Journal of Heuristics (2023) 29:461–485
https://doi.org/10.1007/s10732-023-09517-8

On robust vs fast solving of qualitative constraints

Jan Wehner[1] · **Michael Sioutis**[2] · **Diedrich Wolter**[3]

Received: 4 October 2021 / Revised: 20 January 2023 / Accepted: 4 September 2023 /
Published online: 23 September 2023

Abstract

Qualitative Constraint Networks (QCNs) comprise a Symbolic AI framework for representing and reasoning about spatial and temporal information via the use of natural disjunctive qualitative relations, e.g., a constraint can be of the form "Task *A* is scheduled *after or during* Task *C*". In qualitative constraint-based reasoning, the state-of-the-art approach to tackle a given QCN consists in employing a backtracking algorithm, where the branching decisions during search and the refinement of the QCN are governed by certain heuristics that have been proposed in the literature. Although there has been plenty of research on how these heuristics compare and behave in terms of checking the satisfiability of a QCN fast, to the best of our knowledge there has not been any study on how they compare and behave in terms of obtaining a tractable refinement of a QCN that is also *robust*. In brief, a robust refinement of a QCN can be primarily seen as one that retains as many qualitative solutions as possible, e.g., the configuration "Task *A* is scheduled *after or during* Task *C*" is more robust than "Task *A* is scheduled *after* Task *C*". Here, we make such a preliminary comparison and evaluation with respect to prominent heuristics in the literature, and reveal that there exists a trade-off between fast and robust solving of QCNs for datasets consisting of instances of Allen's Interval Algebra and Region Connection Calculus. Furthermore,

✉ Jan Wehner
J.Wehner@student.tudelft.nl

Michael Sioutis
michael.sioutis@umontpellier.fr
https://msioutis.gitlab.io/

Diedrich Wolter
diedrich.wolter@uni-bamberg.de
https://www.uni-bamberg.de/en/sme/team/diedrich-wolter/

1 Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, Netherlands

2 LIRMM UMR 5506, CNRS, University of Montpellier, Montpellier, France

3 Faculty of Information Systems and Applied Computer Sciences, University of Bamberg, Bamberg, Germany

we investigate reasons for the existence of this trade-off and find that more aggressive heuristics are more efficient at the cost of producing less robust refinements.

Keywords Qualitative constraints · Spatial and temporal reasoning · Robust refinement · Trade-off

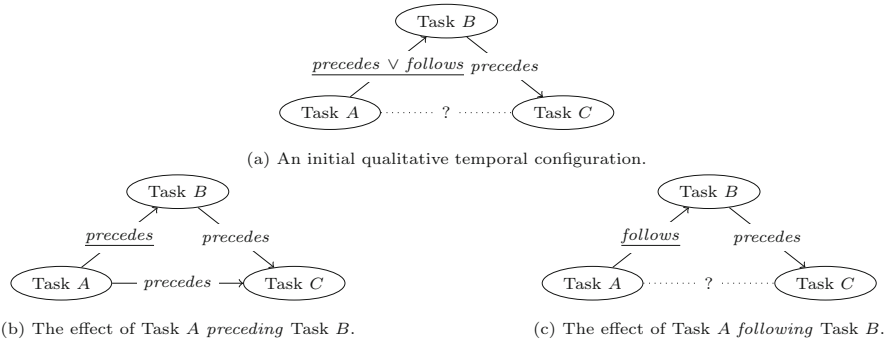
1 Introduction

Qualitative Spatial and Temporal Reasoning (QSTR) is a major and active field of study in AI that deals with the fundamental cognitive concepts of space and time in an abstract, human-like manner, via the use of simple qualitative constraint languages (Ligozat 2013). Such languages consist of symbolic expressions like *inside*, *before*, or *north of* to spatially or temporally relate two or more objects to one another, without involving any quantitative information. By extension, QSTR forms a concise framework that boosts research and applications to a plethora of areas and domains that deal with spatio-temporal information, such as cognitive robotics (Dylla and Wallgrün 2007), stream reasoning (Leng and Heintz 2016), and case-based reasoning and learning (Homem et al. 2020). The interested reader may look into a comprehensive review of the current status and the future directions of QSTR in Sioutis and Wolter (2021).

Qualitative spatial or temporal information may be modeled as a *Qualitative Constraint Network* (QCN), which is a network where the vertices correspond to spatial or temporal entities, and the arcs are labeled with *disjunctive* qualitative spatial or temporal relations respectively, e.g., $x \leq y$ can be a temporal QCN over $\mathbb{Z}$. Given a QCN $\mathcal{N}$, the literature is particularly interested in its *satisfiability problem*, which is the problem of deciding if there exists a spatial or temporal interpretation of the variables of $\mathcal{N}$ that satisfies its constraints, viz, a *solution* of $\mathcal{N}$. For instance, $x = 0 \wedge y = 1$ is one of the infinitely many solutions of the aforementioned QCN, and $x < y$ is the corresponding *scenario* (i.e., qualitative solution) that concisely represents all the cases where x is assigned a lesser value than y. In general, for most widely-adopted qualitative calculi the satisfiability problem is NP-complete (Dylla et al. 2013). In the sequel, we will be using Interval Algebra (IA) (Allen 1983) as an illustrative example of a qualitative calculus.

Motivation and contribution

Let us consider the temporal qualitative configuration of Fig. 1, which can be seen as a simplified QCN of IA as described earlier (these notions are formally defined in the next section). In qualitative constraint-based reasoning, the state-of-the-art approach to tackle such a given QCN consists in employing a backtracking algorithm, where the branching decisions during search and the refinement of the QCN are governed by certain heuristics that have been proposed in the literature (Renz and Nebel 2007); please also consult (Sioutis and Wolter 2020) for a latest overview of this approach. Over the past years, there has been a lot of research on how these heuristics and other related



(a) An initial qualitative temporal configuration.

(b) The effect of Task *A preceding* Task *B*.

(c) The effect of Task *A following* Task *B*.

Fig. 1 In order to obtain a tractable refinement of an initial qualitative configuration, certain disjunctions of possible relations need to be split into (sub-disjunctions of) relations that can guarantee tractability in the general case, e.g., *precedes* ∨ *follows* (a dichotomy) into either *precedes* or *follows*; here, if we place Task *A before* Task *B*, we can deduce a configuration that is very restrictive in terms of allowed relations, yielding a single qualitative solution, yet if we place Task *A after* Task *B*, we can deduce a configuration that is very flexible/robust in terms of allowed relations, yielding multiple qualitative solutions (viz., 13, if we view it as an IA (Allen 1983) configuration)

schemes compare and behave in terms of checking the satisfiability of a QCN fast, see Sioutis and Wolter (2021) and references therein, but, to the best of our knowledge, there has not been any study on how they compare and behave in terms of obtaining a tractable refinement of a QCN that is also *robust*. In brief, here, we consider a robust refinement of a QCN to be primarily one that retains as many qualitative solutions as possible; this notion can be seen as an extension of that introduced in Sioutis et al. (2020), where *single* qualitative solutions are concerned, and no study takes place on heuristics for obtaining/approximating them (the presented algorithm is exhaustive). For example, in Fig. 1, placing Task *A after* Task *B* instead of *before*, allows us to obtain a tractable refinement that maintains 13 possible qualitative solutions (scenarios) of IA. In this case, choosing either *after* or *before* allows us to obtain a tractable refinement; however, in general, choosing the more flexible (sub-)relation in every step of the refinement process, may result in obtaining a tractable refinement, or detecting unsatisfiability, slower than if the choices were more aggressive/restrictive. In this paper, we aim to study whether such a trade-off exists, by comparing prominent heuristics in the literature in relation to the quality of their output, i.e., the robustness and other related measures of the tractable refinements that they help to produce. We think that gaining a better insight into the output of the different heuristics can have important implications in cases where QCNs are coupled with other frameworks, such as planning or machine learning techniques, as constant revision of spatio-temporal information can be typical in systems where such information is communicated by different components, and robust QCNs can better withstand such revision.

Example 1 Let us ground Fig. 1 and the discussion above in a realistic scenario, and imagine that the tasks mentioned in the figure are tasks scheduled in a factory. The temporal configuration itself might well be the result of a human, or an AI system, who took into account certain prerequisites (e.g., that a task has to be executed before another one in the production pipeline) and made an initial incomplete schedule of

how the tasks should be executed. The question is: how should the factory finalize schedule production so that it may be least disturbed in the future? Indeed, in dynamic environments like these, where there might be unpredictable incidents concerning resource availability (e.g., power outages) and/or re-designing the production pipeline can be a frequent occurrence, extracting a complete schedule with no regard for its robustness would hardly be advisable. Instead, on one hand, we argue that some measures to compare the quality of the output of different solving strategies should at least exist, which we introduce in this work, and, on the other hand, that the efficiency of these strategies should be factored in, which is also part of our contribution here. In the discussed example, placing Task *A* *after* Task *B* instead of *before* would allow us to recover a complete schedule as fast as having Task *A* precede Task *B*, but, as explained earlier, the former case would be much more robust compared to the latter one.

Such a study on measures of withstanding perturbations for QCNs, becomes even more important in light of how hard it is in practice to repair QCNs that have been invalidated by some revision(s) (Condotta et al. 2015); specifically, this problem is at least as hard as solving a QCN in the first place in theory, but much more so in practice (Condotta et al. 2015, 2016).

The rest of the paper is organized as follows. In Sect. 2 we give some preliminaries on QSTR. Next, in Sect. 3 we present our study on robust vs fast solving of QCNs and comment on the outcome. Finally, in Sect. 4 we discuss related work, and in Sect. 5 we draw some conclusive remarks and give 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, called *base relations* (Ligozat 2013) and defined over an infinite domain D. 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 $\diamond$ (Ligozat 2013). 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., most restrictive) 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 (1983). IA considers time intervals (as temporal entities) and the set of base relations $\mathsf{B} = \{eq, p, pi, m, mi, o, oi, s, si,$

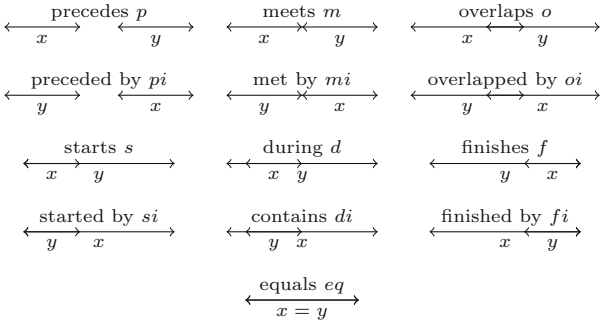


Fig. 2 A representation of the 13 base relations b of IA, each one relating two potential intervals x and y as in $x\ b\ y$; here, bi denotes the converse of b (viz., b^{-1} formally)

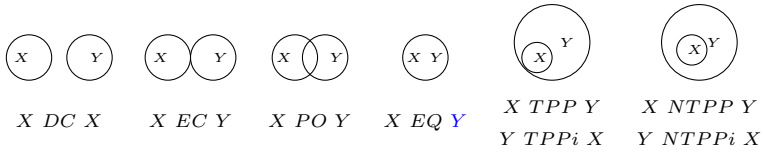


Fig. 3 The base relations of RCC8

$d, di, f, fi\}$ to encode knowledge about the temporal relations between intervals on the timeline, as described in Fig. 2. As another example, the Region Connection Calculus (RCC8) (Randell et al. 1992) considers spatial regions and the set of base relations $\mathsf{B} = \{DC, EC, EQ, PO, TPP, TPPi, NTPP, NTPPi\}$ to reason about topological relations between regions, as can be seen in Fig. 3.

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

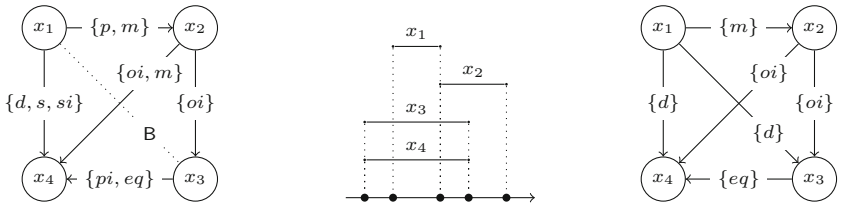
Definition 1 A QCN is a tuple (V, C) where:

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

An example QCN of IA is shown in Fig. 4a; for clarity, converse relations or Id loops are not shown in the figure.

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

- a *solution* of $\mathcal{N}$ is a mapping $\sigma : V \rightarrow \mathsf{D}$ such that $\forall (u, v) \in V \times V, \exists b \in C(u, v)$ such that $(\sigma(u), \sigma(v)) \in b$; and $\mathcal{N}$ is *satisfiable* iff it admits a solution (see Fig. 4b);
- a *refinement* $\mathcal{N}'$ of $\mathcal{N}$, denoted by $\mathcal{N}' \subseteq \mathcal{N}$, is a QCN (V, C') such that $C'(u, v) \subseteq C(u, v)\ \forall u, v \in V$;
- $\mathcal{N}$ is *atomic* iff $\forall v, v' \in V, C(v, v')$ is a *singleton relation*, i.e., a relation $\{b\}$ with $b \in \mathsf{B}$;



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

Fig. 4 Figurative examples of QCN terminology using IA

- a *scenario* $\mathcal{S}$ of $\mathcal{N}$ is an atomic satisfiable refinement of $\mathcal{N}$ (see Fig. 4c);
- 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*, denoted by $\emptyset \in \mathcal{N}$, iff $\exists v, v' \in V$ such that $C(v, v') = \emptyset$.

Given a QCN $\mathcal{N} = (V, C)$ and $v, v' \in V$, the following operation substitutes $C(v, v')$ with a relation $r \in 2^{\mathsf{B}}$ to produce a new, modified, QCN: $\mathcal{N}_{[v,v']/r}$ with $r \in 2^{\mathsf{B}}$ yields the QCN $\mathcal{N}' = (V, C')$, where $C'(v, v') = r$, $C'(v', v) = r^{-1}$ and $C'(u, u') = C(u, u')\ \forall (u, u') \in (V \times V) \backslash \{(v, v'), (v', v)\}$.

We recall the definition of ${}^{\diamond}_{G}$-consistency (Chmeiss and Condotta 2011) [cf. Ligozat (2013)], which entails consistency for all triples of variables in a QCN that form triangles in an accompanying graph G, and is a basic and widely-used local consistency for reasoning with QCNs.

Definition 3 Given a QCN $\mathcal{N} = (V, C)$ and a graph $G = (V, E)$, $\mathcal{N}$ is ${}^{\diamond}_{G}$-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 here that if G is complete, ${}^{\diamond}_{G}$-consistency becomes identical to $\diamond$-consistency (Ligozat 2013), and, hence, $\diamond$-consistency is a special case of ${}^{\diamond}_{G}$-consistency. In the sequel, given a QCN $\mathcal{N} = (V, C)$ of some calculus and a graph $G = (V, E)$, we assume that the closure of $\mathcal{N}$ under ${}^{\diamond}_{G}$-consistency, i.e., the $\subseteq$-maximal ${}^{\diamond}_{G}$-consistent refinement of $\mathcal{N}$, denoted by ${}^{\diamond}_{G}(\mathcal{N})$, is computable; this is true for most widely-adopted calculi (Dylla et al. 2013).

Over the past few years, the use of *chordal* graphs has become prominent in tackling a QCN. In short, a graph is chordal if every cycle of length > 3 has a *chord*, which is an edge that is not part of the cycle but connects two vertices of the cycle. Triangulating the constraint graph of a QCN instead of completing it allows us to reduce the search space, whilst preserving the conditions that can lead to tractability of the QCN; see Chmeiss and Condotta (2011); Huang (2012) in the context of some example calculi.

Note 1 *In the sequel, we will also be considering triangulations of the constraint graphs of* QCN*s, but our approach works for traditional complete graphs too.*

To exploit chordal graphs G in the context of our work, we introduce the notion of a *partial* scenario of a QCN with respect to a graph G, or, simply, a G-scenario of that QCN.

Definition 4 Given a QCN $\mathcal{N} = (V, C)$ and a chordal graph $G = (V, E)$ with $\mathsf{G}(\mathcal{N}) \subseteq G$, a *G-scenario* of $\mathcal{N}$ is a satisfiable refinement (V, C') of $\mathcal{N}$ such that $\forall v \in V$, $C'(v, v) = \{\mathsf{Id}\}$, and $\forall (v, v') \in E$, $C'(v, v') = \{b\}$, where $b \in C(v, v')$.

Informally, a G-scenario of a given QCN can be viewed as a scenario of that QCN (see Definition 2 and Fig. 4c) that is restricted to the edges of some accompanying graph G.

3 Robust vs fast solving of QCNs

In this section we present our study on robust vs fast solving of QCNs. In particular, we build upon the work of Sioutis and Wolter (2020) that introduces and evaluates numerous heuristics for efficiently tackling QCNs in terms of computation time, and introduce and use measures to compare these heuristics in terms of the quality of their output too, i.e., the robustness and other related measures of the results that they help to produce. First, we recall what the state-of-the-art approach for tackling a QCN is and how those heuristics play a role in doing so.

3.1 Identifying satisfiable tractable refinements

In qualitative constraint-based reasoning, the state-of-the-art approach to check the satisfiability of a given QCN $\mathcal{N}$, consists in splitting every relation r that forms a constraint between two variables in $\mathcal{N}$, into a sub-relation $r' \subseteq r$ that belongs to a set of relations $\mathscr{A}$ over which the QCN becomes tractable (Renz and Nebel 2007). In particular, for most widely-adopted qualitative calculi (Dylla et al. 2013), such *split* sets are either known or readily available (Renz 2007), and tractability is then achieved via the use of some local consistency in backtracking fashion; after every refinement of a relation r into a sub-relation r', the local consistency is enforced to know whether that refinement is valid or backtracking should occur and another sub-relation should be chosen Renz and Nebel (2007, Sect. 2). Consult also Algorithm 1 for an example of such a procedure. One of the most essential and widely-used such local consistencies is $^{\diamond}_{G}$-consistency, where G can be either the complete graph on the variables of $\mathcal{N}$ (Ligozat 2013), or a *triangulation* of the constraint graph of $\mathcal{N}$ (Chmeiss and Condotta 2011).

Example 2 The subset $\mathcal{H}_{\mathsf{IA}}$ of the set of relations of Interval Algebra (Nebel 1997) is tractable for $^{\diamond}_{G}$-consistency, i.e., $^{\diamond}_{G}$-consistency is complete for deciding the satisfiability of any QCN defined over $\mathcal{H}_{\mathsf{IA}}$ with respect to a triangulation G of its constraint graph (Chmeiss and Condotta 2011). That subset contains exactly those relations that are transformed to propositional Horn formulas when using the propositional encoding of Interval Algebra (Nebel 1997). Let us consider a constraint that involves the relation $\{mi, di, si, p, m, d, s\}$. This relation does not appear in the subset $\mathcal{H}_{\mathsf{IA}}$ and hence tractability is not guaranteed in general, but it can be split into sub-relations $\{mi\}, \{di, si\}, \{p, m\}, \{d, s\}$ that belong to $\mathcal{H}_{\mathsf{IA}}$.

Algorithm 1: Refinement($\mathcal{N}$, G, $\mathscr{A}$, f)

```
in    : A QCN N = (V, C), a graph G = (V, E), a subset A ⊆ 2^B, and a heuristic f.
out   : A refinement of N w. r. t. G over A, or the empty QCN ⊥^V.
1 begin
2     N ← ◇_G(N);
3     if ∅ ∈ ◇_G(N) then return ⊥^V if ∀{v, v'} ∈ E, C(v, v') ∈ A then return N
4     (v, v') ← {v, v'} ∈ E such that C(v, v') ∉ A;
5     foreach r ∈ subrelationSelection(N, G, A, (v, v'), f) do
6         result ← Refinement(N_{[v,v']/r}, G, A, f);
7         if result ≠ ⊥^V then return result
8     return ⊥^V;
```

Clearly, once all of the relations that form a constraint between two variables in a QCN $\mathcal{N}$ are split into sub-relations that belong to some tractable set of relations, a *tractable refinement* $\mathcal{N}'$ of $\mathcal{N}$ is achieved. At this point, satisfiability of the refined QCN may be decided by a last single application of $\overset{\diamond}{G}$-consistency, and, upon success, G-scenarios of that QCN may be generated if needed. Thus, the choice of the sub-relation in every step of the refinement process is of particular importance, as it directly affects the quality of the produced refined QCN (e.g., in terms of the number of G-scenarios / qualitative solutions it holds). That choice occurs in line 6 of Algorithm 1 and is governed by a heuristic f.

3.2 Branching heuristics

As described in Sect. 3.1, when choosing the next sub-relation to instatiate a constraint with and consequently refine a given QCN, branching heuristics are employed to determine the *least-constraining sub-relation*. In particular, that choice occurs in line 6 of Algorithm 1, and the branching heuristic pertains to the function f given as input to the algorithm. This scheme has been traditionally implemented in a static manner, where base relations are assigned a static weight a priori and the weight of a sub-relation is the cumulative weight of its base relations (van Beek and Manchak 1996). This heuristic is referred to as ***static***.

The dynamic heuristics described in Sioutis and Wolter (2020) decide which sub-relation to choose, based on the count of *local models* of the base relations in the sub-relations. Let $\mathcal{N}{\downarrow_{V'}}$, with $V' \subseteq V$, denote the QCN $\mathcal{N} = (V, C)$ restricted to V', the notion of local models is then defined as follows:

Definition 5 Given a QCN $N = (V, C)$, a graph $G = (V, E)$, and an edge $\{v, v'\} \in E$, the *local models* of a base relation $b \in C(v, v')$ are all the scenarios $S = (V', C')$ of $\mathcal{N} \downarrow_{V'}$, where $V' = \{v, v', u\}$ $\forall u$ such that $\{v, u\}, \{u, v'\} \in E$ (V' corresponds to any triangle in G involving $\{(v, v')\}$), and $C(v, v') = \{b\}$.

The proposed dynamic heuristics in Sioutis and Wolter (2020) are as follows, ranked in terms of efficiency from better to worse performing.

- *dynamic*_min: choose the sub-relation for which the base relation with the fewest local models has the most local models among the respective base relations of the rest of the sub-relations.
- *dynamic*_avg: choose the sub-relation whose base relations have the highest average count of local models.
- *dynamic*_max: choose the sub-relation that contains the base relation with the most local models.
- *dynamic*_sum: choose the sub-relation whose base relations have the highest cumulative count of local models.

Specifically, in an evaluation of the efficiency of those heuristics for hard QCNs of Interval Algebra (Sioutis and Wolter 2020), *dynamic*_min proved to be the fastest in the phase transition, closely followed by *dynamic*_avg, and *static*, *dynamic*_sum, and *dynamic*_max were found to be clearly slower.

3.3 Robustness measures of satisfiable tractable refinements

Generally, robustness can be defined as *"the ability of a system to resist change without adapting its initial stable configuration"* Wieland and Wallenburg (2012); see also Verfaillie and Jussien (2005) and references therein. Naturally, a system being robust, limits as much as possible the need for successive repairs, and hence can play an important role in environments that are prone to perturbation and unexpected change, such as real-life configurations or systems of heterogeneous components (e.g., based on logic and machine learning) that process the same information. In our context, we are interested in measuring the robustness of tractable refinements, or, in other words, the likelihood of those satisfiable tractable refinements to withstand a sequence of *perturbations*, where a perturbation is defined to be the elimination of a base relation from the constraint graph of an initial QCN (from which the refinement was produced).

Definition 6 (*Perturbation*) Given a QCN $\mathcal{N} = (V, C)$, a *perturbation* removes one base relation $b \in C(v, v')$, with $(v, v') \in E(\mathsf{G}(\mathcal{N}))$; we then say that $\mathcal{N}$ has been *perturbed*.

Next, we define certain notions that relate to the concept of a perturbation and that we will use in the sequel.

Definition 7 (*Notions of resistance*) Given a QCN $\mathcal{N} = (V, C)$, a satisfiable tractable refinement $\mathcal{N}' = (V, C')$ of $\mathcal{N}$, and a perturbation to $\mathcal{N}$ that removes its base relation $b \in C(v, v')$, with $(v, v') \in E(\mathsf{G}(\mathcal{N}))$, we say that $\mathcal{N}'$[1]

- *is affected by* the perturbation iff $b \in C'(v, v')$;
- *withstands* the perturbation iff $\mathcal{N}'_{[v,v']/r}$ is satisfiable, where $r = C'(v, v') \backslash \{b\}$;
- *survives* the perturbation iff it both is affected by and withstands that perturbation.

It is straightforward to obtain the following logical implications with respect to the notions of Definition 7:

[1] Note that the perturbation is applied to $\mathcal{N}$, not to its refinement: $\mathcal{N}'$; thus, it is possible that the removed base relation $b \in C(v, v')$ does not belong to $C'(v, v')$, viz., $b \notin C'(v, v')$ (in this case, $\mathcal{N}$ would *not* be affected by the perturbation).

Proposition 1 *Given a* QCN $\mathcal{N} = (V, C)$*, a satisfiable tractable refinement* $\mathcal{N}' = (V, C')$ *of* $\mathcal{N}$*, and a perturbation to* $\mathcal{N}$ *that removes its base relation* $b \in C(v, v')$*, with* $(v, v') \in E(\mathsf{G}(\mathcal{N}))$*:*

- *if* $\mathcal{N}'$ *is not affected by the perturbation, then* $\mathcal{N}'$ *withstands the perturbation;*
- *if* $\mathcal{N}'$ *does not withstand the perturbation, then* $\mathcal{N}'$ *is affected by the perturbation.*

Proof In the first case, $b \notin C'(v, v')$, and as a result $C'(v, v')\backslash\{b\} = C'(v, v')$, thus $\mathcal{N}'_{[v,v']/(C'(v,v')\backslash\{b\})}$ is satisfiable; the second case is the contrapositive of the first one.□

Theoretically, in order to compute the robustness of a tractable refinement of some QCN $\mathcal{N} = (V, C)$, we would need to consider all possible sequences of perturbations[2] pertaining to $\mathcal{N}$, which are $O(|V|^2(|V|^2!))$ in number (i.e., number of permutations of $O(|V|^2)$ constraints, viz., $O(|V|^2!)$, times $O(|V|^2)$ number of constraints). Thus, we consider certain more practical measures for assessing the robustness of a tractable refinement, that we describe as follows.

Amount of G-scenarios in a satisfiable tractable refinement

Our primary measure concerns the total number of G-scenarios that exist in a tractable refinement and that is restricted to the possible combinations of base relations of constraints in the constraint graph of a given QCN (thus, universal relations are disregarded and only true constraints are considered). Intuitively, the more G-scenarios a tractable refinement contains, the more likely it is to remain satisfiable after a sequence of perturbation invalidates some of those scenarios. It follows that the tractable refinement with the biggest number of G-scenarios should be able to withstand a longer sequence of perturbations. Additionally, a bigger set of G-scenarios yields more base relations per constraint on average once these scenarios are unified; this means that the corresponding refinement can stay valid for a wider variety of changes. We note that all G-scenarios of a tractable refinement can be generated with Algorithm 1, by providing the tractable refinement as the input QCN to the algorithm, along with a subset $\mathscr{A}$ that only contains the singleton relations of the calculus (then, every relation is split into singleton relations).

Average withstood perturbations for a satisfiable tractable refinement

Our secondary measure concerns the total number of perturbations that a tractable refinement can *withstand* on average, i.e., the average length of the sequence of perturbations until the refinement becomes unsatisfiable. Although we cannot perfectly compute the robustness of a tractable refinement by considering all possible sequence of perturbations, we can approximate it by averaging over many sampled sequences of perturbations. We compute this value by sampling and applying a sequence of perturbations to the QCN $\mathcal{N}$ until the initially satisfiable tractable refinement $\mathcal{N}'$ is not satisfiable anymore. The length of the sequence indicates how many perturbations $\mathcal{N}'$ *withstood*. Further, we record the number of perturbations in the sequence that

[2] A sequence of perturbations describes a series of perturbations applied to $\mathcal{N}$ one after the other.

this refinement *survives* and is *not affected* by (please see again Definition 7). By Proposition 1 we can infer that every withstood perturbation was either survived by the tractable refinement or did not affect the tractable refinement:

$$\text{avg. \# withstood by } \mathcal{N}' = \text{avg. \# survived by } \mathcal{N}' + \text{avg. \# not affected } \mathcal{N}'$$

When sampling perturbations we assume that all the constraints $C(v, v')$ in a given QCN $\mathcal{N} = (V, C)$ are equally likely to be perturbed, and that all base relations b within a constraint $C(v, v')$ are equally likely to be removed during a perturbation to that constraint.

This average number of withstood perturbations can be seen as a good measure of robustness for a tractable refinement and also complements our primary one, viz., that of calculating the total number of G-scenarios in the tractable refinement. Furthermore, the amount of perturbations that the refinement survives, or is unaffected by, gives us insights into why a tractable refinement is able to withstand perturbations.

Size, restrictiveness, and search behaviour for a satisfiable tractable refinement

It is also of interest to investigate the size of the attained tractable refinements and the restrictiveness of the chosen relations as this could give us insights into the behaviour of the different heuristics. The size of a tractable refinement $\mathcal{N}' = (V, C')$ is simply the count of all of its base relations:

$$|\mathcal{N}'| = \sum_{u,v \in V \wedge u < v} |C'(u, v)|$$

The differences in the size of the tractable refinements between the different heuristics for the same original QCN can be explained by two factors that depend on the choice of sub-relations: (i) the size of the chosen sub-relations, and (ii) how restrictive the chosen sub-relations are.

To measure whether a heuristic favors smaller or bigger sub-relations, the *average size of sub-relations that it chose* will be measured and compared to the *average size of sub-relations that it could have chosen*. Further, a heuristic that chooses more restrictive sub-relations is more *aggressive* in its search. By narrowing down the search space faster, it tends to move towards a solution faster too. A more aggressive heuristic should naturally generate smaller tractable refinements.

Definition 8 Given a QCN $\mathcal{N} = (V, C)$, a graph $G = (V, E)$, and a sub-relation $r \in C(v, v')$ with $v, v' \in V$, the *restrictiveness* of r is $|\mathcal{N}| - |^{\diamond}_{G}(\mathcal{N}_{[v,v']/r})| - |C(v, v') \backslash r|$.

The restrictiveness of a sub-relation captures how many base relations are removed from the other relations after the QCN is refined (by means of $^{\diamond}_{G}$-consistency) with that sub-relation.

It is also of interest to find out how close the heuristics described in Sect. 3.2 are to the most and least restrictive choice. To that end we introduce two additional heuristics that are based on the number of G-scenarios:

- *dynamic*_LCV: choose the sub-relation which retains the most G-scenarios. So this heuristic always chooses the *least constraining value*.
- *dynamic*_MCV: choose the sub-relation which removes the most G-scenarios. So this heuristic always chooses the *most constraining value*.

One would expect that *dynamic*_LCV attains a refinement that is close to optimal in terms of robustness, while *dynamic*_MCV results in very fragile refinements.

To further investigate the behaviour of the heuristics we can track the following measures for every refinement step of the search process: We calculate the *number of solutions* by generating all tractable refinements that are still attainable. This represents the space of valid solutions. We count the *number of combinations* of sub-relations in the constraint graph, which represents the entire search space. For a QCN $\mathcal{N}' = (V, C')$ with constraint graph E it is calculated as follows:

$$\#\ possible\ combinations = \prod_{u,v \in V \wedge u < v \wedge \{u,v\} \in E} |split(C'(u,v))|$$

Finally, we calculate the *fraction of the search space which is valid*. This is the fraction of all possible combinations which are actually valid tractable refinements:

$$\%\ valid = \frac{\#\ \text{contained G-scenarios}}{\#\ possible\ combinations}$$

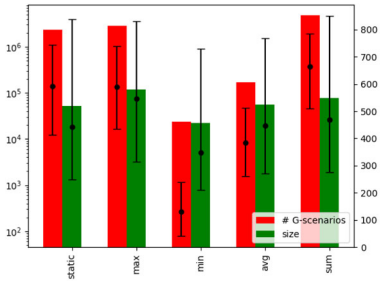
3.4 Evaluation

This section describes four experiments that investigate the robustness of satisfiable tractable refinements, involving Algorithm 1 and each proposed heuristic f described in Sect. 3.2. The first two experiments compare the proposed heuristics on datasets of Interval Algebra based on the measures presented in Sect. 3.3. The third experiment investigates the search behaviour of the proposed heuristics for QCNs of Interval Algebra. The fourth experiment evaluates the heuristics w.r.t. the robustness of tractable refinements for QCNs of RCC8. As per Note 1, the *maximum cardinality search* algorithm (Tarjan and Yannakakis 1984) was used to triangulate the constraint graphs of the involved QCNs.

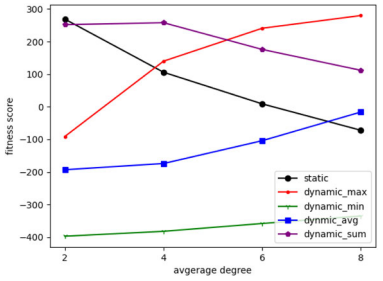
Technical Specifications Experiments were carried out on a computer with a Ryzen 7 4700U CPU, 16 GB of RAM, and the Windows 10 v2004 OS. Algorithms were coded in Python and run using the PyPy intepreter v7.3.3.

Experiment 1: based on amount of G-scenarios

Primarily, this experiment was conducted to assess differences in the amount of G-scenarios contained in the tractable refinements generated by each heuristic. Additionally, the size of the tractable refinement (i.e., its total number of base relations),



(a) Average number of G-scenarios contained in the resulting satisfiable tractable refinements for each heuristic (left axis), and the average size of these refinements (= number of base relations), with the upper and lower quartile and the median (right axis).



(b) Fitness score for each heuristic with regard to the amount of G-scenarios in their resulting tractable refinements; ranking points are given based on how they place for each instance, from 2 (first place) to −2 points with a step of −1.

Fig. 5 Illustrative results from the experiment based on amount of G-scenarios

the average size of the sub-relations chosen by a heuristic, and the average size of the possible sub-relations to be chosen were measured.

Question/answer How robust are the tractable refinements generated by each heuristic as measured by the amount of **G**-scenarios that they contain, and what role does the size of a refinement play for that matter? *Dynamic*_sum and *dynamic*_max generate solutions that have significantly more **G**-scenarios than *dynamic*_min. This can be explained by observing that more robust measures are obtained via the choice of bigger sub-relations, which consequently produce bigger tractable refinements containing more **G**-scenarios.

Dataset A dataset of 800 satisfiable random instances of Interval Algebra was generated using the model $S(n = 10, d, l = 6.5)$ (Nebel 1997; van Beek and Manchak 1996); specifically, 200 QCNs of $n = 10$ variables and $l = 6.5$ base relations per relation on average for each constraint graph degree value $d \in \{2, 4, 6, 8\}$.

Results The main result is that there are stark differences between heuristics in the number of G-scenarios contained in the tractable refinements produced. Figure 5a and Table 1 show that tractable refinements generated by *dynamic*_sum on average contain 71% more G-scenarios than those of any other heuristic. Further, *dynamic*_avg scored 7 times higher than *dynamic*_min, but 14 times lower than *static*, which in turn was close to *dynamic*_max. The fitness scores presented in Fig. 5b reveal that *dynamic*_sum and *static* perform better for smaller degrees, whilst *dynamic*_max and *dynamic*_avg generate comparatively more scenarios for bigger degrees.

The results are presented in further detail and from a different perspective in Table 2. The bigger (in terms of number of base relations) tractable refinements are generated by *dynamic*_max, which is interesting, since it chooses smaller sub-relations than *static* and *dynamic*_sum, both in total and relative to the available options. Further, *dynamic*_min is the only heuristic that chooses smaller than average sub-relations, resulting in the smallest tractable refinements overall. The Spearman's rank correlation between the amount of contained G-scenarios and the size of a tractable refinement is `corr(# G-scenarios, size)` = 0.773. Notably, the average size of possible

Table 1 The amount of G-scenarios that can be generated from a tractable refinement and the size of a tractable refinement of a certain degree; $\frac{\text{avg. \# } G-\text{scenarios}|\text{median \# } G-\text{scenarios}}{\text{avg. size}|\text{median size}}$

Degree	Static	Max	Min	Avg	Sum
2	**1.8** m|**606** k	33*k*|6560	1 k|16	12 k|2616	1.8 m|533 k
	987.88|992	980.62|981	925.25|932	970.45|974	**993.03**|**1** k
4	6.3 m|523 k	5.7 m|504 k	86 k|260	295 k|17 k	**14.3** m|**1.9** m
	577.71|574	**685.13**|**693**	464.53|458	583.15|582	625.49|632
6	764 k|49 k	**4.2** m|**517** k	4.75 k|384	238 k|13 k	2.0 m|268 k
	322.51|312	**411.51**|**416**	269.67|267	338.02|333	355.2|256
8	324*k*|4.5*k*	**1.6** m|**171** k	3.85 k|380	132 k|15 k	968 k|38 k
	193.22|186	**245.73**|**242**	164.55|162	208.86|203	216.3|214
Overall	2.3 m|141 k	2.8 m|133 k	24 k|270	169 k|8.22 k	**4.8 m**|**367 k**
	520.33|443	**580.75**|**547**	456|348	525.12|448	549|468

Best performance numbers are given in bold

Table 2 The average size (= number of base relations) of an attained tractable refinement, the average size of the sub-relations that a heuristic chooses, and the average size of sub-relations from which a heuristic chooses

	Static	Max	Min	Avg	Sum
Avg. size of refinement	520.33	**580.75**	456	525.12	549
Avg. size of chosen sub-rels	3.66	3.06	1.61	2.45	**3.71**
Avg. size of possible sub-rels	2.15	**2.29**	2.07	2.22	2.19

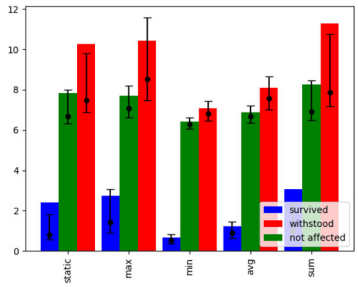
Best performance numbers are given in bold

sub-relations also differed significantly between heuristics, with *dynamic*_max choosing from bigger and *dynamic*_min from smaller ones, indicating that these heuristics are less and more aggressive respectively.

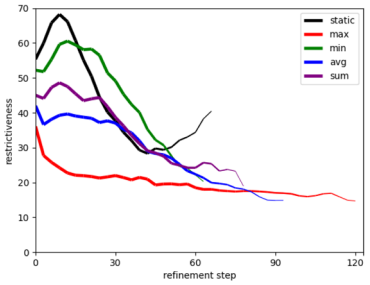
Experiment 2: based on average withstood perturbations

This experiment was conducted to calculate how many perturbations the tractable refinements generated by the heuristics can withstand on average. To this end, in relation to those refinements, the average amount of perturbations the refinement is unaffected by and survives are measured. In addition, the number of nodes that were visited while searching for a tractable refinement, the size of the tractable refinement, the average size of chosen and possible sub-relations, and the restrictiveness of the chosen sub-relation are recorded.

*Question/answer*How robust are the tractable refinements generated by each heuristic against perturbations as measured by the amount of withstood perturbations in a sequence, and how are efficiency and robustness related? Again, *dynamic*_sum and *dynamic*_max are able to withstand the most perturbations, while *dynamic*_min is the least robust. Heuristics that produce more robust tractable refinements need more steps to solve a problem (because they prune relations parsimoniously).



(a) Average, upper, and lower quartile, and the median of the number of perturbations that the resulting tractable refinements for each heuristic *survives*, *withstands* and is *not affected* by.



(b) Average restrictiveness of the chosen sub-relation at each refinement step out of 1000 refinement searches for each heuristic. The width of the graphs represents the amount of ongoing searches at that step and the graphs end when > 90% of searches have found a solution.

Fig. 6 Illustrative results from the experiment based on average withstood perturbations

Dataset A dataset of 1 000 satisfiable random instances of Interval Algebra was generated using the model $S(n = 30, d, l = 6.5)$ (Nebel 1997; van Beek and Manchak 1996); specifically, 200 QCNs of $n = 30$ variables and $l = 6.5$ base relations per relation on average for each constraint graph degree value $d \in \{5, 10, 15, 20, 25\}$

Results The experiment showed clear differences in how many perturbations the tractable refinements of the heuristics can survive and, consequently, withstand, while the likelihood of remaining unaffected was relatively even among heuristics; see Fig. 6a. Just as in the first experiment, *dynamic*_sum performed best, on average withstanding 8% more perturbations than *static* and *dynamic*_max, which were about even. However, *dynamic*_max generates more robust refinements for degrees $d \geq 10$, while *dynamic*_sum only performs best for $d = 5$. While tractable refinements generated by *dynamic*_min survived and withstood the fewest perturbations, they had the highest chance to remain unaffected from a perturbation. This can be explained from the fact that those refinements remained unaffected by many more perturbations than what they survived, indicating that they were unlikely to be affected, but had a low chance of surviving when they were affected. Similarly, *dynamic*_avg generated tractable refinements that survived only 1.03 more perturbations than *dynamic*_min, and that were relatively unlikely to be affected. More perturbations were withstood by the refinement not being affected than by it surviving the perturbation. However, there was a larger difference between heuristics for the number of survived perturbations than for the number of unaffected perturbations.

Much like in the case of the first experiment, see Fig. 5b, the comparative performance by *static* and *dynamic*_sum decreases as degrees get higher, and increases for *dynamic*_max and *dynamic*_avg; the figure here would be qualitatively similar. Most importantly, the average Pearson correlation between the number of withstood perturbations and the number of visited nodes is `corr(avg. # withstood, # visited nodes)` = 0.82. Detailed results for this experiment are displayed in Table 3. The average size measurements of the tractable refinements, the chosen sub-

Table 3 The likelihood that a tractable refinement remains unaffected, the expected amount of survived perturbations, and the expected amount of withstood perturbations; avg.#survived|avg. #withstood|avg. P(not affected) / median # survived|median #withstood|median P(not affected)

Degree	Static	Max	Min	Avg	Sum
5	8.71|21.11|12.4	7.84|18.06|10.22	1.25|8.29|7.04	2.6|10.49|7.89	**10.94|24.43|13.48**
	8.04|20.24|11.86	7.31|17.13|9.9	1.22|8.21|7.03	2.54|10.36|7.81	**10.15|22.95|13.02**
10	1.49|9.02|7.53	**2.7|10.75|8.05**	0.73|7.25|6.52	1.33|8.43|7.1	1.94|9.93|7.99
	1.44|8.9|7.46	**2.59|10.59|7.98**	0.69|7.17|6.5	1.27|8.33|7.06	1.82|9.75|7.9
15	0.79|7.41|6.63	**1.43|8.52|7.09**	0.53|6.8|6.28	0.88|7.52|6.64	0.98|7.81|6.83
	0.76|7.42|6.64	**1.38|8.4|7.03**	0.51|6.77|6.25	0.87|7.5|6.63	0.94|7.71|6.78
20	0.59|6.95|6.36	**0.97|7.67|6.7**	0.41|6.54|6.13	0.68|7.11|6.44	0.75|7.31|6.56
	0.59|6.94|6.36	**0.95|7.72|6.71**	0.4|6.51|6.12	0.67|7.11|6.41	0.73|7.27|6.55
25	0.51|6.78|6.27	**0.73|7.19|6.46**	0.37|6.44|6.07	0.58|6.89|6.32	0.62|7.01|6.39
	0.51|6.76|6.26	**0.7|7.11|6.41**	0.35|6.42|6.06	0.56|6.86|6.31	0.61|7.03|6.39
overall	2.42|10.26|7.84	2.74|10.44|7.7	0.66|7.06|6.41	1.21|8.09|6.88	**3.05|11.3|8.25**
	0.81|7.48|6.69	**1.41|8.53|7.07**	0.53|6.8|6.28	0.89|7.57|6.66	1.01|7.87|6.91

Best performance numbers are given in bold

relations, and the possible sub-relations are qualitatively very similar to those displayed in Table 2, thus a further display and discussion is omitted. The average restrictiveness of chosen sub-relations was highest for *static* (50.64) and *dynamic*_min (49.26), lower for *dynamic*_sum (39.06) and *dynamic*_avg (32.81), and lowest for *dynamic*_max (21.17). Notably, we have that `corr(avg. restrictiveness of chosen sub-relation, # visited nodes)` = 0.92, indicating that more aggressive heuristics solve QCNs more efficiently. Figure 6b shows how the restrictiveness of chosen sub-relations develops as search progresses. It reveals *static* to be quite irregular (too aggressive at the beginning *and* at the end); this is because it is not guided by local models and hence cannot adapt to the particulars of a QCN. On the other hand, *dynamic*_min, even though being almost as aggressive as *static*, can adapt to a QCN and is thus steady. *Dynamic*_max is clearly the least aggressive, and *dynamic*_sum and *dynamic*_avg are similar in terms of aggressiveness.

Experiment 3: $RCC8$

This experiment investigates the robustness of tractable refinements generated by the proposed heuristics for QCNs of the Region Connection Calculus by measuring the average amount of perturbations a tractable refinement can withstand. Additionally, the restrictiveness and size of the chosen sub-relations and the number of visited nodes was recorded.

Question/answer Do the results for Allen's Interval Algebra generalize to the Region Connection Calculus, so that our framework can claim to be generic? Yes, the results for the Region Connection Calculus are very similar (qualitatively) to those for Allen's Interval Algebra.

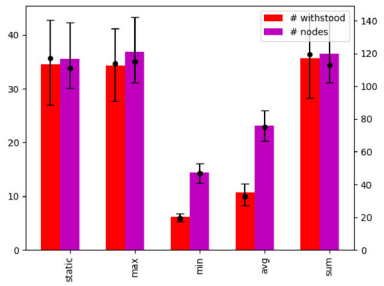
Dataset A dataset of 1000 satisfiable random instances of RCC8 was generated using the model $S(n = 30, d, l = 4)$ (Nebel 1997; van Beek and Manchak 1996); specifically, 200 QCNs of $n = 30$ variables and $l = 4$ base relations per relation on average for each constraint graph degree value $d \in \{5, 10, 15, 20, 25\}$.

Results Overall the evaluation of heuristics for RCC8 gave similar results as the one for IA. Again *dynamic*_min and *dynamic*_avg were most efficient and more aggressive, while *dynamic*_max, *dynamic*_sum and *static* were more robust and less aggressive.

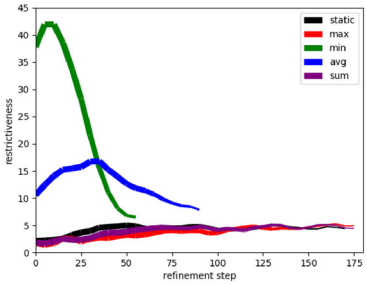
Figure 7a shows that *static*, *dynamic*_max and *dynamic*_sum need to visit roughly 120 nodes on average to get a tractable refinement which can then withstand roughly 35 perturbations. On the other hand it only takes *dynamic*_min 48 nodes on average to attain a tractable refinement which can then withstand only 6.2 perturbations. Similarly *dynamic*_avg is relatively quick (76 visited nodes on average), but produces very fragile tractable refinements (10.7 withstood perturbations on average).

Interestingly, Fig. 7c, d are nearly mirrors of each other. The figures show that *dynamic*_min is the fastest heuristic for any degree. On the other hand, the most robust tractable refinements were generated by *dynamic*_sum. In contrast to Fig. 5b for IA, *dynamic*_max did not produce more robust refinements than *dynamic*_sum for higher degrees.

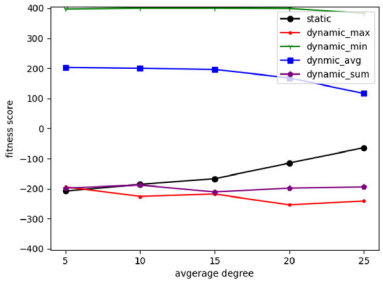
Similarly to the experiments on IA, *dynamic*_min and *dynamic*_avg choose much smaller sub-relations that are also more restrictive and thus attain smaller tractable refinements (see Table 4; Fig. 7b).



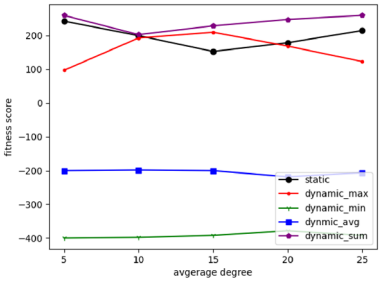
(a) Avg. number of withstood perturbations and avg. number of visited nodes with the upper and lower quartile and median.



(b) Average restrictiveness of the chosen sub-relation at each refinement step out of 900 refinement searches for each heuristic. The width of the graphs represents the amount of ongoing searches at that step and the graphs end when $> 95\%$ of searches have found a solution.



(c) Fitness score for each heuristic with regard to the amount of visited nodes in their resulting tractable refinements; ranking points are given based on how they place for each instance, from 2 (first place) to -2 points with a step of -1.



(d) Fitness score for each heuristic with regard to the amount of withstood perturbations in their resulting tractable refinements; ranking points are given based on how they place for each instance, from 2 (first place) to -2 points with a step of -1.

Fig. 7 Illustrative results from the experiment on instances of RCC8

Table 4 The average size (= number of base relations) of an attained tractable refinement, the average size of the sub-relations that a heuristic chooses,the average number of nodes visited during a search and the number of perturbations the tractable refinements can withstand on average

	Static	Max	Min	Avg	Sum
Avg. size of refinement	4217	4305	2477	3210	4264
Avg. size of chosen sub-rels	3.14	3.15	1.74	2.45	3.15
Avg. # visited nodes	118.5	127.5	48.1	76.5	123.5
Avg. # withstood perturbations	34.54	34.3	6.22	10.69	35.67

Experiment 4: least- and most-restrictive-value

This experiment was devised to compare the proposed heuristics (those presented in Sect. 3.2) to the to the least and most constraining heuristic *dynamic*_LCV and *dynamic*_MCV (see Sect. 3.3) and to investigate the reductions in the search space and the amount of solutions during search. The proposed heuristics were compared to *dynamic*_LCV, by comparing the ordering in which they prioritize the tractable sub-relations to each other. For each decision the spearman rank *correlation of the value orderings* and *whether they placed the same value first* were recorded. The *number of solutions* and the *number of possible combinations* are measured at every refinement step. Additionally, the number of visited nodes, the number of backtrackings that occured during the search and the average number of withstood perturbations were recorded (Table 5).

*Question/answer*How similar are the choices of the heuristics to choosing the *true* least- and most-restrictive sub-relation? How quickly do different heuristics reduce the search space of possible solutions? *Dynamic*_sum chooses most similar to *dynamic*_LCV. More aggressive heuristics like *dynamic*_min reduce the search space faster, making them more efficient in general at finding tractable refinements.

Dataset A dataset of 900 satisfiable random instances of Interval Algebra was generated using the model $S(n = 7, d, l = 6.5)$ (Nebel 1997; van Beek and Manchak 1996); specifically, 300 QCNs of $n = 7$ variables and $l = 6.5$ base relations per relation on average for each constraint graph degree value $d \in \{2, 4, 6\}$. The need for choosing such small QCNs arises, because the number of contained G-scenarios is being calculated at every step for the heuristics *dynamic*_LCV and *dynamic*_MCV, which drastically slows down the search.

Results This experiment grants two main results. First, of the proposed heuristics, *dynamic*_sum is the best approximation of *dynamic*_LCV. Second, more aggressive heuristics reduce the number of solutions and the search space more heavily, resulting in more efficient solving, but less robust solutions.

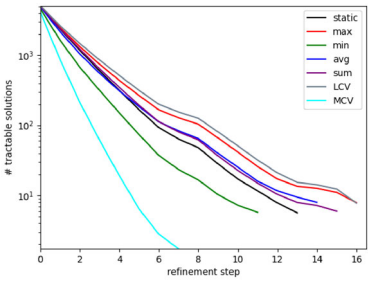
As one would expect, *dynamic*_MCV and *dynamic*_LCV behave opposite to each other and more extreme than the proposed heuristics in every measure.

Of the proposed heuristics, the value orderings of *dynamic*_sum correlate the strongest with those of *dynamic*_LCV (0.641). Also they prioritize the same value in 83.67% of cases. *Static* and *dynamic*_max were also similar to *dynamic*_LCV, showcasing a correlation of 0.504 and 0.459, and choosing the same value in 77.72% and 69.09% of cases respectively, while *dynamic*_avg only correlated weakly (0.301) and *dynamic*_min did not correlate at all (0.029). The value orderings of *dynamic*_MCV and *dynamic*_LCV do not have a perfectly negative spearman rank correlation, because sub-relations with the same amount of G-scenarios are ordered the same for both heuristics.

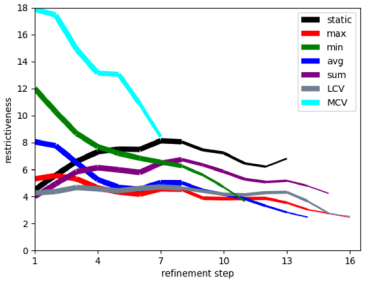
Figure 8a shows that more aggressive heuristics like *dynamic*_MCV and *dynamic*_min reduce the amount of tractable solutions much faster than less aggressive ones like *dynamic*_LCV or *dynamic*_max. A Figure that shows the amount of possible combinations at each step is qualitatively very similar to Fig. 8a and is thus omitted. It would show that aggressive heuristics reduce the search space much faster than less aggressive heuristics. From Fig. 8c one can derive that *dynamic*_min and

Table 5 The average spearman rank correlation between the value orderings of a given heuristic and that of *dynamic*_LCV, the average number of visited nodes, the average number of backtracks, and the average number of withstood perturbations

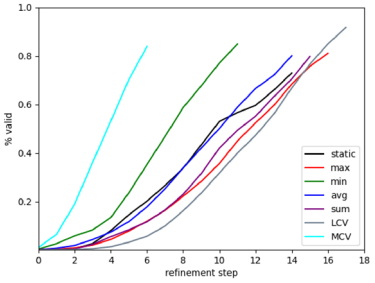
	Static	Max	Min	Avg	Sum	LCV	MCV
Avg. correlation with LCV	0.504	0.459	0.029	0.301	0.641	–	–
Equal first choice with LCV	77.72%	69.09%	38.27%	57.19%	83.67%	–	–
Avg. # visited nodes	10.78	11.85	9.48	11.05	11.51	12.39	6.96
Avg. # backtracks	0.0122	0.0045	0.0144	0.0089	0.0078	0	0.0533
Avg. # withstood	11.97	9.27	6.3	7.77	12.33	5.59	12.86



(a) The average number of tractable solutions that are still possible at each refinement step for each heuristic. The graph ends when $> 95\%$ the 900 of searches have found a solution.



(b) Average restrictiveness of the chosen sub-relation at each refinement step out of 900 refinement searches for each heuristic. The width of the graphs represents the amount of ongoing searches at that step and the graphs end when $> 95\%$ of searches have found a solution.



(c) The average percentage of possible combinations which are consistent tractable refinements at each refinements step for each heuristics for QCNs of degree 6.

Fig. 8 Illustrative results from the experiment including the least- and most-restrictive-value heuristic

*dynamic*_MCV were able to move more quickly to a state in the search where a large percentage of possible combinations were valid ones. The reader should consider that, to make the figure more understandable, only QCNs of degree $d = 6$ are shown in the graph. However, the graph is qualitatively similar for instances of different degrees.

Discussion of results

Despite using different measures of robustness and experimenting with datasets of different numbers of variables, the first and second experiment lead to very similar conclusions regarding the robustness of tractable refinements for QCNs of IA in relation to the different heuristics used. While *dynamic*_sum seems to be the most robust heuristic overall, it is outperformed by *dynamic*_max on instances with high degrees. *Static* proves to be fairly robust, but becomes comparatively less robust for higher degrees. *Dynamic*_avg produces relatively fragile tractable refinements, while *dynamic*_min is the worst heuristic in terms of robustness.

The first experiment shows that heuristics have preferences in the size of sub-relations during the refinement process. In particular, *dynamic*_min favors sub-relations with fewer base relations, as this reduces the likelihood of having a base relation with a low model count (as a reminder, this heuristic prefers sub-relations with the highest lowest model count among sub-relations, see Sect. 3.2). On the other hand, *dynamic*_max chooses sub-relations with more base relations for the exact opposite reason. Further, *static* and *dynamic*_sum prefer bigger sub-relations, as this increases the sum of weights and number of local models respectively. The experiment also showed that *dynamic*_max generated much bigger tractable refinements, while those generated by *dynamic*_min were much smaller.

In the second experiment we can see that the sub-relations chosen by the heuristics also differ with regards to how restrictive they are. *Dynamic*_min and *dynamic*_avg tended to choose more restrictive sub-relations. This is because they chose smaller sub-relations, which leaves less options for base relations in other relations to form consistent scenarios with. Surprisingly, the sub-relations chosen by *static* were also very restrictive, despite having many base relations, which indicates that *static* is unable to adapt well to a given situation. This might be because it does not consider local models in its decision.

The third experiment shows that heuristics behaved similarly when solving QCNs of RCC8 as they do for IA. It supports the observation that *dynamic*_max, *dynamic*_sum and *static* generate more robust tractable refinements, while *dynamic*_min and *dynamic*_avg are more efficient.

The fourth experiment shows that more restrictive heuristics reduce the search space more quickly at the cost of removing more possible solutions. This allows them to move faster to a point in the search where the search space consists in large parts out of valid tractable refinements and thus find a solution more efficiently. This is in part caused by the fact that more restrictive sub-relations reduce the size of "yet to be refined sub-relations" more heavily. When these tractable are finally selected to be refined, they might have already been reduced to tractable sub-relations and will thus not have to be split, resulting in a more shallow search tree. Overall we can gather that *dynamic*_avg and especially *dynamic*_min display a more aggressive search strategy than *dynamic*_max, *dynamic*_sum and *static*.

A trade-off exists The most important finding is that the heuristics that generate more robust tractable refinements, viz., *dynamic*_sum, *dynamic*_max, and *static*, are the least efficient ones Sioutis and Wolter (2020), while *dynamic*_min and *dynamic*_avg, which are the most and second most efficient ones, are the least and second least robust ones, according to our experimentation here. Remarkably, this holds for datasets of different sizes and constraint languages. This indicates a *trade-off* when choosing which heuristic to employ for solving a QCN of Interval Algebra, between finding a solution fast and generating a robust solution. This fact is further supported by the strong positive correlation between the amount of withstood perturbations and visited nodes.

We think that the reason for this lies in the aggressiveness of the heuristics. As can be seen from fourth experiment, heuristics that choose smaller and more restrictive sub-relations move towards a solution more quickly, without causing too much backtracking and are thus more efficient. On the other hand choosing smaller sub-relations

and removing more base relations elsewhere during the search leads to tractable refinements with fewer base relations. As we found out in the first experiment, these smaller tractable refinements are less robust.

4 Related work

As noted in Sect. 3, the study on robust vs fast solving of QCNs presented here builds upon the work of Sioutis and Wolter (2020), which introduces and evaluates numerous heuristics for efficiently tackling QCNs, in that it introduces and uses measures to compare these heuristics in terms of the quality of their output too, i.e., the robustness and other related measures of the results that they help to produce. The work of Sioutis and Wolter (2020) itself was inspired by that of Pesant et al. (2012) in traditional constraint programming, which formalizes a counting-based framework for (finite-domain) CSPs that is adaptive and seeks to make branching decisions that preserve most of the solutions by determining what proportion of local solutions agree with that decision. However, much like Sioutis and Wolter (2020), the work of Pesant et al. (2012) also does not take into account the quality of the different outputs achieved with respect to robustness; it is important to note that the term "robust search" used in Pesant et al. (2012) refers to "search that works well most of the time", across different bechmarks (i.e., reliable or dependable search), and does not relate to our notion of robustness here (see Sect. 3.3).

Even though to the best of our knowledge there has not been any published work on how fast solving compares to robust solving in a given paradigm, be it qualitative constraint-based reasoning, traditional constraint programming, or Boolean satisfiability (SAT) for example, notions of robustness (and stability) alone that are similar to those introduced here (see Sect. 3.3) have been studied quite extensively in the field of traditional constraint programming (Climent 2015; Climent et al. 2014, 2010; Barber and Salido 2015; Hebrard 2007) and SAT theories (Ginsberg et al. 1998; Roy 2006; Bofill et al. 2010) over the past years. We note that in the aforementioned works notions related to robustness, such as *stability*, are studied as well, but these go beyond the scope of this paper. Briefly put, a solution is stable, if in the event of a change that invalidates it, it can be repaired with a minimum number of revisions, whereas a robust solution is more likely to remain valid after the change occurs; the distinction is subtle, but clear.

Finally, limited parts of this work were accepted for publication in the proceedings of the 33rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2021) (Wehner et al. 2021), and what is presented here is a significant extension of that published short manuscript. Specifically, Experiments 3 and 4 in Sect. 3.4 are completely novel here, as are a more comprehensive discussion of the results in that section, a more thorough and self-contained presentation of the theoretical background of the approach in earlier sections, and the Related Work section, viz., Sect. 4, presented here, which establishes connections with works in traditional constraint programming and SAT and broadens the scope of the paper. We should also note that, with the help of our reviewers, we rectified the measure of average withstood perturbations for a satisfiable tractable refinement (see the discussion in Sect. 3.3) and the

subsequent experimental results pertaining to that measure that originally appeared in Wehner et al. (2021). In that sense, this manuscript is also a more robust version of the aforementioned preceding conference paper on robustness.

5 Conclusion and future work

The state-of-the-art approach for tackling a given QCN, consists in employing a backtracking algorithm for obtaining a tractable refinement of it, where the branching decisions during search are governed by certain heuristics proposed in the literature. In this paper, we studied whether a trade-off exists between tackling a QCN fast, and tackling it in a manner that allows us to obtain a *robust* tractable refinement of it, i.e., a refinement that primarily retains as many qualitative solutions as possible. To this end, we introduced some measures of quality for the various tractable refinements that can be produced by the different branching heuristics, viz., their robustness in terms of the qualitative solutions that they contain and their ability to withstand perturbations, and performed an evaluation in relation to these measures. To the best of our knowledge, this is the first work to address this issue in the context of qualitative constraint-based reasoning. Our findings suggest that such a trade-off of speed and robustness indeed exists, at least with respect to the heuristics that have been implemented and are known to date. We think that the reason for this lies in the aggressiveness of the search strategy of the heuristics and that a compromise can be achieved depending on the application of interest (e.g., speed over quality, or vice versa). For future work, we would like to utilize our findings and observations here to investigate and devise *adaptive* techniques that switch among different heuristics when tackling a given QCN (e.g., reducing aggressiveness while nearing a tractable refinement, and so on). It would also be an interesting future direction to explore different probability distributions with regard to perturbations and/or even consider cases where certain constraints can simply not be perturbed (due to encoding the spatial structure of a fixed environment for example).

References

Allen, J.F.: Maintaining knowledge about temporal intervals. Commun. ACM **26**, 832–843 (1983)

Barber, F., Salido, M.A.: Robustness, stability, recoverability, and reliability in constraint satisfaction problems. Knowl. Inf. Syst. **44**, 719–734 (2015)

Bofill, M., Busquets, D., Villaret, M.: A declarative approach to robust weighted Max-SAT. In: PPDP (2010)

Chmeiss, A., Condotta, J.-F.: Consistency of triangulated temporal qualitative constraint networks. In: ICTAI (2011)

Climent, L., Salido, M.A., Barber, F.: Robust solutions in changing constraint satisfaction problems. In: IEA/AIE (2010)

Climent, L.: Robustness and stability in dynamic constraint satisfaction problems. Constraints **20**, 502–503 (2015)

Climent, L., Wallace, R.J., Salido, M.A., Barber, F.: Robustness and stability in constraint programming under dynamism and uncertainty. J. Artif. Intell. Res. **49**, 49–78 (2014)

Condotta, J., Mensi, A., Nouaouri, I., Sioutis, M., Said, L.B.: A practical approach for maximizing satisfiability in qualitative spatial and temporal constraint networks. In: ICTAI (2015)

Condotta, J., Nouaouri, I., Sioutis, M.: A SAT approach for maximizing satisfiability in qualitative spatial and temporal constraint networks. In: KR (2016)
de Leng, D., Heintz, F.: Qualitative spatio-temporal stream reasoning with unobservable intertemporal spatial relations using landmarks. In: AAAI (2016)
Dylla, F., Mossakowski, T., Schneider, T., Wolter, D.: Algebraic properties of qualitative spatio-temporal calculi. In: COSIT (2013)
Dylla, F., Wallgrün, J.O.: Qualitative spatial reasoning with conceptual neighborhoods for agent control. J. Intell. Robot. Syst. **48**, 55–78 (2007)
Ginsberg, M.L., Parkes, A.J., Roy, A.: Supermodels and robustness. In: AAAI/IAAI (1998)
Hebrard, E.: Robust solutions for constraint satisfaction and optimisation under uncertainty. Ph.D. thesis, University of New South Wales, Sydney (2007)
Homem, T.P.D., Santos, P.E., Costa, A.H.R., da Costa Bianchi, R.A., de Mántaras, R.L.: Qualitative case-based reasoning and learning. Artif. Intell. **283**, 103258 (2020)
Huang, J.: Compactness and its implications for qualitative spatial and temporal reasoning. In: KR (2012)
Ligozat, G.: Qualitative Spatial and Temporal Reasoning. Wiley, ISTE (2013)
Nebel, B.: Solving hard qualitative temporal reasoning problems: evaluating the efficiency of using the ORD-horn class. Constraints **1**, 175–190 (1997)
Pesant, G., Quimper, C., Zanarini, A.: Counting-based search: branching heuristics for constraint satisfaction problems. J. Artif. Intell. Res. **43**, 173–210 (2012)
Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. KR **92**, 165–176 (1992)
Renz, J., Nebel, B.: Qualitative spatial reasoning using constraint calculi. In: Handbook of Spatial Logics, pp. 161–215 (2007)
Renz, J.: Qualitative spatial and temporal reasoning: efficient algorithms for everyone. In: IJCAI (2007)
Roy, A.: Fault tolerant Boolean satisfiability. J. Artif. Intell. Res. **25**, 503–527 (2006)
Sioutis, M., Long, Z., Janhunen, T.: On robustness in qualitative constraint networks. In: IJCAI (2020)
Sioutis, M., Wolter, D.: Dynamic branching in qualitative constraint networks via counting local models. In: TIME (2020)
Sioutis, M., Wolter, D.: Qualitative spatial and temporal reasoning: current status and future challenges. In: IJCAI (2021)
Tarjan, R.E., Yannakakis, M.: Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. SIAM J. Comput. **13**, 566–579 (1984)
van Beek, P., Manchak, D.W.: The design and experimental analysis of algorithms for temporal reasoning. J. Artif. Intell. Res. **4**, 1–18 (1996)
Verfaillie, G., Jussien, N.: Constraint solving in uncertain and dynamic environments: a survey. Constraints **10**, 253–281 (2005)
Wehner, J., Sioutis, M., Wolter, D.: On robust vs fast solving of qualitative constraints. In: ICTAI, Short Paper (2021)
Wieland, A., Wallenburg, C.: Dealing with supply chain risks: linking risk management practices and strategies to performance. Int. J. Phys. Distrib. Log. Manag. **42**, 887–905 (2012)