Qualitative Spatial and Temporal Reasoning: Efficient Algorithms for Everyone

Jochen Renz
Research School of Information Sciences and Engineering
The Australian National University
Canberra, ACT 0200, Australia

Abstract

In the past years a lot of research effort has been put into finding tractable subsets of spatial and temporal calculi. It has been shown empirically that large tractable subsets of these calculi not only provide efficient algorithms for reasoning problems that can be expressed with relations contained in the tractable subsets, but also surprisingly efficient solutions to the general, NP-hard reasoning problems of the full calculi. An important step in this direction was the refinement algorithm which provides a heuristic for proving tractability of given subsets of relations. In this paper we extend the refinement algorithm and present a procedure which identifies large tractable subsets of spatial and temporal calculi automatically without any manual intervention and without the need for additional NP-hardness proofs. While we can only guarantee tractability of the resulting sets, our experiments show that for RCC8 and the Interval Algebra, our procedure automatically identifies all maximal tractable subsets. Using our procedure, other researchers and practitioners can automatically develop efficient reasoning algorithms for their spatial or temporal calculi without any theoretical knowledge about how to formally analyse these calculi.

1 Introduction

Dealing with spatial and temporal information is an essential part of any intelligent system and of our everyday lives. In the area of qualitative spatial and temporal representation and reasoning, information is represented in terms of qualitative relationships between spatial or temporal entities, such as "A is left of B" or "C occurs during D". It is distinguished between different aspects of space and time, such as direction, topology, distance, size/duration, or shape. For all these aspects, qualitative relationships can then be defined on a level of granularity that suits a given application.

When developing a qualitative spatial or temporal calculus, one has to select the domain of entities that are being described. This could be points in a d-dimensional space, intervals or line segments, or extended spatial regions. The set of spatial or temporal entities is usually infinite. The next step is to define a pairwise disjoint and jointly exhaustive set of base relations $\mathcal{B}$ (we assume in the following binary relations), i.e., between any two entities of our domain, exactly one of the base relations holds. The best known calculi are the RCC8 calculus [Randell *et al.*, 1992] which consists of eight topological base relations for extended spatial regions and the Interval Algebra [Allen, 1983] which consists of 13 base relations for intervals. Many other calculi are defined in the literature for different entities, aspects and granularities.

Knowledge about the relationships between given entities can be represented in the form of constraints xRy where x and y are variables over the domain of entities and R is a relation of the full set of relations $2^{\mathcal{B}}$. The powerset of the base relations is used in order to be able to express indefinite information in cases where the exact relationship is unknown. Given a set of such constraints we can then formulate different reasoning problems such as deriving unknown relations, eliminating impossible base relations from indefinite relations, computing the minimal representation, or checking the consistency of the given set of constraints. Since most reasoning problems can be polynomially reduced to the consistency problem, this is considered to be the most important reasoning problem. The theoretical properties of the consistency problem depend on the chosen set of relations and the chosen domain. While the consistency problem is usually NP-hard if all relations of $2^{\mathcal{B}}$ are permitted, for many calculi it is tractable when only the base relations $\mathcal{B}$ are used. These two extremes in the complexity of deciding consistency have led to a considerable research effort in the past 20 years to identify larger tractable subsets of a given set of relations $2^{\mathcal{B}}$ or even maximal tractable subsets which mark the boundary between tractable and NP-hard subsets of $2^{\mathcal{B}}$.

The significance of having large tractable subsets is enourmous as it not only allows tractable reasoning in cases where the given information is restricted to relations of a tractable subset, but mainly because it considerably speeds up the time for deciding NP-hard instances of the consistency problem. In emprirical studies [Renz and Nebel, 2001] it has been shown that almost all instances of the NP-hard consistency problem for RCC8, even those instances in the phase transition, which are commonly considered to be the hardest instances, could be solved surprisingly fast by combining different heuristics based on the maximal tractable subsets of RCC8. Similar results were found for the Interval Algebra

[Nebel, 1997]. The reasoning algorithms of these studies can be used for any spatial or temporal calculus provided that tractable subsets are known which contain all base relations. Therefore, tractable subsets are essential for efficient solutions to the spatial and temporal reasoning problems, which in turn is a requirement for applications of these calculi.

Identifying tractable subsets is a very hard problem, in particular when spatial entities like extended spatial regions are involved which cannot be easily represented. However, due to the similar structure of the consistency problem over spatial and temporal relations–they are all based on a set of base relations and the same set of operators–it was possible to develop a general method that can be applied to all sets of relations and which allows to prove tractability of a given subset simply by running an algorithm [Renz, 1999]. The so-called refinement method requires as input two sets of relations $\mathcal{S}$ and $\mathcal{T}$, $\mathcal{S}$ is the set that is being tested for tractability and $\mathcal{T}$ is a set for which it is known that a-closure decides consistency. The refinement method further needs a refinement strategy, which is a mapping of relations $S \in \mathcal{S}$ to relations $T \in \mathcal{T}$ for which T is a refinement of S, i.e, $T \subseteq S$. Since there are $2^{2^{|\mathcal{B}|}}$ different subsets for a given set of base relations $\mathcal{B}$, we cannot apply the refinement method to all subsets, but can only run it for a small number of candidate sets. One way of identifying these candidate sets is to make a number of NP-hardness proofs in order to identify relations that make the consistency problem NP-hard. These relations can be used to restrict the number of candidate sets for tractable subsets. Further, we have to find a refinement strategy for each candidate set to a corresponding set $\mathcal{T}$. Due to the large amount of existing theoretical analysis on RCC8 and the Interval Algebra, e.g., a large number of NP-hard relations was known, it was possible to apply the refinement method to these two calculi and to identify large tractable subsets. But what if we have a calculus where no theoretical results are yet available?

In this paper, we extend the refinement method and present a general procedure by which large tractable subsets of arbitrary spatial and temporal calculi can be identified fully automatically without the need for any manual interaction. This procedure will return one or more tractable subsets of a given spatial or temporal calculus. While we cannot guarantee maximality of the resulting tractable subsets, the procedure identifies all maximal tractable subsets of RCC8 and the Interval Algebra. But even if the resulting tractable subsets are not maximal, they can still give us efficient solutions to the NP-hard consistency problem if they are used as split sets in Renz and Nebel's [2001] reasoning algorithms.

The paper is structured as follows. In section 2 we give an introduction to spatial and temporal reasoning and to the refinement method. In section 3 we describe our procedure, in section 4 we apply it to RCC8 and the Interval Algebra and compare the output of the procedure to the known results. Finally, in section 5 we discuss our results.

2 Background

2.1 Qualitative spatial and temporal reasoning

A qualitative spatial or temporal calculus [Ligozat and Renz, 2004] consists of a domain $\mathcal{D}$ of spatial or temporal entities which is usually infinite and a set of base relations $\mathcal{B}$ which partitions $\mathcal{D} \times \mathcal{D}$ into jointly exhaustive and pairwise disjoint relations, i.e., between any two entities of $\mathcal{D}$ exactly one base relation holds. Since the domains are infinite, we do not work on the tuples contained in the relations but only manipulate the relation symbols. We have different operators for doing so, union ($\cup$), intersection ($\cap$), converse ($\smile$), composition ($\circ$) and weak composition ($\diamond$) of relations. Composition is defined as $R \circ S = \{(a,c) | \exists b.(a,b) \in R \text{ and } (b,c) \in S\}$. Since we cannot deal with the tuples contained in the relations, we can in many cases only use weak composition which is defined as $R \diamond S = \{T | T \in \mathcal{B} : T \cap (R \circ S) \neq \emptyset\}$. The relations used by a spatial and temporal calculus are those contained in the powerset $2^{\mathcal{B}}$ of the base relations. The *consistency problem* $\mathsf{CSPSAT}(\mathcal{S})$ for spatial and temporal calculi, where $\mathcal{S}$ is a subset of $2^{\mathcal{B}}$ is defined as follows:

Instance: A set $\mathcal{V}$ of n variables over a domain $\mathcal{D}$ and a set Θ of constraints xRy where $R \in \mathcal{S} \subseteq 2^{\mathcal{B}}$ and $x, y \in \mathcal{V}$.

Question: Is there an instantiation of all n variables in Θ with values from $\mathcal{D}$ which satisfies all constraints in Θ?

The consistency problem is in general an NP-hard problem, but if a subset $\mathcal{T}$ of $2^{\mathcal{B}}$ is known for which the consistency problem can be solved in polynomial time, a so-called *tractable subset*, then the consistency problem can be solved by splitting Θ into sets of constraints Θ' such that for every constraint $xRy \in \Theta$ there is a constraint $xR'y \in \Theta'$ where R' is a *refinement* of R, i.e., $R' \subseteq R$, and by backtracking over all possible sets Θ' [Nebel, 1997].

A local consistency algorithm for approximating the consistency problem is the algebraic closure algorithm, or a-closure algorithm [Renz and Ligozat, 2005]. It successively makes every triple $\langle x, y, z \rangle$ of variables in Θ a-closed by applying the following operation until a fixpoint occurs: $xRy := xRy \cap (xS \diamond Ty)$ where xSz and $zTy \in \Theta$. If a fixpoint occurs, Θ is called *algebraically closed* or *a-closed*. If the empty relation occurs during this process, Θ is inconsistent. In many cases a-closure decides consistency of $\mathsf{CSPSAT}(\mathcal{B})$. This will be a requirement for applying our methods. In the following we say that a set of relations $\mathcal{S}$ is tractable/NP-hard if $\mathsf{CSPSAT}(\mathcal{S})$ is tractable/NP-hard. Similar, a relation R is said to be tractable/NP-hard, if $\mathsf{CSPSAT}(\mathcal{B} \cup \{R\})$ is tractable/NP-hard.

2.2 The refinement method

The refinement method [Renz, 1999] is a general method for proving tractability of $\mathsf{CSPSAT}(\mathcal{S})$ for given sets $\mathcal{S} \subseteq 2^{\mathcal{B}}$. It requires a subset $\mathcal{T}$ of $2^{\mathcal{B}}$ for which a-closure is already known to decide $\mathsf{CSPSAT}(\mathcal{T})$. Then the method checks whether it is possible to refine every constraint involving a relation in $\mathcal{S}$ according to a given refinement strategy to a constraint involving a relation in $\mathcal{T}$ without changing consistency. It is based on the following definition.

Definition 1 (Reduction by Refinement)
Let $\mathcal{S}, \mathcal{T} \subseteq 2^{\mathcal{B}}$. $\mathcal{S}$ can be reduced by refinement *to $\mathcal{T}$, if the two conditions are satisfied:*

1. *for every $S \in \mathcal{S}$ there is a $T_S \in \mathcal{T}$ with $T_S \subseteq S$,*

Algorithm: CHECK-REFINEMENTS
Input: A set $\mathcal{S}$ and a refinement matrix M of $\mathcal{S}$.
Output: `fail` if refinements in M can make a-closed triples of constraints over $\mathcal{S}$ inconsistent; `succeed` otherwise.

1. changes $\leftarrow true$
2. *while* changes *do*
3. $oldM \leftarrow M$
4. *for every* algebraically closed triple
5. $T = (R_{12}, R_{23}, R_{13})$ of relations over $\mathcal{S}$ *do*
6. *for every* refinement $T' = (R'_{12}, R'_{23}, R'_{13})$ of T
7. with $oldM[R_{12}][R'_{12}] = oldM[R_{23}][R'_{23}] =$
8. $oldM[R_{13}][R'_{13}] = true$ *do*
9. $T'' \leftarrow$ ALGEBRAIC-CLOSURE(T')
10. *if* $T'' = (R''_{12}, R''_{23}, R''_{13})$ contains the empty
11. relation *then return* `fail`
12. *else do* $M[R_{12}][R''_{12}] \leftarrow true$,
13. $M[R_{23}][R''_{23}] \leftarrow true$, $M[R_{13}][R''_{13}] \leftarrow true$
14. *if* $M = oldM$ *then* changes $\leftarrow$ *false*
15. *return* `succeed`

Figure 1: Algorithm CHECK-REFINEMENTS [Renz, 1999]

2. *every a-closed set Θ of constraints over $\mathcal{S}$ can be refined to a set Θ' of constraints over $\mathcal{T}$ by replacing $x_i S x_j \in \Theta$ with $x_i T_S x_j \in \Theta'$ for $i < j$, such that enforcing a-closure to Θ' does not result in the empty relation.*

It is clear that if a-closure decides $\mathsf{CSPSAT}(\mathcal{T})$ for a set $\mathcal{T} \subseteq 2^{\mathcal{B}}$, and $\mathcal{S}$ can be reduced by refinement to $\mathcal{T}$, then a-closure decides $\mathsf{CSPSAT}(\mathcal{S})$. Therefore, it is sufficient for proving tractability of $\mathsf{CSPSAT}(\mathcal{S})$ to show that $\mathcal{S}$ can be reduced by refinement to a set $\mathcal{T}$ for which a-closure decides $\mathsf{CSPSAT}(\mathcal{T})$. This can be shown by the refinement algorithm [Renz, 1999] which uses a *refinement matrix* that manages the different refinements and which initially contains the refinement strategy for every relation $S \in \mathcal{S}$ to a relation $T \in \mathcal{T}$.

Definition 2 (refinement matrix)
A refinement matrix M *of $\mathcal{S}$ has $|\mathcal{S}| \times 2^{|\mathcal{B}|}$ Boolean entries such that for $S \in \mathcal{S}$, $R \in 2^{\mathcal{B}}$, $M[S][R] = true$ only if $R \subseteq S$.*

The algorithm CHECK-REFINEMENTS (see Figure 1) takes as input a set of relations $\mathcal{S}$ and a refinement matrix M of $\mathcal{S}$. This algorithm computes all possible a-closed triples of relations R_{12}, R_{23}, R_{13} of $\mathcal{S}$ (line 4) and enforces a-closure to every refinement $R'_{12}, R'_{23}, R'_{13}$ for which $M[R_{ij}][R'_{ij}] = true$ for all $i, j \in \{1, 2, 3\}, i < j$ (lines 6-9). If one of these refinements results in the empty relation, the algorithm returns `fail`. Otherwise, the resulting relations $R''_{12}, R''_{23}, R''_{13}$ are added to M by setting $M[R_{ij}][R''_{ij}] = true$ for all $i, j \in \{1, 2, 3\}, i < j$ (lines 10-13). This is repeated until M has reached a fixpoint, i.e., enforcing a-closure on any possible refinement does not result in new relations anymore. If no inconsistency is detected in this process, the algorithm returns `succeed`. A similar algorithm, GET-REFINEMENTS, returns the revised refinement matrix M' if CHECK-REFINEMENTS returns `succeed`. If CHECK-REFINEMENTS returns `succeed`, we have pre-computed all possible refinements of every a-closed triple of variables as given in the refinement matrix M'. Thus, applying these refinements to an a-closed set of constraints can never result in an inconsistency when enforcing a-closure.

So if a suitable refinement strategy can be found, then CHECK-REFINEMENTS can be used to immediately verify that reasoning over a given set of relations is tractable.

Corollary 3 ([Renz, 1999]) *Let $\mathcal{S}, \mathcal{T} \subseteq 2^{\mathcal{B}}$ be two sets such that a-closure decides $\mathsf{CSPSAT}(\mathcal{T})$, and let M be a refinement matrix of $\mathcal{S}$. Suppose* GET-REFINEMENTS$(\mathcal{S}, M)$ *returns M'. If for every $S \in \mathcal{S}$ there is a $T_S \in \mathcal{T}$ with $M'[S][T_S] = true$, then a-closure decides $\mathsf{CSPSAT}(\mathcal{S})$.*

3 A general procedure for identifying tractable subsets

In this section we present a procedure for automatically identifying large tractable subsets of a spatial or temporal calculus. We assume that (1) we are given a set of base relations $\mathcal{B}$, that (2) the weak compositions and the converses of the base relations are known, and also that (3) a-closure decides consistency for $\mathsf{CSPSAT}(\mathcal{B})$. A consequence of condition (3) is that the closure $\widehat{\mathcal{S}}$ of a set $\mathcal{S} \subseteq 2^{\mathcal{B}}$ under intersection, converse and weak composition has the same complexity as $\mathcal{S}$ [Renz and Ligozat, 2005]. In particular, it follows that $\widehat{\mathcal{B}}$ is tractable. We further need the following lemma:

Lemma 4 *Let $\mathcal{S} \subseteq 2^{\mathcal{B}}$. If a-closure decides $\mathsf{CSPSAT}(\mathcal{S})$, then a-closure also decides $\mathsf{CSPSAT}(\widehat{\mathcal{S}})$.*

Proof Sketch. Each constraint xRy of Θ over $\widehat{\mathcal{S}}$ is replaced with an equivalent set of constraints Θ_{xy} over $\mathcal{S}$. Θ' over $\mathcal{S}$ is the union of all Θ_{xy} and is equivalent to Θ. Applying a-closure to Θ_{xy} leads to xRy, hence applying a-closure to Θ' leads to the same result as applying it to Θ. ∎

Note that condition (3) is not actually a strict condition. Our procedure can also be applied if this is not yet known. However, the results of our procedure, i.e., tractability of the resulting sets, is only guaranteed if condition (3) is satisfied.

3.1 Identifying tractable relations

Our procedure consists of three different steps which we will describe separately. The first step is to identify single tractable relations, i.e., relations which give a tractable subset of $2^{\mathcal{B}}$ when combined with the base relations. During this step we will also obtain relations which are potentially NP-hard, i.e., single relations that might lead to an NP-hard set of relations when combined with the base relations.

For this we will test every single relation which is not contained in $\widehat{\mathcal{B}}$ and see whether it can be refined to relations of $\widehat{\mathcal{B}}$. For every relation $R \notin \widehat{\mathcal{B}}$, we first compute the closure of $\mathcal{B} \cup \{R\}$ and apply the refinement algorithm to every possible refinement of R which is contained in $\widehat{\mathcal{B}}$. The algorithm is given in Figure 2 and consists of two parts, CHECK-SINGLE-REFINEMENT (C-S-R) which checks all refinements for a single relation R that is added to an existing set of relations $\mathcal{T}$, and FIND-TRACTABLE-SINGLES (F-T-S) which calls C-S-R for all relations not contained in $\widehat{\mathcal{B}}$. Even though the closure of R and $\widehat{\mathcal{B}}$ will likely contain more relations, we only initialise the refinement matrix in C-S-R with a refinement for R. This refinement will propagate to the other relations as they can all be decomposed into relations of $\widehat{\mathcal{B}} \cup \{R\}$.

Algorithm C-S-R has three different outcomes for a relation R and a set $\mathcal{T}$, `succeed`, `fail`, and `unknown`. `succeed` means that there is an initial refinement of R to $R' \in \mathcal{T}$ such that CHECK-REFINEMENTS computes an updated refinement matrix by which all relations of the closure of $\mathcal{T} \cup \{R\}$ can be refined to a relation of $\mathcal{T}$ without changing consistency. If this is the case, and a-closure decides CSPSAT($\mathcal{T}$), then a-closure also decides consistency for the closure of $\mathcal{T} \cup \{R\}$. This is an immediate consequence of Corollary 3. If all possible refinements of R to relations of $\mathcal{T}$ fail, C-S-R returns `fail`. In this case the closure of $\mathcal{T} \cup \{R\}$ cannot be refined to $\mathcal{T}$ and all attempts to refine R lead to an inconsistency. This does not say anything about whether adding R to $\mathcal{T}$ leads to an NP-hard set or not. But it means that tractability for this set cannot be proven using the refinement method and that there is some likelihood that the resulting set is NP-hard. A third possible outcome of C-S-R is `unknown`, which means that there is a refinement of R to $R' \in \mathcal{T}$ which does not result in an inconsistency, but which also does not lead to the refinement of all relations of the closure of $\mathcal{T} \cup \{R\}$ to relations of $\mathcal{T}$. Increasing $\mathcal{T}$ by some relations might lead to a successful refinement of R, so we cannot yet conclude if R leads to a tractable set or not.

F-T-S calls C-S-R for the different relations not contained in $\widehat{\mathcal{B}}$ while keeping track of their outcome. If a relation R returns `succeed` and therefore leads to a tractable subset, then all relations in the closure of $\mathcal{B} \cup \{R\}$ also lead to a tractable subset and don't have to be tested using C-S-R. We conclude this step with the following lemma.

Lemma 5 *Let $\mathcal{T}$ and $\mathcal{H}$ be the result of* F-T-S*($\mathcal{B}$). If a-closure decides* CSPSAT*($\mathcal{B}$), then it also decides* CSPSAT*($\mathcal{B} \cup \{T\}$) for all $T \in \mathcal{T}$.*

3.2 Identifying candidates for tractable subsets

Using the algorithm specified in the previous section, we can identify tractable relations, but we don't know yet how they can be combined to form large tractable subsets. It could be that the whole set $\mathcal{T}$ forms a tractable subset, but it could also be that some of the relations in $\mathcal{T}$ lead to NP-hardness when combined with each other. One possibility to find large tractable subsets would be to test all subsets of $\mathcal{T}$ using the refinement method, but there are obviously too many such sets and we do not know which refinement strategy might work. Our next step will therefore be to find out which relations might lead to intractability when combined with each other. For this purpose we assume that any of the relations of the set $\mathcal{H}$ returned by F-T-S($\mathcal{B}$) leads to intractability when combined with $\mathcal{B}$. Under this assumption, it is clear that any pair of relations of $\mathcal{T}$ that contains a relation of $\mathcal{H}$ in its closure cannot be contained in the same tractable subset and is therefore incompatible. This is done in lines 2-3 of FIND-TRACTABILITY-CANDIDATES (F-T-C) in Figure 3.

F-T-C computes all candidates for tractable subsets starting from the closure of the base relations and successively adding new tractable relations. Each new relation is first tested if it is incompatible with any of the relations already contained in the candidate (line 9). If it is not incompatible, then the closure with the current candidate is computed and

Algorithm: CHECK-SINGLE-REFINEMENT (C-S-R)
Input: A single relation $R \in 2^{\mathcal{B}}$, a tractable set $\mathcal{T} \subset 2^{\mathcal{B}}$ for which $\mathcal{T} \equiv$ closure($\{T\}$) and a refinement matrix M.
Output: `succeed` if the closure of $\{R\} \cup \mathcal{T}$ can be refined to $\mathcal{T}$, `fail` if any refinement of R to $R' \in \mathcal{T}$ leads to an inconsistency, and `unknown` otherwise.

1. $\mathcal{S} \leftarrow$ closure($\{R\} \cup \mathcal{T}$); good $\leftarrow$ *false*
2. *for every* refinement R' of R with $R' \in \mathcal{T}$ *do*
3. $\quad newM \leftarrow M$; $newM[R][R'] \leftarrow true$
4. $\quad$ *if* CHECK-REFINEMENTS($newM, \mathcal{S}$) = `succeed`
5. $\quad\quad$ *then* $M' \leftarrow$ GET-REFINEMENTS($newM, \mathcal{S}$)
6. $\quad\quad$ *if* $\forall i \in \mathcal{S}$ there is a $j \in \mathcal{T}$ s.t. $M'[i][j] = true$
7. $\quad\quad\quad$ *then return* `succeed`
8. $\quad\quad$ *else* good $\leftarrow true$
9. *if* good = $true$ *return* `unknown` *else return* `fail`

Algorithm: FIND-TRACTABLE-SINGLES (F-T-S)
Input: A set of base relations $\mathcal{B}$
Output: A set $\mathcal{T} \subseteq 2^{\mathcal{B}}$ of tractable relations and a set $\mathcal{H} \subseteq 2^{\mathcal{B}}$ of potentially NP-hard relations.

1. $\mathcal{T} \leftarrow$ closure($\mathcal{B}$); $\mathcal{H} = \emptyset$
2. $\forall i, j \in 2^{\mathcal{B}}$: $M[i][j] \leftarrow true$ if $i = j$ and *false* if $i \neq j$
3. *for every* relation $R \notin \mathcal{T}$ *do*
4. $\quad$ result $\leftarrow$ C-S-R(R,$\widehat{\mathcal{B}}$, M)
5. $\quad$ *if* result = `succeed` *then* $\mathcal{T} \leftarrow \mathcal{T} \cup$ closure($\{R\} \cup \mathcal{B}$)
6. $\quad$ *if* result = `fail` *then* $\mathcal{H} \leftarrow (\mathcal{H} \cup \{R\})$
7. *return* $\mathcal{T}$ and $\mathcal{H}$

Figure 2: Algorithms CHECK-SINGLE-REFINEMENT and FIND-TRACTABLE-SINGLES

tested whether the resulting set can be refined to the current candidate using the C-S-R algorithm (line 14). If it doesn't return fail, then the new relation is added to the current candidate. The algorithm keeps track of all relations that cannot be added to the current candidate (lines 10,13,16). If any of these relations is not yet contained in one of the candidates, then a new candidate must be generated which does contain this relation (line 20). The new candidate will be tested in the next while loop of the algorithm. If all tractable relations are contained in at least one candidate, then it is still possible that there are more candidates, namely, if there is a pair of tractable relations which is not incompatible and which is not yet contained together in any of the candidates. The algorithm tests each pair of relations which is not yet contained in a candidate set (line 22,23) and tests whether the closure of both these relations with the base relations can be refined to the closure of one of the relations with the base relations (line 26). If this is not possible and C-S-R returns `fail`, then this pair is added to the list of incompatible pairs (line 27). If a pair is found which can be refined, then a new candidate is formed which contains the pair (line 28) and tested in the next loop of the algorithm. If all pairs are tested, then the algorithm returns all candidates for tractable subsets. All these candidates have in common that they do not contain any incompatible pair, they do not contain a potentially NP-hard relation, and single refinements can be made without resulting in an inconsistency. What we do not know yet, is whether the whole candidate can be refined to a known tractable set.

Algorithm: FIND-TRACTABILITY-CANDIDATES (F-T-C)
Input: A set of tractable relations $\mathcal{T}$ and a set of supposedly NP-hard relations $\mathcal{H}$. (Note: cl = closure)
Output: Candidates for tractable subsets $\mathcal{CS}_i \subseteq 2^{\mathcal{B}}$

1. $\mathcal{L} \leftarrow \emptyset$ (set of incompatible pairs); n $\leftarrow$ 0; max $\leftarrow$ 1
2. *for all* $T_a, T_b \in \mathcal{T}$ *do*
3. *if* cl($\{\{T_a\}, \{T_b\}\} \cup \mathcal{B}$) $\cap \mathcal{H} \neq \emptyset$ *then* $\mathcal{L} \leftarrow \mathcal{L} \cup \langle T_a, T_b \rangle$
4. $\forall r, s \in 2^{\mathcal{B}}$: $M[r][s] \leftarrow true$ if $r = s$ and *false* if $r \neq s$
5. cand $\leftarrow \widehat{\mathcal{B}}$; newcand $\leftarrow \widehat{\mathcal{B}}$
6. *while* ($true$) *do*
7. nextrels $\leftarrow \emptyset$; newrel $\leftarrow \emptyset$; n++
8. *for all* $T_j \in \mathcal{T}$ with $T_j \notin$ cand *do*
9. *if* there is an $R \in$ cand such that $\langle T_j, R \rangle \in \mathcal{L}$
10. *then* nextrels $\leftarrow$ nextrels $\cup \{T_j\}$; continue;
11. newcand $\leftarrow$ cl(cand $\cup \{T_j\}$)
12. *if* newcand $\cap \mathcal{H} \neq \emptyset$ *then*
13. nextrels $\leftarrow$ nextrels $\cup \{T_j\}$; continue;
14. *if* C-S-R(T_j, cand, M) $\neq$ `fail` *then*
15. cand $\leftarrow$ newcand
16. *else* nextrels $\leftarrow$ nextrels $\cup \{T_j\}$;
17. $\mathcal{CS}_n \leftarrow$ cand
18. newrel is any relation in (nextrels $\setminus \bigcup_i \mathcal{CS}_i$)
19. *if* newrel $\neq \emptyset$ *then*
20. cand $\leftarrow$ cl($\mathcal{B} \cup$ newrel); newcand $\leftarrow$ cand; max++
21. *while* max = n *do*
22. $T_k, T_j \in \mathcal{T}$ is a pair s.t. $\langle T_k, T_j \rangle \notin \mathcal{L}$
23. and $\{T_k, T_j\}$ is not contained in any $\mathcal{CS}_i$
24. *if* there is no such pair *then return* all $\mathcal{CS}_i$
25. cand $\leftarrow$ cl($\mathcal{B} \cup \{T_j\}$); newcand $\leftarrow$ cl(cand $\cup \{T_k\}$)
26. *if* C-S-R(T_k, cand, M) = `fail` *then*
27. $\mathcal{L} \leftarrow \mathcal{L} \cup \langle T_k, T_j \rangle$; continue
28. cand $\leftarrow$ newcand; max++

Figure 3: Algorithm FIND-TRACTABILITY-CANDIDATES

3.3 Testing the candidates

In this step we will prove tractability of the candidates we found in the previous section or identify tractable subsets of the candidates in case they are not tractable. We know that CSPSAT($\widehat{\mathcal{B}}$) is tractable and that a candidate set is tractable if it can be reduced by refinement to CSPSAT($\widehat{\mathcal{B}}$). But which refinement strategy can we use? We can extend tractability of CSPSAT($\widehat{\mathcal{B}}$) to a larger set by adding one of the tractable relations and computing its closure. For the next relation to be added, it is sufficient to refine it to the larger tractable subset we obtained in the step before. If a new relation cannot be refined, i.e., if C-S-R returns `fail` we know that we cannot add it to the current tractable subset. If C-S-R returns `unknown`, we will try to add a different relation instead and test this relation again at a later stage.

This is very similar to what has been done in the algorithm F-T-C except for the special treatment of the relations for which the interleaved C-S-R calls return `unknown`. If none of these calls (lines 14 and 26 in F-T-C) returned `unknown` for any of the relations that have been added to a candidate, then this candidate must be tractable and no further processing is required. If `unknown` occured while computing a candidate set, we have to use algorithm TEST-CANDIDATES of Figure 4 which takes a candidate and successively tests for

Algorithm: TEST-CANDIDATES
Input: Candidates for tractable subsets $\mathcal{CS}_i$ of $2^{\mathcal{B}}$
Output: Tractable subsets $\mathcal{TS}_i$ of $2^{\mathcal{B}}$

1. *for all* $\mathcal{CS}_i$ *do*
2. queue $\leftarrow \emptyset$;
3. $\forall r, s \in 2^{\mathcal{B}}$: $M[r][s] \leftarrow true$ if $r = s$ and *false* if $r \neq s$
4. *for each* $R \in \mathcal{CS}_i \setminus \widehat{\mathcal{B}}$ *do* queue $\leftarrow$ queue $\cup \langle R, 0 \rangle$;
5. tractable $\leftarrow \widehat{\mathcal{B}}$;
6. loop $\leftarrow$ 0; changes $\leftarrow$ *false*;
7. *while* queue $\neq \emptyset$ *do*
8. take and delete the first pair $\langle R$,num$\rangle$ from queue;
9. *if* (num $>$ loop and changes = *false*) *then break*;
10. *if* (num $>$ loop) *then* loop $\leftarrow$ num; changes $\leftarrow$ *false*;
11. *if* C-S-R(R, tractable, M) = `succeed` *then*
12. tractable $\leftarrow$ closure(tractable $\cup \{R\}$);
13. delete all $\langle S, n \rangle$ from queue where $S \in$ tractable;
14. changes $\leftarrow$ $true$;
15. *else* add $\langle R$,num+1$\rangle$ to the end of the queue;
16. $\mathcal{TS}_i \leftarrow$ tractable;
17. *return* $\mathcal{TS}_i$ for all i;

Figure 4: Algorithm TEST-CANDIDATES

each relation contained in the candidate whether a refinement to the already known tractable subset is possible. If this is not possible for a relation, the same relation will be tested again at a later stage. In order to prove that this recursive computation indeed proves tractability, we introduce a new notion.

Definition 6 *Given a set* $\mathcal{S} \subseteq 2^{\mathcal{B}}$. *The* closure sequence $\mathcal{C}(\mathcal{S}; R_1, \ldots, R_n)$ *is a recursively defined subset of* $2^{\mathcal{B}}$*:*

1. $\mathcal{C}(\mathcal{S}; R) =$ *closure*$(\mathcal{S} \cup \{R\})$
2. $\mathcal{C}(\mathcal{S}; R_1, .., R_n) =$ *closure*$(\mathcal{C}(\mathcal{S}; R_1, .., R_{n-1}) \cup \{R_n\})$

Lemma 7 *Let* $\mathcal{T} \subseteq 2^{\mathcal{B}}$. *Suppose* $\mathcal{T} \equiv \mathcal{C}(\mathcal{B}; R_1, \ldots, R_n)$ *and* M *is a refinement matrix where only the diagonal is* true. *If* C-S-R$(R_i, \mathcal{C}(\mathcal{B}; R_1, \ldots, R_{i-1}), M)$ *returns* `succeed` *for all* $1 \leq i \leq n$*, then* CSPSAT($\mathcal{T}$) *is tractable.*

Proof. If C-S-R$(R_i, \mathcal{C}(\mathcal{B}; R_1, \ldots, R_{i-1}), M)$ returns `succeed`, then $\mathcal{S}_i = \mathcal{C}(\mathcal{B}; R_1, \ldots, R_i)$ can be reduced by refinement to $\mathcal{S}_{i-1} = \mathcal{C}(\mathcal{B}; R_1, \ldots, R_{i-1})$ and, hence, CSPSAT($\mathcal{S}_i$) is tractable if CSPSAT($\mathcal{S}_{i-1}$) is tractable. Since we know that CSPSAT($\widehat{\mathcal{B}}$) can be decided by a-closure, tractability of CSPSAT($\mathcal{T}$) follows by successively applying C-S-R to the corresponding closure sequence. ■

The whole procedure is then a sequence of the three steps (1) F-T-S, (2) F-T-C, and (3) TEST-CANDIDATES. The input is a set of base relations together with its weak compositions, and the output is one or more tractable subsets. As a consequence of the previous lemmata, we obtain the main results.

Theorem 8 *Given a set of base relations* $\mathcal{B}$ *such that a-closure decides* CSPSAT($\mathcal{B}$)*. For each* $\mathcal{TS}_i \subseteq 2^{\mathcal{B}}$ *returned by our procedure,* CSPSAT($\mathcal{TS}_i$) *is tractable.*

Instead of using $\mathcal{B}$ as an input to our procedure, we can use any other set for which a-closure decides consistency.

Theorem 9 *If a-closure decides* CSPSAT($\mathcal{S}$)*, then any set obtained by applying our procedure to* $\mathcal{S}$ *can be reduced by refinement to* $\mathcal{S}$ *and is therefore tractable.*

3.4 Applying the new procedure

We implemented the procedure in C and used several optimisations such as identifying relations that have the same closure, or more space efficient data structures such as refinement arrays instead of refinement matrices [Renz, 1999]. As a general heuristic for determining the order in which relations are processed we used the size of the closure. We applied the procedure to different spatial and temporal calculi with known maximal tractable subsets in order to test how close the outcome of our procedure is to the actual result.

For RCC5 [Randell *et al.*, 1992], the known maximal tractable subset was found instantly. For RCC8, F-T-S resulted in 76 relations for $\mathcal{H}$ and 179 relations for $\mathcal{T}$. Remarkably, the relations of $\mathcal{H}$ correspond exactly to those relations that are known to be NP-hard [Renz, 1999]. For some relations C-S-R returned `unknown`, but all these relations were added to $\mathcal{T}$ as part of computing the closure of other relations for which C-S-R returned `succeed`. The second step, F-T-C returned three candidates for tractable subsets which actually correspond to the three maximal tractable subsets of RCC8 identified in [Renz, 1999]. Tractability of all three candidates could be shown using TEST-CANDIDATES. The whole procedure ran in less than ten minutes on a Pentium-Duo 1.83 GHz CPU with 512MB RAM.

For the Interval Algebra, F-T-S($\mathcal{B}$) resulted in only 174 relations for $\mathcal{T}$ and a few relations for which C-S-R returned `unknown`. If we add any one of the unknown relations to $\mathcal{B}$ and use it as input to our procedure, then we obtain the known maximal tractable subset ORD-Horn [Nebel and Bürckert, 1995], but this does not prove tractability of ORD-Horn. However, we also obtain ORD-Horn if we use the pointisable subset $\mathcal{P}$ as input to our procedure. Tractability of $\mathcal{P}$ follows immediately from tractability of the Point Algebra and a-closure decides CSPSAT($\mathcal{P}$) [Ladkin and Maddux, 1988], so this does give us a proof of tractability of ORD-Horn. This shows us that ORD-Horn cannot be reduced by refinement to the base relations but only to a subset of $\mathcal{P}$. In all cases, our procedure terminated in less than one hour, which is quite remarkable given the size of the Interval Algebra. This could be because the Interval Algebra seems to have a fairly simple structure: 4885 relations give $2^{\mathcal{B}}$ when closed with the base relations, 60 relations give ORD-Horn, and for all 8192 relations there are only 248 different closures.

4 Discussion & Conclusions

We developed a general procedure which automatically identifies large tractable subsets of a spatial or temporal calculus given only the base relations $\mathcal{B}$ and their weak compositions. The only requirement is that a-closure decides consistency for $\mathcal{B}$. We tested our procedure on RCC8 and the Interval Algebra and obtained all the known maximal tractable subsets in less than one hour. While the resulting sets are tractable, we cannot guarantee maximality in the general case as only sets are found which can be reduced by refinement to the input set. As seen in the case of the Interval Algebra, it can be better to identify tractable subsets of smaller calculi first and then use the results for determining input sets for larger calculi. In any case, the resulting tractable subsets should be enough to give a considerable speed up when solving instances of the NP-hard consistency problem. If it is unknown whether a-closure decides consistency for $\mathcal{B}$, our procedure can still be applied and the resulting sets can still be used for obtaining considerably more efficient reasoning algorithms. If it turns out that the reasoning algorithms give a wrong result for at least one instance, then a-closure does not decide consistency for $\mathcal{B}$.

An interesting consequence of our results is that for all calculi which have the same weak compositions we obtain the same resulting sets, independent of their domain. Further analysis of our procedure is necessary in order to get a clear understanding of the effect of the relations for which C-S-R returns `unknown`. By applying our procedure to other calculi we hope to be able to identify general criteria or heuristics of when relations can be reduced by refinement and when this is not possible. This might give us a measure of how close the resulting sets are to the actual maximal tractable subsets.

One point worth discussing are the consequences of having such a procedure which automatically makes a theoretical analysis of a problem and automatically identifies efficient algorithms. It is certainly very desirable for people working in applications of spatial and temporal information. They only have to run our procedure and don't have to wait for years for some experts to make an analysis. On the other hand what happens to all the experts for doing a theoretical analysis, are they not needed anymore? Will it not be worth a publication anymore to identify tractable subsets analytically if they can just as well be identified using our procedure? Even with our procedure, it is still necessary to show that a-closure decides consistency for the base relations. Showing this is a very hard problem as it is necessary to relate the relations to their semantics and to their domains. A heuristic for how to show this is given in [Renz and Ligozat, 2005]. Also, our procedure does not guarantee maximality. So an accompanying theoretical analysis is still very useful and can give insight into why a problem is hard.

References

[Allen, 1983] J. F. Allen. Maintaining knowledge about temporal intervals. *Comm. ACM*, 26(11):832–843, Nov 1983.

[Ladkin and Maddux, 1988] P. Ladkin and R. Maddux. On binary constraint networks. Technical Report, Kestrel Institute, 1988.

[Ligozat and Renz, 2004] G. Ligozat and J. Renz. What is a qualitative calculus? a general framework. In *PRICAI'04*, 53–64, 2004.

[Nebel and Bürckert, 1995] B. Nebel and H-J. Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *J.ACM*, 42(1):43–66, Jan 1995.

[Nebel, 1997] B. Nebel. Solving hard qualitative temporal reasoning problems: Evaluating the efficiency of using the ORD-Horn class. *CONSTRAINTS*, 3(1):175–190, 1997.

[Randell *et al.*, 1992] D. Randell, Z. Cui, and A. Cohn. A spatial logic based on regions and connection. In *KR'92*, 165–176, 1992.

[Renz and Ligozat, 2005] J. Renz and G. Ligozat. Weak composition for qualitative spatial and temporal reasoning. In *Proc. CP'05*, 534–548, 2005.

[Renz and Nebel, 2001] J. Renz and B. Nebel. Efficient methods for qualitative spatial reasoning. *JAIR*, 15:289–318, 2001.

[Renz, 1999] J. Renz. Maximal tractable fragments of the Region Connection Calculus: A complete analysis. In *Proc. IJCAI'99*, 448–454, 1999.