

Qualitative
Constraint
Calculi
Application & Integration

Workshop at KI 2006

Bremen
June 14, 2006

Stefan Wöfl Till Mossakowski (Eds.)

Qualitative Constraint Calculi

Application and Integration

Workshop at KI 2006

Bremen, Germany, June 14, 2006

Workshop Proceedings

Preface

The term *qualitative constraint calculus* comprises logical formalisms and algorithmic methods used in the domain of *Qualitative Representation and Reasoning*, a steadily growing and vital sub-domain of current AI research. In the past 25 years, since P. J. Hayes' work "Naive physics 1: ontology for liquids" (1978) and J. Allen's work "Maintaining knowledge about temporal intervals" (1983), researchers have intensely studied representations of real-world phenomena by describing features of the world in purely qualitative terms. This is because these formalisms aim at describing the common-sense background knowledge on which our human perspective on the physical reality is based. In fact, although qualitative calculi employ concepts from a wide range of mathematical theories (geometrical notions such as lines, half-planes, and angles, topological terms such as interior, boundary, or connectedness, concepts of size and shape, etc.), qualitative representation formalisms usually are built on vocabularies that are close to expressions in natural languages. This entails that qualitative representation formalisms abstract from metrical aspects of the physical reality and, moreover, from (maybe "over"-) sophisticated concept formations used in mathematics or physics.

From a computational point of view qualitative representation formalism are of particular interest: the fundamental idea underlying qualitative constraint calculi is to restrict the vocabulary of rich mathematical theories about space and time in such a way that diversified aspects of space and time can be treated within distinguished fragments with simple qualitative (in the sense of *non-metrical*) languages, but with good computational properties. For that reason a large number of calculi for efficient reasoning about spatial and temporal entities have been proposed in the literature. To mention just a few of them, the so-called point algebra and Allen's interval algebra describe possible relations between instants or intervals in linear flows of time. Further temporal calculi have been discussed for more general model classes such as tree-like or partial order representations of time, and despite these one-sorted calculi, also many-sorted calculi have been proposed (e. g., Vilain's point-interval calculus). Examples of spatial calculi include mereotopological calculi such as the region connection calculus proposed by D. A. Randell, Z. Cui, and A. G. Cohn, its variants proposed and discussed by B. Bennett, A. Isli, and I. Duentzsch, and extensions investigated by A. Gerevini, J. Renz, and B. Nebel, then M. J. Egenhofer, R. D. Franzosa's 4- and 9-intersection calculi, furthermore Frank's cardinal direction calculus and its variants by S. Skiadopoulos and M. Koubarakis, C. Freksa's double cross calculus, the dipol calculus and its variants by R. Moratz and F. Dylla, G. Ligozat's flip-flop calculus, and many more.

An important application field for qualitative constraint calculi is the domain of human-machine interaction. That is because artificial agents interacting with humans in nondeterministic and uncertain environments must be able to process qualitative spatial and temporal information communicated by humans and thus represented in various conceptual schemata, on different levels of granularity, and within varying reference systems. But applications are, of course, not limited to that domain, since qualitative calculi may also play an interesting rôle in geographic information systems as well as in robotics.

When we started to organize this workshop, we identified a series of problem areas concerning the application and integration of qualitative constraint calculi, which, as we think, should be addressed at the workshop as well as in future research in the field: *Algebraic methods for integrating qualitative calculi*. Researchers in the domain of qualitative reasoning have developed a series of qualitative calculi, each of them dealing with a rather restricted aspect of the world. But many of these calculi are closely related to each other: some are simple extensions of others, some show similarities on the syntactic level, some have related classes of intended models, i. e., they are based on the same background theory. With regard to possible application fields, it seems imperative to provide a unified framework that allows for translating descriptions given in terms of a specific qualitative calculus into terms of another one. This framework should also enable artificial agents to reason with such qualitative descriptions. We think that this problem could be tackled by algebraic, model-theoretical, and category-theoretical methods.

Combinations of qualitative calculi. Information communicated by human agents usually mix concepts dealt with in different calculi. From a logical point of view, a representation of these “mixed” descriptions can be obtained by considering combinations of qualitative calculi. In particular, the analysis of different combinations of topological and geometrical calculi seems a main issue in this area. From an algorithmic point of view, it seems necessary to develop methods for solving constraint networks of combined calculi by using established algorithmic techniques for the component calculi.

Qualitative calculi and ontology. From an ontological point of view, the qualitative representation formalisms may be considered miniature ontologies in the sense of purpose-driven formalizations of human background knowledge. Here the question arises how these qualitative calculi can be integrated into upper ontologies such as DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering, developed at LOA, Trento and Rome). This topic seems an important objective, since ontologies have recently gained a prominent rôle for the mediation, alignment, and negotiation of spatial information in multi-agent systems.

Qualitative calculi in action. We were interested in workshop contributions dealing with interesting case studies in which qualitative calculi have been used for solving particular problems. A main issue here is to identify typical problems that need to be overcome in order to use qualitative calculi in real-world applications. In view of possible applications, we think that the qualitative reasoning domain will also benefit from the development of generic constraint solvers for qualitative calculi (inclusively performance analyses) and from a discussion of interesting benchmark problems that could be used to evaluate such constraint solvers.

The papers now comprised in this workshop proceedings contribute to (at least partially) answer the following research questions: What is a qualitative constraint calculus, and what is it good for? What has been the motivation for many people to study such calculi during the past 25 years? What is the theoretical (in this context: algebraic, logical) basis of these calculi? What are the specific techniques and what are the computational methods used in this domain? What are the applications, where these techniques have been implemented and have proved valuable? What are the tools that can be used by researchers outside the particular community to develop own applications upon established qualitative calculi?

Aspects of qualitative world modeling are addressed in two contributions: S. Schiffer, A. Ferrein, and G. Lakemeyer report on current work in which a qualitative world representation has been successfully used to implement tactics known from human soccer theory. Using qualitative representations leads to significant reduction of the space of possible game situations that need to be considered in soccer games. M. Ragni presents a qualitative calculus that is applicable for representing dynamic aspects of traffic scenarios (such as car movements or the evolution of traffic congestions).

Turning to more foundational aspects, G. Ligozat will address in his talk the relationship of algebraic and categorial concepts by investigating weak representations of binary non-associative relation algebra, which are the basis for many constraint calculi discussed in the literature. Moreover he will address the problem of characterizing so-called tractable subclasses of such relation algebras in a general way. The topic of B. Bennett's talk will be foundational issues in spatial logic as well as some future directions for research in the domain of qualitative reasoning. In continuation of G. Ligozat's work, T. Mossakowski, L. Schröder, and S. Wölfl present a precise representation of binary qualitative constraint calculi from a categorial perspective.

Three contributions discuss algorithmic techniques for qualitative constraint calculi. First, A. Scivos presents techniques that allow for reducing the computational complexity of constraint satisfaction problems, when additional information on the structure of possible solutions can be used. Then J.-F. Condotta, D. D'Almeida, C. Lecoutre, and L. Saïs report on work about how local consistency concepts can be translated between qualitative constraint networks and discrete constraint networks. Finally, J. Renz presents new techniques how large tractable subclasses of binary relation algebras can be computed in an (almost) automatic way.

Last but not least, F. Dylla, L. Frommberger, J. O. Wallgrün, and D. Wolter present a toolbox that integrates a specification language for qualitative constraint calculi as well as a constraint solver that implements some standard techniques discussed in the qualitative reasoning domain.

This workshop was organized to provide a forum for researchers from different subfields of AI research (including Qualitative Reasoning, Logic, Constraint Solving, Computational Linguistics, and Ontologies) to discuss the challenges and open problems within qualitative constraint based reasoning. We would like to thank the authors of the contributions to this workshop as well as the members of the Program Committee, who helped in the review process and provided valuable comments and suggestions to the authors (and the editors). We also owe thanks to the organizers of the KI 2006 conference. Furthermore, we acknowledge the financial support by the Transregional Collaborative Research Center SFB/TR 8 *Spatial Cognition*. Special thanks go to Micha Altmeyer who helped compiling the proceedings of the workshop.

Finally, we hope for stimulating discussions of interesting results at the workshop.

Organization

Organizers

Till Mossakowski (University of Bremen, Germany)
Stefan Wöfl (University of Freiburg, Germany)

Invited Speakers

Brandon Bennett (University of Leeds, UK)
G rard Ligozat (LIMSI-CNRS, Paris, France)

Program Committee

John A. Bateman (University of Bremen, Germany)
Max J. Egenhofer (University of Maine, Orono, USA)
Gerhard Lakemeyer (Aachen University of Technology, Germany)
Reinhard Moratz (University of Bremen, Germany)
Jochen Renz (Australian National University, Australia)

Sponsoring Institutions

Transregional Collaborative Research Center SFB/TR 8 *Spatial Cognition*
funded by *Deutsche Forschungsgemeinschaft (DFG)*

Table of Contents

Invited Talks

- Some Mathematical Aspects of Qualitative Spatial and Temporal Reasoning 1
G erard Ligozat

Qualitative Modeling

- Qualitative World Models for Soccer Robots 3
Stefan Schiffer, Alexander Ferrein, and Gerhard Lakemeyer
- Reasoning in Dynamic Environments: Temporalized Intervals 15
Marco Ragni

Foundations of Qualitative Constraint Calculi

- A Categorical Perspective on Qualitative Constraint Calculi 28
Till Mossakowski, Lutz Schr oder, and Stefan W olfl

Algorithmic Techniques for Qualitative Constraint Calculi

- Structural Approaches for PTIME Solvability of CSPs in Qualitative Reasoning . 40
Alexander Scivos
- From Qualitative to Discrete Constraint Networks 54
*Jean-Fran ois Condotta, Dominique D’Almeida, Christophe Lecoutre,
and Lakhdar Sa s*
- Qualitative Spatial and Temporal Reasoning: Efficient Algorithms for Everyone . 65
Jochen Renz

Tools for Qualitative Constraint Solving

- SparQ: A Toolbox for Qualitative Spatial Representation and Reasoning 79
*Frank Dylla, Lutz Frommberger, Jan Oliver Wallgr un, and Diedrich
Wolter*

Author Index 91

Some Mathematical Aspects of Qualitative Spatial and Temporal Reasoning

Abstract

G rard Ligozat

LIMSI-CNRS, Paris-Sud University
ligozat@limsi.fr

Qualitative Calculi at Large

An impressive variety of qualitative temporal and spatial reasoning calculi have been born since Allen's first introduction of the interval calculus in 1983. To cite only a few, we can mention G sge's two-dimensional rectangle calculus, Ligozat's generalized interval calculi, the RCC-8 and RCC-5 calculi, the cardinal direction calculus, Freksa's double-cross calculus, Mitra's star calculus, Pujari and Sattar's INDU calculus, and several others. Although those calculi are superficially very similar, it has recently been discovered that their finer properties may differ in a significant way. The INDU calculus is a remarkable case in point: for instance, many of the basic properties connecting the classical notions of consistency for constraint networks, which are pleasant properties of Allen's calculus, are no longer true for the INDU calculus. Hence many methods which relied on them become useless. Even on the algebraic side, the corresponding algebra fails to be associative, hence is not a relation algebra in Tarki's sense, as Allen's algebra notoriously is. This kind of realization led to reconsidering the basic facts about qualitative calculi and asking obvious questions: what properties can be safely assumed for any qualitative calculus? In what circumstances do more specific properties appear? How does the presence or lack of particular properties affect the standard approaches to the study of the calculi? Trying to answer such questions in a general context has led us to develop a framework covering all particular cases, including those whose properties differ significantly from those of Allen's calculus.

Besides the interest in defining calculi, a sizeable body of work in the domain of qualitative spatial and temporal reasoning has been devoted to the study of their complexity, especially in connection with the basic problem of deciding whether a given constraint network is consistent. Since for most of the calculi this problem is not tractable, the question of identifying tractable subclasses has come into prominent focus.

This talk will present two directions in which mathematical tools have proved valuable as generic tools for the study of qualitative reasoning formalisms.

Algebra and Category Theory: Weak Representations

The first topic is the use of the tools of universal algebra for a generic and abstract definition of qualitative calculi. The central idea in this respect is the fact that most

of the central notions in qualitative calculi can be expressed in terms of the category of “weak representations” of particular kinds of algebras (of which Tarski’s relation algebras are special cases) called non-associative algebras. We get a common algebraic framework which is able to describe all existing qualitative calculi. The language of the theory of categories provides a clean way of describing a calculus and the basic concepts associated with it, such as various types of consistency. Moreover, it gives a good handle on the study of the representations of the algebra, hence on the properties of the corresponding logical theories.

Geometry and Topology: Conceptual Spaces and Tractability

The second topic is the search for tractable subclasses in the algebras associated to the calculi. Following the initial work on the so-called convex relations in Allen’s algebra, we describe the use of geometric and topological concepts for the characterization of tractable subclasses in a wide range of calculi. A central role is played by various notions of pre-convexity, a property which is connected to the geometry of the set of basic relations in a calculus. The topic is closely related to the notions of neighborhoods of relations as expressed in terms of “conceptual neighborhoods”. It can also be considered as an example of the use of “conceptual spaces” in the sense of Gärdenfors.

References

- [1] G. Ligozat. Categorical methods in qualitative reasoning: The case for weak representations. In A. G. Cohn and D. M. Mark, editors, *Proceedings of COSIT-05, Ellicottville, New York, 2005*, LNCS 3693, pages 265-282. Springer, 2005.
- [2] J. Renz and G. Ligozat. Weak composition for qualitative spatial and temporal reasoning. In *Proceedings of Constraint Programming 2005 (CP 2005), Sitges, Spain*, LNCS 3709, pages 534-548. Springer, 2005.
- [3] G. Ligozat and J.-F. Condotta. On the relevance of conceptual spaces for spatial and temporal reasoning. *Spatial Cognition and Computation*, 5(1):1-27, 2005.
- [4] G. Ligozat, D. Mitra, and J.-F. Condotta. Spatial and temporal reasoning: Beyond Allen’s calculus. *AI Communications*, 17(4):223-233, 2004.
- [5] G. Ligozat and J. Renz. What is a qualitative calculus? A general framework. *Proceedings of PRICAI-04, Auckland, New Zealand*, LNCS 3157, pages 53-64. Springer, 2004.
- [6] G. Ligozat and J. Renz. Problems with local consistency for Qualitative Calculi. In *Proceedings of ECAI-2004, Valencia, Spain*, pages 1047-1048, 2004.

Qualitative World Models for Soccer Robots

Stefan Schiffer, Alexander Ferrein, and Gerhard Lakemeyer

Knowledge-Based Systems Group, Computer Science Department,
RWTH Aachen University, Aachen, Germany
{schiffer, ferrein, gerhard}@cs.rwth-aachen.de

Abstract. Until now world models in robotic soccer have been mainly quantitative in nature, consisting of fine-grained (numerical) estimates of player positions, ball trajectories, and the like. In contrast, the concepts used in human soccer are largely qualitative. Moving to qualitative world models also for robots has the advantage that it drastically reduces the space of possible game situations that need to be considered and, provided the concepts correspond to those in human soccer theory, it eases the task of agent specification for the designer. In this paper we propose qualitative representations using ideas from spatial cognition and employing Voronoi diagrams. We also discuss how reasoning with these representations is achieved within our underlying agent programming framework.

1 Introduction

Until now world models in robotic soccer have been mainly quantitative in nature, consisting of fine-grained (numerical) estimates of player positions, ball trajectories, and the like. In contrast, the concepts used in human soccer are largely qualitative. Moving to qualitative world models also for robots has the advantage that it drastically reduces the space of possible game situations that need to be considered. Provided the concepts correspond to those in (human) soccer theory, it also eases the task of agent specification for the designer.

For example, if we use abstract to positional information like *front-left*, many similar game situations are represented by the same qualitative values whereas all these situations would differ in terms of their numerical values. This is useful, for example, when we formulate the preconditions required to initiate a tactical move. With a qualitative description we cover multiple similar settings, making the specification applicable in many circumstances. We also ease the specification process since we are able to use terms that are much closer to the natural language descriptions commonly used in human soccer theory. Besides, in the majority of cases a tactical instruction just cannot be formulated with precise positions but instead always refers to a set of positions denoted by a qualitative abstraction of regions such as *front* or *left*.

Dylla et al. [9] were among the first to address the question of how insights from human soccer theory can be applied when specifying the behavior of soccer robots. Their proposal is to analyze existing moves from human soccer theory as, for example, described in [17] and to adapt these moves to the abilities of the respective robotic soccer leagues. They identified requirements needed to adapt existing moves to a soccer robot team one of which, they state, is that the robots in the team have to build a qualitative world model. The reason for this is that human soccer knowledge, which is often

represented in the form of diagrams, has an inherent qualitative nature. To encode the moves which are most often depicted only in a prototypical fashion a qualitative world representation is needed to formalize behaviors for cooperative team play.

There already exists work on using qualitative information for autonomous agents. For example, Stolzenburg et al. [21] compared methods how to intercept the ball in the soccer simulator; one of the methods used qualitative abstracted ball coordinates. In [11], Fraser et al. describe the *inReach* predicate for the ball distance while employing a hysteresis function, Stone et al. [22] apply Reinforcement Learning in the Simulation League using some handpicked qualitative predicates for state space abstraction. While in these approaches only some qualitative aspects focusing on a particular task are chosen, none of these fulfill all requirements that are necessary for a complete qualitative framework.

In this paper we follow the ideas by Dylla et al. [9]. Continuing the work on using human soccer knowledge for robotic soccer, we present qualitative enhancements to a world model particularly suited for robots in the MIDDLE SIZE LEAGUE, where up to five robots per team play on a $8m \times 12m$ indoor soccer field. However, the enhancements could be applied to any other ROBOCUPSoccer league just as well. Drawing on previous work in the area of spatial cognition, we present an approach to representing positions on the field qualitatively. We also discuss models for further information needed to transfer human knowledge on soccer such as to decide when a pass can be played to a team-mate. This is done in the context of the logic-based action language READYLOG, a variant of GOLOG [16], which we use for an exemplary specification of the soccer move “kick-off”.

When using a qualitatively abstracted world model, an important issue is drawing appropriate inferences. A simple example is to derive the result of *distance_near* + *distance_far*. One option would be to apply one of the existing qualitative spatial calculi (cf. [4] for a survey of existing calculi). However, this currently does not seem feasible not only for computational reasons. Instead, we propose a hybrid quantitative-qualitative representation of the respective world model information, which allows for a limited form of reasoning about qualitative world model predicates, that seems expressive enough for most soccer applications. To be able to do so we have methods for re-quantifying the qualitative values to their numerical counterparts.

In the next section we define our qualitative world model. In section 3 we briefly introduce the language READYLOG and we show how it can be used to specify abstract soccer moves. In section 4, we present an example illustrating our approach to reasoning about the predicates in our qualitatively enhanced world model. Section 5 concludes the paper.

2 Qualitative Representations

In the following we present the models which we use to abstract the quantitative data gathered from the sensors of the robots and stored in the quantitative world model to a qualitative representation of the world.

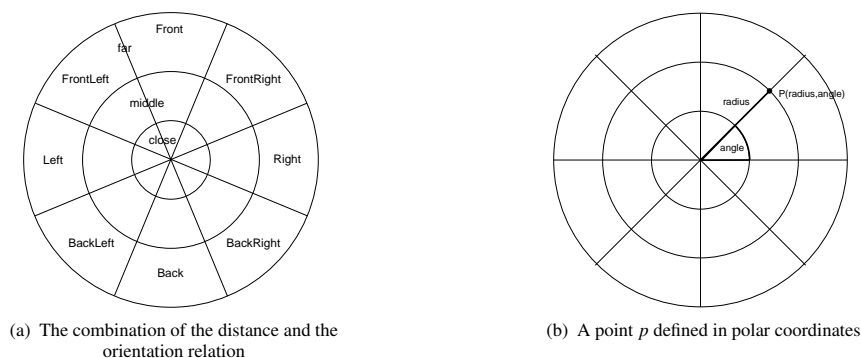


Fig. 1: The combination of distance and orientation relations compared to the polar coordinate system.

2.1 Positional Information

In [3], Clementini, Felici, and Hernandez present a unified framework which allows for qualitative representation of positional information. The framework combines an orientation and a distance relation. The position of a *primary object* is represented by a pair of distance and orientation relations with respect to a *reference object*. Both relations depend on a so-called *frame of reference* which accounts for several factors like the size of objects and different points of view.

The framework also features basic reasoning capabilities such as the composition of spatial relations as well as switching between different frames of reference. Unfortunately, the reasoning features provided are not guaranteed to yield unique¹ results, for instance, for the composition of qualitative terms. However, as for the ROBOCUP soccer domain we depend on unambiguous outcomes of such compositions since they are needed to instruct the robot.

From a quantitative point of view, the combined description of a position with this model can be seen as the representation of a point in polar coordinates. A point p in polar coordinates is defined by the distance r from the origin to this point and the angle φ measured from the horizontal x -axis to the line from the origin to p in the counter-clockwise direction. Thus, the position of a point p is described as (r, φ) . This description directly corresponds to the combination of the distance and the orientation relation. We illustrate this in Figure 1. This correspondence is of particular interest concerning our hybrid approach to reasoning which we will discuss in detail in section 4.

The number of subdivisions, that is the level of granularity within the qualitative description of both distance and orientation can be chosen freely. In this paper we restrict ourselves to one level with eight distinctions although it is possible and might be of benefit to have multiple levels. For a recent approach to qualitative orientation with adjustable granularity see [19].

¹ By *unique* we mean results which contain exactly one relation.

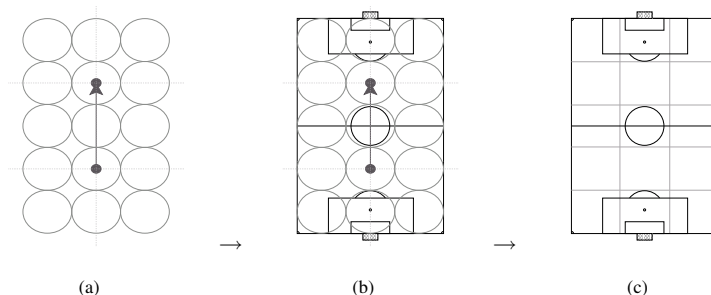


Fig. 2: Semantic regions on the playing field. Figure (a) shows the orientation grid taken from [12]. Figure (b) shows the grid embedded into a soccer field. The resulting semantic regions on the playing field are shown in Figure (c).

Beside the relative positional information described so far positions are also used in a global frame of reference. A qualitative concept which is applied frequently is that of semantic regions on the playing field. These regions are often used as tactical positions corresponding to player roles. To model semantic regions in the sense of global positioning we employ a well-known approach to qualitative representation of positional information proposed by Freksa and Zimmermann in [12, 13]. Qualitative orientation information in two-dimensional space is given by the relation between a vector and a point. The vector consists of a start point A and an end point B . It represents the orientation of a possible movement. Now, imagine a line through A and B and two further lines, one orthogonally going through A and B each. These three lines form an *orientation grid* which has the form of a *double-cross*. Different positions of an additional third point C can then be described with respect to this grid. Altogether the grid leads to 15 different orientation relations.

The grid used for the representation of orientational alignment is based on a (movement) vector. Unfortunately, we do not have an explicit movement in the context of global positioning on the playing field. On the other hand, we can regard the direction of play as a vector. From a tactical point of view one of the main objectives in a game is to advance from a defensive situation in the team's own half to an offensive one in the opponent's half. Thus, we can take the center of each team's half as the start and end point of an imaginary vector. If we place this vector onto the playing field we can relate semantic positions on the field to the orientation relations provided by Freksa's grid. This yields a subdivision of the field into 15 regions. The regions and their derivation are depicted in Figure 2.

Dividing the grid by the field's horizontal and vertical axes results in *zones* for the length of the field and in *sides* for the width. This system roughly corresponds to the Cartesian coordinate system; zones and sides form perpendicular axes with the zones corresponding to the x -axis and the sides corresponding to the y -axis. Thus, we can still specify an object's position in a coordinate system like manner, but by using zones and sides we achieve a qualitative abstraction. The analogy to the Cartesian coordinate system is of importance to our hybrid approach to reasoning again.

In the following we are going to elaborate on possible applications of the qualitative approaches to positional information we just presented.

2.2 Derived Predicates

Besides the greater comfort within the specification of tactical patterns we can use the zone and side information to build further qualitative predicates. Consider, for instance, the course of a soccer game. The overall positioning on the pitch is, besides the ball possession, one of the fundamental indicators upon which to classify whether a team is currently in a defensive or an offensive situation. Moreover, it is an important information whether the game's focus is on one of the sides of the playing field or in the center.

With the zone information we can provide a predicate which we call *game setting*. It expresses where on the pitch the main activity of play is. It can be used to derive positioning instructions or simply to call appropriate sub-procedures in an agent's program. We compute the game setting by a weighted sum of the zone indices of all players and of the ball: $w_{team} \cdot \sum_{i \in team} zone(i) + w_{opp} \cdot \sum_{j \in opps} zone(j) + w_{ball} \cdot zone(ball)$. The center of the possible values lies at a value of *zero* which states that the focus of play is located in the midfield. To be able to perform a classification of the game's focal point in terms of a situation being *offensive*, *balanced*, or *defensive* we need to establish a threshold. Upon this threshold we can decide to which of the above class the current situation belongs. Feasible values for this threshold as well as for the weights for each of the three different objects within the above formula were found empirically in real world experiments. They can, however, also be learned which could lead to more adequate results. Analogously to the game setting we can determine the gist of play with respect to the sides of the pitch. We call this the *game edge*. The game edge renders useful, for instance, if we need to decide which side of the pitch is less occupied and can thus be used to advance into the opponent's field half with a lower risk of being attacked.

2.3 Reachability

Apart from the static semantic regions there are further aspects which can be useful to determine. In particular, we are interested in dynamic spatial properties such as *reachability* of different kinds, which is central for the description and the execution of tactical patterns in soccer. For a discussion of the different forms of reachability we refer to [9]. For all these different reachability relations the individual abilities of a single robot or agent are relevant since even within the same league players of different teams may have unequal capabilities in terms of speed and mobility. Nevertheless, there are some general properties which hold for all players despite their physical abilities.

We consider the concept of Voronoi diagrams to be applicable for modeling a simplified version of the reachability relations required. A Voronoi diagram $V(S)$ of a set S of n point sites is the partitioning of a plane with n points into n convex polygons such that each polygon contains exactly one point and every point in the given polygon is closer to its central point than any other. For a more detailed account on Voronoi diagrams and their dual, the Delaunay triangulation $DT(S)$, see e.g. [1].

We make use of Voronoi diagrams and their dual the Delaunay triangulation to model reachability as they separate the field into non-intersecting regions and we get a connection graph between the players. We take the players as point sites in the plane and construct $V(S)$ and $DT(S)$ with the Euclidean distance thereupon. Then, the Voronoi region of each player is the set of points closer to this player than to any other player. Furthermore, in the dual of the Voronoi diagram $V(S)$, the Delaunay triangulation $DT(S)$, two players' point sites are connected if and only if they share a common boundary in the Voronoi diagram. Our idea of modeling reachability with the aid of $V(S)$ and $DT(S)$ is to consider players to be reachable to each other if their Voronoi regions share a common boundary, that is, they are connected in the Delaunay triangulation $DT(S)$. Fig. 3 depicts the Delaunay triangulation and the Voronoi regions for the geometric structure of several players on a soccer playing field. The yellow lines represent the triangulation, the green and red shaded regions correspond to the Voronoi regions of the attacking team and the defending team, respectively.

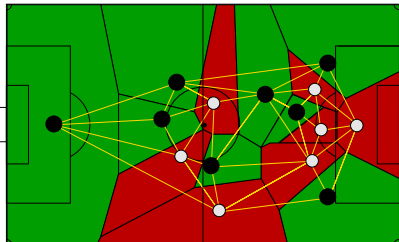


Fig. 3: Voronoi diagram and Delaunay triangulation of a soccer situation

2.4 Free Space

The notion of *free space* is another important aspect which can frequently be found in the description of spatial settings and tactical patterns in soccer. The term *free space* denotes an area which is not occupied by any of the players of the opposing team. Consider the Voronoi diagram being constructed upon all opponent players on the field. That is, each opponent player corresponds to the center of a Voronoi region. The Voronoi edges are formed by points that are equally far away from the two players the edge is between. The Voronoi vertices reflect points in the plane which have the maximal possible distance to even three or more of the surrounding players. These vertices directly match our interpretation of positions in free space as we sketched it above. Please note that we include the four corner points of the playing field into the Voronoi diagram. We need at least three Voronoi regions to obtain a Voronoi vertex. Including the corner points yields a minimum of four regions thus guaranteeing that we do not have less than one Voronoi vertex.

We provide two distinct ways to employ these vertices. First, we consider a classification request. Given a point on the playing field, we can ask how 'free' this point is. To answer such a request we compute the point's distance to the nearest point site in the opponent's Voronoi diagram as well as its distance to the nearest Voronoi vertex. The ratio between these two values is a good criterion on how 'free' the given point is because it reflects whether the point is closer to a free position or to an opponent player. Since the ratio does not reflect the absolute distance to an opponent we additionally take a minimal distance into account which has to be exceeded. Second, we can answer inquiries for free point positions. Most times it is reasonable to specify a region of interest in which to search for a free position. For simplicity we assume that we specify a region of interest by a position in the pitch's coordinate system (along with a maximal

distance to search up to). Given this query, we determine the nearest Voronoi vertex to the position. If the distance between the query point and the vertex is lower than the maximal distance we return the vertex' position. Otherwise, we return the position of a point lying on a line starting at the query point going into the direction of the nearest Voronoi vertex.

2.5 Additional Concepts

Within the course of a soccer game it is a vital piece of information whether or not one's team is in possession of the ball. That is, again, an information we can provide by utilizing the structure of Voronoi diagrams. The simplest way to answer the question of ball possession is to check if the ball is located in the Voronoi region of a player who belongs to one's own team. This is, of course, not always correct. For example, if the player whose Voronoi cell the ball belongs to is not facing the ball it might be the case that another player who has a greater distance to the ball but who is directly facing it can reach it more quickly. It is, however, possible to take this additional information into account and to refine the predicate accordingly.

As an additional qualitative predicate of particular interest in the soccer context we now consider something we call *passway vacancy*. We denote a qualitatively abstracted classification of the amount of space available along a potential pass way by this predicate. That is to say, we classify the degree of exposure of a line segment going from point P_{start} to point P_{end} by examining possible points of interception. We derive our classification by considering a ratio on how likely an interception is. Consider a straight line from P_{start} to P_{end} . We compute the minimal distance of each opposing player to this line, that is either the length of a line perpendicular to the pass way or the distance to the pass way's nearest end point. Further, we compute the distance from each opponent to the starting point of the pass way. Then, we calculate the ratio between this two values. That is to say, we determine if the opponent is so close to the pass way that it can intercept a ball passed along the pass way.

3 Using Readylog For Behavior Specifications

READYLOG [10], a variant of GOLOG, is based on Reiter's Situation Calculus [20, 18], a second-order language for reasoning about actions and their effects. Changes in the world are only due to actions so that a situation is completely described by the history of actions starting in some initial situation. Properties of the world are described by *fluents*, which are situation-dependent predicates and functions. For each fluent the user defines a successor state axiom specifying precisely which value the fluent takes on after performing an action. These, together with precondition axioms for each action, axioms for the initial situation, foundational and unique names axioms, form a so-called *basic action theory* [20].

GOLOG has emerged as an expressive language in recent years. It has imperative control constructs such as loops (*while*), conditionals [16] (*if...then*), and (recursive) procedures (*proc(name(parameters), body)*), but also less standard constructs like the nondeterministic choice of actions ($\langle \rangle$). Extensions exist for dealing with continuous

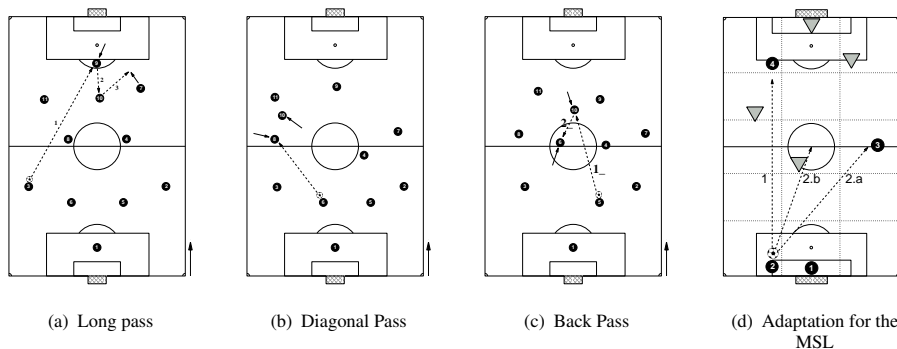


Fig. 4: Example for the “build-up play” move.

change [15] and concurrency [6], allowing for exogenous and sensing actions [5] and probabilistic projections into the future [14]. Another extension provides the facility to do decision-theoretic planning [2] which involves Markov Decision Processes (MDPs) ($solve(p, h)$, where p is a GOLOG program, h is the MDP’s solution horizon). READYLOG integrates these and more features in one agent programming framework [10].

To encode the behavior one has to specify a domain axiomatization including the actions the robot can perform together with their effects, and the fluents which describe the properties of the world like the ball position. Examples of domain descriptions for the soccer domain can be found in [7, 8].

3.1 A Soccer Move Example

We now specify the soccer move “kick-off” in READYLOG and we show that our qualitative world model (READYWORLD) supports the specification. We adapted three possible ways to build up a play as discussed in [17].

The first way to build up play is with a long pass (Fig. 4(a)). We immediately notice that the term *long* is one of the coarse, qualitative notions we need to establish in order to adapt human soccer theory for our autonomous soccer agents. We could also formulate this as passing the ball from a *back* position to a *front* position on the playing field. The second way to build up play is with a diagonal pass as depicted in Fig. 4(b). This time, the term *diagonal* is of qualitative nature. Diagonal means passing to the side being opposite to the current one. Fig. 4(c) shows the last possibility to build up play which is with a *deep* pass (dashed line labeled with 1) followed by a subsequent *back* pass (dashed line labeled with 2). The term *deep* is used to denote the space *behind* or *in between* a group of opponent players. The endpoint of such a pass has to be the most *free* position available *in between* or *behind* the group of opponents.

We now try to adapt as much of these descriptions as possible by integrating their most essential parts into one pattern. All three possibilities have in common that the ball is located in the *back* part on the pitch. According to a role ontology it is a player currently having a defensive role which is about to initiate the pattern to build up play.

```

proc build_up_play_defender ,
if haveBall(ownNumber) then
  getFreeSide(offense,FreeSide); getPassPartner(offense,FreeSide,PassPartner);
  solve (
    if  $\neg$ isKickable(ownNumber) then
      interceptBall
    else if isPassReachable(ownNumber,PassPartner) then
      passTo(ownNumber,PassPartner)
    endif
    endif
    | pickBest(bestSide , {leftSide, middleSide, rightSide}
      dribbleTo(ownNumber,middleZone,bestSide)
      | kickTo(ownNumber,middleZone,bestSide) ) /* end of pickBest */
    | interceptBall ; kickTo( ownNumber, middleZone, middleSide),
  3, func.Reward ) /* end solve with horizon 3 */
else
  interceptBall
endif
endproc

```

Fig. 5: The build-up play program for the defender.

We already characterized the possibility of a long pass as 'bringing' the ball to the *front* part of the pitch. Therefore, in this case the agent chooses to pass to a teammate who is located in the attacking zone. For the two other possibilities the pass' destination is the midfield. The agent can either make a diagonal pass, that is the case if the target position is on the *opposite* side of the field, or it can simply pass to a free area on the same side or in the pitch's center. To illustrate our adaptation of the build up play patterns for the MIDDLE SIZE LEAGUE we depicted a diagram similar to the ones in [17] in Figure 4(d).

Figure 5 shows a program in our action language READYLOG capturing the above example. Note that this specification contains several qualitative elements such as *middleZone*, *leftSide*, and *offense* as well as qualitative predicates such as *isPassReachable*. With our qualitative world model we are able to simply transfer the qualitative notions from the specification in [17]. Moreover, the use of qualitative terms and predicates makes the program applicable in many game situations.

4 Reasoning

When a robot executes a READYLOG program like the one in Figure 5, it needs to evaluate different courses of action and choose the most appropriate one. This happens every time a *solve*-construct is encountered. Roughly, the robot evaluates the different alternatives (nondeterministic actions separated by '|' and nondeterministic choice of arguments (*pickBest*)) and chooses the one that maximizes expected utility in a decision-theoretic fashion (see [2, 10] for details). The evaluation of one alternative involves projecting the effects of the actions it contains into the future, starting from the current (qualitative) world model.

For the purposes of this paper, the main question is how moving-actions involving qualitative spatial terms are projected. For example, suppose the robot is currently in

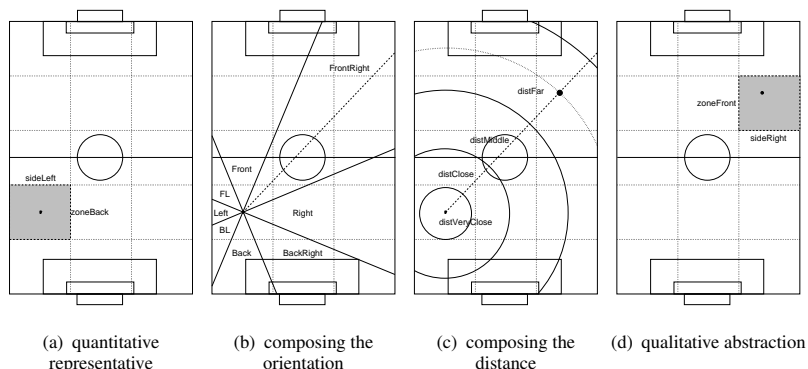


Fig. 6: An exemplary projection of qualitative values in READYLOG.

region $[zoneBack, sideLeft]$ (Figure 6(a)). What should the effect of the action $go(front-right, far)$ be? Depending on the exact position of the robot, it could end up in any of 9 zones on the field.² While this may still be manageable, things quickly get out of hand when we want to project a sequence of actions. In the worst case, the eventual outcome would span all 15 zones, that is, we lose all information about where the robot might end up. Moreover, projecting all possibilities is too costly, as the robot needs to make a decision on what to do next in the order of less than a second.

To avoid these complications and to remain computationally efficient, we simplify the problem in the following drastic way: to project the path of the robot, we simply translate the qualitative information back to numeric (geometric) values, taking as representative the mid-points of zones, sides, directions, and distances. The composition can be computed straight-forwardly using Euclidean geometry. The numerical end-result is then converted back to a qualitative description in the same way we perform the qualitative abstraction in the first place. Figure 6 illustrates this for the action $go(front-right, far)$ starting in region $[zoneBack, sideLeft]$. There are several advantages to this. Rather than having to entertain all possibilities, we only need to compute one. Using mid-points also reduces the error in a reasonable way for practical purposes. Perhaps most importantly, we can use the existing projection mechanism of READYLOG, which requires that the effects of atomic actions are deterministic (in the case of a goto-action, the effect must be a unique location).³ The method is clearly sound, as the result is among those which a purely qualitative reasoner would obtain. It is not complete, as there may be cases where the computed path would take the robot outside the field and hence render the action illegal, even though other legal solutions may exist. Being incomplete is not a big problem in this application. It is more important to obtain a reasonable approximation fast in most circumstances. Also, plans often do not survive very long in soccer, that is, they are often aborted because the world has changed in a

² See also [3] for a discussion of ambiguities in their approach to qualitative representations of positional information.

³ Note that this is different from nondeterministic actions, which are complex, that is, made up of a number of primitive actions.

way that makes the current plan invalid. Hence it is not worth spending too much effort on figuring out what to do. Finally, we remark that, once a course of action has been chosen, we employ the same method that we use during reasoning to compute actual robot trajectories, using the real robot position as the starting location.

5 Conclusions

In this paper we proposed a qualitative spatial world model for soccer playing robots, combining earlier work on semantic regions with that on orientation and distance relations. In addition, we used Voronoi diagrams to provide us with a notion of reachability, which is important in the soccer domain. Computing robot trajectories from ambiguous qualitative descriptions was achieved by mapping the qualitative terms to unique geometric representatives.

The current reasoning scheme is admittedly somewhat ad hoc and was chosen mainly for efficiency reasons. One refinement would be to consider more than one trajectory. Also a comparison with existing spatial calculi [4] is needed. On the practical side, while the above ideas are fully implemented, we need to carry out more experiments to see how the qualitative approach fares in real games.

Acknowledgment

This work was supported by the German National Science Foundation (DFG) in the Priority Program 1125, *Cooperating Teams of Mobile Robots in Dynamic Environments* and a grant by the NRW Ministry of Education and Research (MSWF).

References

- [1] F. Aurenhammer and R. Klein. Voronoi diagrams. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*. Elsevier Science Publishers, 2000.
- [2] C. Boutilier, R. Reiter, M. Soutchanski, and S. Thrun. Decision-theoretic, high-level agent progr. in the situation calculus. In *Proc. AAAI'00*. AAAI Press, 2000.
- [3] E. Clementini, P. D. Felice, and D. Hernandez. Qualitative representation of positional information. *Artificial Intelligence*, 95(2):317–356, 1997.
- [4] A. G. Cohn and S. M. Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29, 2001.
- [5] G. De Giacomo and H. Levesque. An incremental interpreter for high-level programs with sensing. In H. J. Levesque and F. Pirri, editors, *Logical Foundation for Cognitive Agents: Contributions in Honor of Ray Reiter*, pages 86–102. Springer, Berlin, 1999.
- [6] G. De Giacomo, Y. Lésperance, and H. J. Levesque. ConGolog, A concurrent programming language based on situation calculus. *Artificial Intelligence*, 121(1–2):109–169, 2000.
- [7] F. Dylla, A. Ferrein, and G. Lakemeyer. Acting and deliberating using golog in robotic soccer – a hybrid architecture. In *Proc. CogRob02*. AAAI Press, 2002.
- [8] F. Dylla, A. Ferrein, and G. Lakemeyer. Specifying multirobot coordination in ICPGolog – from simulation towards real robots. In *AOS-4 at IJCAI-03*, 2003.

- [9] F. Dylla, A. Ferrein, G. Lakemeyer, J. Murray, O. Obst, T. Röfer, F. Stolzenburg, U. Visser, and T. Wagner. Towards a league-independent qualitative soccer theory for RoboCup. In *RoboCup 2004: Robot World Cup VIII*. Springer, 2005.
- [10] A. Ferrein, C. Fritz, and G. Lakemeyer. On-line Decision-Theoretic Golog for Unpredictable Domains. In *Proc. of 27th German Conference on Artificial Intelligence*, pages 322–336. Springer, 2004.
- [11] G. Fraser, G. Steinbauer, and F. Wotawa. Application of qualitative reasoning to robotic soccer. In *18th Int. Workshop on Qualitative Reasoning*, 2004.
- [12] C. Freksa. Using orientation information for qualitative spatial reasoning. In A. U. Frank, I. Campari, and U. Formentini, editors, *Theories and methods of spatio-temporal reasoning in geographic space*. Springer, Berlin, 1992.
- [13] C. Freksa and K. Zimmermann. On the utilization of spatial structures for cognitively plausible and efficient reasoning. In *IEEE International Conference on Systems Man and Cybernetics*, pages 261–266, Chicago, 1992.
- [14] H. Grosskreutz. Probabilistic projection and belief update in the pgolog framework. In *Proc. 2nd Int. Cognitive Robotics Workshop*, pages 34–41. 2000.
- [15] H. Grosskreutz and G. Lakemeyer. cc-Golog – An action language with continuous change. *Logic Journal of the IGPL*, 2002.
- [16] H. J. Levesque, R. Reiter, Y. Lesperance, F. Lin, and R. B. Scherl. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1-3):59–83, 1997.
- [17] M. Lucchesi. *Coaching the 3-4-1-2 and 4-2-3-1*. Reedswain Publishing, 2001.
- [18] J. McCarthy. Situations, Actions and Causal Laws. Technical report, Stanford University, 1963.
- [19] R. Moratz, F. Dylla, and L. Frommberger. A relative orientation algebra with adjustable granularity. In *Proceedings of the Workshop on Agents in Real-Time and Dynamic Environments (IJCAI 05)*, 2005.
- [20] R. Reiter. On knowledge-based programming with sensing in the situation calculus. *ACM Transactions on Computational Logic (TOCL)*, 2(4):433–457, October 2001.
- [21] F. Stolzenburg, O. Obst, and J. Murray. Qualitative velocity and ball interception. In M. Jarke, J. Köhler, and G. Lakemeyer, editors, *Proc. KI-2002*, LNAI, pages 283–298, Berlin, Heidelberg, New York, 2002. Springer.
- [22] P. Stone, P. Riley, and M. Veloso. The CMUnited-99 champion simulator team. In Veloso, Pagello, and Kitano, editors, *RoboCup-99: Robot Soccer World Cup III*. Springer, 2000.

Reasoning in Dynamic Environments: Temporalized Intervals

Marco Ragni

Albert-Ludwigs-Universität Freiburg, Institut für Informatik
Georges-Köhler-Allee, 79110 Freiburg im Breisgau, Germany
ragni@informatik.uni-freiburg.de

Abstract. Allen’s interval algebra is one of the most prominent calculi in qualitative reasoning. Although that it has been developed for temporal reasoning and representation, it has been soon used also for spatial reasoning, especially in geographical information systems (GIS). Another prototypical example are traffic scenarios where cars can be represented by intervals on a road as the underlying line. Most important are calculi able to model dynamic configurations, i.e. temporalized qualitative spatial calculi. We use intervals to represent spatial objects and construct a spatio-temporal relation algebra. By adding some restrictions on the models, like continuity and size persistence constraints, the calculus becomes more adequate, although the satisfiability problem also looks a lot harder. However, we are able to show that the satisfiability problem is NP-complete.

1 Introduction

Allen’s interval algebra is one of the most prominent calculi in qualitative reasoning. Regardless the fact that it has been developed for temporal reasoning and representation, it has also been used for spatial reasoning [10] especially in geographical information systems (GIS) [8], and for directed intervals [16].

Qualitative Spatial Reasoning (QSR) abstracts from metrical details of the physical world and enables computers to make predictions about spatial relations, even when precise quantitative information is not available [4]. From a practical viewpoint QSR is an abstraction that summarizes similar quantitative states into one qualitative characterization. A complementary view from the cognitive perspective is that the qualitative method *compares* features within the object domain rather than by *measuring* them in terms of some artificial external scale [6]. This is the reason why qualitative descriptions are quite natural for humans.

Balbani and Condotta [2] have systematically investigated the computational complexity of the block calculus, and identified tractable subclasses. Time played no role in their investigations, so what they analyzed was the question, if a given set of constraints is satisfiable, i.e. if it has a scenario. We call such problems *static problems*.

Assume that a scenario, i.e. a qualitative information about positions of objects, is given. So what you may want to know is, if it is possible to transform under specified constraints this scenario in another scenario. A well-known example may be the Tower of Hanoi problem. Such problems are called *dynamic* problems. We will investigate in the following the computational complexity of such dynamic problems with respect

to intervals. In order to investigate this we need to *temporalize* spatial problems. Most important is that some natural additional constraints on the models are satisfied, e.g. the transformation has to be continuous, meaning that objects should change their positions without leaps.

What is the motivation for considering temporalized constraint formalisms? First for modelling real-world problems we need of course temporal aspects. For example, for applications ranging from logistics, planning, to robot navigation, and multi-agent systems such temporal aspects are vital. Other applications range from geographical information systems (GIS) to calculi for directed intervals [16]. The latter work has investigated the question, what happens, if we combine intervals and direction. With respect to more spatial domains, a theory of temporalized intervals may have interesting applications, for example, when the movement of cars on highways, from overhauling to accumulations of cars, have to be modeled. This can be done by identifying cars or accumulation of cars by intervals (cf. Figure 1). Furthermore, a “real-world” model should be, of course, able to differentiate between exceptional cars, e.g. a police car, which is in some states prohibited to overtake. This should also be expressible in such a language.

Another possible application may be in video controlled accumulation warning systems, where “blackouts” occur: A camera films a traffic scenario, then the transmission breaks down, and after a while the camera is restarted and you get another traffic scenario. The question is what has happened between the breakdown and before the restart of the transmission? Such questions can only be answered in temporalized models. This brief discussion already indicates how reasoning with temporalized interval relations could be used for reasoning about traffic networks. The calculus is additionally used in applications dealing with small-scale spaces like Web page design (Borning *et. al.*, 2000), as well as in applications with large-scale spaces like Geographic Information Systems [8]. We see that for both applications a temporalization has its use.

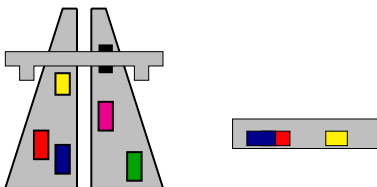


Fig. 1: Intervals in transportation networks

The calculus RCC-8 has been temporalized by a time logic PTL [18] and the interval algebra [7]. Here, we investigate such a temporalization of spatial structures which use intervals, and we aim at a theory sufficient for describing transportation networks. For this reason, we discuss the temporalization of algebras and ask which conditions the models should satisfy. We suggest two main axioms that should be satisfied by any adequate model. These axioms can be described as the *continuity constraint*, which states that any change of scenarios should be continuous and the *size/shape persistence*

constraint, which states that the size (and shape) of the objects should not be changed over time. This is done by using an interval algebra for representing spatial objects (following the idea of [10]) and another interval algebra for modelling temporal aspects. This temporalization of the interval algebra is a first and necessary step for temporalizing objects in two or more dimensions like the block calculus. Some preliminary work for temporalizing relation algebras has been done in [14] and [15]. In the first paper a so-called generalized neighborhood graph has been developed to represent continuous movements and for precisising the notion of consistency in temporalized relation algebras. In the second paper the cardinal direction calculus has been temporalized and connections to classical planning has been outlined.

The paper is organized as follows: In section 2 we review some basic concepts about the interval algebra, and we sketch some results concerning complexity and the neighborhood graph. In a next step, we introduce the notion of temporalization of relation algebras and ask which requirements models of such *dynamic* satisfiability problems should satisfy. In section 4 we work out the particularities of this new structure concerning complexity questions and search unsatisfiable substructures. In particular, we show that the satisfiability problem with respect to the continuity constraint of the models is NP-complete. Finally, section 5 summarizes the results of the paper and gives a short overview of some questions that are left open in this paper.

2 The Interval Algebra

Since the interval algebra [1] is certainly the most prominent relation algebra in AI, we will present here only its essentials. The interval algebra (\mathcal{IA}) has thirteen base relations between pairs of intervals. An interval X is represented as a tuple (x^-, x^+) of real numbers, with $x^- < x^+$, denoting the start and endpoint of the interval respectively, and relations between intervals are composed of disjunctions of base interval relations.

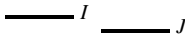
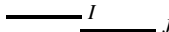





Symbol	Relation	Pictorial Representation
$<$ (conv.: $>$)	I before J	
m (mi)	I meets J	
o (oi)	I overlaps J	
d (di)	I during J	
s (si)	I starts J	
f (fi)	I finishes J	
=	I equals J	

Fig. 2: The thirteen base relations of the interval algebra

Given a spatial or temporal CSP θ , a fundamental reasoning problem¹ is deciding the satisfiability of θ . The satisfiability problem of \mathcal{IA} (short: IASAT) is NP-complete [17]. Assume that two intervals are related by one of the base relations presented in Figure 2. What happens if we move one of the intervals (in very small “steps”) towards the other? This question is usually answered by presenting a neighborhood graph. The neighborhood graph is also often understood as a similarity measure for the conceptual neighborhood of relations.

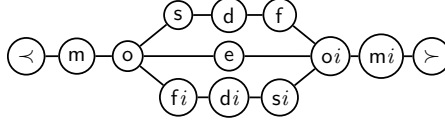


Fig. 3: The neighborhood graph of the interval algebra for intervals with fixed (unchangeable) length.

3 Temporalization of Relation Algebras

Anything which can be described so far by the interval algebra are scenarios, i.e. snapshots of a situation. To model *dynamic* environments, we need in addition the possibility to express time constraints. For this reason, we use another interval algebra for describing temporal relations between the scenarios. From a computational complexity point of view, the question of checking the consistency of one scenario is easier than finding and checking the consistency of a chain of scenarios.

In this section we introduce some technical notions about temporalized constraint formalisms. We adopt here the concepts introduced in [15].

Let V_T and V_S be disjoint sets of variables. A temporal (spatial) \mathcal{IA} constraint is a formula of the form $I R J (X R Y)$, where $I, J \in V_T$ ($X, Y \in V_S$) and $R = \{r_1, \dots, r_n\}$ is a subset of the set of all \mathcal{IA} base relations $\{\prec, m, o, s, f, d, =, \succ, mi, oi, si, fi, di\}$. An \mathcal{IA} constraint network is a finite set of \mathcal{IA} constraints. Let $V(C)$ be the set of variables occurring in a given constraint network C . An *assignment* for an \mathcal{IA} constraint network C is a function $\tau : V(C) \rightarrow \mathbb{R}^2$ that assigns to each variable Z that occurs in C a pair of real numbers $(\tau^1(Z), \tau^2(Z))$ such that $\tau^1(Z) < \tau^2(Z)$. The model relation w. r. t. an assignment τ for C is introduced as follows:

$$\begin{aligned} \tau \models I m J &\iff \tau^2(I) = \tau^1(J) \\ \tau \models I o J &\iff \tau^1(I) < \tau^1(J) < \tau^2(I) < \tau^2(J) \\ \tau \models I d J &\iff \tau^1(J) < \tau^1(I) < \tau^2(I) < \tau^2(J) \\ &\dots \quad (\text{cf. Fig. 2}) \end{aligned}$$

¹ Since all other problems in the CSP context can be reduced in polynomial time to satisfiability [9].

and

$$\tau \models I \{r_1, \dots, r_n\} J \iff \tau \models I r_i J \text{ for some } 1 \leq i \leq n.$$

Finally, an \mathcal{IA} constraint network, C , is satisfiable if there exists an \mathcal{IA} assignment that models all relations of C . In this case the assignment is said to be a *solution* of C . We differentiate between spatial interval variables, which are denoted in the following by X, Y, Z, \dots , and temporal interval variables denoted by I, J, K, \dots .

The temporalized interval calculus \mathcal{TIA} combines constraint formulae of Allen's interval calculus for the spatial and the temporal part. More precisely, we define \mathcal{TIA} constraint formulae as follows:

- Each temporal constraint IRJ is a \mathcal{TIA} constraint, i. e., IRJ is a \mathcal{TIA} constraint for each pair of interval variables $I, J \in V_T$ and each set R of Allen relations.
- For each interval variable I , each pair of spatial variables X and Y , and each set S of \mathcal{IA} relations, $I : x S y$ is a \mathcal{TIA} constraint.

A \mathcal{TIA} constraint network is a finite set of \mathcal{TIA} constraints. A standard interpretation of this constraint formalism is based on the following ingredients: we use the linear ordering of the reals for interpreting interval variables.

A typical example of a \mathcal{TIA} constraint network is the following:

$$I m J, I : X \{m, o\} Y, I : Y \{m\} Z, J : X \{<\} Y, J : Y \{m\} Z,$$

This network expresses that during time interval I , the spatial interval X meets the spatial interval Y , or X overlaps Y and Y meets the interval Z . During time interval J the spatial configuration is different. The spatial interval X is before the spatial interval Y , and so on.

Definition 1 (cp. Ragni and Wöflfl [15]). An *interpretation* of a \mathcal{TIA} constraint network C is an ordered pair $\langle \tau, \gamma \rangle$, where

- τ is an \mathcal{IA} assignment for $V(C) \cap V_T$.
- γ assigns to each instant $t \in \mathbb{R}$ an \mathcal{IA} assignment $\gamma_t : V(C) \cap V_S \rightarrow \mathbb{R}^2$.

Then the function $\hat{\gamma}_X : \mathbb{R} \rightarrow \mathbb{R}^2, t \mapsto \gamma_t(X)$ describes the movement of the object (interval) X . The point $\hat{\gamma}_X(t)$ is referred to as *the position* of X at time point t , and $\hat{\gamma}_X^1(t)$ and $\hat{\gamma}_X^2(t)$ refer to the start and endpoint of the interval, respectively, at instant t .

We then define the model relation as follows:

$$\begin{aligned} \langle \tau, \gamma \rangle \models IRJ &\iff \tau \models IRJ \\ \langle \tau, \gamma \rangle \models I : X S Y &\iff \gamma_t \models X S Y \text{ for each } \tau^1(I) < t < \tau^2(I). \end{aligned}$$

Note that we require that the spatial constraints hold in the interior of the time interval (cp. [15, 7]).

Definition 2. An interpretation $\langle \tau, \gamma \rangle$ for a \mathcal{TIA} constraint network C is said to be a *model* of C if $\langle \tau, \gamma \rangle \models \phi$ for each $\phi \in C$.

The core problem, then, is to find a sequence of scenarios (called *transition chain*), which satisfies a given *TLA* CSP. We can abbreviate the problem for every time interval by a triple $\langle \Sigma, \sigma_s, \sigma_f \rangle$, where σ_s is the start scenario, σ_f is the final scenario, and Σ contains the (spatial) constraints for the objects. A *transition chain* from σ_1 to σ_n is a sequence $\sigma_1, \dots, \sigma_n$ of scenarios for Σ , such that the changes from σ_i to σ_{i+1} satisfy the *size persistence constraint*, *continuity constraint*, and *sequential movement constraint* explained in the following. The central question here is: what are the restrictions we should impose on possible models (interpretations) of *TLA* constraint networks? In the remainder of this paper we will use two “reality”-principles that spatio-temporal models should satisfy and a third constraint, which is introduced for the sake of simplicity. First, we may want to restrict possible changes to those where the intervals do not change their size. This is meant by the *size persistence constraint*. We may also want to restrict possible changes to those that satisfy the principle of *continuous transformation*, i.e. the classical neighborhood graph should be satisfied. In other words, it should not be possible, for two intervals with, e.g. $I \prec J$ (before) to change suddenly into $I \circ J$ (overlaps) without going through $I m J$ (meets) inbetween. The last restriction on the class of models states that at any time only one object should change its position, i.e. there should not be simultaneous movements. This is introduced for controlling the transformations and for assuring the continuity constraint. All these constraints, however, cannot be expressed by *TLA* formulae, but by deduction rules only.

As a consequence of the size persistence constraint it is sometimes possible to deduce relative sizes of intervals (conceived of as spatial objects). For example, if the interval I is during the interval J ($I d J$), we know that the interval I must be smaller than the interval J . As well, if we know from a scenario or from some constraints that the interval I is smaller than an interval J , any solution which consists of at least one violating scenario cannot be a solution. Therefore we have to deduce from the constraints additional constraints on the relative size of intervals.

Since solutions of *TLA* constraint networks have to satisfy these three additional constraints, spatio-temporal problems seem to be much harder than the satisfiability problems of *LA*.

4 The Computational Complexity

Our aim is to prove the following main theorem:

Theorem 1. *TIASAT is NP-complete.*

The NP-hardness of **TIASAT** follows directly from the NP-hardness of **IASAT**.

Given a neighborhood graph G , we define the *neighborhood distance* between spatial relations as follows: For base relations B and B' , $\Delta_G(B, B')$ is defined as the length of the shortest path in G between B and B' . For arbitrary relations S and S' we set

$$\Delta_G(S, S') = \min_{B \in S, B' \in S'} \Delta_G(B, B').$$

Obviously, $\Delta_G(S, S') = 0$ if and only if S intersects with S' , and $\Delta_G(S, S') = 1$ if and only if S and S' are disjoint, but contain base relations B and B' respectively such that B' is a neighbor of B in G . The conceptual distance of two scenarios σ_i, σ_j is the sum of all conceptual distances for all relations in σ_i to σ_j .

4.1 Reasons for Insatisfiability

There are four reasons, why dynamic satisfiability problems of intervals are unsatisfiable: First, there is no path in the classical neighborhood graph for transforming one interval into the other (satisfying all constraints in Σ). Second, if there is a path, a moving interval I changing its position wrt. to an interval J violates its constraints to a third interval K . Third, the constraints induce a change of the sizes of an interval wrt. to another interval. Fourth, the interval is chained, i.e. it is not possible to move an interval because another interval is connected with it and therefore we would have to move both in one step. We provide for any of these problems an example.

We analyze now the difficulty to compute possible paths from an initial scenario to a final scenario. Here the conceptual distance plays an important role. For a given problem $\langle \Sigma, \sigma_s, \sigma_f \rangle$ we can calculate in polynomial time whether any interval in the scenario σ_s can change its actual position (represented by relations to any other interval) according to the neighborhood graph to its position in the scenario σ_f .

Example 1. Assume that the following two scenarios are given (we will use here I_1, I_2, \dots as spatial variables):

$$\begin{array}{l} \sigma_s: \quad \overline{I_1} \quad \overline{I_2} \quad \overline{I_3} \quad \overline{I_4} \quad \overline{I_5} \\ \sigma_f: \quad \overline{I_4} \quad \overline{I_1} \quad \overline{I_5} \quad \overline{I_2} \quad \overline{I_3} \end{array}$$

We assume that all intervals have the same size. As in the scenario σ_s $I_1 \prec I_4$, and in σ_f $I_1 \succ I_4$ hold, obviously

$$I_1 \{ \prec, m, o, =, oi, mi, \succ \} I_4 \in \Sigma.$$

In other words, if only one of these relations for I_1 and I_4 is not in Σ the problem is necessarily unsatisfiable.

Lemma 1. For two spatial intervals I and J with $\sigma_s: Ir_1J$ and $\sigma_f: Ir_2J$, we can check in constant time if the path of the neighborhood graph from r_1 to r_2 is contained in Σ or not.

Let us now investigate the relative size of intervals. We can deduce the relative sizes of some intervals from the constraints in Σ and the base relations which hold in the scenarios σ_s and σ_f . For example, if $I d J$ then we know immediately that the interval I must be smaller than the interval J . The same is true for the relations s, f . Such relative sizes can cause unsatisfiability as we see in the following example.

Example 2. Assume that the following scenarios are given:

$$\begin{array}{ll} \text{The scenario } \sigma_s: & \text{The scenario } \sigma_f: \\ \{J s I, K f I, J m K\} & \{J s K, I f K, J m I\} \end{array}$$



This is an unsatisfiable problem because of the size persistence constraint. Note that initially there are no such *size constraints* given in Σ . Such constraints must be added later.

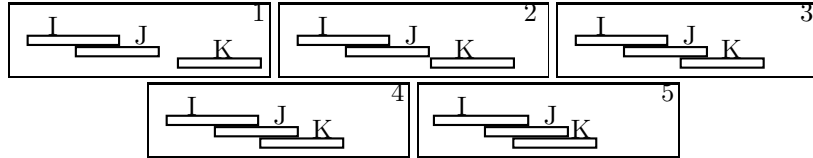
All the different sizes of intervals can be perfectly expressed in the point algebra PA . The relative size of two intervals can only be compared by $\{<, =, >\}$. The satisfiability problem of this substructure can be solved in P .

The sequential movement constraint and the size constraint can imply so-called *immobile intervals*. Such intervals cannot be moved because of their constraints (in Σ); they are related via a *point relation*. We call a relation a *point relation* if it is one of the following relations $\{m, mi, =, s, si, f, fi\}$. If between two intervals only a point relation holds, none of these two intervals will be moved during the whole transition process. We call the other relations *set relations*. Those relations can be moved during the whole transition process as illustrated in the following example.

Example 3. Consider the following problem:

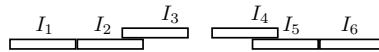
$$\begin{aligned} \sigma_s &= \{I \circ J, J \prec K, I \prec K\} \\ \sigma_f &= \{I \circ J, J \circ K, I \circ K\} \\ \Sigma &= \{I \{o\} J, J \{<, m, o\} K, I \{<, m, o\} K\} \end{aligned}$$

The following five scenarios describe a continuous deformation from σ_s to σ_f :

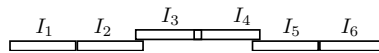


An example of a constraint (in Σ) that implies immobile intervals is $I_1 \{m\} I_2$. In the following, we will provide two examples where through immobile intervals the qualitative size of intervals comes into play. The distances between two immobile intervals can be compared by the length of other intervals. Since these intervals are not measured by a metric, but by their length wrt. other intervals, we call this distance *relative distance*.

Example 4. Consider the constraints $I_1 \{m\} I_2$ and $I_5 \{m\} I_6 \in \Sigma$ and $I_3 \prec I_4 \in \sigma_s$. Assume that the scenario σ_s has the following form:

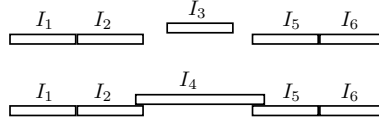


If in the scenario σ_f the relation $I_3 \circ I_4$ should hold, the relative distance of the immobile intervals comes into play.



It is clear that the distance between the two immobile intervals should be smaller than the sum of the sizes of the two intervals I_3 and I_4 . This is true for any constellation between two immobile intervals. Furthermore it is immediately clear that such immobile intervals help to compare different sizes of intervals.

Example 5. Assume two (immobile) intervals I_2 and I_5 (because of the constraints $I_1 \{m\} I_2$ and $I_5 \{m\} I_6 \in \Sigma$) and two intervals I_3 and I_4 passing by:



The two immobile intervals I_2 and I_5 imply a comparison between the length of the two intervals I_3 and I_4 .

Lemma 2. *Length constraints can be calculated in P .*

4.2 Main Theorem

We solve the problem by distinguishing between local and global solution of a problem $\langle \Sigma, \sigma_s, \sigma_f \rangle$. Consider every set V_j to consist of three different variables I_j, J_j, K_j appearing in Σ . Let Σ_j be the set of constraints in Σ involving the three variables in V_j , σ_{ij} and σ_{jf} respectively. C_j , the one-step transition chain for $\langle \Sigma_j, \sigma_{sj}, \sigma_{fj} \rangle$. We will call all the triples $\langle \Sigma_j, \sigma_{sj}, \sigma_{fj} \rangle$ *local transition chain problems*. We will call the solution of such problems *local solution* and the solution of a general satisfiability problem will be called *global solution*.

We investigate now the complexity of **TIASAT**, but, as we have seen, it is sufficient to investigate the complexity of the local transition problem.

A *transition chain* from σ_1 to σ_n is a sequence $\sigma_1, \dots, \sigma_n$ of scenarios for Σ such that the changes from σ_i to σ_{i+1} satisfy the neighborhood graph and such that only one object changes its position.

Instance: An initial scenario σ_s , a final scenario σ_f , and a constraint network Σ .

Question: Is there a transition chain from σ_s to σ_f that does not violate the constraints in Σ ?

This transition chain can be seen as a plan which transforms the objects of σ_s under the model constraints Σ to σ_f . We call such problems *transition chain problems*.

By generating the generalized neighborhood graph [14], more precisely, the (3,1)-neighborhood graph which represents all transformations between scenarios consisting of (exactly) three intervals (with the constraints of section 2), we can prove the existence of a transition chain from each scenario to another scenario of fixed length. The longest path that are in the generalized neighborhood graph, i.e. the transformation distance between a scenario consisting of three intervals to another has the maximal length of 24.

Lemma 3. *Let Σ be a satisfiable spatial CSP involving three variables. If the scenario transformation $\langle \Sigma, \sigma_i, \sigma_f \rangle$ is solvable under the size persistence, sequential movement and continuity constraints, then in Σ there exists a one-step transition chain from σ_i to σ_f of at most 24 scenarios.*

We will call (for two intervals I, J) the relation, which holds in the initial scenario, the *initial relation*, and the relation, which holds in the final scenario, the *final relation*. It becomes clear that during the transition chain both intervals I, J have to traverse

through the relations of the neighborhood graph from the initial relation to the final relation (if there is an interval K in-between I, J , it is clear that we have to traverse first the relations of the neighborhood graph of K , before I can change its relative position to J). We will call these changes *goal-directed* transition.

Lemma 4. *If a transition chain problem is solvable, then there is in any scenario for any two intervals a goal-directed transition possible.*

Proof (sketch). Let us assume that a given satisfiability problem is solvable and there are two intervals I_l and I_k that have to change their position w.l.o.g. from $I_l \prec I_k$ to $I_l \succ I_k$ (assume the identical length of both intervals) only before a certain scenario $\sigma_c \neq \sigma_f$. If there is a solution for this problem, then it must hold that $I_l \{ \prec, m, o, =, oi, mi, \succ \} I_k \in \Sigma$. Because these constraints do not change during the whole transition process, there must be in the interval σ_c , an interval I_m in-between I_l and I_k , i.e. $I_l \prec I_m \prec I_k$, and in Σ is the only relation between I_l and I_m w.l.o.g. \prec . But, if in σ_f the relative position should be $I_m \prec I_k \prec I_l$, then there cannot be a solution, because I_l and I_m had to change their relative position from \prec to \succ . This is not possible because of the constraints in Σ and we get a contradiction. \square

Lemma 5. *1. Two intervals I_1, I_2 cannot change their position, and make a problem unsatisfiable if:*

- (a) *The constraints of I_1 and I_2 in Σ are violated.*
- (b) *The constraints of at least one other interval I_3 with one of the intervals I_1, I_2 , are violated.*

Furthermore a problem can be unsolvable because of:

- (c) *Cycles in the solution path.*
 - (d) *The relative size constraints are violated.*
2. *For a transition chain problem, the problems (a),(b),(c) and (d), can be checked in polynomial time.*

Proof (sketch). We show (a) and (b) simultaneously: For that it is to show that two intervals I_1, I_2 cannot change their relative position (with respect to Σ) because of the constraints between them, or between them and (at least) another interval, but this follows directly from the definition of the transition chain problems. Lemma 1 proves that (a) can be checked in polynomial time, and (b) can be proved by calculating for any three intervals the possibility to change their initial position from σ_s to σ_f , with Lemma 5 this can be done in polynomial time. \square

The algorithm has to handle the above mentioned problems, e.g. to check the consistency of the relative size of the intervals with the point algebra PA . This can be done in polynomial time.

Theorem 2. *If a transition chain problem has a local solution, then it has a global solution.*

Proof (sketch). Let us assume that a transition chain problem has a local solution. Then we show that by synchronizing all C_j (the one-step transition chains) we can find a transition chain from σ_i to σ_f with respect to Σ involving less than $24n^2$ transitions

(scenarios of Σ). The case of parallel transitions is not to be considered, because of the sequential movement constraint, they are not possible. The calculations can be done in the following graph: The vertices are defined as the constraints that are changed in one or more chains. The directed edges correspond to the order of the changed constraints. For instance, if we have given a scenario $\sigma_i = \{I \circ J, J \circ K, I \circ K\}$ and a final scenario $\sigma_f = \{I \circ J, J \circ K, I \prec K\}$, we can compute the needed changes in advance - we know that to reach the final scenario the relations between I and K have to traverse the neighborhood graph from the relation $\{o\}$ to $\{m\}$. Therefore the transition chain has to change first $c_1 = I\{m\}K$, then $c_2 = I\{\prec\}K$ then $c_3 = I\{m\}J$ and finally $c_4 = J\{m\}K$. The synchronization graph contains the vertices c_1, c_2, c_3, c_4 and the directed edges are going from c_1 to c_2 from c_2 to c_3 and from c_3 to c_4 . It is clear that all the vertices with identical content in the transition chains are collapsed into the same vertex, and the edges are appropriately updated. Then we apply topological sort and we get a transition chain with respect to Σ from σ_i to σ_f . With Lemma 4 we know that this chain involves less than $24n^2$ transitions. \square

Theorem 3. *The satisfiability problem of temporalized intervals with respect to the size persistence, the sequential movement and continuity constraint is NP-complete.*

Proof. The NP-hardness follows from the NP-hardness of the satisfiability problem of \mathcal{IA} . Therefore it is sufficient to show that the membership is in NP. We guess a sequence of $24 \cdot n^2$ sets of constraints in which all pairs of spatial variables associated with I are constrained by a base relation. Let $\Theta_b^1, \Theta_b^2, \dots, \Theta_b^m$ be such a sequence, where $m = 24 \cdot n^2$. We check the following in polynomial time:

1. All the generated scenarios satisfy the size persistence constraint. This can be checked by constructing an extended spatial CSP, where the relative size of the constraints and the relative distance of the immobile objects can be expressed in the point algebra and as the satisfiability problem of this algebra lies in P, we can check all these constraints in polynomial time.
2. We check the continuity constraint in polynomial time: First, there is a finite number of scenarios in any transition chain (less than $24n^2$) and second we have only to check if the conceptual distance Δ of the previous scenario to the actual scenario is 1. If the conceptual distance of the previous scenario to the actual scenario is 0, we remove one.
3. Each (guessed) constraint set Θ_b^i in the sequence for a sub-interval I is a scenario for the set of (induced) constraints associated with I , except for the first and the last Θ_b -sets in the sequence, which can also be scenarios for the spatial constraints associated with the predecessor and the successor, respectively, sub-intervals of I (as we have described, we have a sequence of Θ_b -sets for each (induced) sub-interval).
4. The consistency of any scenario can be checked in polynomial time because of the path consistency algorithm. \square

5 Summary and Conclusion

Starting from the question of how qualitative spatial reasoning can be made more “real world” like, we investigated the temporalization of the interval algebra which has been

used as a spatial algebra. Such a temporalization demands from the models certain properties to be really “real-world” descriptions: First the size persistence constraint, i.e. the size of the objects do not change, second the sequential movement constraint, i.e. an object moves after another object and third the continuity constraint, i.e. that the change of the relation between the objects happens continuously. Different problems can make a constraint system unsatisfiable (cf. Examples 1, 2, and 3). Furthermore, we investigated the complexity of the general satisfiability problem for the temporalized interval algebra under the given axioms and showed that it is NP-complete. The proof technique seems to be transferable to other temporalized relation algebras.

This work will be continued as follows: Transfer the results to higher dimensions starting with the block calculus. Apply the proof technique to other relation algebras. In a next step investigate a generalization of the linear time model to nonlinear structures. Use the presented spatio-temporal framework and combine it with the direction calculus [16]. This promises to be a relation algebra suitable for general transportation networks.

Acknowledgements

The author would like to thank Stefan Wöfl and Bernhard Nebel for discussions and two anonymous reviewers for their comments and suggestions. This work was supported by the SFB/TR 8 Spatial Cognition funded by the Deutsche Forschungsgemeinschaft.

References

- [1] J. F. Allen. Maintaining knowledge about temporal intervals. *Comm. of the ACM*, 26(11): 832–843, 1983.
- [2] P. Balbiani, J.-F. Condotta, L. Farinas del Cerro, and A. Osmani. A model for reasoning about generalized intervals. *Proc. WS ECAI 1998*, (1998).
- [3] A. Borning, R. Lin, & K. Marriott. Constraint-based document layout for the Web. *Multimedia Systems*, 8: 177–189, 2000.
- [4] A. G. Cohn. Qualitative spatial representation and reasoning techniques, KI-97. *Advances in Artificial Intelligence*, LNAI, pages 1–30. Springer, 1997.
- [5] A. U. Frank. Qualitative spatial reasoning: Cardinal directions as an example. *International Journal of Geographical Information Science*, 10(3): 269–290, 1996.
- [6] C. Freksa. Spatial Cognition: An AI perspective. *ECAI 2004*, pages 1122–1128, 2004.
- [7] A. Gerevini, B. Nebel. Qualitative spatio-temporal reasoning with RCC-8 and Allen’s Interval Calculus: Computational complexity. In *Proceedings of ECAI 2002*, pages 312–316. IOS Press, 2002.
- [8] OpenGIS Consortium. (1999). OpenGIS: Simple features specification for SQL. *Wayland, MA: OpenGIS Consortium, Inc.*, 2004.
- [9] M.-C. Golumbic and R. Shamir. Complexity and algorithms for reasoning about time: A graph-theoretic approach. *JACM* 40(5): 1108–1133, 1993.

- [10] H. W. Guesgen. Spatial reasoning based on Allen's temporal logic. *International Computer Science Institute technical report*. TR-89-049, Berkeley, CA.
- [11] P. B. Ladkin and R. Maddux: On binary constraint problems. *JACM*, 41(3): 435–469, 1994.
- [12] P. B. Ladkin and A. Reinefeld. Effective solution of qualitative interval constraint problems. *Artificial Intelligence*, 57(1): 105–124, 1992.
- [15] B. Nebel and H.-J. Bürckert: Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *JACM*, 42(1): 43–66, 1995.
- [14] M. Ragni and S. Wöfl. On generalized neighborhood graphs. *KI 2005: Advances in Artificial Intelligence, 28th Annual German Conference on AI*. Ed. U. Furbach. Springer 2005.
- [15] M. Ragni and S. Wöfl. Temporalizing cardinal directions: From constraint satisfaction to planning. *Proceedings of the KR 2006*. (To appear).
- [16] J. Renz. A Spatial Odyssey of the Interval Algebra: 1. Directed intervals. In *Proc. of IJCAI 2001*, pages 51–56, 2001.
- [17] M. B. Vilain, H. A. Kautz, and P. G. van Beek. Constraint propagation algorithms for temporal reasoning: A revised report. In *Readings in Qualitative Reasoning about Physical Systems*. pages 373–381. Morgan Kaufmann, 1989.
- [18] F. Wolter and M. Zakharyashev. Qualitative spatio-temporal representation and reasoning: a computational perspective. In G. Lakemeyer and B. Nebel (eds): *Exploring Artificial Intelligence in the New Millennium*, pages 175–216, Morgan Kaufmann, 2002.

A Categorical Perspective on Qualitative Constraint Calculi

Till Mossakowski¹, Lutz Schröder¹, and Stefan Wölfl²

¹ DFKI Lab Bremen and Department of Computer Science, University of Bremen
P.O. Box 330440, 28334 Bremen, Germany
{till, lschrode}@informatik.uni-bremen.de

² Department of Computer Science, University of Freiburg
Georges-Köhler-Allee, 79110 Freiburg, Germany
woelfl@informatik.uni-freiburg.de

Abstract. In the domain of qualitative constraint reasoning, a subfield of AI which has evolved in the last 25 years, a large number of calculi for efficient reasoning about space and time has been developed. Reasoning problems in such calculi are usually formulated as constraint satisfaction problems. For temporal and spatial reasoning, these problems often have infinite domains, which need to be abstracted to (finite) algebras in order to become computationally feasible.

Ligozat [13] has argued that the notion of *weak representation* plays a crucial rôle: it not only captures the correspondence between abstract relations (in a relation algebra or non-associative algebra) and relations in a concrete domain, but also corresponds to algebraically closed constraint networks.

In this work, we examine properties of the category of weak representations and treat the relations between partition schemes, non-associative algebras and concrete domains in a systematic way. This leads to the notion of *semi-strong* representation, which captures the correspondence between abstract and concrete relations better than the notion of weak representation does. The slogan is that semi-strong representations avoid unnecessary loss of information. Furthermore, we hope that the categorical perspective will help in the future to provide new insights on the important problem of determining whether algebraic closedness decides consistency of constraint networks.

1 Introduction

Qualitative reasoning aims at describing the common-sense background knowledge on which our human perspective on the physical reality is based. Methodically, qualitative constraint calculi restrict the vocabulary of rich mathematical theories dealing with temporal or spatial entities such that specific aspects of these theories can be treated within decidable fragments with simple qualitative (i. e., non-metric) languages. Contrary to mathematical or physical theories about space and time, qualitative constraint calculi allow for rather inexpensive reasoning about entities located in space and time. For this reason, the limited expressiveness of qualitative representation formalism calculi is a benefit if such reasoning tasks need to be integrated in applications. For example, some

of these calculi may be implemented for handling spatial GIS queries efficiently and some may be used for navigating, and communicating with, a mobile robot.

In the past 25 years the number of calculi for efficient reasoning about space and time has grown quite steadily. Examples of temporal calculi include the so-called point algebra, Allen’s interval algebra [2], and Vilain’s point-interval calculus [20]. The most prominent spatial calculi are mereotopological calculi (e. g., [3]), Frank’s cardinal direction calculus [9], Freksa’s double cross calculus [10], Egenhofer and Franzosa’s 4- and 9-intersection calculi [7, 8], Ligozat’s flip-flop calculus [14], and various region connection calculi proposed by Randell et al. [18], Cohn et al. [5], Düntsch et al. [6], and Gerevini and Renz [11].

Reasoning problems in qualitative calculi are usually formulated as so-called constraint satisfaction problems. Starting from a set of base relations (i. e., a family of relations that partitions the set of all tuples of domain elements), a constraint is a formula of the form xRy with variables x and y (taking values in given domains D_x and D_y) and a set of base relations R defined between the domains of x and y —the set of base relations, R , is read disjunctively and hence expresses imprecise knowledge about the concrete scenario described by the constraint formula. The constraint satisfaction problem with respect to a fixed qualitative calculus is to determine for a given constraint network (i. e., a finite set of constraints) whether there exists an assignment to its variables such that all constraints of the network become true. Further typical reasoning tasks are to check that some constraint is entailed by a constraint network, and to compute an equivalent minimal constraint network (all these reasoning tasks are equivalent under polynomial Turing reductions).

A crucial aspect for developing efficient algorithms for qualitative spatial and temporal reasoning is the fact that the underlying model classes usually contain infinite models. Hence, in order to test satisfiability of constraint networks, it is not feasible to enumerate all models and all possible assignments to variables in a fixed model until one finds a satisfying assignment. For this reason other techniques (such as path-consistency algorithms) must be applied for testing satisfiability. These techniques usually depend on the so-called composition table of the calculus at hand (for an example, see section 3). The idea behind these methods is to encode domain-dependent knowledge in a table that lists which relations may possibly hold between two objects a and b , when knowledge about the relations of a resp. b to some other objects is available.

However, there are different possibilities of how to read these composition tables [4]. And as a result of a somehow conceptual confusion, the path consistency method has sometimes been employed, although the underlying interpretation of composition was not justified by the given domain. To clarify this confusion, Ligozat [12] introduced the notion of *weak representation*, which not only captures the correspondence between abstract relations (in a relation algebra or non-associative algebra) and relations in a concrete domain, but also corresponds to algebraically closed constraint networks.

In this paper, we provide an even more abstract, namely a category-theoretical, point of view in order to examine properties of the weak representations and treat the relations between partition schemes, non-associative algebras, and concrete domains in a systematic way (some initial category-theoretic treatment is given by Ligozat [13]). This leads to the notion of *semi-strong* representation, which, in our opinion, captures

the correspondence between abstract and concrete relations better than the notion of weak representation does.

The paper is organized as follows. In section 2, we will provide the basic category-theoretical background. In section 3, we discuss the category of partition schemes and show how qualitative calculi may be represented in a category-theoretical framework. Then in section 4, the category of non-associative relation algebras is introduced. Section 5 and section 6 are dedicated to discuss the relationships between strong and weak composition as well as those between weak and strong representations of abstract (non-associative) relation algebras. In this section we will also discuss a new reasonable notion of representation, which is slightly weaker than the concept of strong representation, but which is conceptually more adequate than the concept of weak representation.

2 Categorical Background

To start with, let us briefly recall some basic notions of category theory which we will refer to in the following sections (for useful introductory texts see [16, 1]).

A *category* \mathbf{C} consists of a class $\text{Ob}(\mathbf{C})$ of *objects*, for each pair (A, B) of objects a class $\text{hom}_{\mathbf{C}}(A, B)$ of *morphisms*, where we write $f : A \rightarrow B$ for $f \in \text{hom}(A, B)$, a choice of an *identity* $\text{id}_A : A \rightarrow A$ for each object A , and a *composition* operation assigning to each pair (g, f) of morphisms $f : A \rightarrow B$ and $g : B \rightarrow C$ its *composite* $g \circ f : A \rightarrow C$. These data are subject to two equational laws: composition is *associative*, i. e., $h \circ (g \circ f) = (h \circ g) \circ f$ whenever these terms are defined, and identities are neutral w. r. t. composition, i. e., $\text{id}_B \circ f = f = f \circ \text{id}_A$ for each morphism $f : A \rightarrow B$.

Given two categories \mathbf{A} and \mathbf{B} , a *functor* $F : \mathbf{A} \rightarrow \mathbf{B}$ assigns to each \mathbf{A} -object A a \mathbf{B} -object FA , and to each \mathbf{A} -morphism $f : A \rightarrow B$ a \mathbf{B} -morphism $Ff : FA \rightarrow FB$ such that identities and composition are preserved, i. e., $F\text{id}_A = \text{id}_{FA}$ for each object A , and $F(g \circ f) = Fg \circ Ff$ whenever $g \circ f$ is defined.

Example 1. The category **Set** of sets and maps has as its objects all sets, and as morphisms $X \rightarrow Y$ all maps $f : X \rightarrow Y$. Composition is the usual composition of maps, and identities are identity maps. **Set** has several *non-full* subcategories (a subcategory \mathbf{A} of a category \mathbf{B} is called *full* if $\text{hom}_{\mathbf{A}}(A, B) = \text{hom}_{\mathbf{B}}(A, B)$ whenever $A, B \in \text{Ob}(\mathbf{A})$), such as the category of sets and injective maps and the category **Set**_{surj} of sets and surjective maps.

The category **Top** of topological spaces has as its objects all topological spaces, and as morphisms $X \rightarrow Y$ all continuous maps $f : X \rightarrow Y$. **Top** is a typical example of a *concrete* category, given in terms of structured objects over a base category, in this case **Set**. Formally, one has a *forgetful functor* $\mathbf{Top} \rightarrow \mathbf{Set}$ that maps every topological space to its underlying set and each continuous map to the map itself.

An important strength of the definition of category is that it is stable under numerous constructions yielding new categories, such as the following. Given a category \mathbf{C} , its *dual* \mathbf{C}^{op} is the category having the same objects as \mathbf{C} , and as morphisms $A \rightarrow B$ the \mathbf{C} -morphisms $B \rightarrow A$. Given functors $F : \mathbf{A} \rightarrow \mathbf{C}$ and $G : \mathbf{B} \rightarrow \mathbf{C}$, the *comma category* (F, G) has triples (A, f, B) as objects, where $A \in \text{Ob}(\mathbf{A})$, $B \in \text{Ob}(\mathbf{B})$, and $f : FA \rightarrow GB$, and pairs (g, h) as morphisms $(A_1, f_1, B_1) \rightarrow (A_2, f_2, B_2)$, where $g : A_1 \rightarrow A_2$, $h : B_1 \rightarrow B_2$,

and $Gh \circ f_1 = f_2 \circ Fg$. An important special case of the latter construction is that F is a *constant* functor taking all objects to a fixed object $C \in \text{Ob } \mathbf{C}$, and all morphisms to id_C . In this case, the comma category (F, G) is denoted more simply as (C, G) , and its description simplifies as follows: objects are pairs (f, B) , where $B \in \text{Ob } \mathbf{B}$ and $f : C \rightarrow GB$, and morphisms $(f_1, B_1) \rightarrow (f_2, B_2)$ are morphisms $h : B_1 \rightarrow B_2$ such that $Gh \circ f_1 = f_2$.

Given two functors $F, G : \mathbf{A} \rightarrow \mathbf{B}$, a *natural transformation* $\eta : F \rightarrow G$ consists of a family $(\eta_A : FA \rightarrow GA)_{A \in \mathbf{A}}$ of \mathbf{B} -morphisms, such that the following naturality condition is fulfilled: for each \mathbf{A} -morphism $f : A_1 \rightarrow A_2$, $Gf \circ \eta_{A_1} = \eta_{A_2} \circ Ff$.

3 Partition Schemes

As stated before, qualitative calculi usually start from a set of so-called *base relations*, which is a family of relations that partitions the set of all tuples of domain elements at hand. This approach can be formally captured by the notion of *partition scheme*.

Definition 1 (cp. Ligozat and Renz [15]). Let U be a non-empty set. A *partition scheme* on U is defined by a finite (index) set I with a distinguished element $i_0 \in I$, a unary operation \smile on I , and a family of binary relations $(R_i)_{i \in I}$ on U such that

- (a) $(R_i)_{i \in I}$ is a partition of $U \times U$ in the sense that the R_i are pairwise disjoint and jointly exhaustive.³
- (b) R_{i_0} is the diagonal relation $\{(x, x) \mid x \in U\}$.
- (c) $R_{i\smile}$ is the (set-theoretical) converse of relation R_i , for each $i \in I$.

The relations R_i are referred to as *basic relations*. In the following we often write

$$U \times U = \bigcup_{i \in I} R_i$$

to denote partition schemes.

Here, we additionally introduce *morphisms of partition schemes*.

Definition 2. Let $(R_i)_{i \in I}$ and $(S_j)_{j \in J}$ be partition schemes on U and V , respectively. A *morphism* $(h, k) : U \times U = \bigcup_{i \in I} R_i \rightarrow V \times V = \bigcup_{j \in J} S_j$ is a pair of functions $h : U \rightarrow V$ and $k : I \rightarrow J$ such that

- (a) $k(i_0) = j_0$,
- (b) $k(i\smile) = k(i)\smile$, and
- (c) $(h \times h)[R_i] \subseteq S_{k(i)}$, i. e., for all $x, y \in U$ with $xR_i y$, $h(x)S_{k(i)}h(y)$ holds.

If such a morphism exists, we also say that $(V \times V = \bigcup_{j \in J} S_j)$ is *refined* (via (h, k)) to $(U \times U = \bigcup_{i \in I} R_i)$ (note that the source scheme is the target of the refinement).

³ Ligozat and Renz require the R_i to be non-empty. This requirement leads to the problem that, for example, the RCC8 calculus leads to a partition scheme only for certain (e.g. connected) topological spaces. We hence drop this requirement in this paper.

Together with pairwise identities and composition, this gives a category **Part** of partition schemes.

Example 2. $RCC5$ is a functor $RCC5: \mathbf{Set}_{surj}^{op} \longrightarrow \mathbf{Part}$, where \mathbf{Set}_{surj} is the category of sets and surjective maps. A set S is sent to the partitioning of $\mathcal{P}(S) \times \mathcal{P}(S)$ into the five relations DR, PO, PP, PPI, and EQ (cf. Table 1). A surjective function $f: S_1 \longrightarrow S_2$ is mapped to (f^{-1}, id) . For such an f , f^{-1} preserves emptiness, and by surjectivity of f also non-emptiness, of sets. Since set-theoretic intersection and difference commute with f^{-1} , f^{-1} preserves the $RCC5$ relations (cf. Table 1).

Table 1: Characteristic properties of $RCC5$ relations. n means “non-empty”, e means “empty”, $?$ means “don’t care”.

relation	$X_1 \cap X_2$	$X_1 \setminus X_2$	$X_2 \setminus X_1$
PO	n	n	n
DR	e	n	n
EQ	$?$	e	e
PP	$?$	e	n
PPi	$?$	n	e

Example 3. $RCC8$ is a functor $RCC8: \mathbf{Top}_{surj,open}^{op} \longrightarrow \mathbf{Part}$, where $\mathbf{Top}_{surj,open}$ is the category of topological spaces and surjective open continuous maps. Let (S, \mathcal{O}) be such a topological space and let Reg denote the set of all non-empty regular closed subsets. The functor sends (S, \mathcal{O}) to the partitioning of $Reg \times Reg$ into the relations PO, EQ, DC, EC, TPP, NTPP, TPPi, and NTPPi (cf. Table 2). A surjective continuous and open function $f: (S_1, \mathcal{O}_1) \longrightarrow (S_2, \mathcal{O}_2)$ is mapped to (f^{-1}, id) . Then f^{-1} commutes with int (additionally to the properties listed in Example 2), and by Table 2, f^{-1} preserves the $RCC8$ relations.

Table 2: Characteristic properties of $RCC8$ relations. n means “non-empty”, e means “empty”, $?$ means “don’t care”.

relation	$X_1 \cap X_2$	$X_1 \setminus X_2$	$X_2 \setminus X_1$	$int(X_1) \cap int(X_2)$	$X_1 \setminus int(X_2)$	$X_2 \setminus int(X_1)$
PO	n	n	n	n	n	n
EC	n	n	n	e	n	n
DC	e	n	n	e	n	n
EQ	n	e	e	n	$?$	$?$
TPP	n	e	n	n	n	n
$NTPP$	n	e	n	n	e	n
$TPPi$	n	n	e	n	n	n
$NTPPi$	n	n	e	n	n	e

Example 4. The refinement of $RCC5$ into $RCC8$ is a natural transformation $\theta: RCC8 \rightarrow RCC5 \circ \mathcal{U}$, where $\mathcal{U}: \mathbf{Top}_{surj} \rightarrow \mathbf{Set}_{surj}$ is the forgetful functor. θ just forgets regular closedness of sets, and sends each $RCC8$ relation to the corresponding $RCC5$ relation (e.g. both TPP and NTPP are sent to PP). The homomorphism property of $\theta_{(S,O)}$ follows from the fact that Table 1 is part of Table 2.

4 Non-Associative Algebras

We are interested in approximating compositions of relations in a finite way. Ligozat and Renz [15] consider so-called non-associative relation algebras in order to capture weak composition (as introduced in the following section).

Definition 3 (Maddux [17]). A *non-associative (relation) algebra* is a tuple $\mathcal{A} = (A, +, -, \cdot, 0, 1, ;, \smile, \Delta)$ such that:

1. $(A, +, -, \cdot, 0, 1)$ is a Boolean algebra.
2. Δ is a constant, \smile a unary and $;$ a binary operation such that, for any $a, b, c \in A$:

$$\begin{aligned} (a) \ (a^\smile)^\smile &= a & (b) \ \Delta; a &= a; \Delta = a & (c) \ a; (b+c) &= a; b + a; c \\ (d) \ (a+b)^\smile &= a^\smile + b^\smile & (e) \ (a-b)^\smile &= a^\smile - b^\smile & (f) \ (a;b)^\smile &= b^\smile; a^\smile \\ (g) \ (a;b) \cdot c^\smile &= 0 \text{ if and only if } & (b;c) \cdot a^\smile &= 0 \end{aligned}$$

Given non-associative algebras A and B , a *homomorphism* from A to B is a homomorphism $h: A \rightarrow B$ on the underlying Boolean algebras such that

- (a) $h(\Delta) \geq \Delta$,
- (b) $h(a^\smile) = h(a)^\smile$ for all $a \in A$, and
- (c) $h(a;b) \geq h(a); h(b)$ for all $a, b \in A$.

Together with set-theoretic identities and composition, this defines the category \mathbf{NA} of non-associative algebras.

A non-associative algebra is a *relation algebra* if the operation $;$ is associative. Let \mathbf{RA} denote the full subcategory of \mathbf{NA} consisting of all relation algebras. A homomorphism of non-associative algebras is *strong* if the above inequalities (a) and (c) are equalities. Let \mathbf{NA}_s be the (non-full) subcategory of \mathbf{NA} consisting of the strong homomorphisms.

An *atomic* non-associative algebra is one that is atomic as a Boolean algebra. The atoms are also called *basic relations* in this case.

5 Strong and Weak Composition

In our setting, *strong* composition can be modelled by a contravariant functor $\mathbf{S}: \mathbf{Set}^{op} \rightarrow \mathbf{RA}$. On objects, it maps a set U to $\mathcal{P}(U \times U)$ equipped with the usual set-theoretic interpretation of $+$ as union, $-$ as set difference, \cdot as intersection, 0 as empty relation, 1 as the universal relation $U \times U$, $;$ as composition, \smile as converse, and Δ as the diagonal relation. Given a function $f: U \rightarrow V$, $\mathbf{S}(f)$ takes inverse images w.r.t. $f \times f$, i.e., a

relation $R \subseteq V \times V$ is taken to $\mathbf{S}(f)(R) = (f \times f)^{-1}[R]$. By abuse of notation, we will denote the composition of \mathbf{S} with the inclusion of \mathbf{RA} into \mathbf{NA} by $\mathbf{S}: \mathbf{Set}^{op} \rightarrow \mathbf{NA}$. The drawback of this construction is that the (usually infinite) space of all relations is not structured. This structuring can be obtained by using a partition scheme.

The notion of weak composition approximates composition of set-theoretic relations (which is strong composition) within a partition scheme, thus leading to a non-associative algebra. The functor $\mathbf{W}: \mathbf{Part}^{op} \rightarrow \mathbf{NA}$ maps a partition scheme

$$U \times U = \bigcup_{i \in I} R_i$$

to the non-associative algebra that has $\mathcal{P}(I)$ as its Boolean algebra component. The converse is given by pointwise application of \smile ; the diagonal is i_0 . Composition is given by *weak composition*:⁴

$$I_1; I_2 = \{i \mid i_1 \in I_1, i_2 \in I_2, (R_{i_1} \circ R_{i_2}) \cap R_i \neq \emptyset\}.$$

Given a morphism (h, k) of partition schemes, $\mathbf{W}(h, k)$ is just k^{-1} . In order to prove that this is a homomorphism of non-associative algebras, note that $(R_{i_1} \circ R_{i_2}) \cap R_i \neq \emptyset$ implies $(S_{k(i_1)} \circ S_{k(i_2)}) \cap S_{k(i)} \neq \emptyset$.

6 Weak, Strong and Semi-Strong Representations

We now discuss representations of abstract non-associative algebras in concrete (i.e. set-theoretic) domains. With the above machinery, we are able to recast the definition of Ligozat and Renz [15] as follows:

Definition 4. A *weak representation* of a non-associative algebra A (in a domain with underlying set U) is a homomorphism of non-associative algebras $\varphi: A \rightarrow \mathbf{S}(U)$. A weak representation is *diagonal-persevering*, if $\varphi(\Delta) = \Delta$.

Note that our notion of weak representation is slightly weaker than that in [13], because we do only require that $\varphi(\Delta)$ contains the diagonal relation, while [13] requires weak representations to be always diagonal-preserving.

Proposition 1. Given two weak representations $\varphi, \psi: A \rightarrow \mathbf{S}(U)$ with

$$\varphi(a) \subseteq \psi(a) \quad (a \in A),$$

we already have

$$\varphi = \psi.$$

Proof. Writing \bar{a} for the complement $1 - a$ of a , we have for $a \in A$ that $\varphi(a) = \varphi(\bar{\bar{a}}) = \overline{\varphi(\bar{a})}$, and similarly $\psi(a) = \overline{\psi(\bar{a})}$. Since $\varphi(\bar{a}) \subseteq \psi(\bar{a})$, we get $\overline{\psi(\bar{a})} \subseteq \overline{\varphi(\bar{a})}$. But this means $\psi(a) \subseteq \varphi(a)$. Altogether, $\varphi(a) = \psi(a)$. \square

⁴ Note that it is common in the category theory community to use \circ for function composition in applicative order, and $;$ for diagrammatic order. By contrast, in the qualitative reasoning community both \circ and $;$ are used for composition of *relations* in *diagrammatic* order. \circ stands for the usual set-theoretic composition, $;$ for weak composition.

Definition 5 (cp. Ligozat [13]). Given weak representations $\varphi: A \rightarrow \mathbf{S}(U)$ and $\psi: A \rightarrow \mathbf{S}(V)$, a *morphism* $h: \varphi \rightarrow \psi$ of weak representations is a function $h: U \rightarrow V$ such that for all $a \in A^5$ and $(x, y) \in U \times U$, if $(x, y) \in \varphi(a)$, then $(h(x), h(y)) \in \psi(a)$.

Proposition 2. A morphism $h: \varphi \rightarrow \psi$ of weak representations equivalently is a function $h: U \rightarrow V$ such that $\varphi = \mathbf{S}(h) \circ \psi$. That is, the category of weak representations $\mathbf{WR}(A)$ is the comma category (A, \mathbf{S}) .

Proof. The implication $(x, y) \in \varphi(a) \Rightarrow (h(x), h(y)) \in \psi(a)$ is equivalent to $\varphi(a) \subseteq \mathbf{S}(h)(\psi(a))$. By Prop. 1, this is equivalent to $\varphi(a) = \mathbf{S}(h)(\psi(a))$. \square

Definition 6. The category \mathbf{WR} of weak representations (over varying non-associative algebras) is a comma category. Objects are weak representations $\varphi: A \rightarrow \mathbf{S}(U)$, and morphisms are commutative squares:

$$\begin{array}{ccc} A_1 & \xrightarrow{k} & A_2 \\ \downarrow \varphi_2 & & \downarrow \varphi_2 \\ \mathbf{S}U_1 & \xrightarrow{\mathbf{S}h} & \mathbf{S}U_2 \end{array}$$

Theorem 1. The functor $\mathbf{W}: \mathbf{Part}^{op} \rightarrow \mathbf{NA}$ (introduced in section 4) can be extended to a functor $\mathbf{W}: \mathbf{Part}^{op} \rightarrow \mathbf{WR}$, by regarding the non-associative algebra of a partition scheme as weakly represented in the underlying set of the partition scheme itself.

Proof. $U \times U = \bigcup_{i \in I} R_i$ can be represented in $\mathbf{S}U$ by just mapping a set $J \subseteq I$ to $\bigcup_{j \in J} R_j$. This clearly is a homomorphism of Boolean algebras, preserves the diagonal as well as converse relations (by the definition of partition scheme), and weakly preserves composition by the definition of weak composition. Given a morphism $(h, k): (U \times U = \bigcup_{i \in I} R_i) \rightarrow (V \times V = \bigcup_{j \in J} S_j)$, let $\mathbf{W}(h, k)$ be $(k^{-1}, \mathbf{S}h)$. To prove that this is a morphism in \mathbf{WR} , given $J_0 \subseteq J$, in light of Prop. 1 it suffices to show that

$$\bigcup_{i \in k^{-1}(J_0)} R_i \subseteq (h \times h)^{-1} \left(\bigcup_{j \in J_0} S_j \right).$$

But this easily follows from $(h \times h)[R_i] \subseteq S_{k(i)}$. \square

A weak representation is *strong* if it is strong as a homomorphism of non-associative algebras. Unfortunately, weak representations arising from partition schemes are usually not strong. However, a weak representation postulates only a very loose connection between abstract and set-theoretic composition. Consider the following examples:

Example 5. Let $RCC5$ be the non-associative algebra of $RCC5$, U be a non-empty set, and $\varphi: RCC5 \rightarrow \mathbf{S}U$ map the base relations as follows: EQ is mapped to $U \times U$, and the other base relations are mapped to \emptyset . This is easily extended to sets of base relations.

⁵ Ligozat requires this only for basic relations, i.e. for the atoms of an atomic non-associative algebra. This seems to be unnaturally weak; in case of finite algebras, it is equivalent to our formulation.

Example 6. Let A be an atomic non-associative algebra such that composition and converse distributes over arbitrary joins (note that this holds in particular for any *finite* non-associative algebra), with Δ atomic (note that this implies $\Delta = \Delta^\vee$). Then we can define a non-associative algebra $Loss(A)$. $Loss(A)$ is like A , but with composition $;$ replaced by composition $\#$, which is defined on basic relations as follows:

$$a\#b = \begin{cases} b, & \text{if } a = \Delta \\ a, & \text{if } b = \Delta \\ 1, & \text{if } a \neq \Delta \neq b, \Delta \leq a; b \\ 1 - \Delta, & \text{otherwise} \end{cases}$$

By the distributivity assumption, the laws for non-associative algebras need only be verified for basic relations, which is not difficult.

Since $Loss(A)$ enlarges the composition of A , the identity is a homomorphism $id : Loss(A) \rightarrow A$. Hence, any weak representation of A leads to a weak representation of $Loss(A)$ by composing with $id : Loss(A) \rightarrow A$. In particular, we get weak representations of $Loss(RCC5)$ and $Loss(RCC8)$.

These weak representations are hardly useful for anything, because abstract composition only provides very little information about concrete composition. While Example 5 is in a sense pathological because the representation is not diagonal-preserving (and this is exploited in an extreme way), the representations of Example 6 *are* diagonal-preserving. In this example, abstract compositions are larger than necessary. Indeed, most information about concrete composition is thrown away, and only information about the diagonal relation is kept. In the light of the possibility to have a better weak representation (namely the standard representations for $RCC5$, $RCC8$ etc.), this must be considered as an unnecessary loss of information. Therefore, we cannot agree with the slogan of [15] “A qualitative calculus is a weak representation.” Apparently, a qualitative calculus has a connection between its abstract non-associative algebra and its concrete domain that is stronger than the one described in terms of weak representations. Hence, we will strengthen the representation condition as follows, in order to capture the situation that no unnecessary loss of information occurs:

Definition 7. Given an atomic non-associative algebra A , a weak representation $\varphi : A \rightarrow \mathbf{S}(U)$ is said to be *semi-strong* if for all $b, c \in A$,

$$b;c = \bigvee \{a \mid a \text{ atomic}, (\varphi(b) \circ \varphi(c)) \cap \varphi(a) \neq 0\}.$$

While the weak representations induced by $RCC5$ and $RCC8$ (Examples 2 and 3) are semi-strong, the weak representations of Examples 5 and 6 are not. The notion of semi-strong representation thus avoids the inclusion of representations that have only a limited connection between the abstract algebra and the concrete representation, while simultaneously providing more flexibility than strong representations, which are too strong to capture weak composition. Indeed, semi-strong representations are precisely the notion that captures weak composition:

Observation 1. *The weak representation induced by a partition scheme is semi-strong.*

Proof. Obvious, by definition of weak composition. □

Recall from [19] that a (finite) constraint network on an atomic non-associative algebra A is a pair $\mathcal{N} = (N, \rho)$, where N is a (finite) set of nodes (or variables) and ρ a map $\rho : N \times N \rightarrow A$. For each pair (i, j) of nodes, $\rho(i, j)$ is the constraint on the arc (i, j) . A network is atomic if ρ is in fact a map into the set of atoms of A . It is normalized if $\forall i, j \in N, \rho(i, j) = \Delta$ if $i = j$, and $\forall i, j \in N, \rho(j, i)^\smile = \rho(i, j)$. A network (N', ρ') is a refinement of (N, ρ) if $\forall i, j \in N$ we have $\rho'(i, j) \leq \rho(i, j)$. Finally, a network is algebraically closed, or a-closed, if $\forall i, j, k \in N, \rho(i, j) \leq \rho(i, k); \rho(k, j)$. (Note that a network can be made a-closed using the path-consistency algorithm.)

The crucial observation of Ligozat and Renz [15] is the following: Given a normalized and atomic constraint network \mathcal{N} over a non-associative relation algebra A , \mathcal{N} is a-closed if and only if it corresponds to a weak representation $\rho^\mathcal{N}$ in $\mathbf{WR}(A)$. Now an a-closed network \mathcal{N} is consistent w.r.t. a given domain of interpretation $\varphi \in \mathbf{WR}(A)$ if and only if there is a morphism of weak representations $h : \varphi \longrightarrow \rho^\mathcal{N}$. This can be summarized as follows:

Observation 2. *For a weak representation that is weakly terminal⁶ in $\mathbf{WR}(A)$, a-closedness decides consistency of constraint networks.*

Ligozat and Renz [15] point out that the question whether a-closedness decides consistency of constraint networks is of fundamental nature, and they give a criterion to determine whether a qualitative calculus enjoys this property. Still, it may be hard to apply their criterion in practice. With our categorical approach via weakly terminal objects, we can try to apply standard categorical, algebraic and coalgebraic methods for answering the question whether a-closedness decides consistency.

7 Conclusion

We have outlined a categorical framework for the unifying treatment of (binary) qualitative constraint calculi. In particular, we have introduced a category of partition schemes and defined standard calculi such as RCC5 and RCC8 as functorial indexings of partition schemes. Moreover, we have identified the category of weak representations of non-associative algebras as a comma category over the category of non-associative algebras, and we have proposed a strengthening of the notion of weak representation: *semi-strong* representations capture the properties of weak composition more precisely than both weak and strong representations do. They allow coarser abstractions than strong representations, but avoid unnecessary loss of information (that can occur with weak representations). We suggest to strengthen the slogan of Ligozat and Renz [15] “A qualitative calculus is a weak representation” in the following way:

A qualitative calculus is a semi-strong representation.

This paper could only present the main ideas of a category-theoretical approach to qualitative constraint calculi. The following questions still remain open:

⁶ A *weakly terminal object* of a category \mathbf{C} is an object T in \mathbf{C} such that to each object A in \mathbf{C} , there exists a morphism $A \rightarrow T$.

- Weakly terminal representations in $\mathbf{WR}(A)$ have the pleasant property that a-closedness decides consistency of constraint networks. Does this characterize weak terminality?
- Given a non-associative algebra, what are its weakly terminal representations? This question, however, is of more theoretical interest, because usually, one does not start with a non-associative algebra, but rather with a concrete domain.
- Hence, the following question is more important: Given a partition scheme, if we apply the functor $\mathbf{W}: \mathbf{Part}^{op} \rightarrow \mathbf{WR}$ to it, how to determine whether the resulting semi-strong representation is weakly terminal? And if it is not weakly terminal, can it be embedded somehow in a semi-strong and weakly terminal representation (possibly using a refinement of the partition scheme)? We hope to apply algebraic and coalgebraic methods to tackle this problem.

A further perspective is indicated by the fact that we had to restrict the functorial representation of typical calculi such as RCC5 and RCC8 to not entirely natural non-full subcategories, e.g. the category of surjective open continuous maps in the case of RCC8. The main suspect as the cause of this technical difficulty is the disjointness requirement in the definition of partition scheme, which forces the use of base relations such as “proper part of” and which itself is motivated by constructions of representations where the base relations play the role of atoms. A technically more pleasing alternative might be to choose a more natural set of base relations, compatible with preimage formation in the relevant category (this would classify relations such as “part of” or “interior part of” as natural, but “proper part of” as unnatural), from which other relations may be built as Boolean combinations. One may then hope to obtain representations using newly constructed atoms, in analogy to the use of maximally consistent sets in propositional logics.

Acknowledgements

We wish to thank Gérard Ligozat for interesting discussions during a mini-workshop on qualitative spatial reasoning in Freiburg in January 2006. Erwin R. Catesbeina deserves thanks for his semi-sharp remarks on some inconsistencies in earlier versions of the paper. This work was partially supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Transregional Collaborative Research Center SFB/TR 8 *Spatial Cognition*.

References

- [1] J. Adámek, H. Herrlich, and G. E. Strecker. *Abstract and Concrete Categories*. Wiley Interscience, 1990.
- [2] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.
- [3] B. Bennett. *Logical Representations for Automated Reasoning about Spatial Relationships*. PhD thesis, School of Computer Studies, The University of Leeds, 1997.

- [4] B. Bennett, A. Isli, and A. G. Cohn. When does a composition table provide a complete and tractable proof procedure for a relational constraint language? In *Proceedings of the IJCAI97 Workshop on Spatial and Temporal Reasoning*, Nagoya, Japan, 1997.
- [5] A. G. Cohn, B. Bennett, J. M. Gooday, and N. Gotts. RCC: A calculus for region based qualitative spatial reasoning. *GeoInformatica*, 1:275–316, 1997.
- [6] I. Düntsch, H. Wang, and S. McCloskey. Relation algebras in qualitative spatial reasoning. *Fundamenta Informaticae*, 39(3):229–249, 1999.
- [7] M. J. Egenhofer. Reasoning about binary topological relations. In O. Günther and H.-J. Schek, editors, *Proceedings of the Second Symposium on Large Spatial Databases, SSD'91 (Zürich, Switzerland)*, Lecture Notes in Computer Science 525, pages 143–160. Springer, 1991.
- [8] M. J. Egenhofer and R. D. Franzosa. Point set topological relations. *International Journal of Geographical Information Systems*, 5:161–174, 1991.
- [9] A. U. Frank. Qualitative spatial reasoning with cardinal directions. In H. Kaindl, editor, *Proceedings of the Seventh Austrian Conference on Artificial Intelligence*, Informatik-Fachberichte 287, pages 157–167. Springer, 1991.
- [10] C. Freksa. Using orientation information for qualitative spatial reasoning. In A. U. Frank, I. Campari, and U. Formentini, editors, *Spatio-Temporal Reasoning*, Lecture Notes in Computer Science 639, pages 162–178. Springer, 1992.
- [11] A. Gerevini and J. Renz. Combining topological and qualitative size constraints for spatial reasoning. In *Proceedings of the 4th International Conference on Principles and Practice of Constraint Programming*, pages 220–234. Springer, 1998.
- [12] G. Ligozat. Weak representations of interval algebras. In *AAAI*, pages 715–720, 1990.
- [13] G. Ligozat. Categorical methods in qualitative reasoning: The case for weak representations. In A. G. Cohn and D. M. Mark, editors, *Spatial Information Theory, International Conference, COSIT 2005, Ellicottville, NY, USA, September 14-18, 2005, Proceedings*, Lecture Notes in Computer Science 3693, pages 265–282. Springer, 2005.
- [14] G. Ligozat. Qualitative triangulation for spatial reasoning. In A. U. Frank and I. Campari, editors, *Spatial Information Theory: A Theoretical Basis for GIS, International Conference COSIT '93, Marciana Marina, Elba Island, Italy, September 19-22, 1993, Proceedings*, Lecture Notes in Computer Science 716, pages 54–68. Springer, 1993.
- [15] G. Ligozat and J. Renz. What is a qualitative calculus? A general framework. In C. Zhang, H. W. Guesgen, and W.-K. Yeap, editors, *PRICAI 2004: Trends in Artificial Intelligence*, Lecture Notes in Computer Science 3157, pages 53–64. Springer, 2004.
- [16] S. Mac Lane. *Categories for the Working Mathematician*. Springer, 1997.
- [17] R. Maddux. Some varieties containing relation algebras. *Transactions of the American Mathematical Society*, 272(2):501–526, 1982.
- [18] D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In B. Nebel, W. Swartout, and C. Rich, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference (KR-92)*, pages 165–176. Morgan Kaufmann, 1992.
- [19] J. Renz and G. Ligozat. Weak composition for qualitative spatial and temporal reasoning. In P. van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005*, Lecture Notes in Computer Science 3709, pages 534–548. Springer, 2005.
- [20] M. B. Vilain. A system for reasoning about time. In D. L. Waltz, editor, *Proceedings of the National Conference on Artificial Intelligence. Pittsburgh, PA, August 18-20, 1982*, pages 197–201. AAAI Press, 1982.

Structural Approaches for PTIME Solvability of CSPs in Qualitative Reasoning

Alexander Scivos

Institut für Informatik, Albert-Ludwigs-Universität Freiburg
Georges-Köhler-Allee 52, 79110 Freiburg, Germany
scivos@informatik.uni-freiburg.de

Abstract. Automatic reasoning tasks are often solved with qualitative calculi based on a non-associative algebra of relations between basic entities. Such entities can be points, employees, program modules, regions, or line segments. In many cases, it is an NP-hard problem to decide consistency for a given constraint system of relations (constraint satisfaction problem, CSP).

Established attempts of reducing the complexity to PTIME reduce the set of relations that are allowed. However, such a modification is not always appropriate. In many applications, e. g. mereology or intra-company communication analysis, additional information on the structure of the solution can be used. In this paper, structure-based approaches of ensuring polynomial-time decidability of CSPs over the full set of relations for such applications with roles and hierarchies are motivated and explained. They work for many calculi.

For dependency networks, a reasoning technique based on scenario types is introduced along with a proof for its tractability.

1 Introduction

Automatic reasoning is an essential task in many applications such as robotics, geographical information systems (GIS), natural language understanding, network design, logistics and planning tasks, expert systems, and automatic program verification. In many of these cases, no detailed quantitative information is available or explicable. For example, visual images display only the relative alignment of objects, text descriptions often contain qualitative descriptions such as “A is inside B”, and causal dependencies can only be described in terms of qualitative information (“B depends on A”).

In such cases qualitative approaches that define formal representations of everyday descriptions are used as an effective way of concluding implications of the given information [4]. Triggered by Allen’s interval calculus IA [1], a series of qualitative temporal and spatial calculi have been successfully proposed and analyzed in the recent decades. Some prominent examples are calculi for qualitative directions [5, 9], point and interval algebras in linear [23], branching [2, 17] and arbitrary time structures [3], the RCC-8 and RCC-5 calculi for reasoning about topological relations [7, 18], and a calculus for dependencies [16]. They share the formal framework of a binary non-associative algebra of relations [10]. It consists of variables for entities or objects and a fixed set of binary relation labels that are used to designate constraints on the possible relations for each pair of objects. In order to specify that different basic relations are possible, all

labels for unions of such basic relations are contained in the set of relation labels. If no restriction is known on the relation between two objects, the universal relation label (\top) is used. All other relation labels stand for restricting relations. Conclusions about the spatial, temporal, causal, or structural relations are drawn by formal rules for converse relations, composition, and intersection of relations [12, 4, 14].

In all these cases, a problem instance is given as a constraint system, i. e. a finite set of variables and relational constraints between them. Calculi are typically used to find sound conclusions. This problem can be polynomially reduced to the problem of checking if a given constraint system is consistent (constraint satisfaction problem, CSP).

1.1 Related Work

For most of the calculi mentioned above, the general problem with all unions of relations is proven to be NP-hard [16]. As NP-hard problems are considered too difficult in practice, several methods have been developed to restrict the problem to a subclass of problems that is in PTIME: tractable subclasses of relations, coarsening the relations, and tree decomposition.

In the first approach the problem is restricted to those with a smaller set of relation labels. For example, for IA and for the RCC-5 and RCC-8 calculi, all tractable subclasses are known [15, 19, 8].

However, there are calculi, e. g. Freksa's double-cross calculus, for which the CSP is NP-hard even over basic relations [21]. Hence, the first approach does not work. Instead of working with this calculus, a calculus with a coarser set of basic relations is chosen for which the CSP is PTIME.

The third method, called tree decomposition, can be applied when the restricting relations are scarce. Let the graph (V, E) for a constraint system be the set of its variables V with edges between those pairs that have a restricting relation label (i. e. not \top). The CSP can be answered in PTIME if the resulting graph has a finite tree width k in the sense of [20] because then it is decomposable into small subproblems (of size $\leq k$) [6].

In all these cases, the PTIME decidability is achieved by restricting the relation labels that are allowed, at least on most of the edges. However, in some situations, none of these methods work, in other cases they involve loss of information.

1.2 The New Approach

Other modifications of standard CSPs will be defined in this paper. For them, PTIME algorithms for solving the CSP are presented. In many situations, a system, e. g. a communication network, has an underlying structure. The idea is to use such structural information of the models that are solutions of the CSP.

For instance, consider the information flow in a company. The dependency calculus for directed graphs [16] with employees as entities suits to determine if some information is passed on to a specific employee. Employees have roles like "boss", "secretary", which determine who informs whom, hence which relations hold formally (cf. Fig. 1). Roles can be organized in a hierarchy.

This observation is generalized. If the relations between some pairs in a solution are required to be the same, the complexity of the CSP might be reduced. Two general

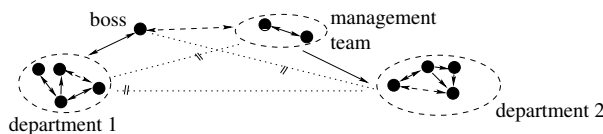


Fig. 1: Roles in a company determine the communication structure.

approaches will be presented: a modification involving only roles, and one with a hierarchical structure. The idea of the role approach is to assign a functional role to each entity. The relation between entities is then determined by their roles. The idea of the hierarchical approach is to define groups of objects, and groups of such groups, and to ensure that on a higher level individual differences of the entities within the group do not play a role. The conditions might seem quite limiting, and indeed they must be restrictive to reduce the computational complexity of the problem. But but they coincide with restrictive conditions frequently found in many fields of application.

For instance, consider mereological descriptions that determine if one part of the human body is contained in another one. Suppose two parts are contained in different organs. Then they are surely separated, no matter how exactly the parts of each organ are arranged. Organs thus correspond to functional roles: They determine the relations. Organs and sub-organs form a structural hierarchy.

Another reasoning task is to determine the locations of vehicles in complex transport systems. Let mobile trains and immobile locations be represented as entities in a point algebra for partial orders like [3]. Assume there are at most k different tracks with certain locations on them. At a fixed point in time, each train can be on one of the k tracks. Even if the order of the stops along a track and the alignment of the tracks are not completely specified, the relations of trains and locations on different tracks only depends on their current tracks (cf. Fig. 2). As will be shown in this paper, such structural conditions can be combined with “traditional” conditions like the actuality that all relations between trains on the same track are relations of linear orderings.

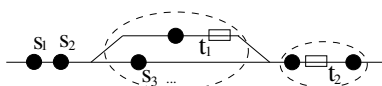


Fig. 2: The train example: Uncertainty only occurs within each oval.

The remainder of this paper is organized as follows: In section 2, the traditional formalism for constraint satisfaction problems is presented, and the two new approaches are introduced from a formal point of view. In section 3, an example of a modified CSP is presented with a proof sketch that it is solved in PTIME. Section 4 summarizes the results and suggests questions for further research.

2 Formalization

2.1 General Non-associative Reasoning Algebras

The basic concepts and ideas will be formalized in this chapter. First, recall the main important terms of qualitative reasoning.

As Ligozat and Renz [10] point out, almost all successful calculi are based on a formalism once described by Maddux [11].

Definition 1 (Non-associative Algebra). A non-associative algebra \mathcal{R} is a tuple $\mathcal{R} = (R, \cup, \cap, \perp, \top, ;, \checkmark, eq)$ such that:

1. $(R, \cup, \cap, \perp, \top)$ is a Boolean algebra.
2. eq is a constant, \checkmark a unary and $;$ a binary operation s. t., for any $a, b, c \in R$:

$(a\checkmark) = a$	$eq; a = a; eq = a$
$a; (b \cup c) = a; b \cup a; c$	$(a; b)\checkmark = b\checkmark; a\checkmark$
$(a \cup b)\checkmark = a\checkmark \cup b\checkmark$	$(a \cap b)\checkmark = a\checkmark \cap b\checkmark$
$(a; b) \cap c\checkmark = \perp$ if and only if $(b; c) \cap a\checkmark = \perp$	

A non-associative algebra is a *relation algebra* in the sense of Tarski [22] if it is associative (i.e. $(a; b); c = a; (b; c)$).

For qualitative reasoning, the elements $a, b, \dots \in R$ of the non-associative algebra are relation labels on which constraints can be defined. Semantically, they should represent relations. If $a \cap b = a$, we write $a \sqsubseteq b$. Recall that b is called atomic or basic if $x \sqsubseteq b$ implies $x \in \{\perp, b\}$. The set of basic relation labels is denoted by $B(\mathcal{R})$.

Definition 2 (Semantics). A *reasoning algebra* is a non-associative algebra together with a *representation* (U, ϕ) . Hereby, U is a set called universe and $\phi : R \rightarrow \mathcal{P}(U \times U)$ maps relation labels to relations and is a homomorphism of Boolean algebras. ϕ is defined on basic labels and extended to the boolean algebra in the natural way. Reversely, the *labeling function* $rel : U \times U \rightarrow B(\mathcal{R})$ maps a pair (u_1, u_2) to the basic relation for which $(u_1, u_2) \in \phi(rel(u_1, u_2))$ holds. Recall that a *constraint system* (or *constraint network*) $\mathcal{C} = (V, c)$ over a reasoning algebra consists of a set of variables V and a function $c : V \times V \rightarrow R$ that assigns a relation label to each pair of variables (written as constraints: $v c_{vw} w$). A constraint system where all labels are basic relations is called a *scenario*.

As mentioned above, it is an important question if such a constraint system is consistent. A constraint system is solved by interpreting the variables with values from the universe U in a way that is consistent with the constraints.

Definition 3 (Satisfiability). An *interpretation* $\theta : V \rightarrow U$ satisfies the constraint system $\mathcal{C} = (V, c)$ iff for all $x, y : (x^\theta, y^\theta) \in \phi(c_{xy})$ (i. e. $rel(x^\theta, y^\theta) \sqsubseteq c_{xy}$).

Each constraint system is an instance of the general constraint satisfaction problem (CSP) of the underlying reasoning algebra \mathcal{R} .

Input: A constraint system \mathcal{C} over \mathcal{R}

Question: Is \mathcal{C} satisfiable?

Formally, a problem can be seen as a pair $\mathfrak{P} = (I, S)$. Hereby, I is a class of input descriptions (instances) and S is the subclass of I containing the inputs with positive answer. The size of a problem instance in case of a CSP is the number of variables n .

2.2 CSPs with Structural Conditions

A structural condition can alter the problem class by imposing additional constraints on the solution. Solutions of the original CSP that do not fulfil the condition are rejected.

Definition 4 (Modified Problem). For a problem $\mathfrak{P} = (I, S)$ and a property p , the p -modification of \mathfrak{P} is the problem $\mathfrak{P}|_p = (I, S|_p)$, whereby $S|_p = \{s \in S \mid p \text{ holds of } s\}$.

Note that the allowed class of inputs is unchanged. Only the understanding of a solution is restricted. Different modifications will be investigated starting with strict preconditions, then gradually relaxing them.

The example from section 1 in which each employee has a functional role may illustrate such a condition. In a solution, only the roles of two employees should determine their relation (cf. Fig. 3). This is formally described by a role consistency property.

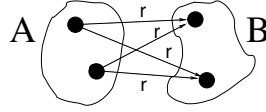


Fig. 3: Between a role A employee and a role B employee always the same relation r holds.

Definition 5 (Role Consistency). Let F be a finite set of functional roles. For a constraint system $\mathcal{C} = (V, c)$, a *role assignment* is a mapping $\rho : V \rightarrow F$.

A *role-consistent* interpretation is an interpretation θ of \mathcal{C} for which holds:

$$\rho(v_1) = \rho(v'_1), \rho(v_2) = \rho(v'_2) \Rightarrow \text{rel}(v_1^\theta, v_2^\theta) = \text{rel}((v'_1)^\theta, (v'_2)^\theta)$$

In other words, for each pair of roles i, j , there is some basic relation $r_{i,j}$ such that

$$\text{For all roles } i, j \in F : \rho(v) = i, \rho(w) = j \Rightarrow \text{rel}(v^\theta, w^\theta) = r_{i,j}.$$

$\text{CSP}|_\rho$ stands for the CSP where solutions are required to be role-consistent with respect to ρ .

Instead of classifying the entities, the relations between entities can be classified.

Definition 6 (Type Consistency). Let $T = \{1, \dots, k\}$ be a finite set of relation types. For a constraint system $\mathcal{C} = (V, c)$, a *type assignment* is a mapping $\tau : V \times V \rightarrow T$.

A *type-consistent* interpretation is an interpretation θ for which holds: For each type t there is a basic relation r_t such that $rel(v^\theta, w^\theta) = r_{\tau(v,w)}$ holds for all v, w .

$CSP|_\tau$ stands for the CSP where the solution is type-consistent with respect to τ .

Lemma 1. *Let \mathcal{R} be a reasoning algebra for which the consistency of a scenario is decidable in PTIME. If there are finitely many types used in a type assignment τ , the corresponding type-consistent $CSP|_\tau$ is in PTIME.*

Proof. For each type, a basic relation must be chosen. Each set of choices yields a scenario. Each scenario can be tested in PTIME. The number of scenario tests only depends on the number of types, not on the problem size (number of constraints). \square

Proposition 1. *Let \mathcal{R} be a reasoning algebra for which the consistency of a scenario is decidable in PTIME. If there are finitely many roles used in a role assignment ρ , the corresponding role-consistent $CSP|_\rho$ is in PTIME.*

Proof. A pair of roles defines a type of relation. A role-consistent solution thus corresponds with a type-consistent solution. If there are finitely many roles, there are finitely many types of pairs. The claim now follows from the previous Lemma 1. \square

Role consistency means that entities of the same role all have the same relation among each other. What happens if role consistency is required only for entities of different roles, but if relations between entities of the same role are not restricted? If no condition is imposed, the problem is generally NP-hard. However, additional conditions on the interplay of relations can still guarantee solvability in PTIME. One idea is to define a set D of relations that may occur between entities with different roles, and a set C of relations allowed between entities of the same role. The CSP modified by such a condition will be abbreviated by $CSP|_{\rho, C, D}$.

A subsystem of a constraint system containing only relations r of C is called a C -block, and its solution is called a C -model. In many reasoning algebras, choices of C and D are possible in a way that a collection of C -models can be composed to a $(C \cup D)$ -model in a way that any D -relation between entities of different C -blocks may hold, no matter which relations hold within the blocks.

Definition 7 (Composed Scenarios). Suppose $\mathcal{D} = (\{w_1, \dots, w_k\}, d)$ is a scenario and for $i \leq k$ let $\mathcal{C}_i = (\{v_{i1}, \dots, v_{ik_i}\}, c^{(i)})$ be scenarios of the same reasoning algebra. Then, the *composed scenario* (cf. Fig. 4) is the scenario

$$\mathcal{C} = \bigcup^{\mathcal{D}} \mathcal{C}_1 \dots \mathcal{C}_k := (\{v_{ij} | i \leq k, j \leq k_i\}, \{c'_{v_{ij}v_{i'j'}}\}) \text{ with } c'_{v_{ij}v_{i'j'}} = \begin{cases} d_{w_i w_{i'}} & (i \neq i') \\ c_{v_{ij}v_{i'j'}}^{(i)} & (\text{else}). \end{cases}$$

Definition 8 (Independence). For a reasoning algebra \mathcal{R} , let two subsets of basic relations be defined, a set $C \subseteq B(\mathcal{R})$ (“close” relations) and a set $D \subseteq B(\mathcal{R})$ (“distant” relations). Let C and D be called *independent* iff it holds:

For each D -block \mathcal{D} and collection $\mathcal{C}_1 \dots \mathcal{C}_k$ of C -blocks, the composed scenario $\mathcal{C} = \bigcup^{\mathcal{D}} \mathcal{C}_1 \dots \mathcal{C}_k$ is consistent iff all individual networks \mathcal{D} and \mathcal{C}_i are consistent.

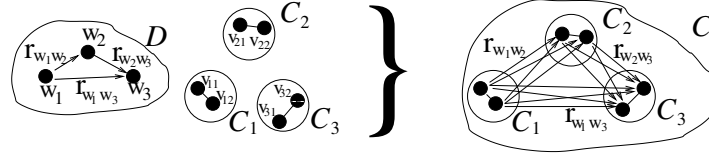


Fig. 4: Scenario composition: For the scenario \mathcal{C} composed of blocks \mathcal{C}_i , the scenario \mathcal{D} defines “exterior” relations across blocks, and all relations within each block remain unchanged.

Independence holds in many cases. In all prominent reasoning algebras, if θ solves a constraint system, its restriction to a subsystem solves the subsystem. The other direction depends on D and C , and often is easily verified.

Example. In the cardinal directions calculus for points in the plane [9] with the basic relations $B(\mathcal{R}) = \{eq, N, NE, E, SE, S, SW, W, NW\}$ (North, Northeast, etc.), the subsets $D = \{NE, SE, SW, NW\}$ and $C = B(\mathcal{R})$ are independent. (cf. Fig. 5)

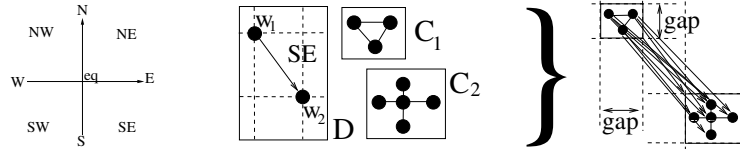


Fig. 5: Independence of $B(\mathcal{R})$ and $D = \{NE, SE, SW, NW\}$ in the cardinal direction calculus: A D -model can be “cut” at any entity w_i parallel to the cardinal directions. All points can be shifted further away from the cut without altering the relations, hence a gap large enough to contain a model of \mathcal{C}_i can be created. Its points hereby inherit the relations with other points of the D -model. This “shifting” technique can be applied in several cases to prove independence.

Lemma 2. Suppose $C \subseteq B(\mathcal{R})$ and $D \subseteq B(\mathcal{R})$ are independent and the CSP over the power set $\mathcal{P}(C)$ is tractable. Then $CSP|_{\rho, C, D}$ is in PTIME.

Proof. The idea is to group all entities of the same role to a “block”, and to solve the problem that occurs by substituting each block by a “top level” entity.

Let an instance $\mathcal{I} = (V, c, \rho, C, D)$ of $CSP|_{\rho, C, D}$ be given. For each role $i \leq k$, let $\rho^{-1}(i) =: V_i \subset V$ be the set of variables (entities) assigned to the role i . Due to the constraint that the solution contains C -relations within each V_i , for all $v, w \in V_i$ trim c_{vw} to $c_{vw} \cap C$. For the instance \mathcal{I} , there is a corresponding “top level instance” $\hat{\mathcal{I}} = (\{V_i | i \leq k\}, \hat{c})$, whereby

$$\hat{c}_{ij} := \bigcap_{v_i \in V_i, v_j \in V_j} c_{v_i v_j} \cap D$$

is the (least restrictive) common constraint of all relevant pairs. Deriving all these constraints has the complexity $O(n^2)$.

Suppose θ is a solution of \mathcal{I} . Then, for all $v_i \in V_i$ and $v_j \in V_j$, $rel(i, j) = rel(v_i^\theta, v_j^\theta) \sqsubseteq c_{v_i v_j}$, hence $rel(i, j) \sqsubseteq \bigcap c_{v_i v_j} = \hat{c}_{ij}$, i. e. $\hat{\mathcal{I}}$ is solvable. Hence, the solvability of $\hat{\mathcal{I}}$ is a necessary condition for the solvability of \mathcal{I} . But, is it sufficient?

Claim: \mathcal{I} is solvable if, and only if, $\hat{\mathcal{I}}$ and each block V_i are solvable.

Suppose $\hat{\mathcal{I}}$ and all blocks have a model. As D is independent from C , there is a composed model. This model fulfils all role constraints. The other direction is trivial. This condition can be checked in PTIME because $\hat{\mathcal{I}}$ is of size k (independent of the problem size n), and all constraints within a block are trimmed to a tractable class. \square

Remark. The proof still works if each block contains only constraints of tractable classes $C_i \subseteq R$. Generally, if the CSP for the blocks is of complexity $O(f(n))$, then the complexity of $CSP|_{\rho, C, D}$ is $O(f(m))$ where m is the maximal size of a block.

From the proof, the following algorithm is extracted:

```

For all  $i, j$  do calculate the trimmed and common relations  $\hat{c}_{ij}$ .
If the top instance  $\hat{\mathcal{I}}$  is inconsistent return “no”.
else for  $i \leq k$  do
  if  $\hat{C}_k$  is inconsistent return false.
  else return “yes”.

```

An application. Recall the example from section 1 in which trains follow different tracks. The description can be formalized in the point algebra [3] for partial orders PA_{po} with the basic relations $R = \{eq, <, >, ||\}$. PA_{po} in general is NP-hard.

The assigned tracks are functional roles. Assume that each track is a linear segment, but the exact location of the trains involves uncertainty. Then, the “close” relations C are the classical point algebra relations $B(PA_{lin})$. PA_{lin} is tractable [23]. Let D be the set of all PA_{po} relations except eq . Then C and D are independent. Hence, by Lemma 2 the consistency of a description can be checked in PTIME.

Slight modifications in the algorithm allow to handle all PA_{po} relations as “distant” relations D without losing the tractability. The modified algorithm works as follows:

```

For all solutions of the “top level CSP”:
  If in the solution for some  $i, j$ :  $rel(w_i^\theta, w_j^\theta) = “=”$ .
    (Temporarily) identify the corresponding roles  $i$  and  $j$ 
    If a it has a solution, return “yes”
    else reverse the identification of roles and loop.
After all top level solutions are tested, return “no”.

```

2.3 Polynomially Structured Hierarchies

In typical cases, D might be chosen as $C \setminus \{eq\}$, or $C = B(\mathcal{R})$. Then $D \subseteq C$. Thus, the composed model again is a C -model and can be a component in a higher level composed model, and so on (cf. Fig. 6). In this section, the following result will be shown: If such a structural hierarchy can be used, a modified CSP can be checked in PTIME even when C is an intractable set. It is sufficient that the set D is tractable, or that on each level

the number of components is less than a global limit k . A special case is when we have precise information about many relations, e. g. how the tracks in the train example (cf. Fig. 2) are related to each other. Only fine grain relations within each block are uncertain. The idea is to limit the size of such blocks to k . The structural design is motivated

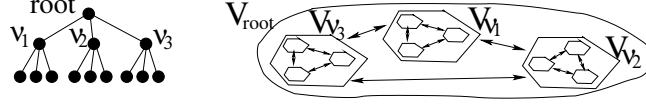


Fig. 6: A structure σ defines how a hierarchy of blocks corresponds with nodes of a tree.

by tasks from real-world applications. As mentioned above, hierarchical structures are often found in companies: Several employees form a work team, several teams form a department, several departments form a larger unit, and so on. The structure can be represented by a tree in which each node represents a unit, and children represent subunits.

The idea of the role assignment will now be generalized to a structure.

Definition 9. A structure σ for a set V of entities is a tree and an assignment of the entities $v \in V$ to complete branches of the tree. For fixed σ , for each node v of the tree let $V_v \subseteq V$ be the set of all entities mapped to a branch containing v (cf. Fig 6). Nodes are called *neighbored* if they are children of the same node.

Let an interpretation of a constraint system be called consistent with a structure σ iff σ reflects how it is composed of subcomponents.

Definition 10 (Structural Consistency). For a constraint system \mathcal{C} and a structure σ for its entities, an interpretation θ of \mathcal{C} is *structurally consistent with σ* iff if for each pair of neighbored nodes v, v' , there is a basic relation $r_{v,v'}$ such that

$$v \in V_v, w \in V_{v'} \Rightarrow \text{rel}(v^\theta, w^\theta) = r_{v,v'}.$$

Let $\text{CSP}|_\sigma$ be the CSP where the solution is structurally consistent with σ and let $\text{CSP}|_{\sigma, \mathcal{C}, D}$ be the CSP such that there is a solution with relations from D that are structurally consistent with σ and with interior relations in \mathcal{C} .

How can such a CSP be solved? If $D \subseteq \mathcal{C}$, \mathcal{C} and D independent, a *decomposition algorithm* similar to the one for $\text{CSP}|_{\rho, \mathcal{C}, D}$ works:

1. Trim the interior constraints: For all leaves v and $v, w \in V_v$ derive $\hat{c}_{vw} := c_{vw} \cap \mathcal{C}$. Check if all the interior blocks are solvable.
2. For all neighbored v', v'' derive $\hat{c}_{v',v''} := (\bigcap_{w' \in V_{v'}, w'' \in V_{v''}} c_{v',w''}) \cap D$.
3. Each v defines a block $\hat{C}_v = (\{v' | v' \text{ child of } v\}, \{\hat{c}_{v',v''} | v', v'' \text{ children of } v\})$. Solve each block individually. The CSP is solvable iff all blocks are solvable.

Without loss of generality, each interior node has at least two children. Then, there are $O(n)$ many nodes, hence $O(n)$ many blocks. The time-critical task is solving the blocks. If each block can be solved in polynomial time, we call the instance polynomially structured.

Definition 11 (Polynomial Structure). A constraint system is called *polynomially k-structured* if for each v :

- v is an interior node having at most k children, or
- v is an interior node and the set of common relations $\{\hat{c}_{v',v''} | v', v'' \text{ children of } v\}$ is contained in a tractable class, or
- v is a leaf and V_v contains at most k entities, or
- v is a leaf and the set of trimmed interior relations $\{\hat{c}_{vw} | v, w \in V_v\}$ is contained in a tractable class.

Note. This structure (which defines which nodes belong to which block) is known, but the relations might contain uncertainties. If the tree is of bounded degree k , no tractability assumption on C, D is necessary. Fig. 7 shows some examples.

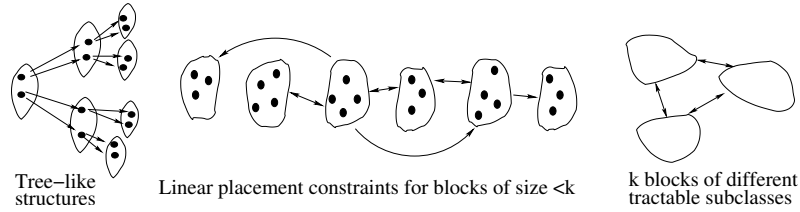


Fig. 7: Trees, chains, and compositions of tractable constraint systems are polynomially k -structured.

Theorem 1. Suppose $D \subseteq C \subseteq B(\mathcal{R})$, C and D are independent. Then $CSP|_{\sigma, C, D}$ for polynomially k -structured constraint systems is in PTIME.

Proof. Similar to the role consistency check, all inner constraints \hat{c}_{vw} and for each pair of neighbored nodes v, v' the common constraints $\hat{c}_{v,v'}$ must be calculated. $O(n^2)$ constraints must be touched once, as can easily be verified. $O(n)$ many polynomially solvable blocks are checked independently. \square

3 A Special Case: Tractability of a Structured CSPs

However, what if C and D are not independent, e. g. if $C = D = B(\mathcal{R})$ is needed? An approach tracking all dependencies is applicable in many cases. For example, consider the calculus $\mathcal{C}\mathcal{D}\mathcal{C}$ [16] with the following 5 basic relations over entities x, y, \dots in a partial order $\langle X, \leq \rangle$:

$x < y$ iff $x \leq y$ and not $y \leq x$.
 $x = y$ iff $x \leq y$ and $y \leq x$.
 $x > y$ iff $y \leq x$ and not $x \leq y$.
 $x \curlyvee y$ iff $\exists z z \leq y \wedge z \leq x$ and neither $x \leq y$ nor $y \leq x$ (then z is called a *witness*).
 $x \curlywedge y$ iff neither $\exists z z \leq y \wedge z \leq x$ nor $x \leq y$ nor $y \leq x$.

Theorem 2 (A Tractability Theorem). *For the dependency calculus $\mathcal{CD}\mathcal{C}$, $\text{CSP}|_\sigma$ for polynomially k -structured constraint systems are solvable in PTIME.*

Proof (sketch, here given for $k = 2$). The idea is, after deriving \hat{c}_{vw} and $\hat{c}_{v'v''}$ like before, a solution is searched from within each block to the top level (by induction).

As dependencies might restrict or enhance the satisfiability, different “types” of scenario are defined. For each block, a list of possible types is generated. A scenario type corresponds with properties that may cause problems when adding more blocks or corresponds with “problems” that must be solved by additional nodes. (In the following proof, most types are defined by restrictions on the allowed basic relations.) In the proof, each scenario type contains either only consistent or only inconsistent scenarios. \square

Definition 12 (Typology). Let $\mathcal{S}(\mathcal{R})$ be the class of all \mathcal{R} scenarios. A *typology* for \mathcal{R} is a finite set $\mathcal{T} = \{\mathcal{T}_i | i \leq t\}$ of *scenario types* partitioning $\mathcal{S}(\mathcal{R})$ such that for each $\mathcal{T}_i, \mathcal{T}_j \in \mathcal{T}$ and $b \in B(\mathcal{R})$ there is a type $\mathcal{T}_{b,i,j} \in \mathcal{T}$ such that for any $\mathcal{S}_i \in \mathcal{T}_i, \mathcal{S}_j \in \mathcal{T}_j$ the composition

$$\mathcal{S} = \bigcup_{(\{s_i, s_j\}, \{b\})} \mathcal{S}_i \mathcal{S}_j$$

of \mathcal{S}_i and \mathcal{S}_j with $(\{s_i, s_j\}, \{b\})$ is in $\mathcal{T}_{b,i,j}$. A type \mathcal{T} is called *allowed* in a constraint system $(V, \{c_{vw} | v, w\})$ iff \mathcal{T} contains a scenario $(V, \{b_{vw} | v, w\})$ with $b_{vw} \sqsubseteq c_{vw} (\forall v, w)$.

This means: The types of two blocks and the common basic relation b between them determine the type of the result. This relationship can be stored in a table.

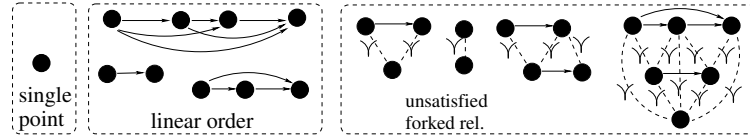


Fig. 8: Some examples for $\mathcal{CD}\mathcal{C}$ types. “linear order” contains all scenarios that only contain the relations $>, =, <$. “single point” only contains the trivial one point scenario. Both consist of consistent scenarios. “unsatisfied forked rel.” contains all scenarios where some \curlyvee relation has no witness. These types can be checked in PTIME.

Lemma 3 ($\mathcal{CD}\mathcal{C}$ Typing is PTIME). *$\mathcal{CD}\mathcal{C}$ has a finite typology of types such that for each scenario \mathcal{S} , its type can be determined in PTIME.*

Proof (sketch). An extensive case analysis yields a list of 8 types that can be determined by looking at the 3-element subnetworks. Fig. 8 shows some examples. \square

Lemma 4. *For any polynomially k -structured $\mathcal{C}\mathcal{D}\mathcal{C}$ constraint system, we can derive in PTIME the list of types allowed in it.*

A CSP $|_{\sigma}$ instance is satisfiable iff a satisfiable type \mathcal{T}_i is allowed in the top level constraint system.

This lemma can be proven by induction over the k -structure.

This allows the following algorithm:

1. **For each** v, w and v', v'' determine \hat{c}_{vw} and $\hat{c}_{v'v''}$.
2. **For each** interior block \mathcal{C}_i and scenario type \mathcal{T}_j check if \mathcal{T}_j is allowed in \mathcal{C}_i .
3. **For each** level:
 - (a) Determine the type list of the next exterior system,
 - (b) Determine (using the table) the type list of the next level, based on the present level type list and the exterior system type list.
4. **If** a satisfiable type is allowed on top level, **return** “yes” **else return** “no”

Fig. 9 shows some example dependencies. The effect of the typology is that some types directly restrict the relations possible in the next level. Some examples:

- If a relation other than $=$ occurs in a block, $=$ cannot occur as an exterior relation.
- If \succ occurs in a block, $>$ cannot occur as a exterior relation.
- If \succ and \succcurlyeq occur in some block below the top level, the system is not satisfiable.

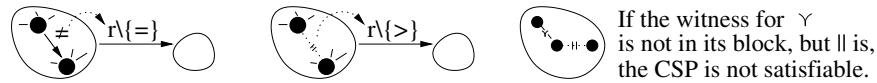


Fig. 9: Some examples for dependencies.

The proof can be generalized to other reasoning algebras in which dependencies between “interior” and “exterior” relations can occur. A precondition is that the types of one hierarchical level determine the types of the next level and that types of consistent scenarios do not contain inconsistent scenarios and vice versa. Hence, a similar result holds for all reasoning algebras in which dependencies are determined by a finite typology of types that are determined in PTIME.

4 Conclusion and Outlook

In order to identify new cases in which a CSP of a generally intractable reasoning algebra can be solved in PTIME, some structural conditions were investigated. The basic idea is to reduce complexity by grouping the entities of a constraint system into blocks. Only scenarios in which the basic relations across blocks is equal are considered as solutions. Three tractable cases were presented:

1. All relations within each block are required to be equal.
2. The set C of basic relations possible within each block and the set D of basic relations possible across blocks are independent.
3. A finite typology of scenarios allows to track dependencies and possible scenarios along a polynomial k -structured hierarchy.

Knowing a hierarchical structure condition of a problem might open the way to solvability in PTIME. These new conditions can be combined with traditional conditions, e. g. the premise that only constraints of a tractable subclass occur within some blocks.

These methods widen the range of problems that can be considered tractable. As general concepts, they are not per se restricted to specific reasoning algebras. For well-known reasoning algebras, it is advisable to investigate which subsets of their basic relations are independent of each other. For other subsets, the question remains if a finite scenario typology exists. First results suggest that many reasoning algebras have such a typology. Sufficient criteria might soon be proven.

Another open research task is the development of a formal logical language to talk and reason about scenario types. With such a framework, an automatic generation of a scenario typology from a reasoning algebra specification could hopefully be feasible.

References

- [1] J. F. Allen. Maintaining knowledge about temporal intervals. *Comm. ACM*, 26(11): 832–843, 1983.
- [2] F. Anger, P. Ladkin, and R. Rodriguez. Atomic temporal interval relations in branching time: Calculation and application. In *Actes 9th SPIE Conf. on Applic. of AI*, 1991.
- [3] M. Broxvall and P. Jonsson. Towards a complete classification of tractability in point algebras for nonlinear time. In *Proc. of CP-99*, pages 129–143, 1999.
- [4] A.G. Cohn. Qualitative spatial representation and reasoning techniques. In *KI-97: Advances in AI*, Brewka, G. and Habel, C. and Nebel, B (eds), LNAI, pages 1–30, 1997.
- [5] A. Frank. Qualitative Spatial Reasoning with Cardinal Directions. In *Proc. of the 7th Austrian Conf. on AI*, 1991.
- [6] R. Dechter, I. Meiri and J. Pearl. Tree decomposition with applications to constraint processing. *Proc. of the 8th National Conf. on Artificial Intelligence*, pages 10-16, 1990.
- [7] M. Egenhofer. Reasoning about binary topological relations. In: O. Günther and H.-J. Schek, editors, *Proceedings of the Second Symposium on Large Spatial Databases, SSD'91*, Lecture Notes in Computer Science 525, pages 143–160. Springer, 1991.
- [8] P. Jonsson and T. Drakengren. A complete classification of tractability in RCC-5 *J. Art. Int. Research*, 6:211–221, 1997.
- [9] G. Ligozat. Reasoning about cardinal directions. *J. of Vis. Lang. & Comp.*, 1(9):23–44. 1998.
- [10] G. Ligozat and J. Renz. What is a Qualitative Calculus? A general framework. In *Proceedings of PRICAI '04*, LNCS 3157, pages 53-64. Springer, 2004.
- [11] R. Maddux. Some varieties containing relation algebras, *Trans. AMS* 272 (1982), 501-526.

- [12] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Inform. Sci.*, 7:95-132,1974.
- [13] R. Moratz, J. Renz, D. Wolter. Qualitative spatial reasoning about line segments. *ECAI 2000*.
- [14] B. Nebel and A. Scivos. Formal properties of constraint calculi for qualitative spatial reasoning, *Künstliche Intelligenz*, 4(02): 14-18, 2002.
- [15] 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, 1995.
- [16] M. Ragni and A. Scivos. Dependency Calculus: Reasoning in a General Point Algebra. *Proc. of IJCAI-05*, pages 1575-1576, 2005.
- [17] M. Ragni, S. Wöfl. Branching Allen: Reasoning with intervals in branching time. Lecture Notes in Computer Science In Christian Freksa et al., *Spatial Cognition 2004 (SC'04)* pages 323 - 345.
- [18] Randell, D. and Cui, Z. and Cohn, A. A spatial logic based on regions and connection. *Proceedings KR-92*, pages 165-176, 1992.
- [19] J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the Region Connection Calculus. *AIJ*, 108(1-2):69-123, 1999.
- [20] N. Robertson, P. Seymour, Graph minors II: algorithmic aspects of tree-width, *J. Algorithms* 7: 309-322, 1986.
- [21] A. Scivos and B. Nebel. Double-Crossing: Decidability and computational complexity of a qualitative calculus for navigation, In *Spatial Information Theory: Found. of Geog. Info. Science (COSIT-2001)*, 2001.
- [22] A. Tarski. On the calculus of relations. *J. of Symb. Logic*, 6: 73-89,1941
- [23] M. Vilain, H. Kautz, and P. van Beek. Constraint propagation algorithms for temporal reasoning: A revised report. *Reasoning about Physical Systems*, pages 373-381, 1989.

From Qualitative to Discrete Constraint Networks

Jean-François Condotta, Dominique D’Almeida,
Christophe Lecoutre, and Lakhdar Saïs

CRIL-CNRS, Université d’Artois
rue de l’Université, 62307 Lens, France
{condotta,dalmeida,lecoutre,sais}@cril.univ-artois.fr

Abstract. In this paper, we present some preliminary results about the connections existing between qualitative and discrete constraint networks. We present a natural encoding of any qualitative network \mathcal{N} into a discrete one \mathcal{P} such that the constraints of \mathcal{N} become the variables of \mathcal{P} and the constraints of \mathcal{P} are defined by the weak composition table of the used qualitative algebra. We then introduce some properties about the (global) consistency of networks, circumscribing conditions under which the two models are equivalent. We also relate some domain filtering consistencies (such as generalized arc consistency) of discrete networks encoding qualitative ones with \circ -consistency, where \circ denotes the weak composition of the qualitative calculus.

1 Introduction

The need for reasoning about time and space arises in many areas of Artificial Intelligence, including computer vision, natural language understanding, geographic information systems (GIS), scheduling, planning, diagnosis and genetics. Numerous formalisms for representing and reasoning about time and space in a qualitative way have been proposed in the past two decades [1, 16, 13, 5, 15, 11, 4].

Those formalisms involve a finite set of basic relations denoting qualitative relationships between temporal or spatial entities. Intersection, overlapping, containment, precedence are examples of such qualitative relationships. For instance, in the field of qualitative reasoning about temporal data, there is a well known formalism called Allen’s calculus [1]. It is based on intervals of the rational line for representing temporal entities and thirteen basic relations between such intervals are used to represent the qualitative situations between temporal entities: an interval can follow another one, meet another one, and so on.

Typically, Qualitative Constraint Networks (QCNs) are used to express information on a spatial or temporal situation. Each constraint of a QCN represents a set of acceptable qualitative configurations between some temporal or spatial entities and is defined by a set of basic relations.

On the other hand, the discrete Constraint Satisfaction Problem (CSP) is at the heart of Constraint Programming. Its task is to determine the satisfiability of a Discrete Constraint Network (DCN), i.e. a network such that each variable takes its values in an associated discrete domain. For solving DCNs, tree search algorithms are commonly used. To limit their combinatorial explosion, various improvements have been

proposed. Such improvements mainly concern ordering heuristics, filtering techniques and conflict analysis, and can be conveniently classified as look-ahead and look-back schemes [8].

In this paper, we report on current work concerning the representation of qualitative networks by discrete ones. More particularly, we define and study a transformation that allows for translating a QCN into a DCN. We show that satisfiability (unlike unsatisfiability) is preserved by this transformation. Moreover, we study the links between local consistency concepts of qualitative and discrete models. The final objective of this work is to detect and import into the qualitative domain the most efficient inference and search methods of the discrete model.

This paper is organized as follows. After introducing some technical background about discrete and qualitative constraint networks, we introduce an encoding of qualitative networks into discrete ones while addressing the issue of satisfiability. Then, we relate local consistencies from the two qualitative and discrete paradigms. Finally, we conclude with some perspectives.

2 Background on Discrete Constraint Networks

Definition 1. A Discrete Constraint Network (DCN) \mathcal{P} is a triple (X, D, C) where:

- X is a finite set of variables;
- D is a mapping which associates to each variable $x \in X$ a finite set of values $D(x)$ called domain;
- C is a finite set of constraints such that each constraint $c \in C$ involves a subset of variables of X , called scope and denoted by $vars(c)$, and has an associated relation, denoted $rel(c)$, which contains the set of tuples allowed for the variables of its scope.

A solution to a discrete constraint network is an assignment of values to all the variables such that all the constraints are satisfied. A constraint network is said to be satisfiable or consistent iff it admits at least one solution. Two discrete constraint networks are equivalent iff they admit the same set of solutions.

Arc Consistency (AC) remains the central property of discrete constraint networks and establishing AC on a given network \mathcal{P} involves removing all values that are not arc-consistent.

Definition 2. Let $\mathcal{P} = (X, D, C)$ be a DCN. A pair (x, a) , with $x \in X$ and $a \in D(x)$, is arc-consistent iff $\forall c \in C \mid x \in vars(c)$, there exists a support of (x, a) in C , i.e. a tuple $t \in rel(c)$ such that $t[x] = a$ and $t[y] \in D(y) \forall y \in vars(c)$ ¹. \mathcal{P} is arc consistent iff $\forall x \in X$, $D(x) \neq \emptyset$ and $\forall a \in D(x)$, (x, a) is arc-consistent.

The definition above is given in the general case, that is to say for instances involving constraints of any arity. Then, one usually talks about Generalized Arc Consistency (GAC) (e.g. see [6]) or hyper-arc consistency (e.g. see [3]). We will say that an assignment of a value to each variable of a set $S \subseteq X$ of variables is consistent iff any

¹ $t[x]$ denotes the value assigned to x in t

constraint $c \in C$ only involving assigned variables of S (i.e. $\text{vars}(c) \subseteq S$) is satisfied. \mathcal{P} is said to be (i, j) -consistent iff any consistent assignment to i variables can be extended to a consistent assignment to j additional variables. Also, the k -consistency concept (with $k \geq 1$) is defined [9] as being equivalent to $(k - 1, 1)$ -consistency. Finally, a DCN is strong k -consistent iff it is j -consistent, for any j in $\{1, \dots, k\}$.

To solve a discrete constraint network, one can apply inference or search methods [8]. Usually, domains of variables are reduced by removing inconsistent values, i.e. values that can not occur in any solution. We can then compare the different states of a network during inference or search by focusing on domains as follows:

Definition 3. Let $\mathcal{P} = (X, D, C)$ and $\mathcal{P}' = (X, D', C)$ be two DCNs. $\mathcal{P}' \subseteq \mathcal{P}$ iff $\forall x \in X, D'(x) \subseteq D(x)$.

3 Background on Qualitative Calculi

3.1 Relations and Operations

A qualitative calculus involves a finite set B of binary² relations, called basic relations, defined on a domain D . The elements of D represent temporal or spatial entities. Each basic relation of B corresponds to a particular possible configuration between two temporal or spatial entities. The relations of B are jointly exhaustive and pairwise disjoint, which means that any pair of elements of D belongs to exactly one basic relation in B . Moreover, for each basic relation $B \in B$ there exists a basic relation of B , denoted by B^\sim , corresponding to the transposition of B . Moreover, we suppose that a particular relation of B is the identity relation on D , we denote this basic relation by Id . The set A is defined as the set of relations corresponding to all unions of the basic relations: $A = \{\bigcup E : E \subseteq B\}$. It is customary to represent an element $B_1 \cup \dots \cup B_m$ (with $B_i \in B$ for each i such that $1 \leq i \leq m$) of A by the set $\{B_1, \dots, B_m\}$ belonging to 2^B . Hence, we make no distinction between A and 2^B in the sequel.

As an example, consider the well known temporal qualitative formalism called Allen’s calculus [2]. It uses intervals of the rational line for representing temporal entities. Hence, D is the set $\{(x^-, x^+) \in \mathbb{Q} \times \mathbb{Q} : x^- < x^+\}$. The set of basic relations consists of a set of thirteen binary relations $B = \{eq, b, bi, m, mi, o, oi, s, si, d, di, f, fi\}$ corresponding to all possible configurations between two intervals. These basic relations are depicted in Figure 1. We have $\text{Id} = eq$.

As a set of subsets, A is equipped with the usual set-theoretic operations including intersection (\cap) and union (\cup). As a set of binary relations, it is also equipped with the operation of converse (\sim) and an operation of composition (\circ) sometimes called weak composition or qualitative composition. The converse of a relation R in A is the union of the transpositions of the basic relations contained in R . The composition $A \circ B$ of two basic relations A and B is the relation $R = \{C \in B \mid \exists x, y, z \in D, x A y, y B z \text{ and } x C z\}$. The composition $R \circ S$ of $R, S \in A$ is the relation $T = \bigcup_{A \in R, B \in S} \{A \circ B\}$. Computing the results of these various operations for relations of 2^B can be done efficiently by using tables giving the results of these operations for the basic relations of B . For instance,

² In this paper, we focus on binary relations but this work can be extended to n -ary relations with $n > 2$.

Relation	Symbol	Converse	Meaning
precedes	b	bi	
meets	m	mi	
overlaps	o	oi	
starts	s	si	
during	d	di	
finishes	f	fi	
equals	eq	eq	

Fig. 1: The basic relations of Allen's calculus.

consider the relations $R = \{eq, b, o, si\}$ and $S = \{d, f, s\}$ of Allen's calculus, we have $R \sim = \{eq, bi, oi, s\}$. The relation $R \circ S$ is $\{d, f, s, b, o, m, eq, si, oi\}$.

3.2 Qualitative Constraint Networks

A qualitative constraint network (QCN) is a pair composed of a set of variables and a set of constraints. The set of variables represents spatial or temporal entities of the system. A constraint consists of a set of acceptable basic relations (the possible configurations) between two variables. Formally, a QCN is defined in the following way:

Definition 4. A QCN is a pair $\mathcal{N} = (V, C)$ where $V = \{v_1, \dots, v_n\}$ is a finite set of n variables and C is a map that assigns to each pair (v_i, v_j) of $V \times V$ a set $C(v_i, v_j) \in 2^{\mathcal{B}}$ of basic relations. In the sequel, $C(v_i, v_j)$ will be also denoted by C_{ij} . C is such that $C_{ii} \subseteq \{\text{Id}\}$ and $C_{ij} = C_{ji}^{\sim}$ for all $v_i, v_j \in V$.

With regard to a QCN $\mathcal{N} = (V, C)$, we have the following definitions. A *solution* of \mathcal{N} is a map σ from V to \mathbb{D} such that $(\sigma(v_i), \sigma(v_j))$ satisfies C_{ij} for all $v_i, v_j \in V$. \mathcal{N} is consistent iff it admits a solution. A QCN $\mathcal{N}' = (V', C')$ is a *sub-QCN* of \mathcal{N} (denoted by $\mathcal{N}' \subseteq \mathcal{N}$) if and only if $V = V'$ and $C'_{ij} \subseteq C_{ij}$ for all $v_i, v_j \in V$. A QCN $\mathcal{N}' = (V', C')$ is *equivalent* to \mathcal{N} if and only if $V = V'$ and both networks \mathcal{N} and \mathcal{N}' have the same solutions. The *minimal* QCN of \mathcal{N} is the smallest (for \subseteq) sub-QCN of \mathcal{N} equivalent to \mathcal{N} . An *atomic* QCN is a QCN such that each C_{ij} contains exactly one basic relation. A *scenario* of \mathcal{N} is an atomic sub-QCN of \mathcal{N} .

Given a QCN \mathcal{N} , the main issue to be addressed is the consistency problem: decide whether or not \mathcal{N} admits (at least) a solution. Most of the algorithms used for solving this problem are based on a method which we call the \circ -closure method. The \circ -closure method is a constraint propagation method allowing to enforce the $(0, 3)$ -consistency of a QCN $\mathcal{N} = (V, C)$, which means that all restrictions of \mathcal{N} to 3-variables are consistent. The \circ -closure method consists in iteratively performing the following operation: $C_{ij} := C_{ij} \cap (C_{ik} \circ C_{kj})$, for all v_i, v_j, v_k of V , until a fix-point is reached. The QCN obtained in this way is a sub-QCN of \mathcal{N} which is equivalent to it, and such that $C_{ij} \subseteq C_{ik} \circ C_{kj}$, for all v_i, v_j, v_k of V .

This latter property is expressed by saying that this sub-network is \circ -closed (to simplify, in the sequel, we will assume that a \circ -closed QCN does not contain the empty relation associated with a constraint). When the QCN obtained in this way contains the empty relation as a constraint, we can assert that the initial QCN is not consistent. However, when it is not the case, we cannot (in the general case) infer the consistency of the network. Despite this, the \circ -closure method is the main constraint propagation method used for qualitative constraint networks.

4 Encoding Qualitative Networks into Discrete Ones

The idea of mapping qualitative networks into discrete ones is quite natural, but, to the best of our knowledge, it has not been formalized and studied in the general case (i.e. for any qualitative algebra). However, we can cite the work of Pham et al. [14] who propose such a transformation for the Interval Algebra (IA). More precisely, any IA network \mathcal{N} can be encoded into a discrete network \mathcal{P} as follows. First, each constraint of \mathcal{N} is mapped to a variable of \mathcal{P} whose domain corresponds to the atomic relations of the constraint (and, as a consequence, a subset of B). Second, each triple of constraints of \mathcal{N} is mapped to a ternary constraint of \mathcal{P} such that the associated relation contains all valid 3-tuples satisfying the weak composition.

In this section, we propose a more preservative encoding of qualitative networks into discrete ones. In our case, a QCN \mathcal{N} is transformed into a ternary DCN \mathcal{P} where the constraints of \mathcal{N} become the variables of \mathcal{P} and the constraints of \mathcal{P} are such that their associated relations are defined by the entire table of weak composition. More formally, we define such a transformation, denoted T_{DCN} , as follows:

Definition 5. Let $\mathcal{N} = (V, C)$ be a QCN. $T_{\text{DCN}}(\mathcal{N})$ is the DCN $\mathcal{P} = (X, D, C')$ defined by:

- for each pair of variables $v_i, v_j \in V$ with $0 < i \leq j \leq n$, X contains a variable x_{ij} . The domain of x_{ij} is defined by C_{ij} ;
- for each triple of variables $v_i, v_j, v_k \in V$ with $0 < i < k < j \leq n$, C' contains a ternary constraint C'_{ijk} involving the three variables x_{ij}, x_{ik}, x_{kj} and defined by $C'_{ijk} = TC$ with $TC = \{(a, b, c) \in B^3 : a \in b \circ c\}$.

Remark that the main difference between the approach that we describe below and the approach of [14] is that the ternary constraints of the discrete network are not reduced by weak composition. Hence, we remain closer to the initial qualitative networks.

Firstly, we can prove that this transformation is sound for the consistency problem:

Proposition 1. Let $\mathcal{N} = (V, C)$ be a QCN. If \mathcal{N} is consistent then $T_{\text{DCN}}(\mathcal{N})$ is consistent.

Proof. Let $\mathcal{N} = (V, C)$ be a QCN and $T_{\text{DCN}}(\mathcal{N}) = (X, D, C')$ be the DCN obtained from \mathcal{N} . If \mathcal{N} is consistent then there exists a consistent scenario $\mathcal{S} = (V, C'')$ of \mathcal{N} . As \mathcal{S} is consistent and atomic, \mathcal{S} is \circ -closed. Now, let us consider the assignment I of the variables X defined by $I(x_{ij}) = b_{ij}$ with $C''_{ij} = \{b_{ij}\}$ for all $x_{ij} \in X$. \mathcal{S} is a

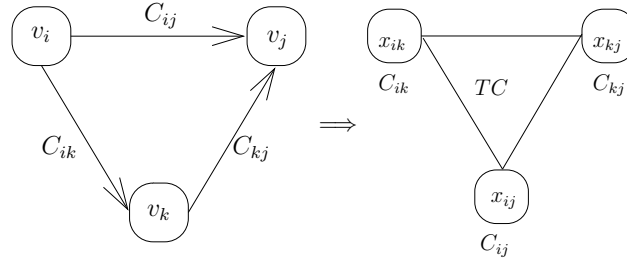


Fig. 2: The transformation T_{DCN} .

subnetwork of \mathcal{N} , hence $I(x_{ij}) \in D(x_{ij})$. Let $x_{ij}, x_{ik}, x_{kj} \in X$ with $0 < i < k < j \leq n$, $C''_{ij} \subseteq C''_{ik} \circ C''_{kj}$ as \mathcal{S} is \circ -closed. As a consequence, $(I(x_{ij}), I(x_{ik}), I(x_{kj})) \in TC$. Hence, $(I(x_{ij}), I(x_{ik}), I(x_{kj})) \in C'_{ijk}$. We can conclude that I is a solution of $T_{\text{DCN}}(\mathcal{N})$. So, $T_{\text{DCN}}(\mathcal{N})$ is consistent. \square

Unfortunately, the encoding is not complete for some qualitative calculi. As an illustration, let us consider the QCN \mathcal{N} depicted in Figure 3 which is defined in the cyclic interval algebra [10, 4]. This qualitative network \mathcal{N} is inconsistent whereas the discrete network $T_{\text{DCN}}(\mathcal{N})$ is consistent. A solution of this DCN is given by instantiating each variable by the value of its domain. Despite this, we have the following weaker property:

Proposition 2. *Let $\mathcal{N} = (V, C)$ be a QCN. If $T_{\text{DCN}}(\mathcal{N})$ is consistent then \mathcal{N} admits a \circ -closed scenario.*

Proof. Let $\mathcal{N} = (V, C)$ be a QCN and $\mathcal{P} = T_{\text{DCN}}(\mathcal{N}) = (X, D, C')$. If \mathcal{P} is consistent then there exists a consistent instantiation I for \mathcal{P} . Let $\mathcal{S} = (V, C'')$ be the QCN defined by: $C''_{ij} = \{I(x_{ij})\}$ for all $0 < i \leq j \leq n$, $C''_{ij} = (C''_{ji})^\sim$ for all $0 < j < i \leq n$. Remark that $C''_{ij} \neq \{\}$ for all $0 < j < i \leq n$. Let $i, j, k \in \{1, \dots, n\}$. Firstly, consider the case where i, j, k are distinct numbers. Suppose without any loss of generality that $i < k < j$. We have $(I(x_{ij}), I(x_{ik}), I(x_{kj})) \in TC$, as a consequence there exists $d_i, d_j, d_k \in D$ such that $d_i C''_{ij} d_j$, $d_i C''_{ik} d_k$, $d_k C''_{kj} d_j$, $d_j C''_{ji} d_i$, $d_k C''_{ki} d_i$ and $d_j C''_{jk} d_k$. Moreover we can remark that $d_i C''_{ii} d_i$, $d_j C''_{jj} d_j$ and $d_k C''_{kk} d_k$ since $C''_{ii} = C''_{jj} = C''_{kk} = \{\text{Id}\}$. From all this we know that \mathcal{S} is an atomic QCN and is consistent on all triples of distinct variables $v_i, v_j, v_k \in V$. It results that $C''_{ij} \subseteq C''_{ik} \circ C''_{kj}$ for all distinct variables $v_i, v_j, v_k \in V$. Now, consider $i, j, k \in \{1, \dots, n\}$ with $i = j$. We have $C''_{ij} = \{\text{Id}\}$. By definition of the weak composition and the converse we know that $\{\text{Id}\} \in b \circ b^\sim$ for all $b \in B$. It results that $C''_{ij} \subseteq C''_{ik} \circ C''_{kj}$ for all $k \in \{1, \dots, n\}$ since $C''_{kj} = C''_{ki} = (C''_{ik})^\sim$. Now suppose that $i, j, k \in \{1, \dots, n\}$ with $i = k$ (resp. $j = k$). We have $C''_{ij} \subseteq C''_{ik} \circ C''_{kj}$ since $C''_{ik} = \{\text{Id}\}$ and $C''_{kj} = C''_{ij}$ (resp. $C''_{kj} = \{\text{Id}\}$ and $C''_{ik} = C''_{ij}$). We can conclude that \mathcal{S} is \circ -closed. \square

A qualitative calculus will be said to be *nice* iff it satisfies the following property: a scenario is consistent if and only if it is \circ -closed. In fact, many qualitative calculi are nice, and in particular the well known Allen's calculus. From Propositions 1 and 2, we can establish the following property (whose proof is immediate):

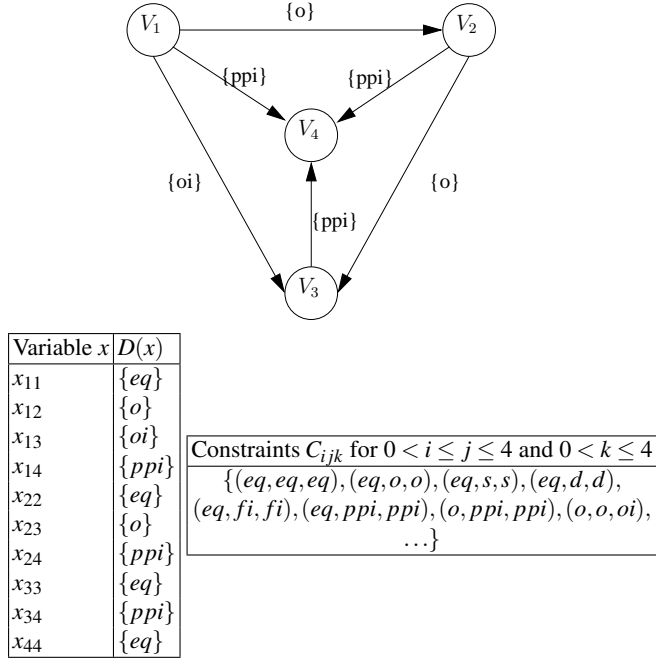


Fig. 3: A QCN \mathcal{N} of the cyclic interval algebra. Although \mathcal{N} is inconsistent, $T_{DCN}(\mathcal{N}) = (X, D, C)$ is consistent.

Proposition 3. *Let \mathcal{N} be a QCN defined in a nice qualitative calculus. \mathcal{N} is consistent iff $T_{DCN}(\mathcal{N})$ is consistent.*

We can also show that the transformation T_{DCN} preserves minimality and equivalence.

Proposition 4. *Let \mathcal{N} be a QCN. \mathcal{N} is minimal iff $T_{DCN}(\mathcal{N})$ is minimal.*

Proposition 5. *Let \mathcal{N} and \mathcal{N}' be two QCNs. If \mathcal{N} and \mathcal{N}' are equivalent then $T_{DCN}(\mathcal{N})$ and $T_{DCN}(\mathcal{N}')$ are equivalent.*

To close this section, we define the converse transformation of T_{DCN} , namely the transformation T_{QCN} .

Definition 6. Let $\mathcal{N} = (V, C)$ be a QCN and $\mathcal{P} = (X, D, C')$ be a DCN such that $\mathcal{P} \subseteq T_{DCN}(\mathcal{N})$. $T_{QCN}(\mathcal{P})$ is the QCN (V, C'') defined by $C''_{ij} = D(x_{ij})$ and $C''_{ji} = (C'_{ij})^\sim$ for all $0 < i \leq j \leq n$.

We have the following properties :

Proposition 6. *Let \mathcal{N} be a QCN.*

- (a) $\mathcal{N} = T_{QCN}(T_{DCN}(\mathcal{N}))$;
- (b) if $\mathcal{P} \subseteq T_{DCN}(\mathcal{N})$ then $T_{QCN}(\mathcal{P}) \subseteq \mathcal{N}$;
- (c) if $\mathcal{N} \subseteq \mathcal{N}'$ then $T_{DCN}(\mathcal{N}) \subseteq T_{DCN}(\mathcal{N}')$.

□

5 Equivalence between Local Consistencies

In this section we study the relationships of qualitative and discrete constraint networks in terms of (local) consistencies.

Proposition 7. *Let \mathcal{N} be a \circ -closed QCN. \mathcal{N} and $T_{\text{DCN}}(\mathcal{N})$ are $(0,3)$ -consistent.*

Proof. For the first claim, let $\mathcal{N} = (V, C)$ be a \circ -closed QCN. We know that $C_{ij} \subseteq C_{ik} \circ C_{kj}$ for all $0 < i, j, k \leq n$. There exists $b_{ij} \in C_{ij}$, $b_{ik} \in C_{ik}$ and $b_{kj} \in C_{kj}$ such that $b_{ij} \in b_{ik} \circ b_{kj}$. By definition of the weak composition, there exist $y_i, y_j, y_k \in D$ such that $y_i b_{ij} y_j$, $y_i b_{ik} y_k$ and $y_k b_{kj} y_j$. Moreover, by definition of QCNs we know that $b_{ij} \in C_{ji}$, $b_{ik} \in C_{ki}$ and $b_{kj} \in C_{jk}$. Hence, by definition of the inverse we have: $y_j b_{ij} y_i$, $y_k b_{ik} y_i$ and $y_j b_{kj} y_k$. Moreover $y_i C_{ii} y_i$, $y_j C_{jj} y_j$ and $y_k C_{kk} y_k$ since $C_{ii} = C_{jj} = C_{kk} = \{\text{Id}\}$. It results that the restriction of \mathcal{N} on v_i, v_j, v_k is consistent for all $0 < i, j, k \leq n$. We can conclude that \mathcal{N} is $(0,3)$ -consistent.

For the second claim let $\mathcal{P} = T_{\text{DCN}}(\mathcal{N}) = (X, D, C')$. Consider three variables x_{ij} , $x_{ik}, x_{kj} \in X$ with $0 < i < k < j \leq n$ (we consider these triples of variables since there are no constraint on other triples of variables). We have $C_{ij} \subseteq C_{ik} \circ C_{kj}$. As a consequence, there exists $b_{ij} \in C_{ij}$, $b_{ik} \in C_{ik}$ and $b_{kj} \in C_{kj}$ such that $b_{ij} \in b_{ik} \circ b_{kj}$. We have $b_{ij} \in D(x_{ij})$, $b_{ik} \in D(x_{ik})$, $b_{kj} \in D(x_{kj})$ and $(b_{ij}, b_{ik}, b_{kj}) \in C'_{ijk}$. We can conclude that \mathcal{P} is $(0,3)$ -consistent. \square

Moreover, we have the following properties.

Proposition 8. *Let \mathcal{N} be a \circ -closed QCN. $T_{\text{DCN}}(\mathcal{N})$ is strongly 3-consistent.*

Proof. Let $\mathcal{P} = T_{\text{DCN}}(\mathcal{N})$ where $\mathcal{N} = (V, C)$ is a \circ -closed QCN. From the fact that each domain of \mathcal{P} is not empty and each constraint is a ternary constraint we can assert that \mathcal{P} is $(0,1)$ -consistent and $(1,1)$ -consistent. Now, let us prove that \mathcal{P} is also $(2,1)$ -consistent. Let us consider three variables $x_{ij}, x_{ik}, x_{kj} \in X$ with $0 < i < k < j \leq n$ (we just consider triples of variables corresponding to the scope of a constraint). Let I be a partial consistent assignment on x_{ij} and x_{ik} . We know that $I(x_{ij}) \in C_{ij}$ and $I(x_{ik}) \in C_{ik}$. Moreover, $C_{ij} \subseteq C_{ik} \circ C_{kj}$. It results that there exists $b_{kj} \in C_{kj}$ such that $I(x_{ij}) \in I(x_{ik}) \circ b_{kj}$. Hence, $(I(x_{ij}), I(x_{ik}), b_{kj}) \in TC$, and besides, $b_{kj} \in D(x_{kj})$. As a consequence, by defining $I(x_{kj})$ with b_{kj} we extend I in a partial consistent assignment to x_{kj} . In a similar way of reasoning, we can extend a partial consistent assignment on x_{ij} and x_{kj} to the variable x_{ik} and extend a partial consistent assignment on x_{ik} and x_{kj} to the variable x_{ij} . Hence \mathcal{P} is $(2,1)$ -consistent. We can conclude that \mathcal{P} is a strongly 3-consistent DCN. \square

Proposition 9. *Let \mathcal{N} be a \circ -closed QCN. $T_{\text{DCN}}(\mathcal{N})$ is generalized arc-consistent.*

Proof. Let $\mathcal{P} = T_{\text{DCN}}(\mathcal{N})$ where \mathcal{N} is a \circ -closed QCN. From Proposition 8, we know that \mathcal{P} is strongly 3-consistent. Since \mathcal{P} only involves ternary constraints, it results that \mathcal{P} is $(1,2)$ -consistent and also generalized arc-consistent. \square

A corollary of these propositions is that if \mathcal{N} is a \circ -closed atomic QCN then $T_{\text{DCN}}(\mathcal{N})$ is a consistent DCN.

Proposition 10. *Let \mathcal{N} be a QCN. If $T_{\text{DCN}}(\mathcal{N})$ is a generalized arc-consistent then \mathcal{N} is \circ -closed.*

Proof. Let $\mathcal{P} = T_{\text{DCN}}(\mathcal{N}) = (X, D, C')$ with $\mathcal{N} = (V, C)$ a QCN. Suppose that \mathcal{P} is generalized arc-consistent. Consider $0 < i, j, k \leq n$. Suppose that $i < k < j$ without any loss of generality. Let $b_{ij} \in C_{ij}$. We have $b_{ij} \in D(x_{ij})$. \mathcal{P} is generalized arc-consistent, it results that there exist $b_{ik} \in D(x_{ik})$ and $b_{kj} \in D(x_{kj})$ with $(b_{ij}, b_{ik}, b_{kj}) \in C'_{ijk}$. As $b_{ik} \in C_{ik}$ and $b_{kj} \in C_{kj}$ we have $b_{ij} \in C_{ik} \circ C_{kj}$. Hence, $C_{ij} \subseteq C_{ik} \circ C_{kj}$. From this we also have $C_{ij}^{\sim} \subseteq (C_{ik} \circ C_{kj})^{\sim}$. Hence, $C_{ji} \subseteq C_{jk} \circ C_{ki}$. Now let $b_{ik} \in C_{ik}$. We have $b_{ik} \in D(x_{ik})$. \mathcal{P} is generalized arc-consistent, it results that there exist $b_{ij} \in D(x_{ij})$ and $b_{kj} \in D(x_{kj})$ with $(b_{ij}, b_{ik}, b_{kj}) \in C'_{ijk}$. From the definition of the weak composition, since $(b_{ij}, b_{ik}, b_{kj}) \in TC$ we can assert that $(b_{ik}, b_{ij}, b_{kj}^{\sim}) \in TC$. As $b_{ij} \in C_{ij}$ and $b_{kj}^{\sim} \in C_{jk}$ we have $b_{ik} \in C_{ij} \circ C_{jk}$. Hence, $C_{ik} \subseteq C_{ij} \circ C_{jk}$. From this we also have $C_{ik}^{\sim} \subseteq (C_{ij} \circ C_{jk})^{\sim}$. Hence, $C_{ki} \subseteq C_{kj} \circ C_{ji}$. With a similar line of reasoning we can prove that $C_{kj} \subseteq C_{ki} \circ C_{ij}$ and $C_{jk} \subseteq C_{ji} \circ C_{ik}$. Now suppose that $i = j$. We have $C_{ij} = C_{ji} = \{\text{Id}\}$. Moreover we know that $\{\text{Id}\} \subseteq b \circ b^{\sim}$ for all $b \in B$. It results that $C_{ij} \subseteq C_{ik} \circ C_{kj}$ and $C_{ji} \subseteq C_{jk} \circ C_{ki}$. Moreover it is easy to see that $C_{ik} \subseteq C_{ij} \circ C_{jk}$, $C_{ki} \subseteq C_{kj} \circ C_{ji}$, $C_{jk} \subseteq C_{ji} \circ C_{ik}$ and $C_{kj} \subseteq C_{ki} \circ C_{ij}$. We obtain the same result with $i = k$ or $k = j$. Finally we can assert that \mathcal{N} is a \circ -closed QCN. \square

As a consequence, a way to obtain the \circ -closure of a QCN is to transform it into a DCN via T_{DCN} . Indeed, we can then apply a GAC algorithm and transform the obtained DCN into a QCN via T_{QCN} (see Figure 4).

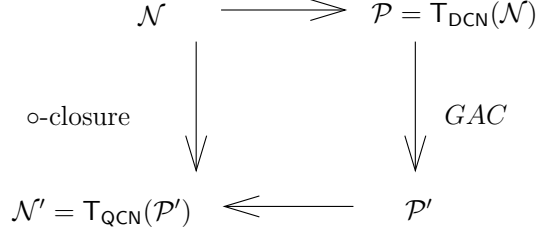


Fig. 4: The \circ -closure through the DCNs.

6 Future Work and Conclusions

Abscon [12] and QAT (Qualitative Algebra Toolkit) [7] are two JAVA constraint programming libraries developed at *CRIL-CNRS*. The first one is dedicated to discrete constraint networks. It can solve instances of any arity and implements state-of-the-art generic filtering (constraint propagation) and search algorithms. The second one is specialized in qualitative constraint networks. It aims to provide open and generic tools for defining and manipulating qualitative algebras and qualitative networks based on

these algebras. QAT also provides several methods to tackle the main centers of interest when dealing with qualitative constraint networks, mainly the consistency problem, the problem of finding one or all solutions, and the minimal network problem.

Currently, using these libraries, we are studying the interest of mapping qualitative networks into discrete ones. One of our ultimate objective is to detect which (inference or search) methods from the discrete CSP community could be efficiently specialized to the qualitative algebras. For example, we project to experimentally determine whether exploiting GAC could be an efficient alternative \circ -closure for qualitative constraints. Another current line of research is the study of SAT encodings for QCNs.

Acknowledgments

This paper has been supported by the CNRS, the “Planevo” project and the “IUT de Lens”.

References

- [1] J. F. Allen. An interval-based representation of temporal knowledge. In *Proceedings of IJCAI'81*, pages 221–226, 1981.
- [2] J. F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, 1983.
- [3] K. Apt. *Principles of Constraint Programming*. Cambridge University Press, 2003.
- [4] P. Balbiani and A. Osmani. A model for reasoning about topologic relations between cyclic intervals. In *Proceedings of KR'00*, pages 378–385, 2000.
- [5] P. Balbiani, J. Condotta, and L. Fariñas del Cerro. A model for reasoning about bidimensional temporal relations. In *Proceedings of KR'98*, pages 124–130, 1998.
- [6] C. Bessière. Constraint propagation. Technical report, LIRMM, Montpellier, 2006.
- [7] J. Condotta, G. Ligozat, and M. Saade. A generic toolkit for n-ary qualitative temporal and spatial calculi. In *Proceedings of TIME'06*, to appear, 2006.
- [8] R. Dechter. *Constraint processing*. Morgan Kaufmann, 2003.
- [9] E. C. Freuder. A sufficient condition for backtrack-free search. *Journal of the ACM*, 29(1):24–32, 1982.
- [10] K. Hornsby and M. E. and P.J. Hayes. Modeling cyclic change. In *Proceedings of REIS'99*, pages 98–109, 2006.
- [11] A. Isli and A. Cohn. A new approach to cyclic ordering of 2D orientations using ternary relation algebras. *Artificial Intelligence*, 122(1–2):137–187, 2000.
- [12] C. Lecoutre, F. Boussemart, and F. Hemery. Abscon 2005. In *Proceedings of CPAI'05*, volume II, pages 67–72, 2005.
- [13] G. Ligozat. Reasoning about cardinal directions. *Journal of Visual Languages and Computing*, 1(9):23–44, 1998.
- [14] D. N. Pham, J. Thornton, and A. Sattar. Modelling and solving temporal reasoning as propositional satisfiability. In *Proceedings of the workshop on Modelling and Reformulating Constraint Satisfaction Problems held with CP'05*, pages 117–131, 2005.

- [15] A. Pujari, G. Kumari, and A. Sattar. Indu: An interval and duration network. In *Proceedings of the Australian Joint Conference on Artificial Intelligence*, pages 291–303, 1999.
- [16] D. Randell, Z.Cui, and A. Cohn. A spatial logic based on regions and connection. In *Proceedings of KR’92*, pages 165–176, 1992.

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
jochen.renz@anu.edu.au

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 achievement in this direction was Renz’s refinement algorithm which provides a heuristic for proving tractability of given subsets of relations. In this paper we modify and 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, our procedure automatically identified all maximal tractable subsets. Using our procedure, other researchers and practitioners can automatically develop efficient 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. When giving route descriptions or describing objects or situations, spatial information of one form or another is usually always contained in the description. While it is common in engineering and other disciplines to represent spatial information in a quantitative way using, for example, coordinate systems, human communication mostly uses a qualitative description which specifies qualitative relationships between spatial entities. A description like “The book is *on* the desk *to the left of* the computer” is much more natural than giving a coordinate description of the location of the book. A qualitative representation is characterised by having a finite and usually small number of different relationships. Even though a qualitative description is less exact than a quantitative description, this is often an advantage, as it gives the possibility to give only as much details as are necessary to identify, e.g., a spatial entity. If more details are required, additional or finer relationships between spatial entities can be specified [2].

In the area of qualitative spatial representation and reasoning, it is distinguished between different aspects of space, such as direction, topology, distance, size, or shape.

For temporal information we can have similar aspects, although they might have different names. For all these aspects, qualitative relationships can then be defined on a particular level of granularity.

When developing a qualitative spatial or temporal calculus [7], 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 n -ary base relations (we assume in the following binary relations), i.e., between any n spatial entities of our domain, exactly one of the base relations holds. 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 our set of relations [5]. Given a set of such constraints we can then formulate different reasoning problems such as deriving unknown relations, eliminating impossible 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, but in most cases it is NP-hard.

In the past years much research effort has been put into identifying subsets of a given set of relations for which the consistency problem can be decided in polynomial time, so-called tractable subsets (e.g. [1, 8, 6]). Of particular interest are maximal tractable subsets which form the boundary between tractable and NP-hard subsets [10, 14, 4]. If the given spatial knowledge can be restricted to relations of these sets, then reasoning is tractable. But the main advantage of tractable subset is that they can be used to considerably speed up the time for deciding instances of NP-hard problems [9, 15]. As shown by Renz and Nebel [15], almost all instances of the NP-hard consistency problem for RCC8 [11], even those instances in the phase transition, which are commonly considered to be the hardest instances, could be solved very fast by combining different heuristics based on the maximal tractable subsets of RCC8. Therefore, a theoretical analysis of the reasoning problems and identifying tractable subsets is essential for obtaining 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, by which tractability of a subset can be proved simply by running an algorithm [12]. The refinement method introduced in Renz [12] 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 path-consistency 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., $T \subseteq S$. The algorithm then checks for all possible path-consistent triples of relations of \mathcal{S} whether the specified refinements can be made and also keeps track of and tests new refinements that are induced by the algorithm. Since

there are $2^{|\mathcal{B}|}$ 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 when combined with the base relations. 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, applying the refinement method to these two calculi and identifying large tractable subsets was relatively easy. But what if we have a new calculus where we don't have all the theoretical results already available?

In this paper, we modify and 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 any additional requirements or interactions. 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 identified all maximal tractable subsets of RCC8. But even if the resulting tractable subsets are not maximal, they can still give us efficient solutions to the NP-hard consistency problems if they are used as split sets in Renz and Nebel's reasoning algorithms [15].

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 compare the output of the procedure to the known results. Finally, in section 5 we give a discussion of our results.

2 Background

2.1 Qualitative Spatial and Temporal Reasoning

A qualitative spatial or temporal calculus 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 [7]. 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 $R \circ S = \{(a, c) \mid \exists b. (a, b) \in R \text{ and } (b, c) \in S\}$. Since we cannot deal with the tuples contained in the relations, we can in many cases only use weak composition which is defined as $R \diamond S = \{T \mid T \in \mathcal{B} : T \cap (R \circ S) \neq \emptyset\}$ [13]. 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 $\text{CSPSAT}(\mathcal{S})$ for spatial and temporal calculi, where \mathcal{S} is a subset of $2^{\mathcal{B}}$ is defined as follows [14]:

Instance: Given a set \mathcal{V} of n variables over a domain \mathcal{D} and a finite set Θ of binary 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' [9]. It has been shown that if large tractable subsets are known, in the ideal case maximal tractable subsets, i.e. tractable subsets that become NP-hard if extended by any other relation, then instances of the NP-hard consistency problem can be solved very efficiently [15].

A popular local consistency algorithm for solving the consistency problem is the algebraic closure algorithm, or a-closure algorithm. 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 fixed point occurs, Θ is called algebraically closed or a-closed. If the empty relation occurs during this process, Θ is inconsistent. If composition \circ is equivalent to weak composition \diamond , then the a-closure algorithm corresponds to the path-consistency algorithm [13]. In many cases a-closure decides consistency of atomic CSPs. This will be a requirement for applying our methods.

2.2 The Refinement Method

The refinement method [12] is a general method for proving tractability of $\text{CSPSAT}(\mathcal{S})$ for 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 $\text{CSPSAT}(\mathcal{T})$. Then the method checks whether it is possible to refine every constraint involving a relation in \mathcal{S} according to a particular refinement scheme 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 relation $S \in \mathcal{S}$ there is a relation $T_S \in \mathcal{T}$ with $T_S \subseteq S$,
2. every algebraically 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 $\text{CSPSAT}(\mathcal{T})$ for a set $\mathcal{T} \subseteq 2^{\mathcal{B}}$, and \mathcal{S} can be reduced by refinement to \mathcal{T} , then a-closure decides $\text{CSPSAT}(\mathcal{S})$. Therefore, it is sufficient for proving tractability of $\text{CSPSAT}(\mathcal{S})$ to show that \mathcal{S} can be reduced by refinement to a set \mathcal{T} for which a-closure decides $\text{CSPSAT}(\mathcal{T})$. Renz [12] developed an algorithm by which this can be shown. The algorithm uses a *refinement matrix* that manages the different refinements and which contains for every relation $S \in \mathcal{S}$ all specified refinements.

Definition 2 (Refinement Matrix).

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

Algorithm: CHECK-REFINEMENTS
Input: A set \mathcal{S} and a refinement matrix M of \mathcal{S} .
Output: `fail` if the refinements specified in M can make an algebraically closed triple 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
 $T = (R_{12}, R_{23}, R_{13})$ of relations over \mathcal{S} *do*
5. *for every* refinement $T' = (R'_{12}, R'_{23}, R'_{13})$ of T
with $\text{oldM}[R_{12}][R'_{12}] = \text{oldM}[R_{23}][R'_{23}] =$
 $\text{oldM}[R_{13}][R'_{13}] = \text{true}$ *do*
6. $T'' \leftarrow \text{ALGEBRAIC-CLOSURE}(T')$
7. *if* $T'' = (R''_{12}, R''_{23}, R''_{13})$ contains the empty
relation *then return* `fail`
8. *else do* $M[R_{12}][R''_{12}] \leftarrow \text{true},$
 $M[R_{23}][R''_{23}] \leftarrow \text{true},$
 $M[R_{13}][R''_{13}] \leftarrow \text{true}$
9. *if* $M = \text{oldM}$ *then* `changes` \leftarrow `false`
10. *return* `succeed`

Fig. 1: Algorithm CHECK-REFINEMENTS

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 algebraically closed triples of relations R_{12}, R_{23}, R_{13} of \mathcal{S} (step 4) and enforces a-closure (using a standard procedure ALGEBRAIC-CLOSURE) to every refinement $R'_{12}, R'_{23}, R'_{13}$ for which $M[R_{ij}][R'_{ij}] = \text{true}$ for all $i, j \in \{1, 2, 3\}, i < j$ (steps 5,6). If one of these refinements results in the empty relation, the algorithm returns `fail` (step 7). Otherwise, the resulting relations $R''_{12}, R''_{23}, R''_{13}$ are added to M by setting $M[R_{ij}][R''_{ij}] = \text{true}$ for all $i, j \in \{1, 2, 3\}, i < j$ (step 8). This is repeated until M has reached a fixed point (step 9), 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 if CHECK-REFINEMENTS returns `succeed` and the basic refinement matrix if CHECK-REFINEMENTS returns `fail`. If CHECK-REFINEMENTS returns `succeed` and GET-REFINEMENTS returns M' , we have pre-computed all possible refinements of every algebraically closed triple of variables as given in the refinement matrix M' . Thus, applying these refinements to an algebraically closed set of constraints can never result in an inconsistency when enforcing a-closure.

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

Corollary 1 ([12]). *Let $\mathcal{S}, \mathcal{T} \subseteq 2^B$ be two sets such that a-closure decides $\text{CSPSAT}(\mathcal{T})$, and let M be a refinement matrix of \mathcal{S} . Suppose $\text{GET-REFINEMENTS}(\mathcal{S}, M)$ returns*

M' . If for every $S \in \mathcal{S}$ there is a $T_S \in \mathcal{T}$ with $M'[S][T_S] = \text{true}$, then a -closure decides $\text{CSPSAT}(\mathcal{S})$.

3 A General Procedure for Identifying Tractable Subsets

In this section we present our new algorithms for identifying large tractable subsets of a spatial or temporal calculus. We assume that (1) we are given a set of jointly exhaustive and pairwise disjoint base relations \mathcal{B} , (2) that the weak compositions and the converses of the base relations are known, and also that (3) a -closure decides consistency for atomic CSPs over \mathcal{B} , i.e., for CSPs that contain exactly one base relation for each pair of variables. As a final condition, we assume that (4) the set $2^{\mathcal{B}}$ preserves complexity for the closure of subsets under intersection, converse and weak composition, i.e., the closure $\widehat{\mathcal{S}}$ of \mathcal{S} has the same complexity as \mathcal{S} .

As shown in [13], weak composition and a -closure are sufficient for the purpose of deciding consistency for most spatial and temporal calculi. In [13] it has also been shown how conditions (3) and (4) can be tested. 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 to be correct if condition (3) is satisfied. We will discuss this point in the last section of this paper. As shown in [13], condition (4) is a direct consequence of condition (3). As a consequence of conditions (3) and (4), we know that the closure of the set of base relations, the set $\widehat{\mathcal{B}}$, is also tractable.

3.1 Identifying Tractable Relations

Our refinement procedure consists of 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 $\widehat{\mathcal{B}} \cup \{R\}$ 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 which checks all refinements for a single relation R that is added to an existing set of relations \mathcal{T} , and FIND-TRACTABLE-SINGLES which calls CHECK-SINGLE-REFINEMENT 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 extend the refinement matrix in CHECK-SINGLE-REFINEMENT with a refinement for R . This refinement will propagate through to the other relations as they can all be decomposed into relations of $\widehat{\mathcal{B}} \cup \{R\}$. A similar algorithm GET-SINGLE-REFINEMENT returns the revised refinement matrix obtained by running CHECK-SINGLE-REFINEMENT.

Algorithm CHECK-SINGLE-REFINEMENT has three different outcomes for a relation R and a set \mathcal{T} , *succeed*, *fail*, and *unknown*. The most desirable one is *succeed* which means that there is an initial refinement of R to $R' \in \mathcal{T}$ such that the

CHECK-REFINEMENTS algorithm computes an updated refinement matrix by which all relations of the closure of $\widehat{\mathcal{T}} \cup \{R\}$ can be refined to a relation of $\widehat{\mathcal{T}}$ without changing consistency. If this is the case, and a-closure decides consistency for \mathcal{T} , then a-closure also decides consistency for the closure of $\widehat{\mathcal{T}} \cup \{R\}$. This is an immediate consequence of Corollary 1. If all possible refinements of R to relations of \mathcal{T} fail, CHECK-SINGLE-REFINEMENT returns *fail*. In this case the closure of $\widehat{\mathcal{T}} \cup \{R\}$ 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 shown using the refinement method and that there is some likelihood that the resulting set is NP-hard. For the purpose of this paper, where we analyse the power of refinements and try to identify large tractable subsets using refinements, we will therefore assume that for any relation R for which CHECK-SINGLE-REFINEMENT(R, \mathcal{T}) returns *fail* the set $\{R\} \cup \mathcal{T}$ is intractable.

A third possible outcome of CHECK-SINGLE-REFINEMENT 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 $\widehat{\mathcal{T}} \cup \{R\}$ to relations of $\widehat{\mathcal{T}}$. Increasing \mathcal{T} by some relations might lead to a successful refinement of R , so we cannot yet conclude whether R leads to a tractable set or not.

FIND-TRACTABLE-SINGLES calls CHECK-SINGLE-REFINEMENT for the different relations not contained in $\widehat{\mathcal{B}}$ while keeping track of their outcome. It is worth noting that if a relation R returns *success* and therefore leads to a tractable subset, then all relations in the closure of $\widehat{\mathcal{B}} \cup \{R\}$ also lead to a tractable subset and don't have to be tested using CHECK-SINGLE-REFINEMENT. We conclude this step with the following lemma.

Lemma 1. *Let \mathcal{B} be a set of base relations and \mathcal{T} and \mathcal{H} be the result of FIND-TRACTABLE-SINGLES(\mathcal{B}). If a-closure decides consistency for CSPSAT(\mathcal{B}), then all relations of \mathcal{T} are contained in tractable subsets of $2^{\mathcal{B}}$.*

3.2 Identifying Candidates for Tractable Subsets

Using the algorithm specified in the previous subsection, we can identify relations that must be contained in a tractable subset, but we don't know yet what the tractable subsets are and how many tractable subsets there are. 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 intractability when combined with each other. One possibility to find an answer to these questions would be to test all subsets of \mathcal{T} using the refinement method, but there are obviously too many such sets. Our next step will therefore be to find out which relations might lead to intractability when combined with the base relations. For this purpose we will assume that any of the relations of the set \mathcal{H} returned by FIND-TRACTABLE-SINGLES(\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 will be intractable as well. This is done by the algorithm FIND-INCOMPATIBLE-PAIRS in Figure 3. Therefore, whenever two relations of \mathcal{T} are incompatible, they must be contained in two different tractable subsets.

Algorithm: CHECK-SINGLE-REFINEMENT
Input: A single relation $R \in 2^{\mathcal{B}}$, a tractable set $\mathcal{T} \subseteq 2^{\mathcal{B}}$ which is closed under weak composition, converse and intersection, 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. $\text{good} \leftarrow \text{false}$
2. $\mathcal{S} \leftarrow \text{closure}(\{R\} \cup \mathcal{T})$
3. for every refinement R' of R with $R' \in \mathcal{T}$ do
4. $\text{newM} \leftarrow M$; $\text{newM}[R][R'] = \text{true}$
5. if $\text{CHECK-REFINEMENTS}(\text{newM}, \mathcal{S}) == \text{succeed}$ then
6. $M' \leftarrow \text{GET-REFINEMENTS}(\text{newM}, \mathcal{S})$
7. if for every $i \in \mathcal{S}$ there is a $j \in \mathcal{T}$ such that $M'[i][j] = \text{true}$ then return succeed
8. else $\text{good} \leftarrow \text{true}$
9. if $\text{good} == \text{true}$ return unknown
10. else return fail

Algorithm: FIND-TRACTABLE-SINGLES
Input: A set of base relations \mathcal{B}
Output: A set of relations $\mathcal{T} \subseteq 2^{\mathcal{B}}$ such that $\text{CSPSAT}(\{T\} \cup \mathcal{B})$ is tractable for every $T \in \mathcal{T}$, and a set of relations $\mathcal{H} \subseteq 2^{\mathcal{B}}$ each of which potentially leads to NP-hardness when combined with \mathcal{B} .

1. $\mathcal{T} \leftarrow \text{closure}(\mathcal{B})$; $\widehat{\mathcal{B}} \leftarrow \text{closure}(\mathcal{B})$; $\mathcal{H} = \emptyset$
2. $M[i][i] = \text{true}$ for all $i \in \mathcal{T}$ and $M[i][j] = \text{false}$ for all $i \neq j$
3. for every relation $R \notin \mathcal{T}$ do
4. $\text{result} \leftarrow \text{CHECK-SINGLE-REFINEMENT}(R, \widehat{\mathcal{B}}, M)$
5. if $\text{result} == \text{succeed}$ then $\mathcal{T} \leftarrow \mathcal{T} \cup \text{closure}(\{R\} \cup \mathcal{B})$
6. if $\text{result} == \text{fail}$ then $\mathcal{H} \leftarrow (\mathcal{H} \cup \{R\})$
7. return \mathcal{T} and \mathcal{H}

Fig. 2: Algorithms CHECK-SINGLE-REFINEMENT and FIND-TRACTABLE-SINGLES

FIND-TRACTABILITY-CANDIDATES computes all candidates for tractable subsets starting from the closure of the base relations and successively adding new tractable relations. Each new relation that is added, is first tested whether it is incompatible with any of the relations already contained in the candidate. If it is not incompatible, then the closure of the current candidate with the new relation is computed and it is tested whether the resulting set can be refined to the current candidate using the CHECK-SINGLE-REFINEMENT algorithm. If it doesn't return fail, then the new relation is added and the current candidate is updated. The algorithm keeps track of all relations that cannot be added to the current candidate. 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. 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: FIND-INCOMPATIBLE-PAIRS
Input: A set of relations \mathcal{T} such that $\text{CSPSAT}(\{T\} \cup \mathcal{B})$ is tractable for all $T \in \mathcal{T}$ and a set of relations \mathcal{H} such that $\text{CSPSAT}(\{H\} \cup \mathcal{B})$ is NP-hard for all $H \in \mathcal{H}$.
Output: A list \mathcal{L} of all pairs $\langle T_1, T_2 \rangle$ with $T_1, T_2 \in \mathcal{T}$ such that $\text{closure}(\{T_1\} \cup \{T_2\} \cup \mathcal{B})$ contains a relation of \mathcal{H} .

1. $\mathcal{L} \leftarrow \emptyset$
2. *for all* $T_i, T_j \in \mathcal{T}$ *do*
3. *if* $\text{closure}(\{T_i\} \cup \{T_j\} \cup \mathcal{B}) \cap \mathcal{H} \neq \emptyset$ *then* $\mathcal{L} \leftarrow \mathcal{L} \cup \langle T_i, T_j \rangle$
4. *return* \mathcal{L}

Algorithm: FIND-TRACTABILITY-CANDIDATES
Input: A set of relations \mathcal{T} such that $\text{CSPSAT}(\{T\} \cup \mathcal{B})$ is tractable for all $T \in \mathcal{T}$, a set of relations \mathcal{H} such that $\text{CSPSAT}(\{H\} \cup \mathcal{B})$ is NP-hard for all $H \in \mathcal{H}$, and a list \mathcal{L} of incompatible pairs.
Output: Candidates for tractable subsets \mathcal{TS}_i of $2^{\mathcal{B}}$

1. $n \leftarrow 0$; $\text{max} \leftarrow 1$
2. $M[i][j] = \text{true}$ for $i = j$ and *false* otherwise
3. $\text{candidate} \leftarrow \hat{\mathcal{B}}$; $\text{newcandidate} \leftarrow \hat{\mathcal{B}}$
4. *while* (*true*) *do*
5. $\text{nextrels} \leftarrow \emptyset$; $n \leftarrow n + 1$
6. *for all* $T_i \in \mathcal{T}$ and $T_i \notin \text{candidate}$ *do*
7. *if* $\langle T_i, R \rangle \notin \mathcal{L}$ for an $R \in \text{candidate}$ *then* $\text{nextrels} \leftarrow \text{nextrels} \cup \{T_i\}$; *continue*;
8. $\text{newcandidate} \leftarrow \text{closure}(\text{newcandidate} \cup \{T_i\})$
9. *if* $\text{CHECK-SINGLE-REFINEMENT}(T_i, \text{candidate}, M) \neq \text{fail}$ *then*
10. $\text{candidate} \leftarrow \text{newcandidate}$
11. $\mathcal{TS}_n = \text{candidate}$
12. newrel is any relation in $(\text{nextrels} \cap \bigcup_i \mathcal{TS}_i)$
13. *if* $\text{newrel} \neq \emptyset$ *then*
14. $\text{candidate} \leftarrow \text{closure}(\mathcal{B} \cup \text{newrel})$; $\text{newcandidate} \leftarrow \text{candidate}$; $\text{max} \leftarrow \text{max} + 1$
15. *while* $\text{max} == n$ *do*
16. $T_k, T_j \in \mathcal{T}$ is a pair such that $\langle T_k, T_j \rangle \notin \mathcal{L}$ and $\{T_k, T_j\}$ is not contained in any \mathcal{TS}_i
17. *if* there is no such pair *then return all* \mathcal{TS}_i
18. $\text{candidate} \leftarrow \text{closure}(\mathcal{B} \cup \{T_j\})$; $\text{newcandidate} \leftarrow \text{closure}(\text{candidate} \cup \{T_k\})$
19. *if* $\text{CHECK-SINGLE-REFINEMENT}(T_k, \text{candidate}, M) == \text{fail}$ *then*
20. $\mathcal{L} \leftarrow \mathcal{L} \cup \langle T_k, T_j \rangle$; *continue*
21. $\text{candidate} \leftarrow \text{newcandidate}$; $\text{max} \leftarrow \text{max} + 1$

Fig. 3: Algorithms FIND-INCOMPATIBLE-PAIRS and FIND-TRACTABILITY-CANDIDATES

algorithm tests each pair of relations which is not yet contained in a tractable subset 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. If this is not possible and CHECK-SINGLE-REFINEMENT returns `fail`, then this pair is added to the list of incompatible pairs. If a pair is found which can be refined, then a new candidate is formed which contains the pair 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. This will be tested in the final step.

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. Tractability can be proven using the refinement method. One way of doing so is to use a heuristic as specified in [12], such as applying the identity refinement matrix to the candidate. The identity refinement matrix refines all relations that contain the identity relation to the refinement where the identity relation is eliminated. This heuristic was very successful for proving tractability of subsets of RCC8 and the Interval Algebra [12], but if this heuristic is not successful, a manual intervention is necessary. We will therefore specify an algorithm which can prove tractability in the general case.

All we know at the moment is that the closure of the base relations is tractable and if we can find a refinement of a candidate set to the closure of the base relations, then we know that the candidate set is tractable. But what refinement strategy can we use? We can extend tractability of the closure of the base relations 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, we know that we cannot add it to the tractable subset, if CHECK-SINGLE-REFINEMENT 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 FIND-TRACTABILITY-CANDIDATES except for the special treatment of the relations for which the interleaved CHECK-SINGLE-REFINEMENT calls return `unknown`. If none of these calls 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.

Definition 3. Given a set of relations $\mathcal{S} \subseteq 2^{\mathcal{B}}$. The *closure sequence* $\mathcal{C}(\mathcal{S}; R_1, \dots, R_n)$ is a subset of $2^{\mathcal{B}}$ which is recursively defined as follows:

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

Lemma 2. Given a set of relations $\mathcal{T} \subseteq 2^{\mathcal{B}}$. Suppose \mathcal{T} is equivalent to the closure sequence $\mathcal{C}(\mathcal{B}; R_1, \dots, R_n)$ and suppose that M is a refinement matrix where the diagonal is true and all other entries are false.

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

1. for all \mathcal{TS}_i do
2. queue $\leftarrow \emptyset$;
3. $M[i][j] = \text{true}$ for $i = j$ and *false* otherwise;
4. for each relation $R \in \mathcal{TS}_i \setminus \widehat{\mathcal{B}}$ do queue \leftarrow queue $\cup (R, 0)$;
5. tractable $\leftarrow \widehat{\mathcal{B}}$;
6. loop $\leftarrow 0$; changes $\leftarrow \text{false}$;
7. while queue $\neq \emptyset$ do
8. take and delete the first pair (R, num) from queue;
9. if $(\text{num} > \text{loop}$ and changes $== \text{false}$) then *break*;
10. if $(\text{num} > \text{loop})$ then loop \leftarrow num; changes $\leftarrow \text{false}$;
11. if CHECK-SINGLE-REFINEMENT(R , tractable, M) $== \text{true}$ then
12. tractable \leftarrow closure(tractable $\cup \{R\}$);
13. delete all pairs (S, n) from queue (for any n) for which $S \in$ tractable;
14. changes $\leftarrow \text{true}$;
15. else add $(R, \text{num} + 1)$ to the end of the queue;
16. $\mathcal{TS}_i \leftarrow$ tractable;
17. return \mathcal{TS}_i for all i ;

Fig. 4: Algorithm TEST-CANDIDATES

If CHECK-SINGLE-REFINEMENT($R_i, \mathcal{C}(\mathcal{B}; R_1, \dots, R_{i-1}), M$) returns *true* for all $1 \leq i \leq n$, then C $\mathcal{SPSAT}(\mathcal{T})$ is tractable.

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

Our algorithm TEST-CANDIDATES in Figure 4 takes each candidate and successively tests for 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.

The whole procedure is then a sequence of the three steps (1) FIND-TRACTABLE-SINGLES, (2) FIND-INCOMPATIBLE-PAIRS and FIND-TRACTABILITY-CANDIDATES, and (3) TEST-CANDIDATES. The input to this procedure is a set of base relations and the composition and converse table, and the output is one or more tractable subsets.

Theorem 1. *Given a set of base relations \mathcal{B} such that algebraic closure decides C $\mathcal{SPSAT}(\mathcal{B})$. For each set $\mathcal{TS}_i \subseteq 2^{\mathcal{B}}$ which is returned by our algorithm, C $\mathcal{SPSAT}(\mathcal{TS}_i)$ is tractable.*

3.4 Applying the New Procedure

We implemented the procedure and tested it for RCC8 in order to see if we can identify the known maximal tractable subsets or how close the results of our method are to the actual maximal tractable subsets. We know that there are three maximal tractable subsets of RCC8 that contain all base relations. Since we are interested in efficient algorithms, we will only consider tractable subsets that contain all the base relations.

Applying the first step of our procedure (FIND-TRACTABLE-SINGLES) to RCC8 resulted in 76 relations that are contained in \mathcal{H} and 179 relations that are contained in \mathcal{T} . Note that the empty relation which is the 256th relation of RCC8 is not considered here. It is remarkable that the 76 relations of \mathcal{H} correspond exactly to those relations that were shown to be NP-hard when combined with the RCC8 base relations in [14, 12]. For some of the relations of \mathcal{T} , CHECK-SINGLE-REFINEMENT returned `unknown`, but all these relations were added to \mathcal{T} as part of computing the closure of other relations for which CHECK-SINGLE-REFINEMENT returned `succeed`. This seems to indicate that a relation is not part of a tractable subset only if CHECK-SINGLE-REFINEMENT returns `fail`, but this has to be analysed in more detail.

The second step, FIND-TRACTABILITY-CANDIDATES returned three candidates for tractable subsets. The second candidate was introduced by a relation that was not contained in the first candidate, while the third candidate was introduced by a compatible pair which was not contained in the previous two candidates. That means the first two candidates cover all relations of \mathcal{T} . What is very remarkable is that the three candidates actually correspond to the three maximal tractable subsets of RCC8 identified in [12]. For the third step, proving tractability of the candidates, we therefore could have used the identity refinement heuristic, but of course we have to assume that we do not know the maximal tractable subsets yet. Therefore, we applied our algorithm TEST-CANDIDATES to the three candidates. Not surprisingly, tractability of all three candidates could be shown using TEST-CANDIDATES. The whole procedure ran in less than one hour on a Pentium-Duo 3.00 GHz CPU with 2GB RAM. By far the most time was spent on the interleaved CHECK-SINGLE-REFINEMENT calls. This could be considerably reduced if we use refinement arrays instead of refinement matrices as introduced in [12].

We also experimented with weaker versions of FIND-TRACTABILITY-CANDIDATES which doesn't use the interleaved calls to CHECK-SINGLE-REFINEMENT. This version was considerably faster and also identified three candidates. The first two candidates were the same, while the third candidate contained 170 relations, ten relations more than in the normal version of FIND-TRACTABILITY-CANDIDATES. These relations were all eliminated by TEST-CANDIDATES and the output of the weaker version of our procedure was again the three known maximal tractable subsets of RCC8. A further speed up was obtained by changing the order in which the relations are tested. Testing those relations first whose closure with the base relations is large avoids computing CHECK-SINGLE-REFINEMENT for all those relations that are already contained in the closure.

4 Conclusions

Using our general procedure, we were able to identify all maximal tractable subsets of RCC8 completely automatically without any manual intervention and without the need to give NP-hardness proofs for some of the relations. All NP-hard relations were identified in the first step by our revised refinement algorithm. Note that we can of course not guarantee that all maximal tractable subsets can be found in all cases or that the relations we assume to be NP-hard are actually NP-hard. We can only guarantee that the sets our procedure finds are tractable subsets. But for the purpose of developing efficient algorithms, the large tractable subsets identified by our procedure should be enough to give a considerable speed up in solving instances of the NP-hard reasoning problem.

Some further analysis of our procedure will be necessary in order to obtain a clear understanding of the effect of the relations for which CHECK-SINGLE-REFINEMENT returns `unknown`. For the cases we tested, this had no effect as all these relations were found to be tractable by other means. For base relations where this is not possible, our procedure could be further extended by testing these relations again once the tractability of the resulting candidates has been verified. Then these relations could be refined to one of the larger tractable subsets instead of the closure of the base relations which could lead to larger tractable subsets. We could also use an optimistic view and add all these relations to the set of tractable relations \mathcal{T} and see in later steps if they have to be removed again. A further improvement could be to integrate step 3 into step 2. We will also implement our procedure using refinement arrays instead of refinement matrices and then apply it to some other sets of spatial or temporal relations. First tests appear to be very promising.

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 maximal tractable subsets analytically if they can just as well be identified using our procedure? The answer to the first question is easy. Even with our procedure, it is still necessary to show that a-closure decides atomic CSPs. 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 [13] but it is still a very challenging problem. This also gives an answer to the second question. Tractable subsets can only be found if it can be shown that a-closure decides atomic CSPs, so this still requires a theoretical analysis. Also, our procedure does not guarantee maximality yet. So an accompanying theoretical analysis is still very useful and gives insight into why a problem is hard.

References

- [1] Philippe Balbiani, Jean-François Condotta, and Luis Farinas del Cerro. A new tractable subclass of the rectangle algebra. In IJCAI-99 [3], pages 442–447.

- [2] Anthony G. Cohn and Shyamanta M. Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29, 2001.
- [3] *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, August 1999.
- [4] Andrei A. Krokhin, Peter Jeavons, and Peter Jonsson. Reasoning about temporal relations: The tractable subalgebras of Allen’s interval algebra. *Journal of the ACM*, 50(5):591–640, 2003.
- [5] Peter B. Ladkin and Roger Maddux. On binary constraint problems. *Journal of the Association for Computing Machinery*, 41(3):435–469, May 1994.
- [6] Gérard Ligozat, Debasis Mitra, and Jean-François Condotta. Spatial and temporal reasoning: Beyond Allen’s calculus. *AI Communications*, 17(4):223–233, 2004.
- [7] Gérard Ligozat and Jochen Renz. What is a qualitative calculus? A general framework. In *PRICAI 2004: Trends in Artificial Intelligence, 8th Pacific Rim International Conference on Artificial Intelligence*, Lecture Notes in Computer Science 3157, pages 53–64. Springer, 2004.
- [8] Isabel Navarrete, Abdul Sattar, Rattana Wetprasit, and Roque Marín. On point-duration networks for temporal reasoning. *Artificial Intelligence*, 140(1/2):39–70, 2002.
- [9] Bernhard Nebel. Solving hard qualitative temporal reasoning problems: Evaluating the efficiency of using the ORD-Horn class. *CONSTRAINTS*, 3(1):175–190, 1997.
- [10] Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen’s interval algebra. *Journal of the Association for Computing Machinery*, 42(1):43–66, January 1995.
- [11] David A. Randell, Zhan Cui, and Anthony G. Cohn. A spatial logic based on regions and connection. In B. Nebel, W. Swartout, and C. Rich, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference*, pages 165–176, Cambridge, MA, October 1992. Morgan Kaufmann.
- [12] Jochen Renz. Maximal tractable fragments of the Region Connection Calculus: A complete analysis. In *IJCAI-99* [3], pages 448–454.
- [13] Jochen Renz and Gérard Ligozat. Weak composition for qualitative spatial and temporal reasoning. In *Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005*, pages 534–548, 2005.
- [14] Jochen Renz and Bernhard Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the Region Connection Calculus. *Artificial Intelligence*, 108(1-2):69–123, 1999.
- [15] Jochen Renz and Bernhard Nebel. Efficient methods for qualitative spatial reasoning. *Journal of Artificial Intelligence Research*, 15:289–318, 2001.

SparQ: A Toolbox for Qualitative Spatial Representation and Reasoning

Frank Dylla, Lutz Frommberger, Jan Oliver Wallgrün, and Diedrich Wolter

SFB/TR 8 Spatial Cognition, Universität Bremen
Bibliothekstr. 1, 28359 Bremen, Germany

{dylla, lutz, wallgruen, dwolter}@sfbtr8.uni-bremen.de

Abstract. A multitude of calculi for qualitative spatial reasoning (QSR) has been proposed during the last two decades. The number of practical applications that make use of QSR techniques is, however, comparatively small. One reason for this may be seen in the difficulty for people from outside the field to incorporate the required reasoning techniques into their software. Sometimes, proposed calculi are only partially specified and implementations are rarely available. With the SparQ toolbox presented in this text, we seek to improve this situation by making common calculi and standard reasoning techniques accessible in a way that allows for easy integration into applications. We hope to turn this into a community effort and encourage researchers to incorporate their calculi into SparQ. This text provides an overview on SparQ and its utilization.

1 Introduction

Qualitative spatial reasoning (QSR) is an established field of research pursued by investigators from many disciplines including geography, philosophy, computer science, and AI [2]. The general goal is to model commonsense knowledge and reasoning about space as efficient representation and reasoning mechanisms that are still expressive enough to solve a given task.

Following the approach taken in Allen’s seminal paper on qualitative temporal reasoning [1], QSR is typically realized in form of calculi over sets of spatial relations (like ‘left-of’ or ‘north-of’). These are called *qualitative spatial calculi*. A multitude of spatial calculi has been proposed during the last two decades, focusing on different aspects of space (mereotopology, orientation, distance, etc.) and dealing with different kinds of objects (points, line segments, extended objects, etc.). Two main research directions in QSR are mereotopological reasoning about regions [13, 4, 15] and reasoning about positional information (distance and orientation) of point objects [5, 10, 9, 12, 14] or line segments [16, 11, 3].

Despite this huge variety of qualitative spatial calculi, the amount of applications employing qualitative spatial reasoning techniques is comparatively small.

We believe that important reasons for this are the following: Choosing the right calculus for a particular application is a challenging task, especially for people not familiar with QSR. Calculi are often only partially specified and usually no implementation is made available—if the calculus is implemented at all and not only investigated theoretically. As a result, it is not possible to “quickly” evaluate how different calculi perform

in practice. Even if an application developer has decided on a particular calculus, he has to invest serious efforts to include the calculus and required reasoning techniques into the application. For many calculi this is a time-consuming and error-prone process (e.g. involving writing down huge composition tables, which are often not even completely specified in the literature).

Overall, we are convinced that the QSR community should strive for making the fruits of its work available to the public in a homogeneous framework. We have thus started the development of a qualitative spatial reasoning toolbox called *SparQ*¹ that aims at supporting the most common tasks—qualification, computing with relations, constraint-based reasoning, etc. (cp. section 3)—for an extensible set of spatial calculi. A complementary approach aiming at the specification and investigation of the interrelations between calculi has been described in [20]. Here, the calculi are defined in the algebraic specification language CASL. In contrast, our focus is on providing an implementation of QSR techniques that is tailored towards the needs of application developers. In its current version, SparQ mainly concentrates on calculi from the area of reasoning about the orientation of point objects or line segments. However, specifying and adding new calculi is in most cases very simple. We hope to turn this into a community effort, encouraging researchers from other groups to incorporate their own calculi.

In this text, we describe SparQ and its utilization. The current version of SparQ and further documentation will be made available at the SparQ homepage². The next section briefly recapitulates the relevant terms concerning QSR and spatial calculi as needed for the remainder of the text. In section 3, we describe the services provided by SparQ. Section 4 explains how new calculi can be incorporated into SparQ and section 5 describes how SparQ can be integrated into own applications.

2 Reasoning with Qualitative Spatial Relations

A qualitative spatial calculus defines operations on a finite set \mathcal{R} of spatial relations. The spatial relations are defined over a particular set of spatial objects, the domain D (e.g. points in the plane, oriented line segments, etc.). While a *binary calculus* deals with binary relations $R \subseteq D \times D$, a *ternary calculus* operates with ternary relations $R \subseteq D \times D \times D$.

A spatial calculus establishes a set of typically jointly exhaustive and pairwise disjoint (JEPD) base relations \mathcal{BR} . The set \mathcal{R} of all relations considered by the calculus contains at least the base relations, the empty relation \emptyset , the universal relation U , and the identity relation Id ; the set \mathcal{R} should be closed under the operations defined in the following. Typically, the powerset of the base relations $2^{\mathcal{BR}}$ is chosen for \mathcal{R} .³

As the relations are subsets of tuples from the same Cartesian product, the set operations union, intersection, and complement can be directly applied:

Union: $R \cup S = \{x \mid x \in R \vee x \in S\}$

¹ Spatial Reasoning done Qualitatively

² <http://www.sfbtr8.uni-bremen.de/project/r3/sparq/>

³ Unions of relations correspond to disjunction of relational constraints and thus we will often simply speak of disjunctions of relations as well and write them as sets $\{R_1, \dots, R_n\}$.

Intersection: $R \cap S = \{ x \mid x \in R \wedge x \in S \}$

Complement: $\bar{R} = U \setminus R = \{ x \mid x \in U \wedge x \notin R \}$

where R and S are both n -ary relations on D . The other operations depend on the arity of the calculus.

2.1 Operations for Binary Calculi

For binary calculi, two other important operations are required:

Converse: $R^\smile = \{ (y, x) \mid (x, y) \in R \}$

(Strong) composition: $R \circ S = \{ (x, z) \mid \exists y \in D : ((x, y) \in R \wedge (y, z) \in S) \}$

For some calculi no finite set of relations exists that includes the base relations and is closed under composition as defined above. In this case, a weak composition is defined instead that takes the union of all base relations that have a non-empty intersection with the result of the strong composition:

Weak composition: $R \circ_{weak} S = \{ d \mid T \in \mathcal{BR} \wedge d \in T \wedge T \cap (R \circ S) \neq \emptyset \}$

2.2 Operations for Ternary Calculi

While there is only one possibility to permute the two objects of a binary relation which leads to the converse operation, there exist 5 such permutations for the three objects of a ternary relation. This results in the following operations⁴ [21]:

Inverse: $INV(R) = \{ (y, x, z) \mid (x, y, z) \in R \}$

Short cut: $SC(R) = \{ (x, z, y) \mid (x, y, z) \in R \}$

Inverse short cut: $SCI(R) = \{ (z, x, y) \mid (x, y, z) \in R \}$

Homing: $HM(R) = \{ (y, z, x) \mid (x, y, z) \in R \}$

Inverse homing: $HMI(R) = \{ (z, y, x) \mid (x, y, z) \in R \}$

Composition for ternary calculi is defined accordingly to the binary case:

(Strong) comp.: $R \circ S = \{ (w, x, z) \mid \exists y \in D : ((w, x, y) \in R \wedge (x, y, z) \in S) \}$

Other ways of composing two ternary relations can be expressed as a combination of the unary permutation operations and the composition [17] and thus do not have to be defined separately. The definition of weak composition is identical to the binary case.

⁴ It is not needed to specify all these operations as some can be expressed by others.

2.3 Constraint Reasoning with Spatial Calculi

Spatial calculi are often used to formulate constraints about the spatial configurations of a set of objects from the domain of the calculus as a constraint satisfaction problem (CSP): Such a spatial constraint satisfaction problem then consists of a set of variables X_1, \dots, X_n (one for each spatial object) and a set of constraints C_1, \dots, C_m (relations from the calculus). Each variable X_i can take values from the domain of the utilized calculus. CSPs are often visualized as constraint networks which are graphs with nodes corresponding to the variables and arcs corresponding to constraints. A CSP is consistent, if an assignment for all variables to values of the domain can be found, that satisfies all the constraints. Spatial CSPs usually have infinite domains and thus backtracking over the domains can not be used to determine global consistency.

Besides global consistency, weaker forms of consistency called *local consistencies* are of interest in QSR. On the one hand, they can be employed as a forward checking technique reducing the CSP to a smaller equivalent one (one that has the same set of solutions). Furthermore, in some cases they can be proven to be not only necessary but also sufficient for global consistency for the set \mathcal{R} of relations of a given calculus. If this is only the case for a certain subset \mathcal{S} of \mathcal{R} and this subset exhaustively splits \mathcal{R} (which means that every relation from \mathcal{R} can be expressed as a disjunction of relations from \mathcal{S}), this at least allows to formulate a backtracking algorithm to determine global consistency by recursively splitting the constraints and using the local consistency as a decision procedure for the resulting CSPs with constraints from \mathcal{S} [8].

One important form of local consistency is *path-consistency* which (in binary CSPs) means that for every triple of variables each consistent evaluation of the first two variables can be extended to the third variable in such a way that all constraints are satisfied. Path-consistency can be enforced syntactically based on the composition operation (for instance with the algorithm by van Beek [19]) in $O(n^3)$ time where n is the number of variables. However, this syntactic procedure does not necessarily yield the correct result with respect to path-consistency as defined above. Whether this is the case or not needs to be investigated for each individual calculus.

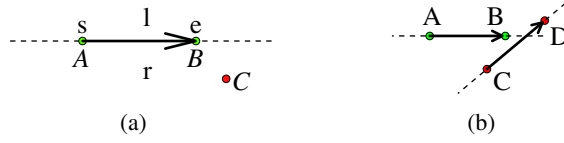
2.4 Supported Calculi

The calculi currently included in SparQ are the FlipFlop Calculus (FFC) [9] with the \mathcal{LR} refinement described in [18], the Single Cross Calculus (SCC) and Double Cross Calculus (DCC) [5], the coarse-grained variant of the Dipole Relation Algebra ($\mathcal{DR}\mathcal{A}_c$) [16, 11], the Oriented Point Relation Algebra $\mathcal{OPR}\mathcal{A}_m$ [12], as well as RCC-5 and RCC-8⁵ [13]. An overview is given in Table 1 where the calculi are classified according to their arity (binary, ternary), their domain (points, oriented points, line segments, regions), and the aspect of space modeled (orientation, distance, mereotopology). As can be seen, mainly calculi for reasoning about orientation have been incorporated so far, but calculi dealing with other kinds of base objects or dealing with other aspects of space can be integrated just as easily. We briefly describe $\mathcal{DR}\mathcal{A}_c$ as it will be used in the examples in the remainder of this text.

⁵ Currently only the relational specification is available for RCC, but no qualifier.

Table 1: The calculi currently included in SparQ

Calculus	arity		domain			aspect of space			
	binary	ternary	point	or. point	line seg.	region	orient.	dist.	mereot.
FFC/LR		✓	✓				✓		
SCC		✓	✓				✓		
DCC		✓	✓				✓		
\mathcal{DRA}_c	✓				✓		✓		
\mathcal{OPRA}_m	✓			✓			✓		
RCC-5/8	✓					✓			✓

**Fig. 1:** Illustration of \mathcal{DRA}_c : (a) The FlipFlop relations used to define Dipole relations. (b) A dipole configuration: $d_{AB} rlll d_{CD}$ in \mathcal{DRA}_c .

\mathcal{DRA}_c A dipole is an oriented line segment as e.g. determined by a start and an end point. We will write d_{AB} for a dipole defined by start point A and end point B . The idea of using dipoles was first introduced by Schlieder [16] and extended in [11]. The coarse-grained dipole calculus variant (\mathcal{DRA}_c) describes the orientation relation between two dipoles d_{AB} and d_{CD} with the preliminary of $A, B, C,$ and D being in general position, i.e., no three disjoint points are collinear. Each base relation is a 4-tuple (r_1, r_2, r_3, r_4) of FlipFlop relations relating one point from one of the dipoles with the other dipole. r_1 describes the relation of C with respect to the dipole d_{AB} , r_2 of D with respect to d_{AB} , r_3 of A with respect to d_{CD} , and r_4 of B with respect to d_{CD} . The distinguished FlipFlop relations are **left**, **right**, **start**, and **end** (see Figure 1 (a)). Dipole relations are usually written without commas and parentheses, e.g. $rlll$. Thus, the example in Figure 1 (b) shows the relation $d_{AB} rlll d_{CD}$.

3 SparQ

SparQ consists of a set of modules that provide different services required for QSR that will be explained below. These modules are glued together by a central script that can either be used directly from the console or included into own applications via TCP/IP streams in a server/client fashion (see section 5).

The general syntax for using the SparQ main script is as follows:

```
$ ./sparq <module> <calculus identifier> <module specific parameters>
```

Example:

```
$ ./sparq compute-relation dra-24 complement "(lrl1 llrr)"
```

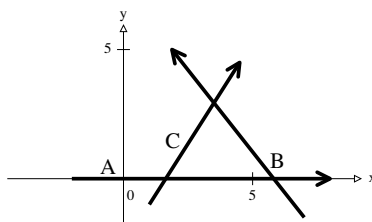


Fig. 2: An example configuration of three dipoles.

where ‘compute-relation’ is the name of the module to be utilized, in this case the module for conducting operations on relations, ‘dra-24’ is the SparQ identifier for the dipole calculus \mathcal{DRAC} , and the rest are module specific parameters, here the name of the operation that should be conducted (*complement*) and a string parameter representing the disjunction of the two dipole base relations $lrll$ and $llrr$ ⁶. The example call thus computes the complement of the disjunction of these two relations. SparQ provides the following modules:

qualify transforms a quantitative geometric description of a spatial configuration into a qualitative description based on one of the supported calculi;

compute-relation applies the operations defined in the calculi specifications (intersection, union, complement, converse, composition, etc.) to a set of spatial relations;

constraint-reasoning performs computations on constraint networks.

Further modules are planned as future extensions. This comprises a quantification module for turning qualitative scene descriptions back into quantitative geometric descriptions and a module for neighborhood-based spatial reasoning. In the following section we will take a closer look at the three existent modules.

3.1 Qualification and Scene Descriptions

The purpose of the qualify module is to turn a quantitative geometric scene description into a qualitative scene description composed of base relations from a particular calculus. The calculus is specified via the calculus identifier that is passed with the call to SparQ. Qualification is required for applications in which we want to perform qualitative computations over objects whose geometric parameters are known.

The qualify module reads a quantitative scene description and generates a qualitative description. A quantitative scene description is a space-separated list of base object descriptions enclosed in parentheses. Each base object description is a tuple consisting of an object identifier and object parameters that depend on the type of the object. For instance, let us say we are working with dipoles which are oriented line segments. The object description of a dipole is of the form ‘(name x_s y_s x_e y_e)’, where name is the identifier of this particular dipole object and the rest are the coordinates of start and end

⁶ Disjunctions of base relations are always represented as a space-separated list of the base relations enclosed in parentheses in SparQ.

point of the dipole. Let us consider the example in Figure 2 which shows three dipoles A, B, and C. The quantitative scene description for this situation would be:

```
( (A -2 0 8 0) (B 7 -2 2 5) (C 1 -1 4.5 4.5) )
```

The qualify module has one module specific parameter:

mode This parameter controls which relations are included into the qualitative scene description: If the parameter is ‘all’, the relation between every object and every other object will be included. If it is ‘first2all’ only the relations between the first and all other objects are computed.

The resulting qualitative scene description is a space-separated list of relation tuples enclosed in parentheses. A relation tuple consists of an object identifier followed by a relation name and another object identifier, meaning that the first object stands in this particular relation with the second object. The command to produce the qualitative scene description followed by the result is⁷:

```
$ ./sparq qualify dra-24 all
$ ( (A -2 0 8 0) (B 7 -2 2 5) (C 1 -1 4.5 4.5) )
> ( (A rllr B) (A rllr C) (B lrrl C) )
```

3.2 Computing with Relations

The compute-relation module allows to compute with the operations defined in the calculus specification. The module specific parameters are the operation that should be conducted and one or more input relations depending on the arity of the operation. Let us say we want to compute the converse of the *llrl* dipole relation. The corresponding call to SparQ and the result are:

```
$ ./sparq compute-relation dra-24 converse llrl
> (rlll)
```

The result is always a list of relations as operations often yield a disjunction of base relations. The composition of two relations requires one more relation as parameter because it is a binary operation, e.g.:

```
$ ./sparq compute-relation dra-24 composition llrr rllr
> (lrrr llrr rllr slsr lllr rllr rlll elll llll lrlr)
```

Here the result is a disjunction of 10 base relations. It is also possible to have disjunctions of base relations as input parameters. For instance, the following call computes the intersection of two disjunctions:

```
$ ./sparq compute-relation dra-24 intersection "(lrrr rlll rllr)"
"(llll rlll)"
> (rlll)
```

⁷ In all the examples, input lines start with ‘\$’. Output of SparQ is marked with ‘>’.

3.3 Constraint Reasoning

The constraint-reasoning module reads a description of a constraint network—which is a qualitative scene description that may include disjunctions and may be inconsistent and/or underspecified—and performs a particular kind of consistency check⁸. Which type of consistency check is executed depends on the first module specific parameter:

action The two actions currently provided are ‘path-consistency’ and ‘scenario-consistency’ and determine which kind of consistency check is performed.

The action ‘path-consistency’ causes the module to enforce path-consistency on the constraint network using van Beek’s algorithm [19] or detect the inconsistency of the network in the process. We could for instance check if the scene description generated by the qualify module in section 3.1 is path-consistent which of course it is. To make it slightly more interesting we add the base relation *ells* to the constraint between *A* and *C* resulting in a constraint network that is not path-consistent:

```
$ ./sparq constraint-reasoning dra-24 path-consistency
$ ( (A rllr B) (A (ells rllr) C) (B lrrl C) )
> Modified network.
> ( (B (lrrl) C) (A (rllr) C) (A (rllr) B) )
```

The result is a path-consistent constraint network in which *ells* has been removed. The output ‘Modified network’ indicates that the original network was not path-consistent and had to be changed. Otherwise, the result would have started with ‘Unmodified network’. In the next example we remove the relation *rllr* from the disjunction between *A* and *C*. This results in a constraint network that cannot be made path-consistent which implies that it is not globally consistent.

```
$ ./sparq constraint-reasoning dra-24 path-consistency
$ ( (A rllr B) (A ells C) (B lrrl C) )
> Not consistent.
> ( (B (lrrl) C) (A () C) (A (rllr) B) )
```

SparQ correctly determines that the network is inconsistent and returns the constraint network in the state in which the inconsistency showed up (indicated by the empty relation *()* between *A* and *C*).

If ‘scenario-consistency’ is provided as argument, the constraint-reasoning module checks if a path-consistent scenario exists for the given network. It uses a backtracking algorithm to generate all possible scenarios and checks them for path-consistency as described above. A second module specific parameter determines what is returned as the result of the search:

return This parameter determines what is returned in case of a constraint network for which path-consistent scenarios can be found. It can take the values ‘first’ which returns the first path-consistent scenario, ‘all’ which returns all path-consistent scenarios, and ‘interactive’ which returns one solution and allows to ask for the next solution until all solutions have been iterated.

⁸ The constraint-reasoning module also provides some basic actions to manipulate constraint networks that are not further explained in this text. One example is the ‘merge’ operation that is used in the example in section 5.

Path-consistency is also used as a forward-checking method during the search to make it more efficient. For certain calculi, the existence of a path-consistent scenario implies global consistency. However, this again has to be investigated for each calculus. As a future extension it is planned to allow to specify splitting subsets of a calculus for which path-consistency implies global consistency and provide a variant of the backtracking algorithm that decides global consistency by searching for path-consistent instantiations that only contain relations from the splitting subset. In the following example, we use ‘first’ as additional parameter so that only the first solution found is returned:

```
$ ./sparq constraint-reasoning dra-24 scenario-consistency first
$ ( (A rele C) (A ells B) (C errs B) (D srsl C) (A rser D) (D rrrl B) )
> ( (B (rlrr) D) (C (slsr) D) (C (errs) B) (A (rser) D) (A (ells) B)
  (A (rele) C) )
```

In case of an inconsistent constraint network, SparQ returns ‘Not consistent.’

4 Specifying Calculi in SparQ

For most calculi it should be rather easy to include them into SparQ. The main thing is to provide the calculus specification. Listing 1 shows an extract of the definition of a simple exemplary calculus for reasoning about distances between three point objects distinguishing the three relations ‘closer’, ‘farther’, and ‘same’. The specification is done in Lisp-like syntax.

```
(def-calculus "Relative distance calculus (reldistcalculus)"
  :arity :ternary
  :base-relations (same closer farther)
  :identity-relation same
  5   :inverse-operation ((same same)
                        (closer closer)
                        (farther farther))
  :composition-operation ((same same (same closer farther))
                          (same closer (same closer farther))
                          10 (same farther (same closer farther))
                          (closer same (same closer farther))
                          (closer closer (same closer farther))
                          (closer farther (same closer farther))
                          [...])
```

Listing 1. Specification of a simple ternary calculus (excerpt).

The arity of the calculus, the base relations, the identity relation and the different operations have to be specified, using lists enclosed in parentheses (e.g. when an operation returns a disjunction of base relations). In this example, the inverse operation applied to ‘same’ yields ‘same’ and composing ‘closer’ and ‘same’ results in the universal relation written as the disjunction of all base relations. Some operations like homing and short cut operations are left out in the example (cmp. section 2.2).

In addition to the calculus specification, it is necessary to provide the implementation of a qualifier function which for an n -ary calculus takes n geometric objects of the corresponding base type as input and returns the relation holding between these objects. The qualifier function encapsulates the methods for computing the qualitative relations from quantitative geometric descriptions. If it is not provided, the qualify module will not work for this calculus.

For some calculi, it is not possible to provide operations in form of simple tables as in the example. For instance, \mathcal{OPRA}_m has an additional parameter that specifies the granularity and influences the number of base relations. Thus, the operations can only be provided in procedural form, meaning the result of the operations are computed from the input relations when they are required. For these cases, SparQ allows to provide the operations as implemented functions and uses a caching mechanism to store often required results.

5 Integrating SparQ into Own Applications

SparQ can also run in server mode which makes it easy to integrate it into own applications. We have chosen a client/server approach as it allows for straightforward integration independent of the programming language used for implementing the application.

When run in server mode, SparQ takes TCP/IP connections and interacts with the client via simple plain-text line-based communication. This means the client sends commands which consist of everything following the ‘./sparq’ in the examples in this text, and can then read the results from the TCP/IP stream.

An example is given in Listing 2: A Python script opens a connection to the SparQ-server and performs some simple computations (qualification, adding another relation, checking for path-consistency). It produces the following output:

```
> ( (A rrl1 B) (A rrl1 C) )
> ( (A rrl1 B) (A rrl1 C) (B eses C) )
> Not consistent.
> ( (B (eses) C) (A () C) (A (rr1) B) )
```

6 Conclusions & Outlook

The SparQ toolbox presented in this text is a first step towards making QSR techniques and spatial calculi accessible to a broader range of application developers. We hope that this initiative will catch interest in the QSR community and encourage other researchers to incorporate their calculi into SparQ.

Besides including more calculi, extensions currently planned for SparQ are a module for neighborhood-based reasoning techniques [6, 3] (e.g. for relaxing inconsistent constraint networks based on conceptual neighborhoods and for qualitative planning) and a module that allows quantification (turning a consistent qualitative scene description back into a geometric representation). This requires the mediation between the algebraic and geometric aspects of a spatial calculus together with the utilization of prototypes. Moreover, we want to include geometric reasoning techniques based on Gröbner bases as a service for calculus developers as these can for instance be helpful

```

# connect to sparq server on localhost, port 4443
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect(('localhost', 4443))
sockfile = sock.makefile('r')
5 # qualify a geometrical scenario with DRA-24
sock.send('qualify dra-24 first2all ')
sock.send('((A 4 6 9 0.5) (B -5 5 0 2) (C -4 5 6 0))')
scene = readline() # read the answer
print scene
10 # add an additional relation (B eses C)
sock.send("constraint-reasoning dra-24 merge")
sock.send(scene + ' (B eses C)')
scene2 = readline() # read the answer
print scene2
15 # check the new scenario for consistency
sock.send('constraint-reasoning dra-24 path-consistency')
sock.send(scene2)
print readline() # print the answer
print readline() # print the resulting constraint network

```

Listing 2. Integrating SparQ into own applications: an example in Python

to derive composition tables [11]. The optimization of the algorithms included in SparQ is another issue that we want to grant more attention to in the future. Finally, we intend to incorporate interfaces that allow to exchange calculus specifications with other QSR frameworks (e.g. [20]).

Acknowledgment

This work was carried out in the framework of the SFB/TR 8 Spatial Cognition, project R3 [Q-Shape]. Financial support by the Deutsche Forschungsgemeinschaft is gratefully acknowledged.

References

- [1] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, pages 832–843, Nov. 1983.
- [2] A. G. Cohn and S. M. Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29, 2001.
- [3] F. Dylla and R. Moratz. Exploiting qualitative spatial neighborhoods in the situation calculus. In Freksa et al. [7], pages 304–322.
- [4] M. J. Egenhofer. A formal definition of binary topological relationships. In *3rd International Conference on Foundations of Data Organization and Algorithms*, pages 457–472, New York, NY, USA, 1989. Springer.
- [5] C. Freksa. Using orientation information for qualitative spatial reasoning. In A. U. Frank, I. Campari, and U. Formentini, editors, *Theories and methods of spatio-temporal reasoning in geographic space*, pages 162–178. Springer, Berlin, 1992.

- [6] C. Freksa. Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 1(54):199–227, 1992.
- [7] C. Freksa, M. Knauff, B. Krieg-Brückner, B. Nebel, and T. Barkowsky, editors. *Spatial Cognition IV. Reasoning, Action, Interaction: International Conference Spatial Cognition 2004*, Lecture Notes in Artificial Intelligence 3343. Springer, Berlin, Heidelberg, 2005.
- [8] P. Ladkin and A. Reinefeld. Effective solution of qualitative constraint problems. *Artificial Intelligence*, 57:105–124, 1992.
- [9] G. Ligozat. Qualitative triangulation for spatial reasoning. In A. U. Frank and I. Campari, editors, *Spatial Information Theory: A Theoretical Basis for GIS, (COSIT'93), Marciana Marina, Elba Island, Italy*, Lecture Notes in Computer Science 716, pages 54–68. Springer, 1993.
- [10] G. Ligozat. Reasoning about cardinal directions. *Journal of Visual Languages and Computing*, 9:23–44, 1998.
- [11] R. Moratz, J. Renz, and D. Wolter. Qualitative spatial reasoning about line segments. In W. Horn, editor, *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, Berlin, Germany, 2000. IOS Press.
- [12] R. Moratz, F. Dylla, and L. Frommberger. A relative orientation algebra with adjustable granularity. In *Proceedings of the Workshop on Agents in Real-Time and Dynamic Environments (IJCAI 05)*, 2005.
- [13] D. A. Randell, Z. Cui, and A. Cohn. A spatial logic based on regions and connection. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR'92)*, pages 165–176. Morgan Kaufmann, San Mateo, California, 1992.
- [14] J. Renz and D. Mitra. Qualitative direction calculi with arbitrary granularity. In C. Zhang, H. W. Guesgen, and W.-K. Yeap, editors, *PRICAI 2004: Trends in Artificial Intelligence, 8th Pacific Rim International Conference on Artificial Intelligence, Auckland, New Zealand, Proceedings*, Lecture Notes in Computer Science 3157, pages 65–74. Springer, 2004.
- [15] J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. *Artificial Intelligence*, 108(1-2):69–123, 1999.
- [16] C. Schlieder. Reasoning about ordering. In *Spatial Information Theory: A Theoretical Basis for GIS (COSIT'95)*, Lecture Notes in Computer Science 988, pages 341–349. Springer, Berlin, Heidelberg, 1995.
- [17] A. Scivos and B. Nebel. Double-crossing: Decidability and computational complexity of a qualitative calculus for navigation. In *Proceedings of COSIT'01*, Berlin, 2001. Springer.
- [18] A. Scivos and B. Nebel. The finest of its class: The practical natural point-based ternary calculus \mathcal{LR} for qualitative spatial reasoning. In Freksa et al. [7], pages 283–303.
- [19] P. van Beek. Reasoning about qualitative temporal information. *Artificial Intelligence*, 58(1-3):297–321, 1992.
- [20] S. Wöflf and T. Mossakowski. CASL specifications of qualitative calculi. In *Spatial Information Theory: Cognitive and Computational Foundations, Proceedings of COSIT'05*, 2005.
- [21] K. Zimmermann and C. Freksa. Qualitative spatial reasoning using orientation, distance, and path knowledge. *Applied Intelligence*, 6:49–58, 1996.

Author Index

Condotta, Jean-François, 53
D'Almeida, Dominique, 53
Dylla, Frank, 79
Ferrein, Alexander, 3
Frommberger, Lutz, 79
Lakemeyer, Gerhard, 3
Lecoutre, Christophe, 53
Ligozat, Gérard, 1
Mossakowski, Till, 28
Ragni, Marco, 14
Renz, Jochen, 64
Saïs, Lakhdar, 53
Schiffer, Stefan, 3
Schröder, Lutz, 28
Scivos, Alexander, 40
Wallgrün, Jan Oliver, 79
Wolter, Diedrich, 79
Wölfl, Stefan, 28