. C. Golumbic, R. Shamir, Complexity and Algorithms for Reasoning about Time: A Graph-Theoretic Approach, J. ACM 40 (1993) 1108–1133.

[21] S. Benzer, On the Topology of the Genetic Fine Structure, Proc. Natl. Acad. Sci. USA 45 (1959) 1607–1620.

[22] R. Lu, S. W. Sadiq, V. Padmanabhan, G. Governatori, Using a temporal constraint network for business process execution, in: ADC, 2006.

[23] D. A. Randell, Z. Cui, A. Cohn, A Spatial Logic Based on Regions & Connection, in: KR, 1992.

[24] B. Bouzy, Les concepts spatiaux dans la programmation du go, Revue d'Intelligence Artificielle 15 (2001) 143–172.

[25] A. D. Lattner, I. J. Timm, M. Lorenz, O. Herzog, Knowledge-based risk assessment for intelligent vehicles, in: KIMAS, 2005.

[26] F. Heintz, D. de Leng, Spatio-Temporal Stream Reasoning with Incomplete Spatial Information, in: ECAI, 2014.

[27] D. A. Randell, A. Galton, S. Fouad, H. Mehanna, G. Landini, Mereotopological Correction of Segmentation Errors in Histological Imaging, J. Imaging 3 (2017) 63.

[28] V. Fenelon, P. E. Santos, H. M. Dee, F. G. Cozman, Reasoning about shadows in a mobile robot environment, Appl. Intell. 38 (2013) 553–565.

[29] P. Rost, L. Hotz, S. von Riegen, Supporting Mobile Robot's Tasks through Qualitative Spatial Reasoning, in: ICINCO, 2012.

[30] Z. Falomir, L. M. Cabedo, V. Castelló, L. G. Abril, Qualitative distances and qualitative image descriptions for representing indoor scenes in robotics, Pattern Recognition Letters 34 (2013) 731–743.

[31] M. Sridhar, A. G. Cohn, D. C. Hogg, From Video to RCC8: Exploiting a Distance Based Semantics to Stabilise the Interpretation of Mereotopological Relations, in: COSIT, 2011.

[32] P. Kordjamshidi, M. Moens, Global machine learning for spatial ontology population, J. Web Sem. 30 (2015) 3–21.

[33] P. Kordjamshidi, J. Hois, M. van Otterlo, M.-F. Moens, Machine learning for interpretation of spatial natural language in terms of QSR, in: COSIT (Extended abstract), 2011.

[34] G. Ligozat, J. Renz, What Is a Qualitative Calculus? A General Framework, in: PRICAI, 2004.

[35] J. Renz, B. Nebel, Qualitative Spatial Reasoning Using Constraint Calculi, in: Handbook of Spatial Logics, 2007, pp. 161–215.

[36] F. Dylla, T. Mossakowski, T. Schneider, D. Wolter, Algebraic Properties of Qualitative Spatio-Temporal Calculi, in: COSIT, 2013.

[37] J. Renz, G. Ligozat, Weak Composition for Qualitative Spatial and Temporal Reasoning, in: CP, 2005.

[38] N. Amaneddine, J.-F. Condotta, M. Sioutis, Efficient Approach to Solve the Minimal Labeling Problem of Temporal and Spatial Qualitative Constraints, in: IJCAI, 2013.

[39] M. Sioutis, S. Li, J.-F. Condotta, Efficiently Characterizing Non-Redundant Constraints in Large Real World Qualitative Spatial Networks, in: IJCAI, 2015.

[40] M. Sioutis, J. Condotta, M. Koubarakis, An Efficient Approach for Tackling Large Real World Qualitative Spatial Networks, Int. J. Artif. Intell. Tools 25 (2016) 1–33.

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

[42] J. Renz, B. Nebel, Efficient Methods for Qualitative Spatial Reasoning, J. Artif. Intell. Res. 15 (2001) 289–318.

[43] B. Nebel, Solving Hard Qualitative Temporal Reasoning Problems: Evaluating the Efficiency of Using the ORD-Horn Class, Constraints 1 (1997) 175–190.

[44] J. Huang, J. J. Li, J. Renz, Decomposition and tractability in qualitative spatial and temporal reasoning, Artif. Intell. 195 (2013) 140–164.

[45] J. Huang, Compactness and its implications for qualitative spatial and temporal reasoning, in: KR, 2012.

[46] J.-F. Condotta, C. Lecoutre, A Class of df-Consistencies for Qualitative Constraint Networks, in: KR, 2010.

[47] H. Bennaceur, M. Affane, Partition-k-AC: An Efficient Filtering Technique Combining Domain Partition and Arc Consistency, in: CP, 2001.

[48] C. Bessière, Arc-Consistency and Arc-Consistency Again, Artif. Intell. 65 (1994) 179–190.

[49] R. Debruyne, C. Bessière, Some Practicable Filtering Techniques for the Constraint Satisfaction Problem, in: IJCAI, 1997.

[50] R. J. Wallace, Neighbourhood SAC: Extensions and new algorithms, AI Commun. 29 (2016) 249–268.

[51] A. Paparrizou, K. Stergiou, On Neighborhood Singleton Consistencies, in: IJCAI, 2017.

[52] A. Balafrej, C. Bessiere, E. Bouyakhf, G. Trombettoni, Adaptive Singleton-Based Consistencies, in: AAAI, 2014, pp. 2601–2607.

[53] C. Bessière, R. Debruyne, Theoretical analysis of singleton arc consistency and its extensions, Artif. Intell. 172 (2008) 29–41.

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

[55] M. B. Vilain, H. A. Kautz, Constraint Propagation Algorithms for Temporal Reasoning, in: AAAI, 1986.

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

[57] A. U. Frank, Qualitative Spatial Reasoning with Cardinal Directions, in: ÖGAI, 1991.

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

[59] R. K. Goyal, M. J. Egenhofer, Consistent Queries over Cardinal Directions across Different Levels of Detail, in: DEXA Workshop, 2000.

[60] S. Skiadopoulos, M. Koubarakis, On the consistency of cardinal direction constraints, AIJ 163 (2005) 91–135.

[61] W. Liu, X. Zhang, S. Li, M. Ying, Reasoning about cardinal directions between extended objects, Artif. Intell. 174 (2010) 951–983.

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

[63] Z. Long, S. Li, On Distributive Subalgebras of Qualitative Spatial and Temporal Calculi, in: COSIT, 2015.

[64] R. Diestel, Graph Theory, 4th Edition, Vol. 173 of Graduate Texts in Mathematics, Springer, 2012.

[65] Z. Long, M. Sioutis, S. Li, Efficient Path Consistency Algorithm for Large Qualitative Constraint Networks, in: IJCAI, 2016.

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

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

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

[69] A. Chmeiss, J. Condotta, Consistency of Triangulated Temporal Qualitative Constraint Networks, in: ICTAI, 2011.

[70] A. Gerevini, Incremental qualitative temporal reasoning: Algorithms for the Point Algebra and the ORD-Horn class, Artif. Intell. 166 (2005) 37–80.

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

[72] P. C. Cheeseman, B. Kanefsky, W. M. Taylor, Where the Really Hard Problems Are, in: IJCAI, 1991.

[73] M. Sioutis, Algorithmic Contributions to Qualitative Constraint-based Spatial and Temporal Reasoning, Ph.D. thesis, Université d'Artois (2017).

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

[75] M. Sioutis, Y. Salhi, J.-F. Condotta, Studying the use and effect of graph decomposition in qualitative spatial and temporal reasoning, Knowl. Eng. Rev. 32 (2016) e4.

[76] J. J. Li, J. Huang, J. Renz, A divide-and-conquer approach for solving interval algebra networks, in: IJCAI, 2009.

[77] M. Bodirsky, S. Wölfl, RCC8 is polynomial on networks of bounded treewidth, in: IJCAI, 2011.

[78] M. Sioutis, A. Paparrizou, J. Condotta, Collective Singleton-Based Consistency for Qualitative Constraint Networks, in: TIME, 2017.

[79] M. Sioutis, A. Paparrizou, J. Condotta, A Lazy Algorithm to Efficiently Approximate Singleton Path Consistency for Qualitative Constraint Networks, in: ICTAI, 2017.