

Qualitative Spatial and Temporal Reasoning with Answer Set Programming

Jason Jingshi Li

Artificial Intelligence Laboratory, École Polytechnique Fédérale de Lausanne
EPFL/IC/IIF/LIA, Bâtiment IN, Station 14, 1015, Lausanne, Switzerland

Email: Jason.Li@epfl.ch

Abstract—Representing and reasoning spatial and temporal information is a key research issue in Computer Science and Artificial Intelligence. In this paper, we introduce tools that produce three novel encodings which translate problems in qualitative spatial and temporal reasoning into logic programs for answer set programming solvers. Each encoding reflects a different type of modeling abstraction. We evaluate our approach with two of the most well known qualitative spatial and temporal reasoning formalisms, the Interval Algebra and Region Connection Calculus. Our results show some surprising findings, including the strong performance of the solver for disjunctive logic programs over the non-disjunctive ones on our benchmark problems.

I. INTRODUCTION

The ability of handling various spatial and temporal knowledge is a key feature of an intelligent system. Human beings often describe spatial or temporal information by using a small number of qualitative relations between the objects of interest. Many of these relations and their semantics are formally encoded as a qualitative spatial or temporal calculus. The best known examples are the Interval Algebra (IA) [1], which represent relations between convex intervals on a directed line; and the Region Connection Calculus (RCC-8) [20], which is used to represent topological relations between spatially extended regions. These qualitative calculi do not specify exact time-points and durations of temporal intervals or the exact coordinates of regions, but instead only the essential relations between the intervals and the regions. This made them useful when precise, numerical information is either not needed or not available. IA and RCC-8 have been used in a diverse number of applications including automated planning [2], computational biology [10], diagnosis [18], natural language processing [24], [16], and spatial information systems [11].

One of the fundamental reasoning tasks with these qualitative calculi is to decide consistency for a given set of constraints. That is, for a given finite set of variables and a set of constraints describing the possible spatial or temporal relations between them, find an assignment of variables to members of the spatial or temporal domain such that all the constraints are satisfied. For example, the sentence “Kate had a coffee some time after lunch, and started a discussions with Chris straight after coffee” described three temporal intervals denoting Kate’s lunch, coffee and discussion with Chris, and the temporal constraints between the intervals. The information is clearly consistent, and one can find an infinite

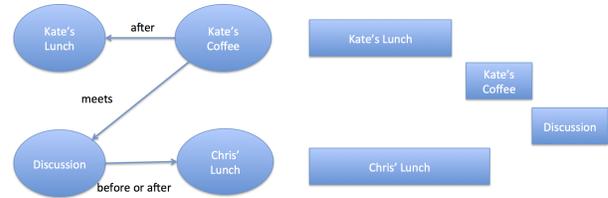


Fig. 1. Left: an example of a constraint network in Interval Algebra, Right: a possible solution to the constraints.

number of actual temporal intervals satisfying the constraints. However, for complex information involving many variables and disjunctive constraints, such as “Chris had a discussion with Kate either before or after his lunch”, a reasoning algorithm is needed to determine whether the information is consistent. A further problem is if the constraints are consistent, how many different qualitatively different scenarios exist. This is related to the well known model enumeration problem and the minimal label problem.

Over the last twenty years, two main type of tools have emerged for deciding consistency for IA and RCC-8 constraints. The first being deploying a backtracking search over tractable sub-instances. Nebel has shown that many difficult instances even in the phase transition region can be solved reasonably efficiently [17], and the later optimized reasoner GQR [6] is able to quickly solve most of the difficult randomly-generated instances. However, this approach observes a heavy-tail problem, where certain instances remain difficult to solve even after a long time. This is somewhat mitigated when no-goods and restarts are introduced in the search [27].

The second approach based on propositional satisfiability (SAT) was first proposed by Pham, Thornton and Sattar [19]. It encodes the constraints into a propositional formulae such that solution finding can be delegated to an off-the-shelf SAT solver. The latest version of such approach [12] showed that it is competitive to GQR for solving the hardest instances at the phase-transition region. The approach was also shown to scale to much larger qualitative calculi [15].

Closely related to propositional satisfiability checking (SAT), Answer Set Programming (ASP) is a form of declarative programming with stable model semantics for solving NP-hard search problems [9]. It represents a problem by a logic program whose answer sets correspond to solutions, and

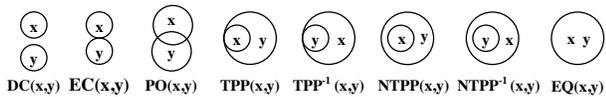


Fig. 2. The 8 base relations of the Region Connection Calculus RCC-8

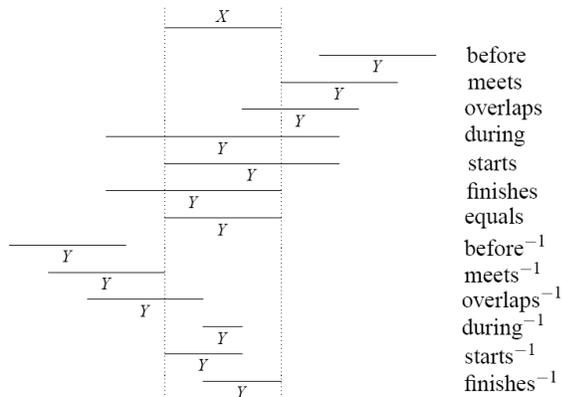


Fig. 3. The 13 base relations of the Interval Algebra

an off the shelf answer set solver can be used for finding the answer sets of a problem. Recent advances in answer set solvers such as *clasp* [7] and *dlv* [13] have shown that the ASP-based approach is promising for solving difficult combinatorial search problems.

While it is possible to use some answer set solvers as SAT solvers and apply the previous methods for solving the problems in qualitative spatial and temporal reasoning, it would be interesting to see how the answer set solvers can solve them as native logic programs. The added benefit being that the higher-level logic programs may be much more human-readable and configurable than propositional formula in DIMACS form accepted by SAT solvers. In this paper, we propose three different approaches for encoding well-known, NP-hard problems in qualitative spatial and temporal reasoning as logic programs for ASP solvers. We evaluate the logic programs with the state of the art grounder/solver *gringo*, *clasp* and *claspD*, and compare the results to existing approaches with the native constraint solver GQR and the SAT encoding with the SAT solver Glucose2 [3].

II. BACKGROUND

A. A Qualitative Calculus

A qualitative calculus formally describes relations between spatial or temporal entities on a possibly infinite domain \mathcal{D} . A set of base relations \mathcal{B} , also known as atomic relations, forms a jointly-exhaustive and pairwise-distinct partition of $\mathcal{D} \times \mathcal{D}$, such that exactly one base relation holds between any two elements of the domain. Disjunction is used to express indefinite information, and we allow all relations $2^{\mathcal{B}}$ from the power set of the set of base relations.

The well known spatial calculus Region Connection Calculus (RCC-8) [20] has eight base relations that describes the relations between regions in a topological space. The

base relations between any two regions is either *disconnected* (*rDC*), *externally connected* (*rEC*), *partial overlap* (*rPO*), *equal* (*rEQ*), *tangential proper part* (*rTPP*), *non-tangential proper part* (*rNTPP*) or the converse relations of the latter two (*rTPPI*, *rNTPPI*) as illustrated in Fig. 2. Similarly, the temporal calculus Interval Algebra (IA) [1] describes the relations between any two intervals along a directed line using thirteen base relations, being *before* (*rb*), *meet* (*rm*), *overlaps* (*ro*), *starts* (*rs*), *finishes* (*rf*), *during* (*rd*), their respective converses (*rbi*, *rim*, *roi*, *rbi*, *rfi*, *rdi*), and the *equal* relation (*req*) as illustrated in Fig. 3. For example, the information “Chris had a discussion with Kate either before or after his lunch” can be described by the relation *discussion* {*before* | *after*} *lunch*.

If the relations between the variables x to y and y to z is known, the set of possible base relations from x to z is given by the weak-composition table of the calculus. More formally, for base relations $R, S \in \mathcal{B}$, we define the weak-composition $R \circ S$ to be the smallest relation $Q \in 2^{\mathcal{B}}$ containing $R \circ S$, where $R \circ S$ is the standard set-theoretic composition of R and S . For example, if we know Kate had a coffee some time after lunch, and that she had a discussion with Chris after the coffee, weak composition of the IA relations allows us to infer that the discussion took place after lunch.

Therefore, instances of spatial and temporal information in QSTR can be described as a qualitative constraint network (QCN), which is a graph where the nodes denote the spatial or temporal variables, and the labels on the directed edges between the nodes describe the possible relations between them. If no constraints are specified between two variables, then the universal relation, which is the disjunction of all possible relation, is placed on the label of the edge between the variables. However, if no label exists between two nodes, then it means that no relation is possible between the two variables, and hence the network is inconsistent.

Definition 1 (Qualitative Constraint Network): For a given qualitative calculus with a set of base relations \mathcal{B} , a qualitative constraint network (QCN) over \mathcal{A} is a pair (V, l) where V is a set of vertices (nodes), and $l : V^2 \rightarrow 2^{\mathcal{B}}$ is any function.

Thus a QCN of a qualitative calculus is a complete directed graph labelled by the power set of the base relations of the calculus. In this paper we would refer to the relation on the label between nodes i, j as R_{ij} .

Similar to other constraint satisfaction problems (CSP), the network is consistent if and only if we can find an assignment of variables to elements of the domain where all specified constraints, either from the labels of the network or the weak-composition table, are satisfied. However, contrary to other CSPs, the domains in QSTR are typically infinite, which renders many standard CSP solving techniques inapplicable.

B. Deciding Consistency

Deciding consistency have been shown to be NP-hard for many qualitative calculi, including IA and RCC-8 [25], [21], [22]. However, one can show that for many qualitative calculi including IA and RCC-8, if the constraint network only allow

base (atomic) relations (we also call such network atomic networks), then the cubic-time *algebraic closure* (a-closure) algorithm is sufficient for deciding consistency [25], [21]. Formally, a QCN (V, l) is algebraically closed (a-closed), if $R_{ik} \subseteq R_{ij} \diamond R_{jk}$ for all $i, j, k \in V$. The a-closure algorithm removes impossible relations between any three variables until a fixed point is reached, at which we say the set of constraints is a-closed. If it terminates early due to the fact that no relation is possible between two variables, then the information is inconsistent. Thus, deciding consistency for a set of constraints in IA or RCC-8 can be reduced to a search for a set of a-closed constraints that was implied by the original set of constraints.

Native constraint solvers such as the Generic Qualitative Reasoner (GQR) [6] use a backtracking search for a tractable fragment that is a logical consequence of the original network. At each step of the search, a-closure is enforced to prune the labels of the edges, and powerful heuristics using *tractable subsets* reduces both the depth and branching factor of the search. The approach has been compared favorably against other existing solvers [26], and delivered the fastest average solving time for most instances. However, it has been observed that for certain instances in the most difficult phase transition region the algorithm may get stuck for hours without finding a solution [14].

Alternatively, the approach based on propositional satisfiability checking (SAT) relies on encoding the problem as a low-level propositional formulae in DIMACS form, and satisfiable models of the formulae corresponds to solutions to the original problem. Pham, Thornton and Sattar [19] first proposed such encodings, and showed that they are competitive against Nebel’s constraint-based solver. Later, Li, Huang and Renz [14] showed that some of the clauses in the earlier encoding is in fact redundant, and proposed more compact encoding that, when combined with the latest SAT solver, is competitive against the GQR on the most difficult instances in the phase-transition region.

III. ENCODING QCN AS LOGIC PROGRAMS

In this paper, we propose three different approaches to encode problems in qualitative spatial and temporal reasoning into logic programs for answer set solvers, each representing a different level of abstraction. The first uses a direct encoding of constraints without disjunctions or negations; the second uses disjunctions to model possible relations from weak composition, while the third mimics a propositional satisfiability formula with low level descriptions of relations between the variables. The first two encodings are applicable to any qualitative spatial and temporal calculus where a-closure decides consistency for atomic networks, while the third encoding in addition requires the qualitative calculus to have the atomic network amalgamation property (aNAP) as described in [14]. Both the RCC-8 and IA satisfy these requirements. The three encoding differs in how the constraints are specified (direct vs. disjunctive), and that encoding 2 and 3 (disjunctive and low level) allow disjunctions.

A. The Direct Encoding

In the first encoding, we directly encode the constraints as integrity constraints. This means specifying everything that is not permitted by the qualitative calculus and the constraint network. It is the only one of the three encoding that does not use disjunctions, which is known to be a cause of complexity in logic programs for ASP.

First, the logic program declares the base relations. So for the IA, the program has the following describing the thirteen base relations:

```
rel(req).
rel(rp). rel(rpi). rel(rd). rel(rdi).
rel(ro). rel(roi). rel(rm). rel(rmi).
rel(rs). rel(rsi). rel(rf). rel(rfi).
```

Similarly for RCC-8, the following is used for describing the eight base relations:

```
rel(rEQ). rel(rDC). rel(rEC). rel(rPO).
rel(rTPP). rel(rNTPP). rel(rTPPI).
rel(rNTPPI).
```

It is then followed by a choice-rule that specifies that between any two variable only one base relation is permitted in any answer set to the program.

```
1{label(X, Y, L) : rel(L)}1
:- node1(X), node2(Y), X < Y.
```

Lastly, we declare the size of the constraint network. So for a network with 20 nodes, we have the following:

```
node1(0..19). node2(0..19).
```

We then specify all the possible constraints arise from weak composition. Recall that weak composition is defined as the set of possible relations between variables x and z if the relations between variables x, y and y, z is known. So we specify all the base relations outside the weak composition as integrity constraints.

For example, in IA, if interval X is *before* Y , and Y is *during* Z , then we can infer that the endpoint of interval X cannot be equal or after the endpoint of Z . This rules out the following relations between X and Z : $\{req, rbi, rdi, rsi, rmi, roi, rf, rfi\}$. Thus, we write down the following integrity constraints:

```
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, req).
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, req).
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, rbi).
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, rdi).
```

```

:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, rsi).
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, rmi).
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, roi).
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, rf).
:- label(X, Y, rb), label(Y, Z, rd),
label(X, Z, rfi).

```

Alternatively, in RCC-8, if region X is *disconnected* from region Y , and region Y *partially overlaps* with region Z , then it can be inferred from weak composition the following:

- region X cannot be *equal* to region Z
- region X cannot be containing region Z . (*inverse tangential proper part* and *inverse non-tangential proper part*)

Hence, we can write down the following integrity constraints:

```

:- label(X, Y, rDC), label(Y, Z, rPO),
label(X, Z, rEQ).
:- label(X, Y, rDC), label(Y, Z, rPO),
label(X, Z, rTPPI).
:- label(X, Y, rDC), label(Y, Z, rPO),
label(X, Z, rNTPPI).

```

Lastly, we write down all the relations ruled out by the QCN specification. Typically, a network is specified by the permitted relations between the nodes, so we take the complement of the specified relations to get the forbidden relations. For example, for the constraints between node 0 and node 2, we permit the following relation $\{rDC, rTPP, rNTPP, rTPPI\}$, we write the following rules into the logic program that rules out the other relations:

```

:- label(0, 2, rEQ).
:- label(0, 2, rEC).
:- label(0, 2, rPO).
:- label(0, 2, rNTPPI).

```

The IA relations can be written in a similar way. Once all the relations specified by the network are translated into integrity constraints, the program is complete. This encoding can be formally defined as the following:

Definition 2 (Direct Encoding): A direct encoding of a QCN (V, l) over a qualitative calculus with base relation \mathcal{B} into a logic program is consisted of the following set of rules:

- Base relation declarations:
 $\{\text{rel}(R) .\} \forall R \in \mathcal{B}$
- Choice rule specifying exactly one relation per edge:
 $1\{\text{label}(X, Y, L) : \text{rel}(L)\}1$
 $:- \text{node1}(X), \text{node2}(Y), X < Y.$
- QCN size
 $\text{node1}(0..|V|-1). \text{node2}(0..|V|-1).$

- The set of integrity constraints from weak-composition
 $\{:- \text{label}(X, Y, R1), \text{label}(Y, Z, R2),$
 $\text{label}(X, Z, R3) .\}$
 $\forall R1, R2, R3 \in \mathcal{B}, R3 \notin R1 \diamond R2$
- The set of integrity constraints from the QCN
 $\{:- \text{label}(X, Y, R)\}$
 $\forall X, Y \in V, R \in \mathcal{B} \text{ s.t. } X < Y, R \notin R_{XY}$

Now we can show the following holds for our encoding:

Theorem 1: A QCN Θ over IA or RCC-8 is consistent if and only if its direct encoding $\text{dir}(\Theta)$ has a solution.

Proofsketch 1: The logic program $\text{dir}(\Theta)$ was created in such way that its solutions corresponds to a-closed atomic networks that are the consequences of Θ . As a-closure decides consistency over atomic networks of IA and RCC-8, Θ is consistent if and only if the encoded logic program has a solution.

B. The Disjunctive Encoding

The second encoding converts a QCN into a disjunctive logic program. Similar to the first encoding, we also declaring the relations, the choice constraint, and the network size in the exact same way. However, when we describe all possible weak composition between the relations, contrary to the direct encoding which rule out impossible relations with integrity constraints, here we describes the relations that are *possible*. So the IA example where interval X is *before* interval Y and Y *during* interval Z , we write:

```

label(X, Z, rb) | label(X, Z, rd) |
label(X, Z, rs) | label(X, Z, rm) |
label(X, Z, ro) :-
label(X, Y, rb), label(Y, Z, rd).

```

Similarly, for the same RCC-8 example where region X is *disconnected* from region Y , and region Y *partially overlaps* with region Z , we write the following rule:

```

label(X, Z, rDC) | label(X, Z, rEC) |
label(X, Z, rPO) | label(X, Z, rTPP) |
label(X, Z, rNTPP) :-
label(X, Y, rDC), label(Y, Z, rPO).

```

This is followed by the description of the network, which is simply list of disjunctions denoting possible relations on given edges. So in the same example of relations $\{rDC, rTPP, rNTPP, rTPPI\}$ between nodes 0 and 2, we have the following rule:

```

label(0, 2, rDC) | label(0, 2, rTPP) |
label(0, 2, rNTPP) | label(0, 2, rTPPI).

```

A similar disjunction rule can be written for IA relations. Once all the relations specified by the network are described in disjunctions, the program is complete. This encoding can be formally defined as the following:

Definition 3 (Disjunctive Encoding): A disjunctive encoding of a QCN (V, l) over a qualitative calculus with base relation \mathcal{B} into a logic program is consisted of the following set of rules:

- Base relation declarations:
 $\{\text{rel}(R) .\} \forall R \in \mathcal{B}$
- Choice rule specifying exactly one relation per edge:
 $1\{\text{label}(X, Y, L) : \text{rel}(L)\}1$
 $:- \text{node1}(X), \text{node2}(Y), X < Y.$
- QCN size
 $\text{node1}(0..|V|-1). \text{node2}(0..|V|-1).$
- The set of disjunctive rules from weak-composition
 $\{ (\bigvee_{R3 \in R1 \diamond R2} \text{label}(X, Z, R3)) :-$
 $\text{label}(X, Y, R1), \text{label}(Y, Z, R2). \}$
 $\forall R1, R2 \in \mathcal{B}$
- The set of disjunctive rules from the QCN
 $\{\bigvee \text{label}(X, Y, R)\}.$
 $\forall X, Y \in V, R \in \mathcal{B} \text{ s.t. } X < Y, R \notin R_{XY}$

Similar to the direct encoding, the solutions of the logic program corresponds to a-closed atomic networks that are the consequences of the original network. Hence we have the following:

Theorem 2: A QCN Θ over IA or RCC-8 is consistent if and only if its disjunctive encoding $dis(\Theta)$ has a solution.

C. The Low-Level Encoding

The third encoding is a low level encoding taken directly from the SAT encoding of [14]. Here, instead of describing weak composition between relations as general rules, the program first perform a-closure preprocessing to eliminate unnecessary relations on the labels, and then encode the weak composition between all triples of nodes except those that can be safely ignored as described in [14]. Contrary to the previous encodings, we do not have to declare relations and labels, but directly use them in the logic program.

Similar to the SAT encoding, instead of writing a choice constraint, we encode the constraint for every edge as a set of rules. The first denoting that on each edge at least one relation holds. So for the example of relations $\{rDC, rTPP, rTPPI\}$ between nodes 0 and 2, we have the following rules:

$$\text{label}(0, 2, rDC) \mid \text{label}(0, 2, rTPP) \mid \text{label}(0, 2, rTPPI).$$

For the same edge, we then have the set of rules that describe that at most one of the relations holds, which is the following in our example:

$$\begin{aligned} &:- \text{label}(0, 2, rDC) \mid \text{label}(0, 2, rTPP). \\ &:- \text{label}(0, 2, rTPP) \mid \text{label}(0, 2, rTPPI). \\ &:- \text{label}(0, 2, rDC) \mid \text{label}(0, 2, rTPP). \end{aligned}$$

Similarly, in for the IA relation $\{rb, rbi\}$ holds between intervals 0 and 2, we write the following:

$$\begin{aligned} &\text{label}(0, 2, rb) \mid \text{label}(0, 2, rbi). \\ &:- \text{label}(0, 2, rb) \mid \text{label}(0, 2, rbi). \end{aligned}$$

We then proceed to encode the weak-composition constraints. For example, for the triples of nodes 0, 1, 2 with the specified relations $\text{label}(0, 1, rDC)$, $\text{label}(1, 2, rPO)$, and $\text{label}(0, 2, rEQ) \mid \text{label}(0, 2, rDC) \mid \text{label}(0, 2, rEC)$, we write the rule that permits the remaining relations not ruled out by weak-composition:

$$\begin{aligned} &\text{label}(0, 2, rDC) \mid \text{label}(0, 2, rEC) :- \\ &\text{label}(0, 1, rDC), \text{label}(1, 2, rPO). \end{aligned}$$

Here, $\text{label}(0, 2, rEQ)$ is ruled out by weak-composition, and hence don't appear in the implication. Similarly in IA with triples of nodes with relations $\text{label}(0, 1, rb)$, $\text{label}(1, 2, rd)$ and $\text{label}(0, 2, rb) \mid \text{label}(0, 2, rbi)$, from the previous example we know that the relation $\text{label}(0, 2, rbi)$ is ruled out, and we write:

$$\begin{aligned} &\text{label}(0, 2, rb) :- \text{label}(0, 1, rb), \\ &\text{label}(1, 2, rd) \end{aligned}$$

Once this is done for all triple of nodes that may not be ignored as described in [14], the programs complete. Formally, the encoding is defined as the following:

Definition 4 (Low-Level Encoding): A low-level encoding of a QCN (V, l) over a qualitative calculus with base relation \mathcal{B} into a logic program is consisted of the following set of rules:

- At-least-one (ALO) rules. Every edge has at least one base relation after a-closure preprocessing.
 $\{\bigvee \text{label}(a, b, R)\}.$
 $\forall a, b \in V, a < b, R \in R_{ab},$
 $R \in R_{ac} \diamond R_{cb} \forall c \in V$
- At-most-one (ALO) rules. Every edge has at most one base relation.
 $\{:- \text{label}(a, b, R1), \text{label}(a, b, R2).\}$
 $\forall a, b \in V, a < b, R1, R2 \in R_{ab},$
 $R1, R2 \in R_{ac} \diamond R_{cb} \forall c \in V$
- Weak composition triples. Specifying possible relations for all triples that are not ignored in the scheme [14].
 $\{ (\bigvee_{R3 \in R1 \diamond R2} \text{label}(a, c, R3)) :-$
 $\text{label}(a, b, R1), \text{label}(b, c, R2). \}$
 $\forall a, b, c \in V, R1 \in R_{ab}, R2 \in R_{bc},$
 $(a, b, c) \text{ is not ignored.}$

As the low-level encoding corresponds to the SAT encoding of [14], we have the following corollary:

Corollary 1: A QCN Θ over IA or RCC-8 is consistent if and only if its low-level encoding $low(\Theta)$ has a solution.

IV. EMPIRICAL EVALUATION AND DISCUSSIONS

We evaluated our approach on well known qualitative spatial and temporal calculi RCC-8 and Interval Algebra. All the encoded, high-level logic programs were first grounded to *smodel* format using the grounder *gringo* [8]. We then used

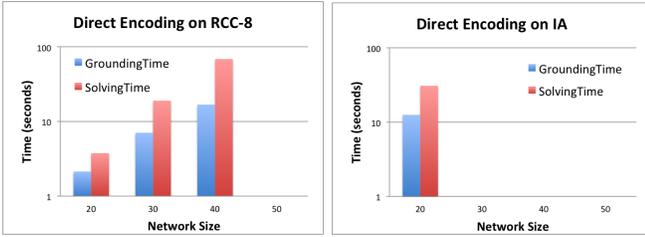


Fig. 4. Grounding and solving times for the direct encoding of RCC-8 (left) and IA (right)

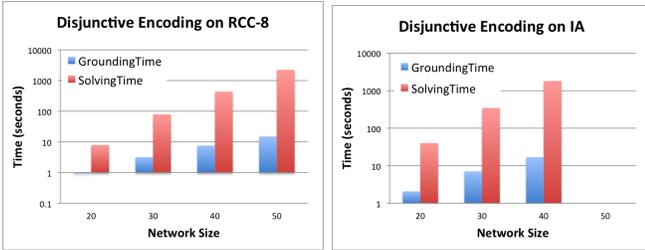


Fig. 5. Grounding and solving times for the disjunctive encoding of RCC-8 (left) and IA (right)

the answer set solver *clasp* [7] for solving the instances with the direct encoding, and the solver *claspD* [5] for the instances with the disjunctive and low-level encodings. The encoders, written in `Perl` for the direct and disjunctive encoding and `C++` for the low-level encoding, translate weak-composition tables and constraint networks from the standard format in GQR to logic programs. As the encoders finish instantaneously for all instances, we will be focused on the grounding and solving time (their sum being the total solving time) in our evaluation.

We first evaluated our three encodings against a benchmark of RCC-8 instances with only difficult relations in the phase-transition region. For network sizes 20, 30, 40 and 50, we created 200 randomly generated instances for each network size. We then evaluate the encodings against a benchmark of difficult IA instances also in the phase-transition region, with 100 randomly generated instances for each network size.

We observed that the direct encoding failed all RCC-8 instances of size 50, and all IA instances of size 30 and above;

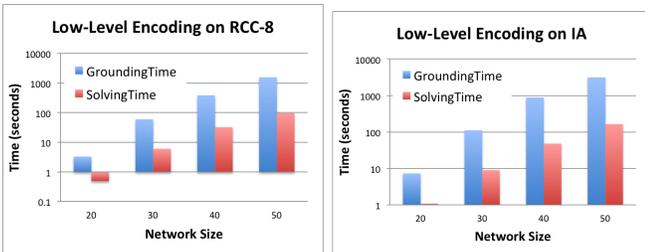


Fig. 6. Grounding and solving times for the low-level encoding of RCC-8 (left) and IA (right)

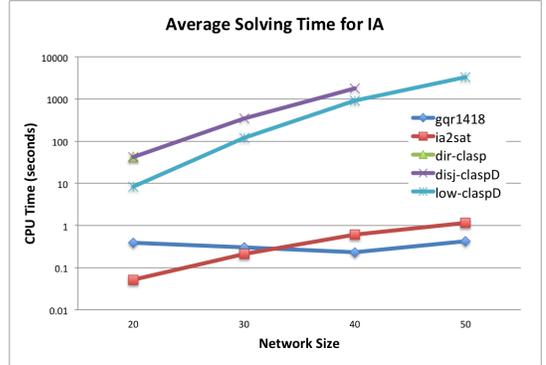
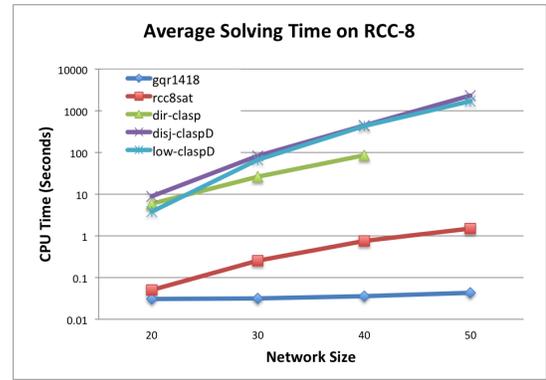


Fig. 7. Comparison of the average CPU time of all solvers in RCC-8 (top) and IA (bottom)

the disjunctive encoding solve all instances except IA instances of size 50; and the low-level encoding failed 18 out of the 100 IA instances size 50. We also note that while the solver on the direct encoding and disjunctive encoding spent most of its time solving, the solver on the low-level encoding actually spent most of its time grounding. One somewhat surprising result is the strong performance of the disjunctive approaches to the non-disjunctive one, as disjunction is usually associated with costs to the performance in answer set solving.

We then compare the performance of the ASP solvers against the current state of the art solvers from both constraint-based approaches and SAT-based approaches (Fig. 7). For constraint search we used GQR1418 [6], and for SAT we used the encoder from [14] with the SAT solver Glucose2 [3]. Here we note the existing gap in performance between the state of the art approaches and the current ASP solvers. However, the performance are expected to improve as ASP solvers become more efficient.

V. CONCLUSION

We proposed three novel approaches for encoding problems of deciding consistency of qualitative constraint networks in IA and RCC-8 as logic programs for ASP solvers. We evaluated the grounding and solving time of the three encodings, and found that the use disjunctions in the ASP encoding actually allows the problems to be solved more efficiently. While current ASP solvers are not as fast as the state-of-the-art constraint or SAT approaches for solving these problems,

these encodings nevertheless provide an interesting class of benchmark problems for the ASP research community.

Possible future work includes making use of the known maximal tractable subsets of the qualitative calculi to improve solving performance similar to that from the search-based approaches [17], encode different classes of consistencies [4], and handling the implicit constraints of qualitative calculi where a-closure does not decide consistency [23].

ACKNOWLEDGMENT

The author would like to thank Francesco Ricca and Guohui Xiao for their discussions that led to this work. This work was supported by the OpenSense project funded by the Nanotera.ch program.

REFERENCES

- [1] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.
- [2] J. F. Allen. Planning as temporal reasoning. In James F. Allen, Richard Fikes, and Erik Sandewall, editors, *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pages 3–14. Morgan Kaufmann, 1991.
- [3] G. Audemard and L. Simon. Predicting Learnt Clauses Quality in Modern SAT Solver. *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI'09)*, 2009.
- [4] J.-F. Condotta and C. Lecoutre. A Class of \diamond_f -Consistencies for Qualitative Constraint Networks. *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference (KR'10)*, Toronto, Canada, 2010.
- [5] C. Drescher, M. Gebser, T. Grote, B. Kaufmann, A. König, M. Ostrowski and T. Schaub, Conflict-Driven Disjunctive Answer Set Solving. *Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, pages 422–432, 2008.
- [6] Z. Gantner, M. Westphal, and S. Wöfl. GQR—A fast reasoner for binary qualitative constraint calculi. In *Proceedings of AAAI'08 Workshop on Spatial and Temporal Reasoning*, 2008.
- [7] M. Gebser, B. Kaufmann, A. Neumann and T. Schaub Conflict-Driven Answer Set Solving *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 386–392, 2007.
- [8] M. Gebser, R. Kaminski, A. König and T. Schaub. Advances in gringo Series 3 *Proceedings of the Eleventh International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'11)*, pages 345–351, 2011.
- [9] M. Gelfond and V. Lifschitz The stable model semantics for logic programming. In Kowalski, R., and Bowen, K., eds., *Proceedings of International Logic Programming Conference and Symposium*, 10701080. MIT Press. 1988.
- [10] M. C. Golumbic and R. Shamir. Complexity and algorithms for reasoning about time: A graph-theoretic approach. *Journal of the ACM*, 40(5):1108–1133, 1993.
- [11] V. Haarslev, C. Lutz, and R. Möller. Foundations of spatioterminological reasoning with description logics. In *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 112–123. 1998.
- [12] J. Huang, J. J. Li and J. Renz Decomposition and Tractability in Qualitative Spatial and Temporal Reasoning To appear in *Artificial Intelligence*, doi:10.1016/j.artint.2012.09.009.
- [13] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. The DLV system for knowledge representation and reasoning. *ACM Transactions on Computational Logic*, 7(3):499–562, 2006.
- [14] J. J. Li, J. Huang, and J. Renz. A divide-and-conquer approach for solving interval algebra networks. *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI'09)*, pages 572–577, 2009.
- [15] J. J. Li and J. Renz. In defense of large qualitative calculi. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI'10)*, pages 315–320, 2010.
- [16] W. Maab, P. Wazinski, and G. Herzog. Vitra guide: multimodal route descriptions for computer assisted vehicle navigation. In *Proceedings of the 6th international conference on Industrial and engineering applications of artificial intelligence and expert systems*, pages 144–147. 1993.
- [17] B. Nebel. Solving hard qualitative temporal reasoning problems: Evaluating the efficiency of using the ord-horn class. *Constraints*, 1(3):175–190, 1997.
- [18] K. Nökel. *Temporally Distributed Symptoms in Technical Diagnosis*, volume 517 of *Lecture Notes in Computer Science*. Springer, 1991.
- [19] D. N. Pham, J. Thornton, and A. Sattar. Modelling and solving temporal reasoning as propositional satisfiability. *Artificial Intelligence*, 172(15):1752–1782, 2008.
- [20] D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *Principle of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference (KR'92)*, pages 165–176, 1992.
- [21] J. Renz. Maximal tractable fragments of the region connection calculus: A complete analysis. In Thomas Dean, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 448–455. Morgan Kaufmann, 1999.
- [22] J. Renz and J. J. Li. Automated complexity proofs for qualitative spatial and temporal calculi. In Gerhard Brewka and Jérôme Lang, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 11th International Conference (KR'08)*, pages 715–723. AAAI Press, 2008.
- [23] J. Renz. Implicit Constraints for Qualitative Spatial and Temporal Reasoning In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference (KR'12)*, Rome, Italy, June 10–14, 2012.
- [24] F. Song and R. Cohen. The interpretation of temporal relations in narrative. In *Proceedings of the 7th National Conference on Artificial Intelligence (AAAI'88)*, pages 745–750, 1988.
- [25] M. B. Vilain and H. A. Kautz. Constraint propagation algorithms for temporal reasoning. In *Proceedings of the 5th National Conference on Artificial Intelligence (AAAI'86)*, pages 377–382, 1986.
- [26] M. Westphal and S. Wöfl. Qualitative CSP, Finite CSP, and SAT: Comparing Methods for Qualitative Constraint-based Reasoning. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI'09)*, pages 628–633, 2009.
- [27] M. Westphal, S. Wöfl and J. J. Li. Restarts and Nogood Recording in Qualitative Constraint-based Reasoning. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI'10)*, Lisbon, Portugal, August 2010.