Linköping Studies in Science and Technology

Dissertation No. 779

A Study in the Computational Complexity of Temporal Reasoning

by

Mathias Broxvall

INSTITUTE OF TECHNOLOGY
LINKÖPINGS UNIVERSITET

Department of Computer and Information Science
Linköpings universitet
SE-581 83 Linköping, Sweden

Abstract

Reasoning about temporal and spatial information is a common task in computer science, especially in the field of artificial intelligence. The topic of this thesis is the study of such reasoning from a computational perspective. We study a number of different qualitative point based formalisms for temporal reasoning and provide a complete classification of computational tractability for different time models. We also develop more general methods which can be used for proving tractability and intractability of other relational algebras. Even though most of the thesis pertains to qualitative reasoning the methods employed here can also be used for quantitative reasoning. For instance, we introduce a tractable and useful extension to the quantitative point based formalism $\mathrm{STP}^{\neq}$. This extension gives the algebra an expressibility which subsumes the largest tractable fragment of the augmented interval algebra and has a faster and simpler algorithm for deciding consistency.

The use of disjunctions in temporal formalisms is of great interest not only since disjunctions are a key element in different logics but also since the expressibility can be greatly enhanced in this way. If we allow arbitrary disjunctions, the problems under consideration typically become intractable and methods to identify tractable fragments of disjunctive formalisms are therefore useful. One such method is to use the independence property. We present an automatic method for deciding this property for many relational algebras. Furthermore, we show how this concept can not only be used for deciding tractability of sets of relations but also to demonstrate intractability of relations not having this property. Together with other methods for making total classifications of tractability this goes a long way towards easing the task of classifying and understanding relational algebras.

The tractable fragments of relational algebras are sometimes not expressive enough to model real-world problems and a backtracking solver is needed. For these cases we identify another property among relations which can be used to aid general backtracking based solvers to find solutions faster.

Acknowledgments

This thesis work has been conducted at the Theoretical Computer Science Lab at the Department of Computer and Information Science, Linköping University. I would like to thank my primary supervisor Peter Jonsson and my secondary supervisors Ulf Nilsson and Anders Haraldsson for letting me do this research, as well as the other members of the laboratory for providing a stimulating research environment. I would also like to thank Jochen Renz for co-authoring one of the papers in this thesis and for providing many useful and interesting comments.

Other important contributors are the many anonymous reviewers who have provided many useful comments on the various papers that are part of this thesis and Ivan Rankin for his many corrections and suggestions for improvements on the final thesis. My funding has graciously been provided by the ECSEL graduate student program and I would like to thank all the teachers both within ECSEL and the university in general who have taught the many courses I've been taking during both my undergraduate and my graduate studies. Last but not least I would also like to thank all the administrative staff at IDA for dealing with all the practical problems encountered during these years as a student.

Mathias Broxvall
Linköping, 2002

List of Papers

The thesis includes the following papers:

I. Mathias Broxvall and Peter Jonsson. Point Algebras for Temporal Reasoning: Algorithms and Complexity.

   The paper is a revised and extended version of the following three papers:

   - Mathias Broxvall and Peter Jonsson. Towards a Complete Classification of Tractability in Point Algebras for Nonlinear Time. In *Proceedings of the 5th International Conference on Principles and Practice of Constraint Programming (CP-99)*, pp. 129–143, Alexandria, VA, USA, Oct, 1999.
   - Mathias Broxvall and Peter Jonsson: Disjunctive Temporal Reasoning in Partially Ordered Time Structures. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pp. 464–469, Austin, Texas, USA, Aug, 2000.
   - Mathias Broxvall. The Point Algebra for Branching Time Revisited. In *Proceedings of the Joint German/Austrian Conference on Artificial Intelligence (KI-2001)*, pp. 106–121, Vienna, Austria, Sep, 2001.

II. Mathias Broxvall, Peter Jonsson and Jochen Renz: Disjunctions, Independence, Refinements. *Artificial Intelligence* 140(1–2): 153–173, 2002.

   This article is an extended version of the paper:

   > Mathias Broxvall, Peter Jonsson and Jochen Renz: Refinements and Independence: A Simple Method for Identifying Tractable

Disjunctive Constraints. In *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP-2000)*, pp. 114–127, Singapore, Sep, 2000.

III. Mathias Broxvall. Constraint Satisfaction on Infinite Domains: Composing Domains and Decomposing Constraints. In *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR-2002)*, pp. 509–520, Toulouse, France, Apr, 2002.

IV. Mathias Broxvall. A Method for Metric Temporal Reasoning. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, pp. 513–518, Edmonton, Canada, Jul, 2002.

Introduction

Many problems in computer science and artificial intelligence include a component of temporal or spatial reasoning, cf. Golumbic and Shamir [GS93] for an extensive list of examples from a wide variety of application areas. For this purpose, many different formalisms for modeling and reasoning about real world problems have been proposed. Some of the best known formalisms include Allen's Interval algebra [All83], the region connection calculus [RCC92] and various point algebras [VKvB89, AMR98].

In this thesis we will consider temporal reasoning from a computational perspective. That is, we do not examine or introduce new formalisms for modeling problems but rather examine already established formalisms from a computational perspective. We also demonstrate how these formalisms may be extended and analyze how this affects the computational properties.

The two most common computational tasks when reasoning about temporal or spatial information are that of determining whether the information is consistent and that of deducing new information. For most formalisms these two problems are polynomial time equivalent and we will therefore only consider the satisfiability problem, i.e. deciding whether or not some given information is consistent. If we wish to determine whether a certain relation is entailed by some temporal information, this can easily be done by checking for inconsistency when adding the negation of the entailed relation to the original information.

To give a concrete example of where temporal reasoning may be applied consider the following scenario. Professor Hill, Dr. Green and Mr. Smith all have attended a small conference with three sequential sessions X,Y and Z. We know for sure that there was no break between Y and Z but we have no other knowledge of the order of how X,Y and Z were scheduled. Professor Hill arrived well before the conference and left shortly after session Z, telling Mr. Smith that he felt ill. Dr Green's flight arrived late, and he met Prof.

Basic relation		Example	Endpoints
x before y	b	xxx yyy	$x^+ < y^-$
y after x	a		
x meets y	m	xxxx yyyy	$x^+ = y^-$
y met by x	m^{-1}		
x overlaps y	o	xxxx yyyy	$x^- < y^- < x^+$, $x^+ < y^+$
y overl. by x	o^{-1}		
x during y	d	xxx yyyyyyy	$x^- > y^-$, $x^+ < y^+$
y includes x	d^{-1}		
x starts y	s	xxx yyyyyyy	$x^- = y^-$, $x^+ < y^+$
y started by x	s^{-1}		
x finishes y	f	xxx yyyyyyy	$x^+ = y^+$, $x^- > y^-$
y finished by x	f^{-1}		
x equals y	eq	xxxx yyyy	$x^- = y^-$, $x^+ = y^+$

Table 1: The thirteen basic relations of the interval algebra. The endpoint relations $x^- < x^+$ and $y^- < y^+$ that are valid for all relations have been omitted

Hill in the entrance as Prof. Hill was leaving the conference. Furthermore, we know that Dr. Green attended at least parts of sessions X,Y and that he also talked briefly to Mr. Smith during the conference even though Mr. Smith left the conference early during session X.

The question now is, in what order were the sessions scheduled and could Prof. Hill possibly have attended session X before leaving? To answer this question we can use Allen's interval algebra [All83] which is a temporal formalism which has been extensively studied. Allen's interval algebra is a relational algebra consisting of 13 atomic relations and their disjunctions. The relations of the interval algebra operate over intervals in a real valued domain and are the following: *before (b), after (a), meets (m), met-by (m^{-1}), overlaps (o), overlapped-by (o^{-1}), during (d), includes (d^{-1}), starts (s), started-by (s^{-1}), finishes (f), finished-by (f^{-1})* and *equals (eq)*. An explanation of these atomic relations can be found in Table 1. We can model temporal problems using the Allen algebra by expressing them as interval algebra networks. An interval algebra network is a set of variables and a set of binary constraints over the variables where each constraint is a disjunc-

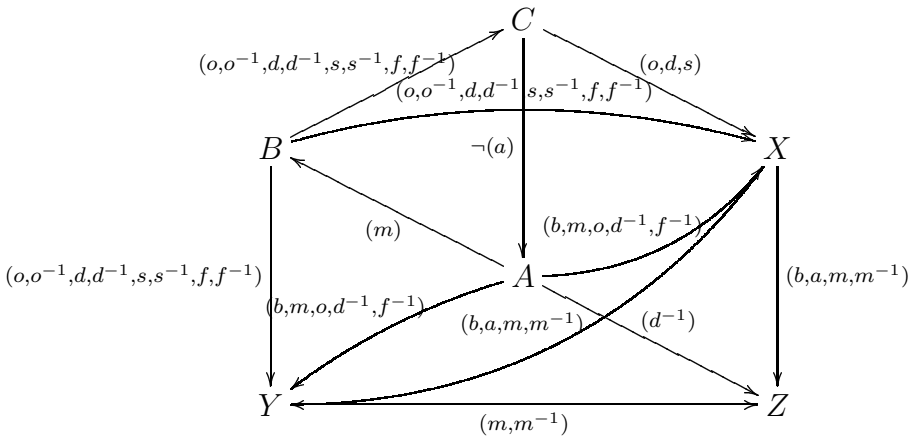


Figure 1: The interval algebra network from the example

tion of the atomic relations. The disjunction of the relations *before (b)* and *after (a)* is written as *(b,a)* and the constraint $x(b, a)y$ is satisfied whenever interval x occurs strictly before or strictly after y.

We can now model this problem using Allen's interval algebra; we let the variables X, Y, Z represent the time intervals of the sessions and A, B, C represent the durations of Prof. Hill, Dr. Green and Mr. Smith's presence at the conference. It is now easy to translate the problem described above into interval constraints: For instance, since we know that there was no break between sessions Y and Z we have the constraint

$$Y\ (m, m^{-1})\ Z$$

which should be interpreted as "interval Y meets or is met-by Z". That is, interval Y either follows directly after interval Z or interval Z follows directly after interval Y. Furthermore, from the problem description we learn that Prof. Hill arrived strictly before session X started but we have no further information of the relationship between interval A and X. We therefore have the constraint:

$$A\ (b, m, o, d^{-1}, f^{-1})\ X$$

since these disjunctive atomic interval relations all fulfill the requirement that A began strictly before X while the other 7 atomic relations do not. In Figure 1 the complete interval algebra network for this problem can be

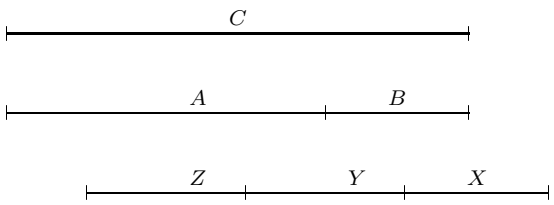


Figure 2: A possible assignment to the intervals

found. There are many different solvers made for testing satisfiability of such networks. Depending on the type of relations occurring in the interval networks different methods can be employed in such solvers. Two commonly occurring methods are to enforce *path-consistency* or to use *backtracking* to non-deterministically choose relations satisfying the constraints. An interval network is said to be path-consistent if every triplet of intervals is consistent. Checking for path-consistency can easily be done in cubic time and for many classes of interval networks this is both a sound and complete method of deciding global consistency. For those cases where path-consistency isn't sufficient for ensuring global consistency we can employ a backtracking method which tries every possible choice of constraining relation between every pair of intervals.

By using a solver employing these two methods we can easily check that the problem described this far is satisfiable and we can identify consistent assignments to the intervals. One such possible assignment can be found in Figure 2 where we clearly see the different schedules of the sessions and the individuals presence at the conference. We can also apply these solvers to deduce further information. For instance, by noting that the set of constraints becomes unsatisfiable if we add the constraint that X occurred during Prof. Hill's presence at the conference

$$X \ (d, s, f) \ A$$

we know for sure that Prof. Hill missed that session. In the same manner we can entail that the interval Z meets Y and that Y meets or comes before X *in all consistent assignments*.

By studying the computational complexity of different formalisms, such as the interval algebra, we get methods for solving these kinds of problems and can evaluate which formalisms are best suited for different domains. Even though we have mainly focused on qualitative temporal reasoning in

this thesis the proof techniques and some of the results are of a more general nature and can be applied to other forms of reasoning. For instance, in papers II and III we apply our results to deduce new tractable classes and to better handle the intractable classes of some spatial algebras and in paper IV we extend a quantitative point based algebra $\mathrm{STP}^{\neq}$ to yield a new tractable and expressive quantitative framework for temporal reasoning.

Contributions

An important aspect of different formalisms are the computational properties of reasoning in the given framework. Since the time in which we seek solutions to our problems generally is restricted, it is worthwhile to classify different frameworks according to their time complexity. We say that problems which can be solved in polynomial time are tractable and call problems in the complexity class **NP** (or higher) intractable. That is, we assume as usual that the classes **NP** and **P** are not the same.

The task of temporal reasoning is typically viewed as a constraint satisfaction problem. Since the general constraint satisfaction problem is intractable, much effort has been made to find necessary and sufficient conditions for ensuring tractability of different constraint satisfaction problems. The two main approaches here are to either identify tractable problems by restrictions on the overall structure of the problem [Fre85, GLS99, PJ97], or to put restrictions on the constraints considered [PJ97, DBvH99, JC95, vBD95]. In the former case it is often the constraint graph of problem instances which must have certain properties such as bipartiteness while in the later case the constraint relation must generally be chosen from a specific set of allowed relations. These two approaches for identifying tractable problems are also used for temporal and spatial reasoning. That is, tractable temporal and spatial reasoning problems are identified either by limitations on *which* variables may be related (cf. Dechter *et al.* [DMP91]) or *how* these variables are constrained (cf. [KJJ01, DJ97, Ren99]). In this thesis we will only consider the later approach, ensuring tractability by restricting the allowed constraints, since this method has proven successful for many different formalisms and since it allows us to make total classifications of tractability. That is, we not only show that the problems considered are tractable when they only contain relations from certain sets, but also that as soon as other relations are considered the problems become intractable.

Since we consider many different formalisms with different restrictions on them we choose to look at all these formalisms from a common constraint satisfaction viewpoint. The computational problem at hand, that of deciding satisfiability, can easily be described as a constraint satisfaction problem with the domain and restrictions on allowed constraints dependent on the formalism. Given an implicit domain and a set Γ of relations over the domain we write CspSat(Γ) for the computational task of deciding whether a set of constraints over the relations in Γ is satisfiable or not. For instance, the domain can be the set of all real numbers R and the relations the full set of 8 relations which can be formed by disjunctions of the 3 atomic point relations *less-than* ($<$), *greater-than* ($>$) and *equality* ($=$), in which case CspSat is a tractable problem. Using this unifying viewpoint of the different formalisms for temporal reasoning we can describe problem instances Π of CspSat(Γ) as a set of variables V and a set of constraints C where each constraint consists of one relation from Γ and one or more variables from V. The problem instance is satisfiable iff there exists a mapping from the variables onto the domain such that for each constraint the image of all the variables in the constraint is part of the constraint's relation. Such a satisfying mapping from the variables onto the domain is called a *model* of the problem instance.

Using this view on the problems at hand it is only natural to consider the question of the complexity when we extend the allowed relations in Γ. Since many formalisms only allow binary constraints, their expressibility can be greatly enhanced by allowing disjunctions of relations in Γ. For instance, by extending various point algebras with disjunctions we have identified many new tractable problems with a good expressibility.

For temporal reasoning, Allen's interval algebra [All83, KJJ01] and the point algebra [VKvB89] have been the most dominant formalisms and are well understood. In terms of the constraint satisfaction viewpoint above the domain for these algebras is the set of all intervals and the set of real numbers respectively. The set of relations we consider are subsets of the full set of 8192 relations which can be formed by disjunctions of the 13 atomic Allen relations and subsets of the set of 8 relations which can be formed by disjunctions of the 3 atomic point relations *less-than* ($<$), *greater-than* ($>$) and *equality* ($=$).

For the interval algebra $\mathcal{IA}$ a complete classification of tractability has been given by Krokhin *et al.* [KJJ01]. Tractability of the full point algebra is easy and it has been shown [Kou96, JB98] that it can be extended with disjunction of disequality while preserving tractability. That is, we can han-

dle in polynomial time not only binary constraints such as $x \leq y$ but also non-binary constraints such as:

$$x \leq y \ \vee \ z \neq w \ \vee \ s \neq t$$

This is interesting since one of the most useful tractable fragments of the interval algebra, Ord-Horn [NB95], can compactly be described as those relations which can be expressed by such disjunctions on the end-points.

One limitation with Allen's interval algebra and the point algebra is that they only consider a linear model of time. However, it is clear that more complex time models are needed in a variety of applications such as the analysis of concurrent and distributed systems, certain planning domains, robot motion problems and cooperating agents [CES86, EH86, McD82, DB88, Lam78, Ang89, ES89, Win89]. Some concrete examples of problems encountered for these applications are presented in [RA98]. A number of alternative time models suitable for these applications have been proposed in the literature. The two perhaps best known models are *partially-ordered* time and *branching* time. The partially-ordered model of time has mainly been used for studying distributed systems (e.g. cooperating agents) [Ang89, Lam86]. The branching time model has been used, for instance, in planning [DB88, McD82] and in the analysis of concurrent systems [ES89]. It should also be noted that several logics based on branching time (such as CTL and CTL*) have been thoroughly investigated, cf. the tutorial by Emerson and Srinivasan [ES88]. Furthermore, the point algebra for branching time has been examined by Düntsch *et al.* [DWM99] from an algebraic point of view. Other examples of interesting time models include the *parallel worlds* model [Lam78], *relativistic time* [Lam86] and the *directed intervals* model [Ren01].

In this thesis we investigate the computational tractability of the point algebra for some of these different time domains. Similarly to the point algebra for linear time the point algebra for partially ordered time is a relational algebra consisting of a number of atomic relations. The domain is that of partially ordered points and the atomic relations are *less-than* ($<$), *greater-than* ($>$), *equality* ($=$) and *parallel* ($\|$). Branching time is defined similarly to partially ordered time but with the added restriction that the partial orders which constitute the interpretations must have a tree structure. For partially ordered time with bounded dimensions the partial orders under consideration must also be of a fixed finite dimension.

We choose to investigate the point algebra rather than the interval algebra and extend it with disjunctions for several reasons. Most importantly

because of the complexity of the interval based approach for nonlinear time models, the interval algebra for branching and partially ordered time contains 2^{19} and 2^{29} relations respectively [AR96]. Using this approach we also have high hopes of being able to compactly express the tractable interval relations as disjunctions of end-point relations. For instance, since we can handle disjunctions of disequality in polynomial time for partially ordered time, the equivalent of the Ord-Horn fragment also exists for this time model. Furthermore, by allowing arbitrary disjunctions we can also express constraints which cannot be handled by an interval based approach (without disjunctions).

In paper I we present a total classification of tractability for these point algebras. We do this for several reasons. Firstly, by identifying several tractable classes for each algebra we can choose to model problems using only relations from these classes and thus be certain that the problems can be solved efficiently. Secondly, even when the problems under consideration cannot be modeled using only relations from the tractable sets, knowledge of which sets of relations are tractable can be used to enhance a backtracking based solver. It is faster to backtrack over the relations in some tractable set of relations rather than over the primitive relations of the domain. Thirdly, by using some of the methods presented later in this thesis it is possible to combine these tractable classes to yield new even more expressive constraint languages, sometimes even extended with disjunctions of constraints from multiple domains. Last but not least, by proving that these tractable classes are the only tractable cases we gain a better understanding of the underlying problem and know that there is no need to put further effort into discovering new tractable classes for these domains.

For the purpose of finding tractable disjunctive extensions of tractable problems, the k-independence property [CJJK01] has proven useful. This is a property of relations which allows us to extend tractable sets of relations with disjunctions. For instance, since disequality has been proven 1-independent of the other point relations for linear time, conjunctions of constraints of the form $x_1 \leq y_1 \vee x_2 \neq y_2 \vee \cdots \vee x_n \neq y_n$ can be solved in polynomial time. This is very useful since such constraints can express, for instance, every relation in one of the most interesting tractable fragments of Allen's interval algebra, the Ord-Horn fragment. As we see in paper II the k-independence property can not only be used for proving tractability of constraints but also intractability of certain disjunctive constraints.

Considering the *ad hoc* nature of the many tractability classifications

which have been made for various formalisms it is worthwhile developing more systematic tools for proving tractability or intractability of sets of relations. For the classifications of tractability in paper I we develop a few methods for proving intractability. In paper II we look closer at the concept of refinement which is a way of using sets of relations decided by path consistency to automatically identify new tractable sets of relations. We develop a connection between this method and the concept of independence in order to provide an automatic tool for finding tractable sets of disjunctive relations. A few more results for determining tractability can be found in paper III where we look at the case of combining disjoint domains.

Since it is not always possible to restrict the relations used when modeling problems to ensure tractability we also look at methods to more efficiently solve intractable cases. One such method is to employ a property of relations called the *partitioning* property which we introduce in paper III. Using this property it is possible to decompose complex intractable problems into simpler problems and thus able to solve larger problems.

For certain tasks it is not sufficient to only model the problems in a qualitative algebra. Instead, a quantitative algebra allowing metric information is needed. There exist many different quantitative formalisms for temporal reasoning and the augmented interval algebra suggested by Condotta [Con00] is one of them. In this framework the interval algebra of Allen has been extended with metric information between the endpoints of intervals. By using a fairly complex algorithm that repeatedly propagates information between the qualitative and quantitative parts of a problem instance; it is possible to determine satisfiability of problem instances in this framework whenever their qualitative part belongs to a tractable fragment of the interval algebra.

In the last paper of this thesis we demonstrate an alternative approach with a greater expressibility than the augmented interval algebra and with a simpler and faster algorithm for deciding satisfiability. This approach is based on the $\mathrm{STP}^{\neq}$ framework of Gerevini and Cristani [GC97] which we extend with disjunctions of disequality. This way we get a point algebra STP^{*} which has both a fast and simple algorithm for deciding satisfiability as well as a good expressibility subsuming the augmented Ord-Horn fragment of the interval algebra.

Structure of the Thesis

In this introductory part of the thesis we have first presented a very brief overview of our results in the context of earlier works within the field of temporal and spatial reasoning. We continue with more on the results of each separate paper of the thesis.

The rest of the thesis consists of four independent papers containing all the definitions and proofs needed for the results of each paper. The papers are presented in approximately the same order as they have been written but since papers I and II are extended versions of several conference papers, the order of some of the results can be somewhat non-obvious. The first two conference papers contained in paper I were written before paper II and the last one after. Although this means that there are references in both directions between the conference papers constituting papers I and II, it is recommended to read paper I first. The other papers in this thesis can be read in any order but it may be beneficial to read paper I before paper III and paper II before paper IV.

Summary of the Papers

We give here a brief summary of the topic and the results of each paper included in this thesis. We try to do this on a sufficiently abstract level not to require the actual definitions needed for the formal results but a quick glance at the definitions of the full papers might come in handy for a reader not familiar with the field.

Paper I - Point Algebras for Temporal Reasoning: Algorithms and Complexity

In this paper we examine the point algebras for different domains. By extending them to allow for disjunctions we can express non-binary constraints and more complex relations. For instance, by using disjunctions of constraints between endpoints of intervals, we can express interval relations. We give a total classification of tractability for the point algebra extended with disjunctions for the following time models:

1. Totally ordered time.
2. Partially ordered time.
3. Partially ordered time with bounded dimension.
4. Branching time.

For the first domain, the full point algebra (without disjunctions) is tractable and when extended with disjunctions there are exactly two tractable classes.

When looking at partially ordered time the picture becomes a little bit more complicated since the point algebra is not tractable. When disjunctions are not allowed we have three different tractable subsets of the relations and when extended with disjunctions we have a total of four different maximal tractable sets.

In some real-world problems involving e.g. unsynchronized clocks, time can be modeled as a partial order of bounded dimension. However, the computational complexity of such problems becomes very hard. For this time model, even the set of basic point relations is NP-hard. We demonstrate that this formalism has exactly three maximal tractable sets of relations for the point algebra case and three corresponding maximal sets for the disjunctive case.

Finally, the case of branching time is also of interest since the full (binary) point algebra is tractable although it has been demonstrated that k-consistency for any k is not sufficient to determine global consistency. This is interesting since path consistency is sufficient to determine consistency for all the other tractable sets of binary relations considered in this paper. Also, although this algebra when extended with disjunctions has five maximal tractable sets of relations, the full set of binary relations cannot be extended with disjunctions tractably. Apart from making a classification of tractability for this formalism we also suggest an improved algorithm for deciding consistency of the full point algebra. This new algorithm runs in $O(n^{3.376})$ time which should be compared with the previous algorithm by Hirsch [Hir97] running in $O(n^5)$ time.

We also exhibit a connection between the point algebra for partially ordered time and the Λ-problem which is a simple qualitative algebra for spatial reasoning. Using this connection we can easily translate the tractability results from the point algebra to the Λ-problem.

Paper II - Disjunctions, Independence, Refinements

Here we investigate disjunctions of constraints more deeply. We consider three properties of relations known as the guaranteed satisfaction (GS) property, 1-independence and 2-independence [CJJK01]. Let Γ and Δ be two sets of relations such that the CSP problem over $\Gamma \cup \Delta$ is tractable. In short, we prove the following:

- Let the set Δ^* contain all possible finite disjunctive relations over Δ. The CSP problem for this set is tractable if and only if Δ has the GS property.
- Let the set $\Gamma^{\veebar}\Delta^*$ contain all disjunctive relations over $\Gamma \cup \Delta$ where relations in Γ are allowed to appear at most once in a disjunction (compare with the Horn fragment of propositional logic). The CSP problem for this set is tractable if and only if Δ is 1-independent of Γ.
- Consider the set $\Gamma \cup \Delta^2$ where Δ^2 contains all disjunctive relations over Δ containing at most two disjuncts (compare with the Krom fragment of propositional logic). The CSP problem for this set is tractable if and only if Δ is 2-independent of Γ.

Considering that these results allows us to decide tractability or intractability of different sets of relations it would be useful to have an automatic method to identify the GS, 1- and 2-independence properties. We look at a connection between the concept of refinements, a method for automatically identifying sets of relations that are decided by path consistency, and 1-independence. We demonstrate how this connection can be used to automatically identify relations having the 1-independence property. This is a powerful method which allows us to automatically identify, for instance, all the tractable sets of relations for the point algebras (with disjunctions) for totally ordered and partially ordered time.

To see the applicability of these results we continue with the example given in the introduction. Using the automatic method given in this paper it can be noted that the relation *not-equal* is 1-independent of the tractable Ord-Horn fragment of Allen's interval algebra. Thus we can tractably solve problems involving constraints such as: "Either Mr. Smith came to the conference before Prof. Hill *or* session W and session X were not scheduled at the same time".

Paper III - Constraint Satisfaction on Infinite Domains: Composing Domains and Decomposing Constraints

Much effort has been spent on discovering tractable sets of relations for relational algebras such as Allen's interval algebra [KJJ01], various point algebras [Kou96, JB98, AMR98] and spatial algebras such as RCC-5 and RCC-8 [Ren99]. There has also been some effort to combine tractable sets of relations from different domains to yield new tractable cases. For instance, Cohen *et al.* [CJG00] demonstrate how tractable classes from disjoint domains can be combined using the multiple relational union operator ($\bowtie$).

In this paper we demonstrate how several properties such as decidability by path-consistency and independence are preserved when combining disjoint domains with the $\dot{\bowtie}$ operator. We also introduce a new property of relations called the partitioning property. This property can be used to decompose complex problems over one or several domains into smaller subproblems, something which is especially useful when dealing with hard constraint problems.

For instance, if we were to extend the example from the introduction further by adding variables and complex constraints on when different persons attended last year's conference, common sense tells us that we should be able to solve the problem in two parts, first handling all the constraints on events from last year and then the constraints for this year. If there are no constraints at all between the original variables and the variables from last year's conference, this is trivial to realize since the problem evidently consists of two disjoint parts. However, even if we have constraints such as "Session A was scheduled before session X" where A is a session in last year's conference, we can solve the problem as two disjoint subproblems since "before" is a partitioning relation in Allen's interval algebra.

By noting that the independence property is preserved when we combine disjoint domains we can also solve constraints involving disjunctions and multiple domains such as:

$$I_1 \textit{ comes before } I_2 \textit{ or } R_1, R_2 \textit{ are disjoint}$$

where I_1, I_2 represent time intervals (e.g. the duration of Dr Green's and Mr. Smiths presence at some conference) and R_1, R_2 represent regions (e.g. *which* conference they attended).

Furthermore, we run some tests for evaluating the efficiency of using this method on different domains such as Allen's interval algebra, the point algebra for various time domains and two spatial algebras, RCC-5 and RCC-8. Since it has been noted [Wal01] that real-world constraint problems rarely have a completely random underlying structure, we choose to run these tests not only on purely random problem instances but also instances with some structure. Many different methods for generating random problem instances have been suggested in the literature [BA99, WS98, Hog96]. For these investigations we choose to generate purely random constraint graphs, graphs with a powerlaw [BA99] distribution and smallworld [WS98] graphs. As can be expected these tests indicate that the decomposition method is most successful for domains with a large number of relations with the partitioning property and for problem instances with a powerlaw or smallworld graph structure.

Paper IV - A Method for Metric Temporal Reasoning

Several methods for metric temporal reasoning using a linear model of time have been proposed. One method is Horn Disjunctive Linear Relations [JB98, Kou01], Horn DLRs for short. This method is generally considered hard to implement since it builds on fairly complicated polynomial time algorithms for linear programming. One method which has been suggested as an alternative is to augment traditional qualitative reasoning problems such as Allen's interval algebra and the rectangle algebra with metric information [Con00]. In this paper, we present a new point-based approach STP* for metric temporal reasoning which subsumes the augmented interval algebra in terms of expressibility but is subsumed by Horn DLRs. The advantage of using this method over both the augmented interval algebra and Horn DLRs is that it is very fast and easy to implement. We run a number of tests on random problem instances and note that the practical cost of the algorithm is fairly low. Note that altough similar in definition to the temporal constraint formalism of Stergiou and Koubarakis [SK00] the expressibility of STP* and their formalism are only partially overlapping.

A natural example of where these kind of metric constraints can be used is to extend the example of the conference attendees with metric information such as: Mr. Smith arrived at the conference site three hours before session Y and there was a five minute coffee break between sessions X and Y.

Future work

We will here recapitulate on some of the more central questions raised in this thesis. In paper II we have shown three properties to be essential and sufficient for tractability of three different forms of sets of relations. Since we have also provided results simplifying the identification of these properties, this makes it easier to demonstrate tractability or intractability of new sets of relations. Since there is at least one more interesting set of relations not covered by our results we pose the following open questions:

Open question 1. Assume CspSat($\Gamma \cup \Delta$) is tractable. What is a necessary and sufficient condition for tractability of CspSat($\Gamma^{\veebar}\Delta$)?

Applying our automatic method of deciding the 1-independence property on the point algebras for totally-ordered and partially-ordered time detects all tractable sets of disjunctive relations. Since we have neither been able to prove nor disprove this in the general case, this naturally raises the following question:

Open question 2. Assume path-consistency decides CspSat(Γ) and let M^{Δ} be the matrix and Check-Refinements the algorithm defined in paper II. Is it true that $\Delta \subseteq \Gamma$ is 1-independent of Γ if and only if the algorithm Check-Refinements(Γ, M^{Δ}) returns succeed?

Since the refinement based method is restricted to binary relations only and path-consistency must decide the underlying CspSat problem, it would be of interest to find a more general algorithm without these restrictions.

Open question 3. Given arbitrary sets Γ,Δ of relations, is there an algorithm for deciding whether Δ is 1-independent of Γ or not?

Of course, this question could be generalized and it would be of interest to find an automatic method for deciding k-independence for arbitrary k. In our classifications in paper I we have come a very long way using 1- and 2-independence for proving tractability and intractactability. We therefore pose also this open question:

Open question 4. Given arbitrary sets Γ,Δ of relations, is there an algorithm for deciding whether Δ is 2-independent of Γ or not?

Note that there exists a trivial automatic but not complete method for proving that Δ is not 2-independent of Γ by generating problems and testing for satisfiability. This method can, for instance, be used for most of the NP-hardness proofs for relations in paper I involving disjunctions.

One interesting follow up of these investigations would be to develop an automatic tool aiding classifications of new domains by using the refinement and independence based methods for discovering tractable and intractable sets of relations as well as the methods of paper I for making total classifications when disjunctions are allowed. Such a tool could greatly ease the task of classifying new domains and constructing efficient solvers for them.

In paper III we present the partitioning property of relations which can be used to enhance backtracking based solvers and give a method to automatically identify this property when path consistency is sufficient to determine consistency for some basic relations. This naturally raises the question whether there exists an automatic method for identifying this property in the general case.

Open question 5. Given arbitrary sets Γ,$\varphi \subset \Gamma$ of relations, is there an algorithm for deciding whether φ partitions Γ or not?

The partitioning property cannot only be used on qualitative algebras but also for quantitative ones. For instance, if we consider STP constraints (and their closure under intersection and composition) as intervals of real numbers and define $\varphi = \{]-\infty, r], [r, +\infty[| r \in R\}$ we can prove that φ partitions the full set of STP (and $\text{STP}^{\neq}$) constraints. Thus we can employ the decomposition method to easier solve metric point algebra problems involving disjunctions. This makes the STP^* method of paper IV even more promising and it would be interesting to develop a solver which backtracks on arbitrary constraints involving disjunctions of $\text{STP}^{\neq}$ constraints until only STP^* constraints are left. By employing the decomposition method this solver could probably be made very efficient.

Bibliography

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[AMR98] Frank D. Anger, Debasis Mitra, and Rita V. Rodriguez. Temporal constraint networks in nonlinear time. Technical report, ECAI Workshop on Temporal and Spatial Reasoning, Brighton, UK, 1998.

[Ang89] Frank D. Anger. On Lamport's interprocessor communication model. *ACM Transactions on Programming Languages Systems*, 11(3):404–417, 1989.

[AR96] Frank Anger and Rita Rodriguez. The lattice structure of temporal interval relations. *Journal of Applied Intelligence*, 6(1):29–38, 1996.

[BA99] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[CES86] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages Systems*, 8(2):244–263, 1986.

[CJG00] David Cohen, Peter Jeavons, and Richard Gault. New tractable classes from old. In *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming*, pages 160–171, Singapore, 2000.

[CJJK01] David Cohen, Peter Jeavons, Peter Jonsson, and Manolis Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47(5):826–853, 2001.

[Con00] Jean-Francois Condotta. The augmented interval and rectangle networks. In *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*, pages 571–579, Trento, Italy, 2000. Morgan Kaufmann Publishers.

[DB88] Thomas Dean and Mark Boddy. Reasoning about partially ordered events. *Artificial Intelligence*, 36:375–399, 1988.

[DBvH99] Yves Deville, Olivier Barette, and Pascal van Hentenryck. Constraint satisfaction over connected row convex constraints. *Artificial Intelligence*, 109(1–2):243–271, 1999.

[DJ97] Thomas Drakengren and Peter Jonsson. Eight maximal tractable subclasses of Allen's algebra with metric time. *Journal of Artificial Intelligence Research*, 1997.

[DMP91] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial Intelligence*, 49(1-3):61–95, 1991.

[DWM99] Ivo Düntsch, Hui Wang, and Stephen McCloskey. Relations algebras in qualitative spatial reasoning. *Fundamenta Informaticae*, 39(3):229–248, 1999.

[EH86] Allen E Emerson and Joseph Y Halpern. "Sometimes" and "not never" revisited: On branching versus linear temporal logic. *Journal of the ACM*, 33(1):151–178, January 1986.

[ES88] E. Allen Emerson and Jai Srinivasan. Branching time temporal logic. In *Proceedings of REX Workshop 1988*, pages 123–172, 1988.

[ES89] E. Allen Emerson and Jai Srinivasan. Branching time temporal logic. In J. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, pages 123–172, New York, 1989. Springer-Verlag.

[Fre85] Eugene C. Freuder. A sufficient condition for backtrack-bounded search. *Journal of the ACM*, 32:755–761, 1985.

[GC97] Alfonso Gerevini and Matteo Cristani. On finding a solution in temporal constraint satisfaction problems. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI 97)*, pages 1460–1465, Nagoya, Japan, 1997.

[GLS99] Georg Gottlob, Nicola Leone, and Francesco Scarcello. A comparison of structural CSP decomposition methods. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 394–399, Stockholm, Sweden, 1999.

[GS93] Martin Charles Golumbic and Ron Shamir. Complexity and algorithms for reasoning about time: A graph-theoretic approach. *Journal of the ACM*, 40(5):1108–1133, 1993.

[Hir97] Robin Hirsch. Expressive power and complexity in algebraic logic. *Journal of Logic and Computation*, 7(3):309–351, 1997.

[Hog96] Tad Hogg. Refining the phase transition in combinatorial search. *Artificial Intelligence*, 81(1-2):127–154, 1996.

[JB98] Peter Jonsson and Christer Bäckström. A unifying approach to temporal constraint reasoning. *Artificial Intelligence*, 102(1):143–155, 1998.

[JC95] Peter Jeavons and Martin C. Cooper. Tractable constraints on ordered domains. *Artificial Intelligence*, 79:327–339, 1995.

[KJJ01] Andrei Krokhin, Peter Jeavons, and Peter Jonsson. Reasoning about temporal relations: The tractable subalgebras of Allen's interval algebra. Technical Report PRG-RR-01-12, Computing Laboratory, Oxford University, 2001. Available from `web.comlab.ox.ac.uk/oucl/publications/tr/rr-01-12.html`.

[Kou96] Manolis Koubarakis. Tractable disjunctions of linear constraints. In *Proceedings of the 2nd Conference on Principles and Practice of Constraint Programming (CP'96)*, pages 297–307, Boston, MA, 1996.

[Kou01] Manolis Koubarakis. Tractable disjunctions of linear constraints: basic results and applications to temporal reasoning. *Theoretical Computer Science*, 266(1-2):311–339, 2001.

[Lam78] Leslie Lamport. Time, clocks and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, 1978.

[Lam86] Leslie Lamport. The mutual exclusion problem: Part I—a theory of interprocess communication. *Journal of the ACM*, 33(2):313–326, 1986.

[McD82] Drew McDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.

[NB95] Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *Journal of the ACM*, 42(1):43–66, 1995.

[PJ97] Justin Pearson and Peter Jeavons. A survey of tractable constraint satisfaction problems. Master Thesis Report CSD-TR-97-15, Royal Holloway Univ. of London, 1997.

[RA98] Rita V. Rodriguez and Frank. D. Anger. Using constraint propagation to reason about unsynchronized clocks. *Constraints*, 2:191–202, 1998.

[RCC92] David A. Randell, Zhan Cui, and Anthony G. Cohn. A spatial logic based on regions and connection. In *Proceedings of the 3rd International Conference on Principles on Knowledge Representation and Reasoning (KR-92)*, pages 165–176, Cambridge, MA, USA, October 1992. Morgan Kaufmann.

[Ren99] Jochen Renz. Maximal tractable fragments of the region connection calculus: A complete analysis. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 448–454, Stockholm, Sweden, 1999.

[Ren01] Jochen Renz. A spatial odyssey of the interval algebra: 1. Directed intervals. In *Proceedings of the 17th International Joint*

Conference on Artificial Intelligence (IJCAI-2001), pages 51–56, Seattle, WA, USA, 2001.

[SK00] Kostas Stergiou and Manolis Koubarakis. Backtracking algorithms for disjunctions of temporal constraints. *Artificial Intelligence*, pages 81–117, 2000.

[vBD95] Peter van Beek and Rina Dechter. On the minimality and decomposability of row-convex constraint networks. *Journal of the ACM*, 42(3):543–561, 1995.

[VKvB89] Marc B. Vilain, Henry A. Kautz, and Peter G. van Beek. Constraint propagation algorithms for temporal reasoning: A revised report. In Daniel S. Weld and Johan de Kleer, editors, *Readings in Qualitative Reasoning about Physical Systems*, pages 373–381. Morgan Kaufmann, San Mateo, CA, 1989.

[Wal01] Toby Walsh. Search on high degree graphs. In *Proceedings of the 17th International Joint Conferance on Artificial Intelligence (IJCAI-2001)*, pages 266–271, Seattle, WA, USA, 2001.

[Win89] Glynn Winskell. An introduction to event structures. In J. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, pages 364–397, 1989.

[WS98] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

Paper I

Point Algebras for Temporal Reasoning: Algorithms and Complexity

Mathias Broxvall and Peter Jonsson

Abstract

We investigate the computational complexity of temporal reasoning in different time models such as totally-ordered, partially-ordered and branching time. Our main result concerns the satisfiability problem for point algebras and point algebras extended with disjunctions—for these problems, we identify all tractable subclasses. We also provide a number of additional results; for instance, we present a new time model suitable for reasoning about systems with a bounded number of unsynchronized clocks, we investigate connections with spatial reasoning and we present improved algorithms for deciding satisfiability of the tractable point algebras.

Contents

1 Introduction

Reasoning about temporal knowledge is a common task in computer science and elsewhere, cf. Golumbic and Shamir [GS93] for an extensive list of examples from a wide variety of application areas. In most applications, knowledge of temporal constraints is expressed in terms of collections of relations between time points and/or time intervals. Typical reasoning tasks include determining the satisfiability of such collections and deducing new relations from those that are known.

Research on automated reasoning about temporal constraints has largely concentrated on simple, linear models of time and the computational properties of such constraints have been investigated in depth. For example, it is known that the point algebra is tractable [VKvB89] and all tractable fragments of the point-interval and interval-interval relations have been identified [JDB99, KJJ01]. However, it has been observed many times that more complex time models are needed in, for instance, the analysis of concurrent and distributed systems, certain planning domains, robot motion problems and cooperating agents [Ang89, CES86, DB88, EH86, ES89, Lam78, McD82, Win89]. Typical problems include the traditional difficulties of synchronization and communication problems, network and bus protocol correctness and decision making in the absence of a global clock. Some concrete examples of these problems are presented in [RA98].

Consequently, a number of alternative time models have been proposed and the two most well-known are *partially-ordered* time and *branching* time. The partially-ordered model of time has mainly been used for studying distributed systems (e.g. cooperating agents) [Ang89, Lam86]. The branching time model has been used, for instance, in planning [DB88, McD82] and in the analysis of concurrent systems [ES89]. It should also be noted that several logics based on branching time (such as CTL and CTL*) have been thoroughly investigated, cf. the tutorial by Emerson and Srinivasan [ES88]. Furthermore, the point algebra for branching time has been examined by Düntsch *et al.* [DWM99] from an algebraic point of view. Other examples of interesting time models include the *parallel worlds* model [Lam78], *relativistic time* [Lam86] and the *directed intervals* model [Ren01].

Most of the previous research on non-standard time models has concentrated on modeling properties and theoretical foundations while computational aspects have received relatively little attention. There are a few papers that address such questions, though. Clarke *et al.* [CES86] have provided

an apparently efficient algorithm that can check the correctness of certain circuits and algorithms. The method builds on a propositional branching-time logic based on CTL (Computation Tree Logic). CTL (or extended versions of it) is frequently used in model-checking—a good survey can be found in [McM92]. Anger *et al.* [AMR98] have studied whether standard constraint-based approaches can be applied to reasoning about nonlinear time; their results show that this is possible in certain restricted cases. Computational properties of the point algebra for branching time have been investigated by Hirsch [Hir97]. He shows, among other things, that the full point algebra is tractable (satisfiability can be decided by an $O(n^5)$ time algorithm) but k-consistency (for arbitrary k) is insufficient for determining satisfiability. This can be compared to many other relational algebras where path consistency decides consistency for the tractable sets of relations, for instance, the point algebra for totally ordered time [VKvB89] and all known tractable subclasses of RCC-5 [BJR00, Paper II] and RCC-8 [Ren99].

The aim of this paper is to study different time models from a computational perspective. More specifically, we do the following:

1. We identify all tractable subclasses of the point algebra for partially-ordered time and we identify all tractable subclasses of the point algebra extended with disjunctions for partially-ordered, totally-ordered and branching time. Note that it has been proven earlier that the full point algebras for totally-ordered and branching time are tractable [VKvB89, Hir97].

2. We argue that a certain time model, *partially ordered time with bounded dimension*, is suitable when reasoning about systems containing a finite number of unsynchronized clocks. We show that reasoning in this model is unusually hard when the dimension is > 1 (since even the fragment only containing basic relations is NP-complete) and give a full classification of tractability for the point algebra as well as for the point algebra extended with disjunctions.

3. We demonstrate how our results can be used for proving complexity results in other domains. As a concrete example, we use our results on partially-ordered time for studying a simple spatial reasoning formalism (whose objects consists of non-degenerate two-dimensional convex sets). By exhibiting a connection between these two formalisms, we ar-

rive at a complete classification of tractability in the spatial formalism almost for free.

4. Finally, we present an improved algorithm for the full point algebra for branching time running in $O(nM(n))$ time where $M(n)$ is the time complexity of multiplying two $n \times n$ integer matrices. Coppersmith and Winograd [CW90] have shown that there exists such an algorithm running in $O(n^{2.376})$ time. By using their algorithm, our algorithm runs in $O(n^{3.376})$ time which is comparable to that of path consistency checking algorithms, $O(n^3)$. It is thus a significant improvement over Hirsch's [Hir97] $O(n^5)$ algorithm.

The reader may rightfully ask why we give so much attention to disjunctive extensions of point algebras. Instead of considering all possible subclasses of a temporal algebra, one can concentrate on subclasses that are 'interesting' in some sense. Whether disjunctive subclasses are interesting or not is of course a matter of taste but we can provide some evidence that they are worth studying. First, simple constraint languages extended with disjunctions have historically proved to have appealing properties. For instance, the Horn DLRs [JB98] which subsumes almost all previously presented temporal languages for total-ordered time can be viewed as a point algebra extended with disjunctions. Several other similar examples are given in Cohen *et al.* [CJJK01]. Secondly, disjunctions can compactly describe complex relations. Consider for example the ORD-Horn algebra [NB95] which is a tractable subclass of Allen's algebra. It contains 868 different relations and is consequently quite difficult to remember. Defining the ORD-Horn with the aid of disjunctions is much easier: ORD-Horn contains exactly the Allen relations which can be expressed by disjunctions of the form $x_1 \leq y_1 \vee x_2 \neq y_2 \vee \ldots \vee x_n \neq y_n$. Interval algebras for time models other than totally-ordered time have been studied in [AR96]. They show, for instance, that there are 19 and 29 basic relations in the interval algebra for branching and partially-ordered time, respectively. The large number of relations (2^{19} and 2^{29}) in the full algebras clearly motivates concise descriptions of tractable subclasses.

Another reason is that we would like to answer open questions posed by Broxvall *et al.* [BJR02, Paper II]. Their paper provides necessary and sufficient conditions for tractability of two types of disjunctive constraint languages ($\Gamma^{\times}_{\vee}\Delta^*$ and $\Gamma \cup \Delta^2$, the exact definitions can be found in Section 2). One of the open questions is whether there exist tractable disjunctive languages of other 'types' (such as $\Gamma^{\times}_{\vee}\Delta$ is tractable, but $\Gamma^{\times}_{\vee}\Delta^*$ is not) or not.

To answer such questions, we need to thoroughly analyze disjunctive constraints. In an attempt to be systematic in our search, we consider point algebras with very different computational properties.

1. tractable point algebras that are solvable by enforcing path-consistency (totally-ordered time);
2. tractable point algebras that are not solvable by enforcing path-consistency (branching time);
3. intractable point algebras with a tractable fragment containing all basic relations (partially-ordered time); and
4. intractable point algebras without a tractable fragment containing all basic relations (partially-ordered time with bounded dimension).

The findings made in this article support the conjecture that there are just a few classes of tractable disjunctive constraint languages — no 'structurally' new constraint classes were discovered.

Yet another reason is that disjunctive constraint classes can effectively be embedded in different logics (where the disjunction operator typically is a key ingredient in the logic language). Very clear examples of this approach have been given by Drakengren and Bjäreland [DB99a, DB99b] who show how expressive temporal logics for reasoning about action and change can be based on disjunctive constraints.

The paper is structured as follows: Section 2 contains basic definitions and some auxiliary results. Sections 3–6 contains the results for totally- and partially-ordered time, partially ordered time with bounded dimension and branching time, respectively. Finally, some concluding remarks are collected in Section 7. Parts of this article have previously appeared in three conference papers [Bro01, BJ99, BJ00].

2 Preliminaries

2.1 Point algebras

A point algebra is based on the notion of *relations* between pairs of variables interpreted over a partially-ordered set. We consider four *basic relations* which we denote by $<, >, =$ and $\|$. If x, y are points in a partial order

$(T, \leq)$, then we define these relations in terms of the partial ordering $\leq$ as follows:

1. $x < y$ iff $x \leq y$ and not $y \leq x$
2. $x > y$ iff $y \leq x$ and not $x \leq y$
3. $x = y$ iff $x \leq y$ and $y \leq x$
4. $x \| y$ iff neither $x \leq y$ nor $y \leq x$

These basic relations are a set of jointly exhaustive and pairwise disjoint (JEPD) relations. The complete set of relations we consider are disjunctions of basic relations and they are represented as unions of basic relations. Since we have 4 different basic relations we get $2^4 = 16$ possible relations. The set of basic relations is denoted $\mathcal{B}$ and the set of all 16 relations is denoted by $\mathcal{PA}$. We use a self-explanatory, shorthand notation for relations, for example, $< \cup =$ is written as $\leq$ and $= \cup \|$ as $\underline{\|}$. The empty relation is denoted by $\perp$ and is usually omitted in our definitions of the tractable sets of relations for clarity. Note that a tractable set of relations extended with the empty relation is still tractable.

The basic computational problem is the satisfiability problem where we have a set of variables, a set of constraints over the variables and the question is whether there exists a mapping from the variables to some partial order such that all constraints are satisfied.

Definition 1 Let $X \subseteq \mathcal{PA}$ be a set of point relations and p a class of partial orders. A problem instance $\Pi = (V, C)$ of $\textsc{Psat}_p(X)$ is a set of variables V and a set of binary constraints C of the form xry where $x, y \in V$ and $r \in X$. A tuple $\langle f, \langle T, \leq \rangle \rangle$ where $f : V \to T$ is a total function and $(T, \leq) \in p$ is called an interpretation of Π.

A problem instance Π is satisfiable iff there exists an interpretation $M = (f, (T, \leq))$ such that $f(x)\ r\ f(y)$ holds for every constraint xry in C. Such an M is called a *model* of Π.

We note once and for all that $\textsc{Psat}_p(X)$ is in NP for all choices of p and X considered in this paper. Given a problem instance $\Pi = (V, C)$, let $\mathrm{Var}(\Pi) = V$. By slightly abusing the notation, we write $c \in \Pi$ to denote the fact that the constraint c is a member of C. The size of a problem instance can

either be regarded as the total number of variables and constraints or (as is common when studying algorithms for enforcing path consistency) simply the total number of variables. We choose the later approach since this gives stronger time bounds than the alternative definition.

A point algebra $\mathcal{A}$ consists of the 16 possible relations between points together with the operations *converse* $\cdot^{-1}$, *intersection* $\cap$ *and composition* $\circ$ which are defined as follows:

$$\forall x, y: \ xr^{-1}y \Leftrightarrow yrx$$

$$\forall x, y: \ x(r \cap s)y \Leftrightarrow xry \text{ and } xsy$$

$$\forall x, y: \ x(r \circ s)y \Leftrightarrow \exists z: \ (xrz \text{ and } zsy)$$

It follows that the converse of $r = b_1 \cup \ldots \cup b_n$ is equal to $b_1^{-1} \cup \ldots \cup b_n^{-1}$. The intersection of two relations can be expressed as the usual set-theoretic intersection. Using the definition of composition, it can be derived that given two relations $r = b_1 \cup \ldots \cup b_n$ and $r' = b'_1 \cup \ldots \cup b'_m$,

$$r \circ r' = \bigcup \{b_i \circ b'_j \mid i \leq n \text{ and } j \leq m\}.$$

Given a set X of relations, we define the *closure* of X, $C(X)$, as the least set X' of relations closed under converse, intersection and composition and having the property $X \subseteq X'$. We say that $X \subseteq \mathcal{PA}$ is a *subalgebra* if $C(X) = X$. We have the following result [RN99].

Theorem 2 $\textsc{Sat}_P(X)$ is tractable (resp. NP-complete) iff $\textsc{Sat}_P(C(X))$ is tractable (resp. NP-complete).

We say that a set of relations X is a *maximal tractable* subclass iff $\textsc{Sat}_p(X)$ is tractable and there exist no X' such that (1) $X \subset X' \subseteq \mathcal{PA}$; and (2) $\textsc{Sat}_p(X')$ is tractable. If X is a maximal tractable subclass, then $C(X) = X$.

2.2 Disjunctions

We will now show how to extend point algebras with disjunctions. This section contains the basic definitions together with some results needed for proving the classification theorems.

Definition 3 Let R_1, R_2 be two relations of arity i, j over a domain D and define the disjunction $R_1 \vee R_2$ of arity $i + j$ as follows:

$$R_1 \vee R_2 = \{(x_1, \ldots, x_{i+j}) \in D^{i+j} | (x_1, \ldots, x_i) \in R_1 \vee (x_{i+1}, \ldots, x_{i+j}) \in R_2\}$$

Thus, the disjunction of two relations with arity i, j is the relation with arity $i+j$ satisfying either of the two relations. To give a concrete example of how the disjunction operator is used, consider the following example.

Example 2.1 Let $D = \{0, 1\}$ and the relations $\mathsf{And} = \{(1, 1)\}$ and $\mathsf{Xor} = \{(0, 1), (1, 0)\}$ be given. The disjunction of And and Xor is:

$$\mathsf{And} \vee \mathsf{Xor} = \left\{ \begin{array}{c} (0,0,0,1), (0,1,0,1), (1,0,0,1), (1,1,0,1), \\ (0,0,1,0), (0,1,1,0), (1,0,1,0), (1,1,1,0), \\ (1,1,0,0), (1,1,0,1), (1,1,1,0), (1,1,1,1) \end{array} \right\}$$

We see that the constraint x And $y \vee x$ Xor z encoded by $(\mathsf{And} \vee \mathsf{Xor}, x, y, x, z)$ is satisfied when x, y and z have been instantiated to, for instance $1, 0, 0$ respectively.

Let Γ_1, Γ_2 be sets of relations and define the disjunction of two sets of relations $\Gamma_1 {}^{\times}\!\!\vee \Gamma_2$ as follows:

$$\Gamma_1 {}^{\times}\!\!\vee \Gamma_2 = \Gamma_1 \cup \Gamma_2 \cup \{R_1 \vee R_2 | R_1 \in \Gamma_1, R_2 \in \Gamma_2\}$$

The disjunction of two sets of relations $\Gamma_1 {}^{\times}\!\!\vee \Gamma_2$, first introduced by Cohen *et al.* [CJJK01], is the set of disjunctions of each pair of relations in Γ_1, Γ_2 plus the sets Γ_1, Γ_2. It is natural to include Γ_1 and Γ_2 since one wants to have the choice of using the disjunction or not. The fact that if $R_1 \vee R_2$ is included in a set of relations, then both R_1 and R_2 are in the set is a property which we refer to as the ${}^{\times}\!\!\vee$-closure property. In the sequel, we will tacitly assume that all sets of relations have this property.

We shall frequently be concerned with constraints that are specified by disjunctions of an arbitrary number of relations. Thus, we make the following definition: for any set of relations, Δ, define $\Delta^* = \bigcup_{i=0}^{\infty} \Delta^i$ where $\Delta^0 = \{\bot\}$ and $\Delta^{i+1} = \Delta^i {}^{\times}\!\!\vee \Delta$.

The previously defined concepts of problem instances, models and so on can obviously be extended to disjunctions in a natural way. It is also worth noticing that the problem $\textsc{Psat}_p$ is still in NP if we allow the use of disjunctions.

We say that a set of relations Γ is *maximal tractable* for $\textsc{Psat}_p$ iff $\textsc{Psat}_p(\Gamma)$ is tractable and for every set $X \supset \Gamma$ of relations from $\mathcal{PA}^*$ it holds that $\textsc{Psat}_p(X)$ is not tractable. Given a complete classification of $\textsc{Psat}_p$ with disjunctions, we can easily find a complete classification of the point algebra over p.

Theorem 4 Let M be the set of maximal tractable sets of relations for $\textsc{Psat}_p(\mathcal{PA}^*)$. Then, the set $M' = \{X \cap \mathcal{PA} \mid X \in M\}$ contains every maximal tractable set of $\textsc{Psat}_p(\mathcal{PA})$.

Proof: Let $X' \subseteq \mathcal{PA}$ be a maximal tractable set. There exists an $X \in M$ such that $X' \subseteq X$ since M contains the maximal tractable sets of $\mathcal{PA}^*$ (and $\mathcal{PA} \subseteq \mathcal{PA}^*$). Hence, $X' \subseteq X \cap \mathcal{PA}$. It cannot be the case that $X' \subset X \cap \mathcal{PA}$ since $X \cap \mathcal{PA}$ is tractable and X' is a maximal tractable set. Consequently, $X' = X \cap \mathcal{PA}$ and $X' \in M'$. □

It follows immediately that the maximal tractable sets of $\textsc{Psat}_p(\mathcal{PA})$ are the maximal elements of the partial order $(M', \subseteq)$.

Next, we introduce a construction which simplifies maximality proofs by allowing us to only consider a small number of disjunctions. Let Γ and Δ be arbitrary sets of relations such that $\Delta \subseteq \Gamma$.

Definition 5 Let $\mathcal{T} = \Gamma^{\times}_{\vee}\Delta^*$. We define $\overline{\mathcal{T}}$ as

$$\Big[(\mathcal{PA} - \Gamma) \cup (\Gamma - \Delta)^{\times}_{\vee}(\Gamma - \Delta)\Big] - (\Gamma - \Delta)$$

Lemma 6 If $\mathcal{T} = \Gamma^{\times}_{\vee}\Delta^*$ and $\mathcal{T}' \not\subseteq \mathcal{T}$, then there exists $C \in \overline{\mathcal{T}}$ such that $C \in \mathcal{T}'$.

Proof: Arbitrarily choose a $C \in \mathcal{T}'$ such that $C \notin \mathcal{T}$ and choose $C_1, \cdots, C_n \in \mathcal{PA}$ such that $C_1 \vee \cdots \vee C_n = C$. Assume first that there exists some i such that $C_i \notin \Gamma$. The definition of $\overline{\mathcal{T}}$ implies that $C_i \in \overline{\mathcal{T}}$ and the $^{\times}_{\vee}$-closure property implies that $C_i \in \mathcal{T}'$.

Assume instead that $C_1, \cdots, C_n \in \Gamma$. Since $\Gamma \subseteq \mathcal{T}$ and $C \notin \mathcal{T}$ we know that $n > 1$. If all or all but one $C_i \in \Delta$, then we know that $C \in \mathcal{T}$. Hence, there exists at least two i, j such that $C_i, C_j \notin \Delta$. Now, $C_i \vee C_j \in \mathcal{T}'$ by the $^{\times}_{\vee}$-closure property and the definition of $\overline{\mathcal{T}}$ gives that $C_i \vee C_j \in \overline{\mathcal{T}}$. □

For the tractability results we will use the concept of *k-independence* [CJJK01]. This property is defined as follows

Definition 7 For any sets of relations Γ and Δ, define $\text{PSAT}_{p,\Delta\leq k}(\Gamma \cup \Delta)$ to be the subproblem of $\text{PSAT}_p(\Gamma \cup \Delta)$ consisting of all instances containing at most k constraints over the relations in Δ. We say that Δ is *k-independent* of Γ if the following condition holds: any set of constraints C in $\text{PSAT}_p(\Gamma \cup \Delta)$ has a solution provided every subset of C belonging to $\text{PSAT}_{p,\Delta\leq k}(\Gamma \cup \Delta)$ has a solution.

We will refer to 1-independence as simply independence. The following result by Cohen *et al.* [CJJK01] demonstrates the usefulness of the k-independence property:

Theorem 8 Let Γ and Δ be sets of relations such that $\text{PSAT}_p(\Gamma \cup \Delta)$ is tractable. If Δ is 1-independent of Γ, then $\text{PSAT}_p(\Gamma^{\diamondsuit}\Delta^*)$ is tractable. If Δ is 2-independent of $\emptyset$, then $\text{PSAT}_p(\Delta^{\diamondsuit}\Delta)$ is tractable.

Example 2.2 We see that $\{=\}$ is not 1-independent of $\{=, \neq\}$ since the constraints $X = \{x = y, y = z, x \neq z\}$ are not satisfiable although the constraints $X' = \{x = y, x \neq z\}$ and $X'' = \{y = z, x \neq z\}$ are both satisfiable. Note that $\{=\}$ is 1-independent of $\{=\}$ since all problem instances containing only equality constraints are satisfiable.

We continue by proving that the 1-independence property is preserved under the closure operator.

Theorem 9 Let Δ be a set of relations 1-independent of Γ. Then, $C(\Delta)$ is 1-independent of $C(\Gamma)$.

Proof: We begin by showing that Δ is 1-independent of $C(\Gamma)$. By inspecting Renz and Nebel's [RN99] proof of Theorem 2, it follows that any instance of $\text{PSAT}_p(C(\Gamma))$ can be transformed to an equivalent instance of $\text{PSAT}_p(\Gamma)$. In fact, each constraint xRy where $R \in C(\Gamma) - \Gamma$ is merely replaced by a fixed instance of $\text{PSAT}_p(\Gamma)$ over the variables x, y and (possibly) a number of auxiliary variables. It follows that Δ is 1-independent of $C(\Gamma)$ since Δ is 1-independent of Γ.

Now, let Π be an arbitrary instance of $\text{PSAT}_p(C(\Gamma) \cup C(\Delta))$ and let S be the subinstance of Π only containing relations from $C(\Gamma) - C(\Delta)$. Assume that every subset $S \cup \{x_1\delta_1 y_1\}, \ldots, S \cup \{x_n\delta_n y_n\}$ of Π where $\delta_i \in C(\Delta)$ is satisfiable. Each such set of constraints can be rewritten in the equivalent form $S \cup \Pi_i$ where Π_i is a CSP instance over Δ. Obviously, each set $S \cup \Pi_i$

is satisfiable as well as all the relaxed sets $S \cup \{\pi_{i,1}\}, \ldots, S \cup \{\pi_{i,m}\}$, where $\{\pi_{i,1}, \ldots, \pi_{i,m}\}$ are the constraints in Π_i. Since Δ is 1-independent of $C(\Gamma)$ and Π_i only contains relations from Δ, we know that $S \cup H$ where $H = \{\pi_{i,j} | i \leq n, j \leq m\}$ is satisfiable. Since $S \cup H$ is equivalent to the instance Π, we have shown that Π is satisfiable and, hence, that $C(\Delta)$ is 1-independent of $C(\Gamma)$. □

For the total classifications in Section 3–6 we need to prove that the satisfiability problem $\text{PSAT}_p(\{R_1, R_2, R_3 \vee R_3\})$ is NP-hard for different instantiations of p, R_1, R_2 and R_3. For this purpose, we provide a result inspired by Broxvall *et al.* [BJR02, Paper II]. Note that $\text{PSAT}_p(\{R_1, R_2\} \cup \{R_3\}^2)$ can be reduced to $\text{PSAT}_p(\{R_1, R_2, R_3 \vee R_3\})$ in polynomial time.

Theorem 10 $\text{PSAT}_p(\Gamma \cup \Delta^2)$ is tractable only if Δ is 2-independent of Γ. Otherwise, $\text{PSAT}_p(\Gamma \cup \Delta^2)$ is NP-complete.

Proof: Assume to the contrary that $\text{PSAT}_p(\Gamma \cup \Delta^2)$ is tractable but Δ is not 2-independent of Γ. This implies that there exists a set of constraints X over Γ and a set $H = \{h_1, \ldots, h_n\}$ of constraints over Δ such that $X \cup \{h_i, h_j\}$ is satisfiable for every $1 \leq i, j \leq n$ but $X \cup H$ is not satisfiable. Choose X and H such that $|H|$ is as small as possible and note that $|H| \geq 3$. The existence of a set $H' \subset H$ such that $X \cup H'$ is not satisfiable contradicts the minimality of H so $X \cup H'$ is satisfiable for all $H' \subset H$. Thus, we can define the satisfiable set $\mathcal{X} = X \cup \{h_1 \vee h_2, h_1 \vee h_3, h_2 \vee h_3, h_4, \ldots, h_n\}$ which has the following property: In every model of $\mathcal{X}$, exactly one of h_1, h_2, h_3 is not satisfied.

To prove the result, we show that 3-Colourability can be transformed to $\text{PSAT}_p(\Gamma \cup \{R \vee R\})$ in polynomial time. Arbitrarily choose an undirected graph $G = (V, E)$ such that $V = \{v_1, \ldots, v_k\}$. We will construct an instance of $\text{PSAT}_p(\Gamma \cup \{R \vee R\})$ that is satisfiable iff G is colourable with three colours.

For each vertex v_i, introduce a fresh copy of the set $\mathcal{X}$ where we denote the constraints h_1, h_2, h_3 as h_1^i, h_2^i, h_3^i, respectively. As we have already noted, this will force exactly one of h_1^i, h_2^i, h_3^i not to hold in every model. We interpret this to mean that h_j^i does not hold as 'vertex v_i has colour j'.

For each edge $(v_i, v_j) \in E$, we add the disjunctions $h_1^i \vee h_1^j$, $h_2^i \vee h_2^j$ and $h_3^i \vee h_3^j$ which ensures that v_i and v_j are not assigned the same colour. The resulting set of constraints can be computed in polynomial time, it is an

instance of $\textsc{Psat}_p(\Gamma \cup \Delta^2)$ and it is satisfiable iff G is 3-colourable, which concludes the proof. □

To simplify the presentation of some tractable sets of relations, we recall the definition of the *guaranteed satisfaction property* [CJJK01].

Definition 11 Let Δ be a set of relations. If every instance of $\textsc{Psat}_p(\Delta)$ which does not contain a constraint of the form $x \perp y$ is satisfiable, then we say that Δ has the *guaranteed satisfaction* (GS) property.

Clearly, $\textsc{Psat}_p(\Delta)$ and $\textsc{Psat}_p(\Delta^*)$ are tractable if Δ has the GS property. Define the sets $\Delta_=$ and $\Delta_\parallel$ such that $\Delta_= = \{\perp\} \cup \{R \in \mathcal{PA} \mid (=) \subseteq R\}$ and $\Delta_\parallel = \{\perp\} \cup \{R \in \mathcal{PA} \mid (\parallel) \subseteq R\}$ Since we encounter these sets of relations frequently, we note once and for all that $\Delta_=$ and $\Delta_\parallel$ have the GS property. We also note that whenever an instance of $\textsc{Psat}_p(\Delta_=)$ has a model, it has a one-point model. Similarly, a satisfiable instance of $\textsc{Psat}_p(\Delta_\parallel)$ always has a model where the points are unrelated. Also note that we can always extend a tractable set of relations with the equality relation, i.e. $\textsc{Psat}_p(\Delta \cup \{=\})$ is tractable whenever $\textsc{Psat}_p(\Delta)$ is tractable.

3 Totally Ordered Time

We will now identify the maximal tractable subclasses of the totally ordered point algebra. The domain we are considering is the set of total orders, *to*, and the corresponding decidability problem is denoted $\textsc{Psat}_{to}$. The methods and the results presented in this section will be used many times in later sections when we classify point algebras over other types of partial orders.

Note that the basic relation $||$ is unnecessary when dealing with total orders so we only have three basic relations ($<$, $=$ and $>$) and eight possible disjunctions of these relations. Let $\mathcal{PA}_{to}$ denote the set of these eight relations and define $\mathcal{X}_1 = \mathcal{PA}_{to}{}^{\times}_{\diamond}\{\neq\}^*$ and $\mathcal{X}_2 = \Delta_=^*$. As we will see later on, $\mathcal{X}_1$ and $\mathcal{X}_2$ are the only maximal tractable subclasses of $\textsc{Psat}_{to}$.

Lemma 12 $\textsc{Psat}_{to}(\mathcal{X}_i)$, $1 \leq i \leq 2$, are tractable problems.

Proof: Tractability of $\mathcal{X}_1$ has been proved by Jonsson and Bäckström [JB98] and Koubarakis [Kou96] while the tractability of $\mathcal{X}_2$ follows trivially since $\Delta_=$ has the GS property. □

Lemma 13 $\textsc{Psat}_{to}(\mathcal{N}_i)$, $1 \leq i \leq 5$, is NP-complete where

$$\begin{array}{ll} \mathcal{N}_1 = \{< \vee <\} & \mathcal{N}_2 = \{\neq, \leq \vee \leq\} \\ \mathcal{N}_3 = \{<, = \vee =\} & \mathcal{N}_4 = \{\neq, = \vee =\} \\ \mathcal{N}_5 = \{<, \leq \vee \leq\}. & \end{array}$$

Proof: We begin by proving that Δ_i is not 2-independent of Γ_i for Δ_i, Γ_i as defined below:

$$\Delta_1 = \{<\}, \Delta_2 = \{\leq\}, \Delta_3 = \{=\}, \Delta_4 = \{=\}, \Delta_5 = \{\leq\}$$

$$\Gamma_1 = \emptyset, \Gamma_2 = \{\neq\}, \Gamma_3 = \{<\}, \Gamma_4 = \{\neq\}, \Gamma_5 = \{<\}$$

Together with Theorem 10, this proves that the correspodning satisfiability problems are NP-complete. The proof is straightforward: the following sets of constraints are not satisfiable although every subinstance of them containing at most two relations from Δ_i are satisfiable.

1. $x_1 < x_2 < x_3 < x_1$
2. $x_1 \leq x_2 \leq x_3 \leq x_1, x_1 \neq x_2$
3. $x_1 = x_2 = x_3 = x_4, x_1 < x_4$
4. $x_1 = x_2 = x_3 = x_4, x_1 \neq x_4$
5. $x_1 \leq x_2 \leq x_3 \leq x_1, x_1 < x_2$ □

We are now ready to prove the main theorem.

Theorem 14 $\mathcal{X}_1$ and $\mathcal{X}_2$ are the only maximal tractable disjunctive subclasses of $\textsc{Psat}_{to}$.

Proof: Assume that there exists a maximal tractable algebra $\mathcal{X}$ such that $\mathcal{X} \not\subseteq \mathcal{X}_1$ and $\mathcal{X} \not\subseteq \mathcal{X}_2$. By Lemma 6, there exists γ_1, γ_2 in $\mathcal{T}$ such that $\gamma_1 \in \overline{\mathcal{X}_1}$ and $\gamma_2 \in \overline{\mathcal{X}_2}$. It is easy to see that $\overline{\mathcal{X}_1} \subseteq \{r_1, \ldots, r_6\}$ where

$$\begin{array}{llll} r_1 & = & (< \vee <) & r_2 = (\leq \vee \leq) \\ r_3 & = & (= \vee =) & r_4 = (\leq \vee =) \\ r_5 & = & (< \vee =) & r_6 = (< \vee \leq) \end{array}$$

and $\overline{\mathcal{X}_2} \subseteq \{<, \neq, < \vee <, < \vee \neq, \neq \vee \neq\}$. By Lemma 22, we know that $\textsc{Psat}_{to}\{(< \vee <)\}$ is NP-complete so the disjunction $r_1 = (< \vee <)$ can be excluded immediately. We will show that if γ_1 equals one of $r_2, \ldots, r_6$ and γ_2 equals $<$ or $\neq$, then $\textsc{Psat}_{to}(\{\gamma_1, \gamma_2\})$ is NP-complete.

1. $\gamma_1 \in \{r_2, r_3\}$. Then, the NP-completeness of $\textsc{Psat}_{to}(\{\gamma_1, \gamma_2\})$ is an immediate consequence of Lemma 13.

2. $\gamma_1 = r_4$. Note that r_3 can be obtained from r_4 so $\textsc{Psat}_{to}(\{\gamma_1, \gamma_2\})$ is NP-complete by case 1.

3. $\gamma_1 = r_5$ and $\gamma_2 = <$. We show how to obtain the disjunctive relation $a < b \vee c < d$ with γ_1 and γ_2: introduce a fresh variable t and the following two relations: $a < t$ and $t = b \vee c < d$. NP-completeness follows from case 1.

4. $\gamma_1 = r_5$ and $\gamma_2 = \neq$. The relation $a < b$ can easily be obtained by the relations $t_1 \neq t_2$ and $t_1 = t_2 \vee a < b$ where t_1 and t_2 are fresh variables. NP-completeness follows from the previous case.

5. $\gamma_1 = r_6$. The NP-completeness of $\textsc{Psat}_{to}(\{\gamma_1, \gamma_2\})$ is a consequence of cases 3 and 4 and the observation that r_5 can be obtained from r_6.

Now, $<$ can be obtained from $< \vee \neq$ since $x < y \vee z \neq z$ (where z is a fresh variable) and $x < y$ are equivalent. Similarly, $\neq$ can be obtained from $\neq \vee \neq$. NP-hardness follows from the cases above so $\textsc{Psat}_{to}(\{\gamma_1, \gamma_2\})$ is NP-complete for all $\gamma_1 \in \overline{\mathcal{X}_1}$ and $\gamma_2 \in \overline{\mathcal{X}_2}$. This contradicts our initial assumptions and proves the theorem. □

4 Partially Ordered Time

We will now consider the class *po* of all partial orders. The results can be summarized as follows: the point algebra extended with disjunctions contains the following four maximal tractable subclasses:

$$\mathcal{S}_A = \Gamma_A{}^{\times}_{\vee}\Delta^*_A,\ \mathcal{S}_B = \Gamma_B{}^{\times}_{\vee}\Delta^*_B,\ \mathcal{S}_C = \Gamma_C{}^{\times}_{\vee}\Delta^*_C,\ \mathcal{S}_D = \Delta^*_D.$$

The different sets of relations are defined in Table 2 and a composition table can be found in Table 1. An immediate consequence of Theorem 4 is that

	$<$	$>$	$=$	$\parallel$
$<$	$<$	$\mathcal{B}$	$<$	$< \parallel$
$>$	$\mathcal{B}$	$>$	$>$	$> \parallel$
$=$	$<$	$>$	$=$	$\parallel$
$\parallel$	$< \parallel$	$> \parallel$	$\parallel$	$\mathcal{B}$

Table 1: Composition of basic relations in partially-ordered time.

	Γ'_A	Δ'_A	Γ_A	Δ_A	Γ'_B	Δ'_B	Γ_B	Δ_B	Γ'_C	Δ'_C	Γ_C	Δ_C	Δ_D
$<$			•				•		•		•		
$\leq$	•		•		•		•						•
$<>$							•	•					
$<=>$					•	•	•	•					•
$\parallel$	•	•	•	•							•	•	
$\underline{\parallel}$	•		•								•	•	•
$=$			•				•		•		•		•
$\neq$	•	•	•	•	•	•	•	•	•	•	•	•	
$< \parallel$			•	•							•	•	
$\leq \parallel$			•						•	•	•	•	•

Table 2: Tractable classes in partially-ordered time.

the point algebra (without disjunctions) contains three maximal tractable subclasses Γ_A, Γ_B and Δ_D.

We begin by making a number of observations. For the non-disjunctive case, Γ_A is the only maximal tractable set containing all basic relations. For the point algebra extended with disjunctions, $\mathcal{S}_A$ and $\mathcal{S}_C$ are the only maximal tractable sets containing all the basic relations. Furthermore, a satisfiable problem instance over Γ_B or $\mathcal{S}_B$ always has a model that is a total order; similarly, a satisfiable problem instance over Δ_D or $\mathcal{S}_D$ has a one-point model since $\mathcal{S}_D = \Delta_=$.

4.1 Tractability Results

We will now show that the four classes of disjunctive relations $\mathcal{S}_A, \ldots, \mathcal{S}_D$ (as defined in the beginning of Section 4) are tractable. We begin by showing that

Γ_A, Γ_B and Δ_D are tractable. The tractability result follows from combining these results with a series of independence results and applying Theorem 8.

To prove the tractability of Γ_A and Γ_B, we present algorithm A (Figure 1) and claim that it correctly solves the problems $\textsc{Psat}_{po}(\Gamma'_A)$ (where $\Gamma'_A = \{\leq, \|, \neq\}$) and $\textsc{Psat}_{po}(\Gamma'_B)$ (where $\Gamma'_B = \{\leq, <=>, \neq\}$). By proving that $C(\Gamma'_A) = \Gamma_A$ and $C(\Gamma'_B) = \Gamma_B$ (and using Theorem 2), we are done.

We need some definitions before we present the algorithm. Given an instance Π of $\textsc{Psat}_p(X)$ and two variables x, y we write $x \leq^+ y$ to say that there exists zero or more variables $z_1, \ldots, z_n$ such that

$$x \leq z_1 \wedge z_1 \leq z_2 \wedge \cdots \wedge z_{n-1} \leq z_n \wedge z_n \leq y$$

and we write $x \leq^* y$ to say $x \leq^+ y$ or $x = y$.

The *components* of a graph $G = (V, E)$ are the equivalence classes of vertices that are mutually reachable in the symmetric closure of G, i.e. the graph $G = (V, \{(v, w), (w, v) \mid (v, w) \in E\})$.

An R-subinstance Π' of Π is defined as $(V, C \cap (V \times \{R\} \times V))$. A variable v is *R-minimal* in the instance Π iff v is minimal in the R-subinstance of Π. The *R-components* of an instance Π are the components of the R-subinstance of Π. An instance Π is $(\leq)$-acyclic iff the $(\leq)$-subinstance of Π is acyclic.

If f is a function which is undefined for n, then $(f|n \longmapsto c)$ is defined as the function $f \cup \{(n, c)\}$. Assume n_1, n_2 are variables in an instance $\Pi = (V, C)$ and let $\Pi' = (V - \{n_2\}, C')$ where

$$C' = C \cup \{n_1 r x \mid n_2 r x \in C\} \cup \{x r n_1 \mid x r n_2 \in C\} - \{x r n_2, n_2 r x \mid x \in V\}.$$

We say that Π' is obtained by *contracting* n_1 and n_2. That is, we identify the variables n_1, n_2 by n_1. Note that there may appear constraints of the form xrx after a contraction. We will use a function of the form contract(Π, n_1, n_2) for computing the instance resulting from contracting the variables n_1, n_2 in Π.

The following simple result is needed several times for proving algorithm correctness.

Lemma 15 Let Π be an instance of $\textsc{Psat}_p(X)$ such that for every model M of Π it holds that $M(n_1) = M(n_2)$, and let Π' be the instance resulting from contracting the variables n_1, n_2 in Π. Then, Π is satisfiable iff Π' is satisfiable.

```
1  algorithm A(Π)
2  Input: An instance Π of PSAT_po(Γ_A) or PSAT_po(Γ_B)
3   repeat
4     Π' ← Π
5     for each pair of variables n_1, n_2 ∈ Var(Π)  do
6        if n_1 ‖ n_2 and n_1 ≤* n_2 in Π  then  return false
7        if n_1 ≤* n_2, n_2 r n_1 ∈ Π and r ∈ {≤*, ‖}  then
8           if n_1 ≠ n_2 in Π  then  return false
9           else  Π' ← contract(Π, n_1, n_2)
10       end if
11    end for
12 until Π' = Π
13 return true
```

Figure 1: The algorithm for solving $\text{PSAT}_{po}(\Gamma'_A)$ and $\text{PSAT}_{po}(\Gamma'_B)$

Proof: Assume that Π has a model M. Clearly M is also a model of Π'. Assume that Π' has a model $M' = (f, (T, \leq))$, the following is a model of Π:

$$M = ((f|n_2 \longmapsto f(n_1)), (T, \leq))$$

□

Theorem 16 $\text{PSAT}_{po}(\Gamma_A)$ is tractable.

Proof: Algorithm A obviously runs in polynomial time since every loop removes at least one element from the given instance Π, so at most $|\Pi|$ loops can be made. We continue by showing that algorithm A solves $\text{PSAT}_{po}(\Gamma'_A)$ correctly.

If the algorithm contracts two variables n_1, n_2 in an instance Π, then Π is satisfiable iff the contracted instance is satisfiable by Lemma 15 since either $n_1 \leq^* n_2, n_2 \leq^* n_1$ or $n_1 \| n_2, n_2 \leq^* n_1$ which both lead to $M(n_1) = M(n_2)$ in every model M of Π. Thus, if the algorithm rejects Π, then the original instance cannot have a model.

Assume to the contrary that the algorithm accepts an instance. Consider the instance Π after the last iteration and recall that the original instance is satisfiable iff Π is satisfiable. The $(\leq)$-subinstance of Π is acyclic and there exist no n_1, n_2 such that $n_1 \| n_2$ and $n_1 \leq^* n_2$. Let,

$$M = (f, (T, \leq)) = (\{(n_i, p_i) | n_i \in \Pi\}, (\{p_i | n_i \in \Pi\}, \{(p_i, p_j) | n_i \leq^* n_j\}))$$

and note that $(T, \leq)$ is a partial order. Arbitrarily choose xRy in Π and assume without loss of generality that x, y are distinct variables. If R is $\neq$ the relation holds since every variable n_i is mapped to a unique point p_i, if R is $\underline{\|}$ the relation holds since neither $x \leq^* y$ nor $y \leq^* x$ in Π. The last case when R equals $\leq$ trivially holds. Consequently M is a model of Π and the original problem instance is satisfiable.

Finally, we note that $C(\Gamma'_A) = \Gamma_A$ since $(<) = (\leq \cap \neq)$, $(\leq \|) = (\leq \circ \underline{\|})$, $(< \|) = (\leq \| \cap \neq)$ and $(=) = (\leq \cap \geq)$. □

Many different algorithms exist for solving $\textsc{Psat}_{po}(\Gamma_A)$. Anger *et al.* [AMR98] have presented an alternative algorithm and Broxvall *et al.* [BJR02, Paper II] have shown that path consistency decides this problem. We have chosen to present algorithm A for mainly two reasons; (1) it shows how the structure of the partially-ordered time model can be exploited in an algorithm; and (2) the algorithm can be used for showing the independence results needed in the next section (where we study disjunctive constraints). The independence results can also be shown by an automatic method [BJR02]; however, this method gives no clue whatsoever why the independence property holds. When we turn our attention to branching time in Section 6, the reader will benefit from having seen how an algorithm can be used for proving independence results—in that case, the automatic method does not work.

Theorem 17 $\textsc{Psat}_{po}(\Gamma_B)$ is tractable.

Proof: We show that algorithm A correctly solves the $\textsc{Psat}_{po}(\Gamma'_B)$ problem. If the algorithm contracts two variables n_1, n_2 in an instance Π, then Π is satisfiable iff the contracted instance is satisfiable by Lemma 15 since there exists a cycle $n_1 \leq^* n_2 \leq^* n_1$ which leads to $M(n_1) = M(n_2)$ in every model M of Π. After having made this observation, it is trivial to show that if the algorithm rejects an instance Π, then it does not have a model.

Assume to the contrary that the algorithm accepts the instance Π and let Π' denote the instance after the last iteration of the algorithm. We can construct a model for Π' by induction over the number of variables in Π'. If Π' is empty then $M = (\emptyset, (\emptyset, \emptyset))$ is a model of Π'. Assume we can construct a model for every instance with k or fewer variables. Let Π' be an arbitrary instance containing $k + 1$ variables. Note that the $(\leq)$-subinstance of Π' is acyclic and thus contains at least one minimal variable x. By the induction hypothesis there exists a model $M' = (f, (T, \leq))$ of $\Pi' - \{x\}$. Let,

$$M = ((f|n \longmapsto c), (T \cup \{c\}, \leq \cup \{(c, x) | x \in T\}))$$

```
1  algorithm B(Π)
2  Input: An instance Π of PSAT_po(Γ'_C)
3   repeat
4     Π' ← Π.
5     if ∃n1, n2 ∈ Var(Π) s.t. n1 = n2 ∈ Π  then Π ← contract(Π, n1, n2)
6     end if
7  until Π' = Π
8  if ∃n ∈ Var(Π) such that n ≠ n or n < n in Π then  return false
9  if ∃n1, n2 ∈ Var(Π) s.t. n1 <* n2 ∈ Π ∧ n1 R n2 ∈ Π where <∉ R then
   return false
10 return true
```

Figure 2: The algorithm for solving $\text{PSAT}_{po}(\Gamma'_C)$

where c is a fresh point. Clearly M is a model of Π' and thus Π is satisfiable.

To conclude the theorem, $C(\Gamma'_B) = \Gamma_B$ since $(<) = (\leq \cap \neq)$, $(=) = (\leq \cap \geq)$ and $(<>) = (<=> \cap \neq)$. □

Observe that the model constructed in the previous proof is a total-order and whenever the algorithm rejects an instance, there is no partial order that can serve as a model for the instance. This proves the following corollary.

Corollary 18 $\text{PSAT}_p(\Gamma_B)$ is tractable whenever $to \subseteq p$.

Finally, $\text{PSAT}_{po}(\Delta_D)$ is a tractable problem since $\Delta_D = \Delta_=$. We can now continue with disjunctive constraints.

Lemma 19 $\Delta'_A, \Delta'_B, \Delta'_C$ are independent of Γ'_A, Γ'_B and Γ'_C, respectively.

Proof: We begin by showing that Δ'_A is independent of Γ'_A; the proof for Γ'_B is similar.

Let Π be an instance of $\text{PSAT}_{po}(\Gamma'_A \cup \Delta'_A)$ and assume that every instance Π' such that Π' only contains constraints present in Π and at most one constraint from Δ'_A is satisfiable. We prove Π to be satisfiable.

Assume to the contrary that Π is not satisfiable. Then, there exists at least one constraint $x\ R\ y \in \Pi$ where R is $\neq$ or $\|$ which causes algorithm A to reject. Let Π' denote the problem instance containing all constraints from Π of the form $z\ R'\ w$ where R' is $\leq$ or $\underline{\|}$ plus the constraint $x\ R\ y$. Note

that Π' causes the algorithm to reject. This is a contradiction, since Π' is a problem instance previously assumed to be satisfiable and the algorithm correctly solves $\text{PSAT}_{po}(\Gamma'_A \cup \Delta'_A)$. Hence, Δ'_A is independent of Γ'_A.

To prove that Δ_C is independent of Γ'_C, we present an alternative tractable algorithm for Γ'_C. Obviously. $\Gamma_C \subseteq \Gamma_A$ so we could in principle use algorithm A—it is, however, easier to prove the result this way. A polynomial-time algorithm for $\text{PSAT}_{po}(\Gamma'_C)$ is presented in Figure 2. Obviously algorithm B rejects only unsatisfiable instances of $\text{PSAT}_{po}(\Gamma'_C)$.

Assume algorithm B accepts a certain instance Π and let Π' be the instance resulting from the contractions made by the algorithm in lines 3-6. Observe that Π' does not contain the relation $(=)$ and if Π' is satisfiable, then Π is satisfiable. Let I be the interpretation given by the function $f(x) = x$ and the partial order $(V, \{(x, y) \mid x <^* y \text{ in } \Pi'\})$ where V is the set of variables appearing in Π'.

Each $\neq$- and $<$-constraint is satisfied by I and if there exists a constraint $x \leq \|y$ in Π' then either $x < y$ or $x \| y$ under I which guarantees that each such constraint is satisfied so I is a model and Π is satisfiable. Thus, $\text{PSAT}_{po}(\Gamma'_C)$ is tractable.

Proving that Δ'_C is independent of Γ'_C with the aid of algorithm B is analogous to the previous independence proofs. □

We can now show that the four classes are tractable.

Theorem 20 $\text{PSAT}_{po}(\mathcal{S}_A), \ldots, \text{PSAT}_{po}(\mathcal{S}_D)$ are tractable problems.

Proof: That $\text{PSAT}_{po}(\Gamma'_A{}^{\times}_{\vee}\Delta'^{*}_A)$ is tractable follows from Lemmata 16, 19 and Theorem 8. That $\Gamma_A = C(\Gamma'_A)$ was shown in the proof of Theorem 16. Let Π be an arbitrary instance of $\text{PSAT}_{po}(\mathcal{S}_A)$. We will show how the relation $< \|$ can be replaced by relations in $\Gamma'_A{}^{\times}_{\vee}\Delta'^{*}_A$ which implies the tractability of $\text{PSAT}_{po}(\mathcal{S}_A)$ by Theorem 9.

Choose an arbitrary disjunction $\gamma = x\ R_1\ y \vee \ldots \vee x_n\ R_n\ y_n$ in Π where $R_1 = (< \|)$. Introduce a fresh variable $t_{x,y}$, add the constraint $x \leq t_{x,y}$ and replace γ by the disjunction $\gamma' = t_{x,y} \parallel y \vee \ldots \vee x_n\ R_n\ y_n$. Repeat this transformation until no $(< \|)$ remains—a process which clearly takes polynomial time. Let Π' denote the resulting instance. Since the $(\leq \circ \|) =< \|$, the constraints $x \leq t_{x,y}, t_{x,y} \| y$ implies the relation $x < \| y$. It follows that Π' is satisfiable iff Π is satisfiable and that $\text{PSAT}_{po}(\mathcal{S}_A)$ is tractable.

The tractability of $\text{PSAT}_{po}(\Gamma'_B{}^{\times}_{\vee}\Delta'^{*}_B)$ follows from Theorem 17, Lemma 19 and Theorem 8, and that $C(\Gamma'_B) = \Gamma_B$ was shown in the proof of Theorem 17.

The two constraints $x \neq y$, $x <=> y$ are satisfiable iff $x <> y$ which shows that $C(\Delta'_B) = \Delta_B$. This fact combined with Theorem 9 proves that $\text{PSAT}_{po}(\mathcal{S}_B)$ is tractable.

To show that $\mathcal{S}_C$ is tractable we begin by noting that $C(\Delta'_C) = \Delta_C$ since $(\|) = (\leq \| \ \cap \ (\leq \|)^{-1})$, $(\|) = (\neq \cap (\leq \|) \ \cap \ (\leq \|)^{-1})$ and $< \| = (\neq \cap \leq \|)$. Hence, Δ_C is independent of Γ'_C by Lemma 19 and Theorem 9. That $\text{PSAT}_{po}({\Gamma'_C}^{\times}_{\vee}\Delta^*_C)$ is tractable follows from Theorem 8. The theorem follows since $\Gamma_C = \Gamma'_C \cup \Delta_C$ which implies that $\mathcal{S}_C = {\Gamma'_C}^{\times}_{\vee}\Delta^*_C = {\Gamma_C}^{\times}_{\vee}\Delta^*_C$.

Finally, $\text{PSAT}_{po}(\mathcal{S}_D)$ is tractable since Δ_D has the GS property. □

4.2 Intractability Results

This section contains the NP-completeness results needed for the classification theorem. Our hardness results rely on the fact that $\text{PSAT}_{po}(\mathcal{A}_2)$ is NP-complete where $\mathcal{A}_2 = \{<>, \|\}$. We show hardness for NP by a polynomial-time reduction from the BETWEENNESS problem which is defined as follows:

INSTANCE: A pair (A, T) consisting of a finite set A and a collection T of ordered triples (a, b, c) of distinct elements from A.
QUESTION: Is there a one-to-one mapping $f : A \to \{1, 2, \ldots, |A|\}$ such that for each $(a, b, c) \in T$ we have either $f(a) < f(b) < f(c)$ or $f(c) < f(b) < f(a)$.

The BETWEENNESS problem is known to be NP-complete [GJ79, page 279]. Given an arbitrary instance (A, T) of BETWEENNESS where $A = \{a_1, \ldots, a_k\}$, construct an instance Π of $\text{PSAT}_{po}(\mathcal{A}_2)$ as follows:

1. for each $a_i \in A$, introduce a fresh variable a'_i;

2. for each pair of distinct $a_i, a_j \in A$, add the constraint $a_i <> a_j$;

3. for each triple $t_m = (a_i, a_j, a_k) \in T$, introduce two fresh variables x'_m, y'_m and add the following constraints

$$x'_m \| a'_i, \ x'_m \| a'_j, \ x'_m <> a'_k$$

$$y'_m <> a'_i, \ y'_m \| a'_j, \ y'_m \| a'_k$$

We show that Π is satisfiable iff (A, T) has a solution. For the if-direction, assume that (A, T) is satisfiable and let $\mathcal{P}_{|A|}$ be the partial order $(V, <)$ where $V = \{x_1, \ldots, x_{|A|}, y_1, \ldots, y_{|A|}, z_1, \ldots, z_{|A|}\}$ and:

$$x_i < y_j, y_i < y_{j+1}, y_i < z_j \text{ iff } i \leq j$$
$$x_i \| x_j, x_i < z_j, z_i \| z_j \text{ for all } i, j$$

Furthermore, let f be a mapping from A to $\{1, 2, \ldots, |A|\}$ satisfying each triple in T. We build a mapping f' from the variables in Π as follows:

1. $f'(a'_i) = y_l$ where $f(a_i) = l$
2. For each $t_m = (a_i, a_j, a_k) \in T$ such that $f(a_i) < f(a_j) < f(a_k)$ we let $f'(x'_m) = x_{f(a_k)}, f'(y'_m) = z_{f(a_i)}$
3. For each $t_m = (a_i, a_j, a_k) \in T$ such that $f(a_i) > f(a_j) > f(a_k)$ we let $f'(x'_m) = z_{f(a_k)}, f'(y'_m) = x_{f_(a_i)}$

If we examine all the constraints in Π it is obvious that f' satisfies them all.

To clarify the steps of the model construction above we provide a small example. The Betweenness problem instance

$$(\{a_1, a_2, a_3, a_4\}, \{(a_1, a_4, a_2), (a_2, a_3, a_4)\})$$

gives the point algebra problem instance Π in figure 3 and has a solution $g = \{(a_1, 1), (a_2, 4), (a_3, 3), (a_4, 2)\}$. From the solution of the Betweenness problem we can construct a model M of Π depicted in the same figure. Transitive edges are omitted.

We continue by showing the only-if direction. Assume that Π has a model M. Note that the variables $a'_1, a'_2, \ldots$ in Π corresponding to variables $a_1, a_2, \ldots$ in A are totally ordered, i.e. for arbitrary distinct a'_i, a'_j, either $M(a'_i) < M(a'_j)$ or $M(a'_j) < M(a'_i)$. Choose indices $m_1, \ldots, m_k$ such that $a'_{m_1} < \cdots < a'_{m_k}$. Define a solution $f : A \to \{1, \ldots, k\}$ of (A, T) as follows: $f(a_i) = m_i$. Clearly, this function is a one-to-one mapping from A to $\{1, \ldots, k\}$.

Arbitrarily choose a triple $(a_i, a_j, a_k) \in T$. We show that $f(a_i) < f(a_j) < f(a_k)$ or $f(a_k) < f(a_j) < f(a_i)$. Assume to the contrary that $f(a_j) < f(a_i) < f(a_k)$. Then, the model M implies that $M(a'_j) < M(a'_i) < M(a'_k)$ under the

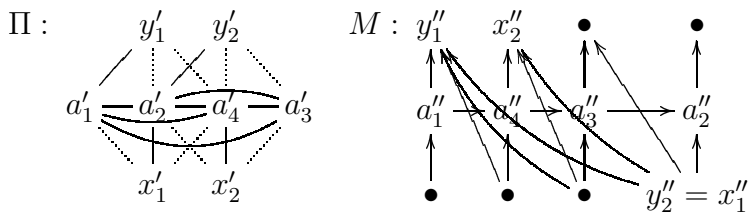


Figure 3: The problem instance Π and its model M. To simplify the picture, dotted arrows have been used to mark the relation $\|$ and filled arrows for the relation $<>$

partial order of M. By the construction of Π we know that there exists a variable y'_m such that $M(y'_m) <> M(a'_i)$, $M(y'_m) \| M(a'_j)$ and $M(y'_m) \| M(a'_k)$. Assume first that $M(y'_m) < M(a'_i)$, then $M(y'_m) < M(a'_k)$ which contradicts the fact that $M(y'_m) \| M(a'_k)$. Similarly, if $M(y'_m) > M(a'_i)$ then $M(a'_j) < M(y'_m)$ and we have a contradiction since $M(y'_m) \| M(a'_j)$. The remaining three cases can be proven analogously.

Thus, we have proved the main part of the following lemma.

Lemma 21 If X is a set of relations containing $<>$ and one of $\|, < \|$, $\underline{\|}$ or $\leq \|$, then $\textsc{Psat}_{po}(X)$ is NP-complete.

Proof: The case when $(\|) \in X$ was proved above. Assume $< \|$ or $\underline{\|}$ is in X. Then, $(\|) = (< \|) \cap (< \|)^{-1}$ or $(\|) = (<> \circ \underline{\|}) \cap \underline{\|}$. If X contains $\leq \|$, then $(\underline{\|}) = (\leq \|) \cap (\leq \|)^{-1}$ and NP-completeness follows from the previous case. □

We also need a number of hardness results for disjunctive relations.

Lemma 22 The following problems are NP-complete:

1. $\textsc{Psat}_{po}(\{< \vee <\})$;
2. $\textsc{Psat}_{po}(\{R_1, \leq \vee \leq\})$ if $R_1 \in \{\neq, <>\}$;
3. $\textsc{Psat}_{po}(\{R_2, = \vee =\})$ if $=$ is not a subset of R_2;
4. $\textsc{Psat}_{po}(\{R_3, R_4, \underline{\|} \vee \underline{\|}\})$ if $R_3 \in \{\leq, <=>\}$ and $R_4 \in \{<, <>, \neq, \| <\}$.

Proof: Define Δ_i, Γ_i as follows:

$$\Delta_1 = \{<\}, \Delta_2 = \{\leq\}, \Delta_3 = \{=\}, \Delta_4 = \{\underline{\|}\}$$

$$\Gamma_1 = \emptyset, \Gamma_2 = \{R_1\}, \Gamma_3 = \{R_2\}, \Gamma_4 = \{R_3, R_4\}$$

We demonstrate that Δ_i is not 2-independent of Γ_i by giving examples that are not satisfiable although all subinstances containing at most two relations from Δ_i are satisfiable.

1. $x_1 < x_2 < x_3 < x_1$
2. $x_1 \leq x_2 \leq x_3 \leq x_1, x_1\ R_1\ x_2$
3. $x_1 = x_2 = x_3 = x_4, x_1\ R_2\ x_4$
4. $x_1 \underline{\|} x_2 \underline{\|} x_3 \underline{\|} x_4, x_1\ R_3\ x_2\ R_3\ x_3\ R_3\ x_4, x_1\ R_4\ x_4$

Since Δ_i is not 2-independent of Γ_i, NP-hardness follows from Theorem 10. □

4.3 Maximality

We are now ready to prove that $\mathcal{S}_A, \ldots, \mathcal{S}_D$ are the only maximal tractable sets of the disjunctive point algebra for partially ordered time. To reduce the number of NP-completeness results needed in the classification, we will employ a modified closure operator $C_{\mathrm{po}}(\cdot)$ which is defined with the aid of the standard operators converse, intersection and composition together with a number of equivalences. These equivalences and their properties are stated in the next lemma. The straightforward proof is only sketched.

Lemma 23 Let $R_i \in \mathcal{PA}$ and $\Gamma_j \subseteq \mathcal{PA}$. The following problems are equivalent up to polynomial-time reductions.

1. $\textsc{Sat}_{po}(\Gamma_1 \cup \{< \vee R_1\})$ and $\textsc{Sat}_{po}(\Gamma_1 \cup \{< \vee R_1,\ R_1 \vee R_1\})$;
2. $\textsc{Sat}_{po}(\Gamma_1 \cup \{= \vee R_1)$ and $\textsc{Sat}_{po}(\Gamma_1 \cup \{= \vee R_1,\ R_1 \vee R_1\})$ when one of $<$, $<>$, $\neq$ or $< \|$ is in Γ_1.
3. $\textsc{Sat}_{po}(\Gamma_1 \cup \{R_1 \vee R_3, R_2 \vee R_3\})$ and $\textsc{Sat}_{po}(\Gamma_1 \cup \{R_1 \vee R_3, R_2 \vee R_3, (R_1 \oplus R_2) \vee R_3\})$ when $\oplus \in \{\cap, \circ\}$;

Proof sketch: The proofs of these equivalences are fairly similar so we only prove the first one. It is sufficient to note that $aR_1b \vee cR_1d$ is equivalent to

$$\{e < f \vee aR_1b,\ f < e \vee cR_1d\}$$

where e, f are fresh variables. □

If Γ is a set of relations, we define $C_{\mathrm{po}}(\Gamma)$ to be the least set X such that

1. $\Gamma \subseteq X$;
2. X is closed under converse, intersection and composition of point relations; and
3. X is closed under the rules in the previous lemma.

By using Lemma 23, the $\overset{\times}{\diamond}$-closure property and the properties of converse, intersection and composition, it is a routine verification to show that if $\mathrm{PSAT}_{po}(\Gamma)$ is NP-complete, then $\mathrm{PSAT}_{po}(C_{\mathrm{po}}(\Gamma))$ is also NP-complete.

We can now prove our classification theorem.

Theorem 24 $\mathcal{S}_A, \mathcal{S}_B, \mathcal{S}_C$ and $\mathcal{S}_D$ are the only maximal tractable subclasses of PSAT_{po}.

Proof: Suppose to the contrary that there exist another maximal tractable algebra $\mathcal{T}$. It follows that there exist $\gamma_A, \cdots, \gamma_D$ in $\mathcal{T}$ such that $\gamma_A \in \overline{\mathcal{S}}_A, \cdots, \gamma_D \in \overline{\mathcal{S}}_D$ by Lemma 6. Note that there exists only a finite number of possible values for $\gamma_A, \ldots, \gamma_D$.

To prove the result, a machine-assisted case analysis of the following form was performed: each admissible choice of $\gamma_A, \ldots, \gamma_D$ was generated and $X = C_{\mathrm{po}}(\{\gamma_{\mathrm{A}}, \ldots, \gamma_{\mathrm{D}}\})$ was computed. Each such set X was examined and it was found that at least one of the NP-complete sets of Lemma 22 was a subset of X. Thus, $\mathrm{PSAT}_{po}(\mathcal{T})$ is NP-complete and the theorem follows. □

4.4 Partial orders and spatial reasoning

This section contains an example where we use the complexity results for point algebras in order to show computational properties of a spatial reasoning problem. The basic idea is to exhibit a connection between partial orders and convex sets in the plane. When this is done, the translation of complexity results is fairly trivial.

Let E^n denote the n-dimensional Euclidean space. Given a set $S \subseteq E^n$, we say that S is *convex* iff for every two points $x, y \in S$ and every $0 \leq \theta \leq 1$, $\theta \cdot x + (1 - \theta) \cdot y \in S$. Define the set Θ such that $x \in \Lambda$ if and only if x satisfies the following properties:

1. x is a convex set in the plane, i.e. a convex subset of E^2; and
2. the area of x is strictly larger than 0.

We require the area of objects to be > 0 in order to rule out degenerate (i.e. lines and points) objects. In the Λ-*problem*, we consider objects taken from Λ related by the following basic relations:

1. $x \subset y$ iff $x \subset y$ (i.e. x is a strict subset of y);
2. $x \supset y$ iff $x \supset y$;
3. $x = y$ iff $x = y$;
4. $x \| y$ iff $x \not\subset y$, $x \not\supset y$ and $x \neq y$.

The possible relations between intervals are the 2^4 possible unions of these basic relations. The satisfiability problem is that of deciding whether there exists elements in Λ that can be assigned to the variables such that all of the relations are satisfied. We will now prove that there is a close connection between the Λ-problem and reasoning about partially-ordered time. More precisely, we prove the following theorem.

Theorem 25 Let Π be an instance of the Λ-problem. Then, Π is satisfiable iff the following $\textsc{Psat}_{po}(\mathcal{PA})$ problem Π' is satisfiable: for each variable x in Π, introduce a variable x and for each constraint xRy in Π, impose the constraint $x \bigcup_{r \in R} \sigma(r) y$ where $\sigma(\subset) = (<)$, $\sigma(\supset) = (>)$, $\sigma(=) = (=)$ and $\sigma(\|) = (\|)$.

This theorem combined with the results in the previous sections tells us, for instance, that the Λ-problem has three maximal tractable subclasses (corresponding to Γ_A, Γ_B and Δ_D) and four maximal disjunctive tractable subclasses (corresponding to $\mathcal{S}_A, \ldots, \mathcal{S}_D$).

We need a couple of definitions and lemmata to prove this theorem. Given two points a, b in the plane, the *(Euclidean) distance* between them is

$$d(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2}.$$

The distance between a and b equals 0 iff $a = b$. Let $C \subseteq E^2$ denote the unit circle, i.e. $\{(x,y) \mid x^2 + y^2 = 1\}$. Given a set $S \subseteq E^n$, $\mathrm{cnv}(S)$ denotes the *convex hull* of S, i.e.

$$\mathrm{cnv}(S) = \{x\theta + y(1-\theta) \mid x, y \in S \text{ and } 0 \leq \theta \leq 1\}.$$

Note that if S is a subset of C and $|S| > 2$, then the area of $\mathrm{cnv}(S)$ is > 0.

Lemma 26 If $x, a, b \in C$ and $x = \theta a + (1-\theta)b$ for some $0 \leq \theta \leq 1$, then $x = a$ or $x = b$.

Proof: If $\theta = 0$, then $x = b$ and if $\theta = 1$, then $x = a$. Assume that $0 < \theta < 1$. If $z = (z_1, z_2) \in C$, then $z_1^2 + z_2^2 = 1$ by the definition of the unit circle. Hence,

$$\begin{array}{ll}
 & x_1^2 + x_2^2 = 1 \\
\Leftrightarrow & (\theta a_1 + (1-\theta)b_1)^2 + (\theta a_2 + (1-\theta)b_2)^2 = 1 \\
\Leftrightarrow & \theta^2 \underbrace{(a_1^2 + a_2^2)}_{1} + (1-\theta)^2 \underbrace{(b_1^2 + b_2^2)}_{1} + 2\theta(1-\theta)(a_1b_1 + a_2b_2) = 1 \\
\Leftrightarrow & \theta^2 + (1-\theta)^2 + 2\theta(1-\theta)(a_1b_1 + a_2b_2) = 1 \\
\Leftrightarrow & 2\theta(\theta - 1) + 2\theta(1-\theta)(a_1b_1 + a_2b_2) = 0 \\
\Leftrightarrow & 2\theta(1-\theta) = 2\theta(1-\theta)(a_1b_1 + a_2b_2) \\
\Leftrightarrow & 1 = a_1b_1 + a_2b_2 \text{ (since } 0 < \theta < 1 \text{ implies } 2\theta(1-\theta) \neq 0).
\end{array}$$

We calculate the Euclidean distance between a and b:

$$\begin{array}{rcl}
d(a,b) & = & \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \\
 & = & \sqrt{a_1^2 + b_1^2 - 2a_1b_1 + a_2^2 + b_2^2 - 2a_2b_2} \\
 & = & \sqrt{\underbrace{a_1^2 + a_2^2}_{1} + \underbrace{b_1^2 + b_2^2}_{1} - \underbrace{2(a_1b_1 + a_2b_2)}_{2}} \\
 & = & 0.
\end{array}$$

Consequently, $a = b$. Furthermore, $x = \theta a + (1-\theta)b$ so $x = a = b$. $\square$

Lemma 27 Let P, Q be arbitrary subsets of C and $r \in \{\subset, \supset, =, \|\}$. If PrQ, then $\text{cnv}(P)\ r\ \text{cnv}(Q)$.

Proof: The lemma holds trivially if $r = (=)$. Assume now that $r = (\subset)$. Arbitrarily choose $x \in \text{cnv}(P)$. Clearly, $x = \theta a + (1-\theta)b$ for some $a, b \in P$ and $0 \leq \theta \leq 1$. It follows immediately that $x \in \text{cnv}(Q)$ since $P \subset Q$ and $a, b \in Q$.

Now, choose $x \in Q - P$. We show that $x \notin \text{cnv}(P)$ and thus prove the lemma. Assume to the contrary that $x = \theta a + (1-\theta)b$ for some $a, b \in P$ and $0 \leq \theta \leq 1$. Since $x, a, b \in C$, we can apply Lemma 26 and conclude that $x = a$ or $x = b$. Consequently, $x \in P$ which leads to a contradiction. The case $r = \supset$ is analogous.

$P \| Q$ implies the existence of x, z such that:

- $x \in P$, $x \notin Q$;
- $y \notin P$, $y \in Q$;

We show that $x \notin \text{cnv}(Q)$ and $y \notin \text{cnv}(P)$ which establishes the result. Assume to the contrary that $x \in \text{cnv}(Q)$. Then, there exist $a, b \in Q$ such that $x = \theta a + (1-\theta)b$ for some $0 \leq \theta \leq 1$. Now, $x, a, b \in C$ and $x = a$ or $x = b$ by Lemma 26. Thus, $x \in Q$, which contradicts our initial assumptions, showing that $y \notin \text{cnv}(P)$ is analogous. □

Proof: (of Theorem 25)

only-if: The structure $(\Lambda, \subseteq)$ is a partial order so Π' is satisfiable if Π is satisfiable.

if: Assume Π' contains the variables $X = \{x_1, \ldots, x_n\}$ and that $f : X \to (T, \leq)$ is a model of Π'. We assume without loss of generality that $T = \{2, \ldots, n+1\}$. Construct the function $f' : X \to 2^T$ as follows:

$$f'(x_i) = \{0, 1, f(x_i)\} \cup \{j \in T \mid j \text{ is below } f(x_i) \text{ in } (T, \leq)\}.$$

It is easy to realize that if $f(x) <_T f(y)$ implies $f'(x) \subset f'(y)$, $f(x) \|_T f(y)$ implies $f'(x) \| f'(y)$ and so forth. Arbitrarily choose an injective function

$g : \{0, \ldots, n+1\} \to C$. Construct an interpretation I of Π as follows: for $1 \leq i \leq k$,

$$I(x_i) = \mathrm{cnv}(\{g(t) \mid t \in f'(x_i)\}).$$

Since $I(x_i)$ is the convex hull of three or more points of the unit circle, it is a convex set in the plane with area > 0. It remains to show that if $x_i r x_j \in \Pi$, then x_i and x_j are related by r under the interpretation I. However, this follows immediately from the definition of function σ and Lemma 27. □

5 Partial Orders with Bounded Dimension

We begin this section by considering a simple scenario: a number of sensors $S_1, \ldots, S_k$ are observing a system and they transmit time-stamped data to a central computer. The system is highly asynchronous so each sensor runs a different clock and these clocks are not synchronized. Assume a certain event A occurs within the system. Each sensor S_i observes A and gives it a time-stamp t_i. Since there is no global notion of time, the best we can do is to assign A the time-stamp $(t_1, \ldots, t_k)$, i.e. a vector containing the time-stamps of the different sensors.

Now, assume we have two events A, A' with time-stamps $(t_1, \ldots, t_k)$ and $(t'_1, \ldots, t'_k)$, respectively. If $t_i = t'_i$ for $1 \leq i \leq k$, we know that A and A' occurred simultaneously; if $t_i \leq t'_i$, $1 \leq i \leq k$, and there exists a j such that $t_j < t'_j$, we know that A occurred strictly before A' and so on.

To reason about such systems, we can construct time point algebras where the possible time points are real vectors containing k elements (i.e. the set of time points equals the k-dimensional Euclidean space) and the basic relations are defined as follows: for time points $x = (x_1, \ldots, x_k), y = (y_1, \ldots, y_k)$:

1. $x < y$ iff $\forall i : x_i \leq y_i$ and $\exists j : x_j < y_j$
2. $x > y$ iff $\forall i : x_i \geq y_i$ and $\exists j : x_j > y_j$
3. $x = y$ iff $\forall i : x_i = y_i$
4. $x \| y$ otherwise

This time model is closely related to certain restricted partial orders and this connection gives us the opportunity to study the complexity of such point algebras. We need a few definitions:

Let $P = (T, \leq)$ be a partial order. A *linear extension* of P is a partial order $P' = (T, \leq')$ such that $\leq \subseteq \leq'$ and for every $x, y \in T$, either $(x, y) \in \leq'$ or $(y, x) \in \leq'$.

Given a set of partial orders $\mathcal{T} = \{(T, \leq_1), \ldots, (T, \leq_n)\}$, the *intersection* $\bigcap \mathcal{T} = (T, \bigcap_{i=1}^{n} \leq_i)$ is also a partial order. The *dimension* $d(P)$ of a partial order is the least k such that there exists a set $\mathcal{T}$ of total orders over T satisfying

1. $|\mathcal{T}| = k$; and
2. $P = \bigcap \mathcal{T}$.

Note that every finite partial order has a well-defined dimension. The dimension of a partial order $P = (T, \leq)$ can be given a well-known geometric interpretation [Ore62]: let π be a mapping from T to distinct points of the d-dimensional Euclidean space E^d. Let $P(\pi) = (T, \leq')$ denote the partial order defined by: $(x, y) \in \leq'$ if and only if each coordinate of $\pi(x)$ is less than or equal to the corresponding coordinate of $\pi(y)$. The dimension of P is then the minimum d for which there exists such a mapping π with $P(\pi) = P$.

Hence, the time model presented above coincides with partially-ordered time where the set of partial orders have bounded dimension. Note that the ordinary partially-ordered time model corresponds to a system containing an infinite number of unsynchronized clocks. Let po/k denote the class of partial orders with dimension bounded by k and note that the set of total orders equals the set $po/1$. Thus, $\textsc{Psat}_{po/1}(\mathcal{PA})$ is the satisfiability problem for the totally ordered time point algebra.

We continue our study of point algebras with bounded dimension by demonstrating the hardness of reasoning about satisfiability even for basic relations. This is followed by a complete classification of the tractable sets of relations and, finally, we also give a complete classification for the case when disjunctions are allowed.

5.1 Basic relations

We will now show that reasoning about partial orders of bounded dimensions is NP-hard even when restricted to the two basic relations $<, \|$. Apart from

demonstrating the difficulty of reasoning about satisfiability over this domain this also lays the foundation for the total classifications in the next two sections.

Theorem 28 Let $X = \{<, \|\}$. Then, $\text{PSAT}_{po/k}(X)$ is NP-complete whenever $k \geq 2$.

The fact that we cannot solve the satisfiability problem for the basic relations implies, among other things, that straightforward backtracking algorithms for $\text{PSAT}_{po/k}(\mathcal{PA})$, $k \geq 2$, will not work. Such algorithms successively refine the relations in the given instance until only the basic relations remain and this modified problem is solved by using some tractable method (such as enforcing path consistency).

The proof of Theorem 28 is relatively easy when $k \geq 3$.

Lemma 29 $\text{PSAT}_{po/k}(X)$ is NP-complete when $k \geq 3$.

Proof: Yannakakis [Yan82] has proved that deciding whether a given partial order has dimension k or not is an NP-complete problem for every fixed $k \geq 3$. Thus, given a partial order $P = (T, \leq)$, we can in polynomial time construct an instance of $\text{PSAT}_{po/k}(X)$ that is satisfiable iff the given partial order has dimension k. For each $t \in T$, introduce a variable t. If two points t and t' are related under $<$, add the constraint $t < t'$; if they are incomparable, add the constraint $t \| t'$. □

When $k = 2$, we use a connection between intervals on the real line and partial orders of dimension 2.

Theorem 30 (Baker, Fishburn and Roberts [BFR72]) A partial order $(T, \leq)$ has dimension 2 iff for each $t \in T$, there exists an interval I_t on the real line such that $t \leq t' \Leftrightarrow I_t \subseteq I_{t'}$.

The problem can now be proved by using Allen's algebra. Allen's interval algebra [All83] is based on the notion of *relations between intervals*. An interval x is represented as a tuple (x^-, x^+) of real numbers with $x^- < x^+$, denoting the left and right endpoints of the interval, respectively. The possible relations between intervals are the 2^{13} possible unions of 13 *basic interval relations*, which are shown in Table 3. The problem of *satisfiability* ($\mathcal{A}$-SAT(S)) of a set of interval variables with relations from S between them

Basic relation		Example	Endpoints
x precedes y	p	xxx yyy	$x^+ < y^-$
y preceded by x	p^{-1}		
x meets y	m	xxxx yyyy	$x^+ = y^-$
y met by x	m^{-1}		
x overlaps y	o	xxxx yyyy	$x^- < y^- < x^+$, $x^+ < y^+$
y overl. by x	o^{-1}		
x during y	d	xxx yyyyyyy	$x^- > y^-$, $x^+ < y^+$
y includes x	d^{-1}		
x starts y	s	xxx yyyyyyy	$x^- = y^-$, $x^+ < y^+$
y started by x	s^{-1}		
x finishes y	f	xxx yyyyyyy	$x^+ = y^+$, $x^- > y^-$
y finished by x	f^{-1}		
x equals y	$\equiv$	xxxx yyyy	$x^- = y^-$, $x^+ = y^+$

Table 3: The thirteen basic relations. The endpoint relations $x^- < x^+$ and $y^- < y^+$ that are valid for all relations have been omitted

is that of deciding whether there exists an assignment of intervals on the real line for the interval variables, such that all of the relations between the intervals are satisfied. The complexity of this problem has been determined for all choices of S [KJJ01].

Theorem 31 $\textsc{Psat}_{po/2}(X)$ is NP-complete.

Proof: The problem $\mathcal{A}$-$\textsc{sat}(\{(\mathsf{dsf}), (\mathsf{pp}^{-1}\mathsf{mm}^{-1}\mathsf{oo}^{-1})\})$ is NP-complete [KJJ01]. Given an instance of this problem, it can trivially be transformed to an instance of $\textsc{Psat}_{po/2}(X)$:

- for each constraint $x(\mathsf{dsf})y$, add the constraint $x < y$; and
- for each constraint $x(\mathsf{pp}^{-1}\mathsf{mm}^{-1}\mathsf{oo}^{-1})y$, add the constraint $x \| y$.

By Theorem 30, the resulting instance is satisfiable iff the given instance of Allen's algebra is satisfiable. □

5.2 Total classification of the point algebra

We continue by giving a total classification of the point algebra. We prove that the following sets of relations are the only maximal tractable sets of relations for $\text{PSAT}_{po/k}$ for $k \geq 2$.

1. $\mathcal{A}_1 = \{<, \leq, <>, < \| >, <=>, =\}$
2. $\mathcal{A}_2 = \{\|, < \|, < \| >, =, \underline{\|}, \leq \|\}$
3. $\mathcal{A}_3 = \{=, \leq, <=>, \underline{\|}, \leq \|\}$

Corollary 18 implies the tractability of $\text{PSAT}_{po/k}(\mathcal{A}_1)$. Tractability for the two other sets of relations is trivial since $\mathcal{A}_2 = \Delta_{\|} \cup \{=\}$, $\mathcal{A}_3 = \Delta_{=}$ and $\Delta_{\|}, \Delta_{=}$ both have the GS property. We need a few auxiliary results to show that these are the only maximal tractable sets of relations.

Lemma 32 Let $P = (V, <_P)$ be a partial order of dimension $k \geq 2$. Then the following partial orders are also of dimension k.

1. $P' = (P \cup \{s\}, <_P \cup \{(y, s) | y <_P x\})$ for any $x \in V$.
2. $P'' = (P \cup \{s\}, <_P \cup \{(s, y) | x <_P y\})$ for any $x \in V$.

Proof: We only prove the first case since the second case can be proven analogously. Let $<_1, \ldots, <_k$ be k linear extensions of $<_P$ such that $<_P = <_1 \cap \cdots \cap <_k$. Furthermore, let $<'_1$ be $<_1 \cup \{(a, s), (s, b) | a, b \in P, a \leq_1 x, x <_1 b\}$ and let $<'_i$ be $<_i \cup \{(a, s) | a \in P\}$ when $i > 1$. Obviously, $<'_1, <'_i$ are total orders and $<_{P'} = <'_1 \cap \cdots \cap <'_k$ which demonstrates that P' has the dimension k. □

Intuitively, the previous lemma states that we can always add a new maximal (case 1) or minimal (case 2) element connected to exactly one $x \in P$ (not counting the transitive relations) without increasing the dimension of the partial order.

Lemma 33 $\text{PSAT}_{po/k}(\{<, \underline{\|}\})$ is NP-complete for $k \geq 2$.

Proof: We prove this by a reduction from the previously proven NP-hard problem $\text{PSAT}_{po/k}(\{<, \|\})$. Let Π be a $\text{PSAT}_{po/k}(\{<, \|\})$ instance and let Π' be the result of replacing each occurrence $x \| y$ of $\|$ in Π with the constraints $x < s, y < t, s \| t, s \| y, t \| x$ where s, t are two fresh variables. We prove that Π is satisfiable iff Π' is satisfiable. For the if-direction assume that Π' is satisfiable and has a model M. Then it must hold that $x \| y$ under M so M is also a model of Π.

For the only-if case assume that Π is satisfiable and has a model M. Let M' be the result of extending M with the two points s, t such that $x < s, y < t$. Obviously this interpretation satisfies all the constraints in Π' and Lemma 32 proves that this extension does not increase the dimension of the partial order. Thus, M' is a model of Π'. □

Lemma 34 $\text{PSAT}_{po/k}(\{\leq, \|\})$ is NP-complete for $k \geq 2$.

Proof sketch: A reduction similar to that in the previous lemma from the $\text{PSAT}_{po/k}(\{<, \|\})$ case can be made by replacing constraints $x < y$ with the constraints $x \leq y, x \| s, s \leq y$ where s is a fresh variable. □

Lemma 35 $\text{PSAT}_{po/k}(\{\|, <>\})$ is NP-complete for $k > 1$.

Proof: In Section 4.2, it was proven that $\text{PSAT}_{po}(\{\|, <>\})$ is NP-hard by a reduction from the NP-complete problem BETWEENNESS. The reduction creates an $\text{PSAT}_{po}(\{\|, <>\})$ instance Π for each BETWEENNESS instance (A, T) and proves that Π is satisfiable iff (A, T) is satisfiable. Obviously, even if we regard Π as an $\text{PSAT}_{po/k}$ problem instance the only-if part still holds.

For the if part we prove that $\mathcal{P}_{|A|}$ is of dimension two and hence, that the proof of the if part in Section 4.2 holds also for all other finite dimensions. Let g, h be the two injective mappings from the points of $\mathcal{P}_{|A|}$ onto $\{1, \ldots, 3|A|\}$ such that:

$$g(x_i) = |A| + 1 - i, h(z_i) = 3|A| + 1 - i$$
$$g(y_i) = |A| + 2i - 1, h(y_i) = 2i$$
$$g(z_i) = |A| + 2i, h(x_i) = 2i - 1$$

These two mapping give us two total orders $<_1 = \{(s, t) | g(s) < g(t)\}$, $<_2 = \{(s, t) | h(s) < h(t)\}$. A visualization of $\mathcal{P}_4$ and the corresponding linear extensions can be found in Figure 4. We see that $<_1 \cap <_2 = \mathcal{P}_{|A|}$ and thus $\mathcal{P}_{|A|}$ is of dimension two. Hence, the reduction from BETWEENNESS holds also for partial orders of fixed dimension ≥ 2. □

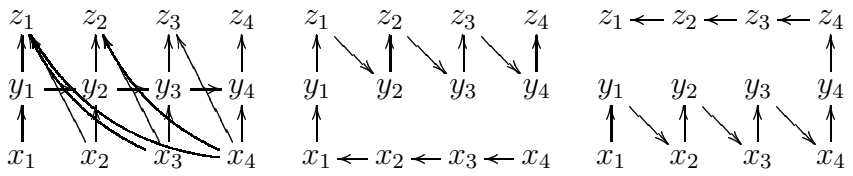


Figure 4: The partial order $\mathcal{P}_4$ and corresponding linearisations

Lemma 36 $\text{PSAT}_{po/k}(\{<=>, \|\})$ is NP-complete for $k \geq 2$.

Proof: We make a reduction from $\text{PSAT}_{po/k}(\{<>, \|\})$ by replacing each constraint $x <> y$ with the constraints $x <=> s, s\|y, x <=> y$ where s is a fresh variable. Since $x = y$ contradicts $x <=> s, s\|y$ we see that the original problem instance is satisfiable if this problem instance is. For the only-if direction assume that the original problem instance has a model $M = (f, (T, <_M))$. We construct an interpretation $M' = (f, (T \cup \{s\}, <'_M))$ where:

$$<_{M'} = \begin{cases} (T \cup \{s\}, <_M \cup \{(z, s) | z <_M x\}) & \text{if } x <_M y \\ (T \cup \{s\}, <_M \cup \{(s, z) | x <_M z\}) & \text{otherwise} \end{cases}$$

Obviously M' satisfies all the constraints in the original problem instance and it also satisfies all the new constraints introduced in the reduction. From Lemma 32 we see that the dimension of M' is the same as that for M. Hence, the reduced problem is satisfiable iff the original problem is. □

We are now ready to prove the main theorem of this subsection. By proving that there exists no other maximal tractable set of relations than $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ we have given a total classification for the point algebra for partial orders of bounded dimensions.

Theorem 37 $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ are the only maximal tractable subclasses of $\text{PSAT}_{po/k}$, for $k \geq 2$.

Proof: Assume that there exists some other maximal tractable set of relations $\mathcal{A}$. From Lemma 6, it follows that the following must hold:

1. $\exists r_1 \in \overline{\mathcal{A}}_1 = \{\|, < \|, \underline{\|}, \leq \|\}$ and $r_1 \in A$
2. $\exists r_2 \in \overline{\mathcal{A}}_2 = \{<, <>, \leq, <=>\}$ and $r_2 \in A$

3. $\exists r_3 \in \overline{\mathcal{A}}_3 = \{<, <>, < \|, < \| >, \|\}$ and $r_3 \in A$

If r_1 is $\|$ or $< \|$, then (regardless of the choice of r_2) $\text{PSAT}_{po/k}(\{r_1, r_2\})$ is NP-complete according to the previous lemmata. Otherwise, r_1 is $\underline{\|}$ or $\leq \|$. If r_2 is $<$ or $<>$, then $\text{PSAT}_{po/k}(\{r_1, r_2\})$ is also NP-complete according to the previous lemmata. If r_2 is $\leq$ or $<=>$ we either have that r_3 is $\|$ which makes $\text{PSAT}_{po/k}(\mathcal{A})$ NP-complete for the same reason as when $r_1 = \|$, or we have that r_3 is $< \|$ or $< \| >$. We see that $r_2 \cap r_3$ is $<$ or $<>$ and $\text{PSAT}_{po/k}(\mathcal{A})$ is NP-complete since $\text{PSAT}_{po/k}(\{r_1, r_2 \cap r_3\})$ is NP-complete. □

5.3 Disjunctions

In this subsection we extend the results of the previous classification to allow disjunctions. We prove that the following sets of relations are the only maximal tractable subclasses of $\text{PSAT}_{po/k}$ extended with disjunctions:

1. $\mathcal{B}_1 = \Gamma_1 {}^{\times}_{\vee} \Delta_1^*$ where $\Delta_1 = \{<>, < \| >, <=>\}$ and $\Gamma_1 = \Delta_1 \cup \{<, \leq, =\}$
2. $\mathcal{B}_2 = \Gamma_2 {}^{\times}_{\vee} \Delta_2^*$ where $\Delta_2 = \{\|, < \|, < \| >, \underline{\|}, \leq \|\}$ and $\Gamma_2 = \Delta_2 \cup \{=\}$
3. $\mathcal{B}_3 = \Delta_3^*$ where $\Delta_3 = \{=, \leq, <=>, \underline{\|}, \leq \|\}$

Tractability of $\text{PSAT}_{po/k}(\mathcal{B}_1)$ follows from tractability of $\text{PSAT}_{po/1}(\mathcal{X}_1)$ by using Corollary 18. Furthermore, since Δ_2 and Δ_3 have the GS property, tractability of $\text{PSAT}_{po/k}(\mathcal{B}_2)$ and $\text{PSAT}_{po/k}(\mathcal{B}_3)$ follows immediately. We continue with a few NP-hardness results needed for the classification result.

Lemma 38 $\text{PSAT}_{po/k}(\{r_1, r_2\})$, $k \geq 2$, is an NP-complete problem whenever $r_1 \in \{<, <>, < \|, \neq, \|\}$ and $r_2 \in (\Gamma_1 - \Delta_1)^2 - (\Gamma_1 - \Delta_1)$.

Proof: First, choose γ_1, γ_2 such that $r_2 = \gamma_1 \vee \gamma_2$. In order to reduce the number of cases needed to be examined we note that $\gamma_1 \vee \gamma_2 = \gamma_2 \vee \gamma_1$. Furthermore if γ_1 is $\leq$ then $\text{PSAT}_{po/k}(\{r_1, r_2\})$ can trivially be reduced to $\text{PSAT}_{po/k}(\{r_1, = \vee \gamma_2\})$. It is therefore sufficient to only examine the cases where r_2 is one of $< \vee <$, $= \vee =$ or $< \vee =$.

For the first and the second case, it has already been proven that r_2 is not 2-independent of r_1 in Lemma 22 for the case when $k = \infty$. Obviously, the very same counterexamples also hold for partial orders of bounded dimensions so $\text{PSAT}_{po/k}(\{r_1, r_2\})$ is NP-hard.

For the third case, we make a reduction from the case when r_2 is $= \vee =$ by replacing each constraint $c : x = y \vee z = w$ with the two constraints $c_1 : x = y \vee t < s$ and $c_2 : s < t \vee z = w$ where s, t are two fresh variables. Note that every model satisfying c_1 and c_2 also satisfies c. We prove the other direction by noting that any model $M = (f, (T, <_M))$ satisfying c can be extended to a model $M' = (f \cup f', (T \cup \{u, v\}, <_M \cup \{(u, v), (u, a), (v, a) | a \in T\}))$ where $f' = \{(s, u), (t, v)\}$ if $f(x) = f(y)$ and $f' = \{(s, v), (t, u)\}$ otherwise. Obviously, M' satisfies all the constraints and Lemma 32 implies that M' has the same dimension as M. Thus, this problem instance is satisfiable iff the original problem instance is. □

Together with the total classification of the previous subsection this lemma can now be used to prove NP-hardness of every set of relation not included in $\mathcal{B}_1, \mathcal{B}_2$ or $\mathcal{B}_3$.

Theorem 39 $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ are the only maximal tractable sets of relations for $\text{PSAT}_{po/k}$ extended with disjunctions for all $k \geq 2$.

Proof: Let $\mathcal{B}$ be a set of relations which is not a subset of one of $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$. It follows from Lemma 6 that there exists $x_1 \in \overline{\mathcal{B}}_1, x_2 \in \overline{\mathcal{B}}_2, x_3 \in \overline{\mathcal{B}}_3$ such that $x_1, x_2, x_3 \in \mathcal{B}$. We will examine every possible choice of x_1, x_2, x_3 and demonstrate that deciding satisfiability for $\mathcal{B}$ is NP-hard. We have two cases.

First, assume that $x_1 \notin \mathcal{PA} - \Gamma_1$. We must then have that $x_1 \in (\Gamma_1 - \Delta_1)^2 - (\Gamma_1 - \Delta_1)$ and from Lemma 38 we see that deciding satisfiability for $\{x_1, x_3\}$ is NP-hard. Furthermore, if $x_2 \notin \mathcal{PA} - \Gamma_2$ then x_2 is $= \vee =$ and Lemma 38 can be applied again which implies NP-hardness of $\{x_2, x_3\}$.

Thus we must have $x_1 \in \mathcal{PA} - \Gamma_1, x_2 \in \mathcal{PA} - \Gamma_2, x_3 \in \mathcal{PA} - \Delta_3$. From Theorem 37 we see that deciding satisfiability for $\{x_1, x_2, x_3\}$ is NP-hard. □

6 Branching Time

We will now turn our attention to the point algebra for branching time. This point algebra restricts the class of partial orders $(T, \leq)$ to those that satisfy the *branching* condition:

$$\forall x, y, z \in T : \text{if } x \| y \text{ and } y < z \text{ then } x \| z.$$

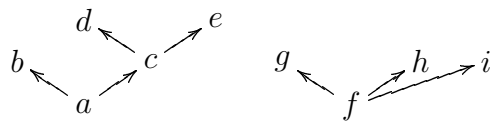


Figure 5: A small subset of **br**.

The set of partial orders satisfying this condition is denoted *br*. Given a partial order $(T, \leq) \in br$, we note that two incomparable points cannot have an upper bound. This tree-like structure motivates the name 'branching'. We also note that $po/1 \subset br \subset po$ but br and po/k are incomparable when $1 < k < \infty$.

In the following, we will view *br* as a forest containing all finite trees infinitely many times; we denote this set **br**. We choose this definition since it simplifies the proofs and does not change the basic model of time. Consequently, we define the basic relations as follows: given arbitrary points x, y in **br**,

1. $x < y$ iff x is below y in **br**.
2. $x > y$ iff y is below x in **br**.
3. $x = y$ iff x and y are the same point.
4. $x \| y$ iff x, y belong to different branches or trees.

Example 6.1 In the subset of **br** (depicted in Figure 5) some of the relations holding between points are: $a < c$, $b \| c$, $b \| e$, $a < d$ and $c \| f$.

A point $n \in \mathbf{br}$ related to some point in the image of f by $<$ or $>$ but not itself in the image is said to be a *redundant* point. If there exist no redundant points for f, i.e. if f satisfies the following:

$$f(V) = \{a \in \mathbf{br} \mid \exists x \in V : a \leq f(x) \vee f(x) \leq a\}$$

we say that f is a *non-redundant* model of Π. Note that if Π has a model, then Π also has a non-redundant model since it is always possible to remove an arbitrary point from a tree and preserve the relations between remaining points.

	$<$	$>$	$\|$	$=$
$<$	$<$	$<=>$	$< \|$	$<$
$>$	$\top$	$>$	$\|$	$>$
$\|$	$\|$	$> \|$	$\top$	$\|$
$=$	$<$	$>$	$\|$	$=$

Table 4: The composition table for the point algebra for branching time.

Example 6.2 Let Π be a problem instance over five variables $\{x_0, \ldots, x_4\}$ and the constraints: $x_0 < x_3, x_2 \neq x_3, x_4 \underline{\|} x_3, x_2 \leq x_4, x_4 \leq x_2, x_1 \| x_4, x_0 <> x_1$ and $x_0 <> x_4$. Furthermore let, f_1, f_2 and f_3 be the following interpretations:

$$f_1(x_0) = a, f_2(x_0) = f, f_3(x_0) = a$$

$$f_1(x_1) = b, f_2(x_1) = g, f_3(x_1) = g$$

$$f_1(x_2) = d, f_2(x_2) = h, f_3(x_2) = d$$

$$f_1(x_3) = e, f_2(x_3) = i, f_3(x_3) = e$$

$$f_1(x_4) = d, f_2(x_4) = h, f_3(x_4) = d$$

where $a, \ldots, i$ are the point in **br** given in Figure 5. We have that f_1 and f_2 are models of Π but f_3 is not since the $x_0 <> x_1$ constraint is not satisfied by f_3. Furthermore f_2 is a non-redundant model while f_1 is redundant since c is connected to the nodes of f_1 but not itself part of the image of f_1.

6.1 Tractability results

We will now show that the sets of relations defined below (where $\Psi_A, \ldots, \Omega_E$ are defined in Table 5) are tractable.

$$\mathcal{T}_A = \Psi_A, \;\; \mathcal{T}_B = \Psi_B {}^{\times}\!\vee \Omega_B^*, \;\; \mathcal{T}_C = \Omega_C^*, \;\; \mathcal{T}_D = \Psi_D {}^{\times}\!\vee \Omega_D^*, \;\; \mathcal{T}_E = \Psi_E {}^{\times}\!\vee \Omega_E^*.$$

We note that all basic relations are included in $\mathcal{T}_A$ and $\mathcal{T}_E$. The subclass $\mathcal{T}_A$ is a bit unusual since it cannot be extended by disjunctions at all—in the case of partially-ordered time, every maximal tractable subclass can be extended

	Ψ_A	Ψ_B	Ω_B	Ω_C	Ψ_D	Ω_D	Ψ_E	Ω_E
$<$	•	•					•	
$\leq$	•	•		•			•	
$<>$	•	•	•				•	
$<=>$	•	•	•	•			•	
$\parallel$	•				•	•	•	
$\underline{\parallel}$	•			•	•	•		
$=$	•	•		•	•		•	
$\neq$	•	•	•		•	•	•	•
$< \parallel$	•				•	•	•	
$\leq \parallel$	•			•	•	•		

Table 5: Tractable classes in branching time.

by disjunctions. The class $\mathcal{T}_B$ always has a model that is a total order by Corollary 18. Furthermore, tractability of $\mathcal{T}_C$ and $\mathcal{T}_D$ is trivial since Ω_C and Ω_D have the GS property. These sets of relations will be proven to be the unique maximal sets of tractable relations for PSAT_{br} in Sections 6.2-6.3.

The tractability of Ψ_A follows from Hirsch's [Hir97] $O(n^5)$ algorithm. We present a considerably faster $O(n^{3.326})$ algorithm for this problem in Section 6.4. We will now proceed to the four other tractable sets of relations.

Theorem 40 The satisfiability problem for $\mathcal{T}_B, \ldots, \mathcal{T}_E$ is tractable.

Proof: We note that every problem instance of $\text{PSAT}_{po}(\mathcal{T}_B)$ and $\text{PSAT}_{po}(\mathcal{T}_C)$ is satisfiable iff it has a totally ordered model (by Corollary 18) or a one-point model, respectively. Hence, we can solve $\text{PSAT}_{\text{br}}(\mathcal{T}_B)$ and $\text{PSAT}_{\text{br}}(\mathcal{T}_C)$ problem instances as problem instances of the point algebra for totally ordered time as described in Section 3.

The following two steps are all that is needed for an algorithm determining satisfiability for problem instances Π of $\text{PSAT}_{\text{br}}(\Psi_D)$.

1. For each constraint $c = (=, x, y)$, contract x and y.

2. If there exists a constraint (R, x, x) such that $= \not\subseteq R$ then reject, else accept.

If the algorithm rejects an instance, it is clearly not satisfiable. Otherwise, it is satisfiable since all remaining variables can be mapped to root nodes in disjoint trees. By an analysis of the algorithm it is obvious that Ω_D is independent of Ψ_D and, hence, $\Psi_D{}^{\times}\!\!\vee\Omega_D^*$ is tractable.

Finally, a tractable algorithm for Ψ'_E can be found in Section 6.4. By analyzing this algorithm in the same way as we analyzed algorithm A in the proof of Lemma 19, we see that $\neq$ is independent of Ψ'_E and hence, ${\Psi'_E}^{\times}\!\!\vee\{\neq\}$ is tractable. Finally, it is easy to show that $\Psi_E = C(\Psi'_E)$ so $\mathcal{T}_E$ is tractable by Theorem 2. □

It is important to note that the automatic method for proving independence suggested by Broxvall *et al.* [BJR02, Paper II] cannot be used for proving the previous theorem. In order to use this method, the satisfiability problem of the underlying set of relations must be decided by path consistency. However, Hirsch [Hir97] has proved that path consistency does not decide $\textsc{Psat}_{\text{br}}(\{<>, < \|\})$.

6.2 Intractability results

This section contains the NP-completeness results which are needed for proving the classification result.

Theorem 41 $\textsc{Psat}_{\text{br}}$is NP-hard for the following sets of relations:

1. $\{< \vee <\}$
2. $\{= \vee =, R_1\}$ where $=$ is not a subset of R_1
3. $\{\| \vee \|, <=>\}$
4. $\{\neq \vee \neq, \underline{\|}, <>\}$
5. $\{\underline{\|} \vee \underline{\|}, <, <=>\}$
6. $\{R_2 \vee R_2, \|\}$ where R_2 is $<>$ or $<=>$
7. $\{<> \vee <=>, \|\}$

Proof: NP-hardness of the first six sets of relations will be proven in a fashion similar to that in Lemma 13. We begin by defining Δ_i, Γ_i as follows:

$$\Delta_1 = \{<\}, \Delta_2 = \{=\}, \Delta_3 = \{\|\}, \Delta_4 = \{\neq\}, \Delta_5 = \{\underline{\|}\}, \Delta_6 = \{R_2\}$$

$$\Gamma_1 = \emptyset, \Gamma_2 = \{R_1\}, \Gamma_3 = \{<=>\}, \Gamma_4 = \{\underline{\|}, <>\}, \Gamma_5 = \{<, <=>\}, \Gamma_6 = \{\|\}$$

Now consider the six problem instances below.

1. $x_1 < x_2 < x_3 < x_1$
2. $x_1 = x_2 = x_3 = x_4, x_1\ R_1\ x_4$
3. $x_1 <=> x_2 <=> x_3 <=> x_4 <=> x_5 <=> x_6 <=> x_1, x_1 \| x_4, x_2 \| x_5, x_3 \| x_6$
4. $x_1 <> x_2 <> x_3 <> x_4 <> x_5 <> x_6 <> x_1, x_1 \underline{\|} x_4, x_2 \underline{\|} x_5, x_3 \underline{\|} x_6, x_1 \neq x_4, x_2 \neq x_5, x_3 \neq x_6$
5. $x_1 <=> x_2 < x_3 <=> x_4 < x_5 <=> x_6 < x_1, x_1 \underline{\|} x_2, x3 \underline{\|} x_4, x_5 \underline{\|} x_6$
6. $x_1\ R_2\ x_2\ R_2\ x_3\ R_2\ x_4\ R_2\ x_5\ R_2\ x_6\ R_2\ x_1, x_1 \| x_4, x_2 \| x_5, x_3 \| x_6$

By using the algorithm in Figure 6, it can be shown that these instances are unsatisfiable even though all subinstances of them containing at most two constraints from Δ_i are satisfiable. Thus, Δ_i is not 2-independent of Γ_i and Theorem 10 implies that deciding satisfiability for the first six sets of relations is NP-hard.

The NP-hardness proof for the final set of relations cannot be derived in the same way. We therefore make a reduction from the NP-hard problem $\text{PSAT}_{\text{br}}\{<> \vee <>, \|\}$. Let C be a set of constraints over these relations. We construct a new set of constraints C' by replacing every occurrence of $c : a <> b \vee c <> d$ with the constraints $\{c_1, c_2\} \cup S$:

$$c_1 : a <> b \vee x_1 <=> x_2, \quad c_2 : c <> d \vee x_2 <=> x_3$$

$$S = \{x_3 <=> x_4 <=> x_5 <=> x_6 <=> x_1, x_1 \| x_4, x_2 \| x_5, x_3 \| x_6\}$$

where $x_1, \ldots, x_6$ are fresh variables. By using the algorithm in Figure 6, it is fairly easy to see that the constraints $\{x_1 <=> x_2\} \cup S$ and $\{x_2 <=> x_3\} \cup S$ are both satisfiable but not $\{x_1 <=> x_2, x_2 <=> x_3\} \cup S$. It follows that any model of C easily can be extended to a model of C'. We also see that in any model of C' we must either have that $a <> b$ or $c <> d$ is satisfied. Thus a model of C' is also a model of C. □

6.3 Maximality

We will now demonstrate that the five sets of relations proven tractable in Section 6.1 are the only maximal tractable sets. To do so, we need a number of equivalence results.

Lemma 42 Let $R_i \in \mathcal{PA}$ and $\Gamma_j \subseteq \mathcal{PA}$. The following problems are equivalent up to polynomial-time reductions.

1. $\textsc{Psat}_{\text{br}}(\{< \vee R_1\} \cup \Gamma_1\})$ and $\textsc{Psat}_{\text{br}}(\{(<) \vee R_1, R_1 \vee R_1\} \cup \Gamma_1)$.
2. $\textsc{Psat}_{\text{br}}(\{= \vee R_1, R_2\} \cup \Gamma_1)$ and $\textsc{Psat}_{\text{br}}(\{= \vee R_1, R_2, R_1 \vee R_1\} \cup \Gamma_1)$ where $= \not\subseteq R_2$.
3. $\textsc{Psat}_{\text{br}}(\{\| \vee R_1, R_2\} \cup \Gamma_1)$ and $\textsc{Psat}_{\text{br}}(\{\| \vee R_1, R_2, R_1 \vee R_1\} \cup \Gamma_1)$ where R_2 is $(<>)$ or $(<=>)$.
4. $\textsc{Psat}_{\text{br}}(\{\underline{\|} \vee R_1, < \|, R_2\} \cup \Gamma_1)$ and $\textsc{Psat}_{\text{br}}(\{\underline{\|} \vee R_1, < \|, R_2, R_1 \vee R_1\})$ where R_2 is $(<>)$ or $(<=>)$.
5. $\textsc{Psat}_{\text{br}}(\{\neq \vee R_1, \|, R_1\} \cup \Gamma_1)$ and $\textsc{Psat}_{\text{br}}(\{\neq \vee \neq, \neq \vee R_1, \|, R_1\} \cup \Gamma_1)$ whenever R_1 is the relation $(<>)$ or $(<=>)$;
6. $\textsc{Psat}_{\text{br}}(\{R_1 \vee R_2, R_1 \vee R_3\} \cup \Gamma_1)$ and $\textsc{Psat}_{\text{br}}(\{R_1 \vee R_2, R_1 \vee R_3, R_1 \vee (R_2 \oplus R_3)\} \cup \Gamma_1)$ when $\oplus \in \{\circ, \cap\}$.

Proof: The first two cases were proved in Lemma 23 for partially-ordered time and it is easy to show that they hold also for branching time. For the third case consider the following set of relations:

$$a\ R'\ b,\ b\ R'\ c,\ c\ R'\ d,\ x\ R\ y \vee a \| c,\ z\ R\ w \vee b \| d$$

and note that at most one of $a\|c$ and $b\|d$ can hold. Hence, $x\ R\ y \vee z\ R\ w$ must hold but without further restricting the choices of x, y, z, w. We use this in order to make a polynomial reduction from $\textsc{Psat}_{\text{br}}(\{\| \vee R, R', R \vee R\} \cup \Gamma)$ to $\textsc{Psat}_{\text{br}}(\{\| \vee R, R'\} \cup \Gamma)$. Let Π be an arbitrary problem instance of the first set of relations and replace each $R \vee R$ constraint of Π with the constraints above where a, b, c, d are fresh variables and x, y, z, w are the variables of the $R \vee R$ constraint. Clearly, the resulting set of constraints is an instance of $\textsc{Psat}_{\text{br}}(\{\| \vee R, R'\} \cup \Gamma)$ that is satisfiable iff the original instance is satisfiable. Correctness of the next two points follows by similar reasoning. The final point follows from the following two equivalences.

- $x\ \gamma_1\ y \vee z\ (\gamma_2 \cap \gamma_3)\ w$ is equivalent to $x\ \gamma_1\ y \vee z\ \gamma_2\ w$ and $x\ \gamma_1\ y \vee z\ \gamma_3\ w$
- $x\ \gamma_1\ y \vee z\ (\gamma_2 \circ \gamma_3)\ w$ is equivalent to $x\ \gamma_1\ y \vee z\ \gamma_2\ t$ and $x\ \gamma_1\ y \vee t\ \gamma_3\ w$ where t is a fresh variable.

□

If Γ is a set of relations, we define $C_{\mathrm{br}}(\Gamma)$ as the least set X such that

1. $\Gamma \subseteq X$;
2. X is closed under converse, intersection and composition on point relations; and
3. X is closed under the equivalences in the previous lemma.

It is a routine verification to show that if $\mathrm{PSAT}_{po}(\Gamma)$ is tractable, then the problem $\mathrm{PSAT}_{po}(C_{\mathrm{po}}(\Gamma))$ is also tractable. We can now prove the main result by using a computer-assisted case analysis as in the proof of Theorem 24.

Theorem 43 $\mathcal{T}_A, \mathcal{T}_B, \mathcal{T}_C, \mathcal{T}_D$ and $\mathcal{T}_E$ are the only maximal tractable disjunctive subclasses of $\mathrm{PSAT}_{\mathrm{br}}$.

6.4 Algorithm for Ψ_A

We will describe an alternative algorithm for Ψ_A in this section; the algorithm can be found in Figure 6. It runs in $O(nM(n))$ time, where $M(n)$ denotes the time complexity of multiplying two $n \times n$ matrices. Using the $O(n^{2.376})$ algorithm for matrix multiplication proposed by Coppersmith and Winograd [CW90], this algorithm is a significant improvement over the $O(n^5)$ algorithm given by Hirsch [Hir97].

The algorithm works by first partitioning the problem instance into sets of variables which can be mapped to disjoint trees, i.e. all constraints between the partitions include the relation $\|$. Next, the algorithm tries to identify a variable *root* which can be mapped to the root node for each partition. Note that there exists exactly one such node for each satisfiable partition since each partition maps to a distinct tree.

Let x_1 be a candidate variable to be mapped to the root node in a satisfiable problem instance. We note that if there exists a chain of constraints $x_1\ R_1\ x_2\ \cdots\ x_{n-1}\ R_n\ x_n$ such that $< \not\subseteq R_i$ we must also map $x_2, \ldots, x_n$ to the root node. Existence of the constraints $x_1\ R_1\ x_2\ \cdots\ x_{n-1}\ R_n\ x_n$ for each

pair of variables x_1, x_n is checked by looking at the adjacency matrix M of the transitive closure for the graph G_i constructed by the algorithm for every partition V_i. If $M[x, y] = 1$ and x is mapped to the root node then so must y be.

Thus, for x to be mapped to the root node there must not exist a pair of variables which both are identified with x and for which we have a constraint not allowing the equality relation, i.e. there must not exist variables y, z and a constraint $c = yRz$ such that $= \not\subseteq R$, $M[x, y] = 1$ and $M[x, z] = 1$. Conveniently, the existence of z and c such that $= \not\subseteq R$ and $M[x, z] = 1$ can be computed by matrix multiplication. This is done in step 18, where $P[x, y] = 1$ iff there exists such a variable z and constraint c. After identifying a variable *root* which can be mapped to the root node, every variable which must be identified with *root* is removed from the partition and the algorithm is recursively called for this new problem instance.

Before proving correctness of the algorithm we recall the definition of matrix multiplication.

$$C = A \cdot B \Leftrightarrow C[i, j] = \sum_{k=1}^{n} A[i, k] \cdot B[k, j] \tag{1}$$

Lemma 44 If algorithm C rejects a problem instance then it is not satisfiable.

Proof: Assume to the contrary that there exists a satisfiable problem instance Π which is rejected by the algorithm. Then there exists a component V_i of a subset of the original problem instance such that the algorithm rejects in line 27. Note that rejection on line 26 only occurs if the algorithm rejected on line 27 after some recursions. Let Π' denote the problem instance (V_i, C_i) where V_i, C_i are the variables and constraints making the algorithm reject on line 27. Trivially, the algorithm rejects also Π'. Note that Π' is a subinstance of Π and, hence satisfiable.

Since Π' is satisfiable there exists a non-redundant model f of Π'. Assume that there exists more than one minimal point in the image of Π' under f and let $f(x)$ and $f(y)$ denote two such points. Note that minimal points in **br** are the roots of distinct trees. Since the graph G constructed by the algorithm for Π' contains only one component then there exists a path of constraints $x\ R_1\ x_1\ \cdots\ x_n\ R_{n+1}\ y$ such that $\| \not\subseteq R_1, \ldots, R_{n+1}$. Since $f(x)$ and $f(y)$ are the roots of different trees we obviously have some x_i, x_{i+1} whose images lies

```
1  Algorithm C(Π)
2  Input: Problem instance Π = (V, C) of Ψ_A
3  Let G be the undirected graph (V, ∅)
4  for each constraint xRy ∈ C such that ∥ ⊈ R do
5    add the edge {x, y} to G
6  Partition G into components V_1, ..., V_k
7  Partition C into C_1, ..., C_k such that:
8  C_i = {xRy ∈ C | x, y ∈ V_i}
9  for each component V_i do
10   Let G_i be the directed graph (V_i, ∅)
11   for each c = xRy ∈ C_i such that < ⊈ R do
12     add the (directed) edge (x, y) to G_i
13   Let M be the adjacency matrix of the transitive
14   and reflexive closure of G_i
15   N ← new empty matrix indexed by V_i
16   for each xRy ∈ C_i such that =⊈ R do
17     N[x, y] ← 1
18   P ← M · N
19   root ← ⊥
20   for each x ∈ V_i do
21     if ∀y ∈ V_i : P[x, y] = 0 or M[x, y] = 0 then
22       root ← x
23   if root ≠ ⊥ then
24     V_i' ← V_i − {c | M[root, c] = 1}
25     C_i' ← {xRy ∈ C_i | x, y ∈ V_i'}
26     if Branch((V_i', C_i')) rejects then return false
27   else return false
28 return true
```

Figure 6: Algorithm for determining satisfiability for Ψ_A

in different trees thus violating the constraint $x_i \ R_{i+1} \ x_{i+1}$. Contradiction, and we have only one minimal point which must be the root of all other points in the image in Π'.

Let x denote a variable in Π' such that $f(x)$ is the minimal point. Since the algorithm rejects, we know that there exists some $y \in V_i$ such that $P[x, y] \geq 1$ and $M[x, y] = 1$. From the definition of matrix multiplication (1) follows the equations (2) and (3) which obviously are equivalent.

$$P[x, y] = \sum_{z \in V_i} M[x, z] \cdot N[z, y] \geq 1 \tag{2}$$

$$\exists z \in V_i : M[x, z] = N[z, y] = 1 \tag{3}$$

Hence, we have $z \in V_i$, $c = zRy \in C_i$ such that $M[x, z] = 1$ and $= \not\subseteq R$. Since $M[x, y] = 1$, we have a chain of constraints $x \ R_1 \ y_1 \ R_2 \ \cdots \ R_{n+1} \ y$ such that $< \not\subseteq R_i$. Note that $f(x) \leq f(y_i)$ and hence, $f(x) = f(y_1)$ and $f(y_1) \leq f(y_i)$ which gives $f(x) = f(y_1) = f(y_2)$ and $f(y_2) \leq f(y_i)$ etc. This leads to $f(x) = f(y_i) = f(y)$. Analogously, $f(x) = f(z_i) = f(z)$. Contradiction, since $f(z) = f(x) = f(y)$ violates the constraint c. Thus, Π' and Π are not satisfiable. □

Having proven that the algorithm rejects only unsatisfiable instances, we will now demonstrate that the algorithm correctly identifies the satisfiable problem instances.

Lemma 45 Algorithm C only accepts satisfiable problem instances.

Proof: We prove the result by induction over the number of variables in the given problem instance. Obviously, all problem instances accepted by the algorithm containing zero variables are satisfiable. Assume that acceptance of the algorithm implies satisfiability for all problem instances of size n or less. Let Π be a problem instance of size $n+1$ which is accepted by the algorithm. We construct an interpretation f of the problem instance as follows:

1. For each component i, let f_i' denote a model of (V_i', C_i). Note that $|V_i'| < |V|$ since at least one variable is removed on line 24. Existence of f_i' follows from the induction hypothesis.

2. Let V_i' be an arbitrary component and T_i the subset of **br** that f_i' maps V_i' to; i.e. **br** restricted to $\{x \in \mathbf{br} \mid x = f_i'(v) \text{ and } v \in V_i'\}$. Consider another subtree U_i of **br** that is isomorphic to T_i with one exception; it contains a fresh root point t_i (which is below all other points). Modify f_i' to map V' to $U_i - \{t_i\}$ instead.

   We define an interpretation f_i of (V_i, C_i) as follows:

$$f_i(x) = \begin{cases} t_i & \text{iff } M[root, x] = 1 \\ f_i'(x) & \text{otherwise} \end{cases}$$

3. Define f as the union of the (disjoint) interpretations f_i.

We continue by demonstrating that each constraint $C = xRy$ is satisfied by f. Assume $x \in V_i$ and $y \in V_j$. We have two cases.

1. If $i = j$ there are four cases to analyze. If $M[root, x] = M[root, y] = 0$ then C is included in (V_i, C_i) and hence, satisfied by f_i',f_i and thus f. When $M[root, x] = M[root, y] = 1$ we have $P[root, y] = 0$ which implies $N[x, y] = 0$. Hence, we have $= \subseteq R$ and thus, C is satisfied by f_i and f. Finally, when $M[root, x] = 1$ and $M[root, y] = 0$ we have $< \subseteq R$ since there otherwise would exist a path from $root$ to y in the graph G_i constructed by the algorithm. Hence, f_i and f satisfies C. The case $M[root, x] = 0$ and $M[root, y] = 1$ can be shown analogously.

2. Assume $i \neq j$. Since the algorithm has partitioned x and y into different components we know that $\| \subseteq R$ and c is satisfied by f.

□

By demonstrating that the algorithm runs in polynomial time we conclude that the satisfiability problem for $\mathcal{T}_A$ is tractable. This is done in the following theorem.

Theorem 46 Algorithm C runs in $O(nM(n))$ time.

Proof: We begin by noting that the initial graph partitioning can be performed in $O(n^2)$ steps by applying a standard graph partitioning algorithm, cf. Baase [Baa88]. Also, the transitive closure of G_i on line 13 can be computed in the same time as the boolean matrix multiplication on line

18 [Baa88]. Hence, steps 10-26 can all be performed within $O(M(n_i))$ time. The function $f(n)$ is an upper bound of the total running time of algorithm C:

$$f(n) \leq c_1 n^2 + \sum_{i=1}^{k} (c_2 M(n_i) + f(n_i - 1))$$

for some constants c_1, c_2 and where n_i denotes the size of component V_i. Note that we have $n_1 + \ldots + n_k = n$. Furthermore, for all $c \geq 0$ we have that $n_1^c + \cdots + n_k^c \leq (n_1 + \cdots + n_k)^c$. Hence, $\sum_{i=1}^{k} c_2 M(n_i) \leq c_2 M(\sum_{i=1}^{k} n_i) = c_2 M(n)$. This gives us:

$$f(n) \leq (c_1 + c_2) M(n) + \sum_{i=1}^{k} f(n_i - 1)$$

and it can be shown that $f(n)$ is a polynomial with positive coefficients by induction over n. Consequently, the worst case appears when $k = 1$, that is,

$$f(n) \leq (c_1 + c_2) M(n) + f(n-1) = \sum_{i=1}^{n} (c_1 + c_2) M(i).$$

Given a polynomial p that is everywhere greater than zero, it holds that $\sum_{i=1}^{n} p(i) \leq n \cdot p(n)$. Hence,

$$f(n) \leq \sum_{i=1}^{n} (c_1 + c_2) M(i) \leq (c_1 + c_2) n M(n)$$

□

6.5 Algorithm for Ψ_E

This section contains the final part of the proof of Theorem 40; that is, we present a polynomial-time algorithm for deciding satisfiability of $\text{PSAT}_{\text{br}}(\Psi'_E)$ (where $\Psi'_E = \{\leq, \|, \neq\}$).

A variable n is said to be *superminimal* in a problem instance Π iff n is a $(\leq)$-minimal variable of Π and there exists no x such that $n \| x$. A *semipath* between two variables n_1, n_2 in an instance Π is a set of variables $a_1, \ldots, a_k, b_1, \ldots, b_k$ such that:

$$n_1 \leq^* a_1 \geq^* b_1 \leq^* \ldots \leq^* a_k \geq^* b_k \leq^* n_2$$

Note that if two variables n_1, n_2 are part of the same $(\leq)$-component, then there exists a semipath between n_1, n_2. Before we can prove the main result of this section, we need several lemmata.

Lemma 47 If $a \leq^* b$ and $b <=> c$, then $a <=> c$.

Proof: Follows from the composition table for branching time. □

In the point algebra of branching time we have an important property which helps us prove correctness of algorithm D. Only superminimal variables in a problem instance Π may be mapped to a minimal point in any model of Π, which the following lemma proves.

Lemma 48 Assume that Π is a problem instance containing variables n_1, n_2 such that $n_1 \| n_2$ and there exists a semipath between n_1, n_2, the image of n_1 cannot be a minimal node in any model of Π.

Proof: Assume M is a model of Π such that $M(n_1)$ is a minimal node. Since there exists a semipath between n_1, n_2 there exists $a_1, \ldots, a_k, b_1, \ldots, b_k$ such that:

$$n_1 \leq^* a_1 \geq^* b_1 \leq^* \ldots \leq^* a_k \geq^* b_k \leq^* n_2$$

Lemma 47 gives $n_1 <=> b_1$, $M(n_1)$ minimal gives $n_1 \leq b_1$. Note that $n_1 \leq b_1 \leq^* a_2$ and $b_2 \leq^* a_2$, Lemma 47 gives $n_1 <=> b_2$ thus $n_1 \leq b_2$. Repeat for each b in $b_2, \cdots, b_k$ and we have $n_1 \leq b_k \leq^* n_2$. Hence $n_1 \leq^* n_2$ and $n_1 \| n_2$, contradiction. Thus there exists no model M such that $M(n_1)$ is a minimal node. □

Lemma 49 Let Π be a problem instance of $\textsc{Psat}_{br}(\Psi'_E)$ such that Π is $(\leq)$-acyclic and f is a non-redundant model of Π. Then there exists at least one minimal point p in **br** which is the image of a $(\leq)$-minimal variable in Π.

Proof: Arbitrarily choose a variable $v \in \text{Var}(\Pi)$ and let T be the unique subtree of **br** such that $f(v) \in T$. Let p be a minimal point in T.

Let $N = n_1, \cdots, n_k$ be the variables in Π that are mapped to p by f. We know that M is nonredundant so $N \neq \emptyset$. Assume now that no member of N is $(\leq)$-minimal. Π is acyclic so there exists a $(\leq)$-minimal variable n' such that $n' \leq^* n_1 \in N$. The $(\leq)$-minimality of n' implies that $n' \notin N$ and $f(n') \neq p$. This leads immediately to a contradiction since $f(n') < f(n_1) = p$ and p is minimal in $(T, \leq)$. Consequently, there exists a $(\leq)$-minimal variable in N that is mapped to p. □

Now, we can prove that algorithm D (see Figure 7) correctly solves $\textsc{Psat}_{br}(\Psi'_E)$. Given a problem instance Π and a variable v, we write $\Pi - \{n\}$ to denote the problem instance $(V - \{v\}, C')$ where

$$C' = \{xry \in C \mid x \neq v \text{ and } y \neq v\}.$$

Lemma 50 Algorithm D correctly solves the $\textsc{Psat}_{br}(\Psi'_E)$ problem.

Proof: We prove correctness of the algorithm by induction over the number of variables in the instance. The algorithm accepts all instances Π containing zero variables and such instances are trivially satisfiable.

Assume the algorithm correctly solves the problem for every instance containing k variables. Let Π be an arbitrary instance containing $k+1$ variables. We consider four cases:

1. (lines 5-7) There exist two variables n_1, n_2 such that $n_1 \leq^* n_2, n_2 \leq^* n_1$. If $n_1 \neq n_2$ in Π, then there can be no model of Π and the algorithm rejects. Otherwise, by Lemma 15 Π is satisfiable iff contract(Π, n_1, n_2) is satisfiable and the algorithm is correct by the induction hypothesis.

2. (lines 10-11) There exists at least two $(\leq)$-components $C_1, \cdots, C_n$ of Π. If the algorithm rejects some C_i, then C_i cannot have a model by the induction hypothesis. Since C_i is a subinstance of Π, there exists no model of Π.

   We show that if C_i is satisfiable for every i then Π is satisfiable. For each $i \leq n$, let f_i denote a model of C_i and assume without loss of generality that the ranges of these models are different subtrees of **br**.

   Create a new interpretation f by taking the union of $f_1, \ldots, f_n$. Note that the only possible relation r between two distinct variables $n_i \in C_i, n_j \in C_j$ can be $\neq$ or $\|$. Obviously, $f(n_i) r f(n_j)$ holds since n_i and n_j are mapped to different trees. Thus, it follows that f is an interpretation of Π since every f_i is an interpretation of C_i.

3. (lines 12-13) There exists only one $(\leq)$-component of Π which has a superminimal variable n. If the algorithm rejects, then $\Pi - \{n\}$ is unsatisfiable by the induction hypothesis and Π cannot have a model.

   If $\Pi - \{n\}$ is satisfiable, then Π is satisfiable: let f' denote a model of $\Pi - \{n\}$ and T_i the subset of **br** that f' maps the variables in Π

to. Consider another subtree U_i of **br** that is isomorphic to T_i with one exception; it contains a fresh root point t (which is below all other points). Modify f' to map the variables to $U_i - \{t\}$ instead. Now, simply define an interpretation of Π as follows: $f(n) = t$ and $f(v) = f'(v)$ for all other variables.

Clearly, f is an interpretation of $\Pi - \{n\}$ since the only added point t is a minimal point. Assume $nrv \in \Pi$. If r is $\neq$, then the relation holds trivially. Otherwise r is $\leq$ since the superminimality of n prohibits r from being either $\|$ or $\geq$. Hence, the relation holds since $f(n)$ is a minimal point.

4. (line 15) There exists only one ($\leq$)-component of Π and Π contains no superminimal variable. Assume Π is satisfiable. Then, there exists a non-redundant model M of Π. By Lemma 49, M has a minimal point p which is the image of some ($\leq$)-minimal variable n. Since n is a minimal variable but not a superminimal variable there exists an x such that $n \| x$. Since Π only has one component there exists a ($\leq$)-semipath between n and x. By Lemma 48 the image p of n cannot be a minimal variable which leads to a contradiction. Hence, there cannot exist a model of Π.

□

Lemma 51 $\textsc{Psat}_{br}(\Psi'_E)$ is tractable.

Proof: The correctness of algorithm D follows from Lemma 50. To show that D runs in polynomial time, let $p(n)$ be a polynomial that is an upper bound for the time complexity of which steps 5 and 9 can be performed. The time complexity of the algorithm is bounded from above by the solution to the following recursive equation:

$$\begin{aligned} f(1) &= 1 \\ f(n) &= \max \begin{cases} p(n) + f(n-1) \\ p(n) + \sum_{i=1}^{k} f(c_i) & : k \geq 2, c_i > 0, \sum_{i=1}^{k} c_i = n \end{cases} \end{aligned}$$

The case $f(n) = p(n) + f(n-1)$ occurs when the algorithm recurses at line 13 and the case $f(n) = \sum f(c_i)$ occurs when the algorithm recurses at line 11 (where c_i denotes the size of component C_i). Proving that $f(n) \leq c \cdot np(n)$ for some constant c is analogous to the proof of Theorem 46. □

```
1  Algorithm D(Π)
2  Input  An instance Π of PSAT_br(Ψ'_E)
3  if Π is empty  then
4    return true
5  elsif ∃n1, n2 ∈ Var(Π) such that n1 ≤* n2, n2 ≤* n1 ∈ Π  then
6    if n1 ≠ n2 ∈ Π  then return false
7    else return D(contract(Π,n1,n2))
8  else
9    Identify the (≤)-components C1, ···, Ck of Π.
10   if k > 1  then
11     return D(C1) ∧ D(C2) ∧ ··· ∧ D(Ck)
12   elsif ∃n ∈ Var(Π) such that n is superminimal  then
13     return D(Π − {n})
14   else
15     return false
16   end if
17 end if
```

Figure 7: The algorithm for solving $\textsc{Psat}_{br}(\Psi'_E)$

7 Concluding Remarks

We have studied the computational complexity of a number of temporal reasoning problems in different time models such as totally-ordered, partially-ordered and branching time. Primarily, we have considered the satisfiability problem for point algebras and point algebras extended with disjunctions—for these problems, we have identified all tractable subclasses. We have also considered a number of other issues such as a new time model suitable for reasoning about systems with a bounded number of unsynchronized clocks, improved algorithms for branching time and examples of how our complexity results can be transferred to other domains.

We have only considered qualitative relations between time points in this article. For some applications this is satisfactory but for others we must have the ability to reason about metric time. Reasoning about metric time in the totally-ordered model of time is straightforward from a conceptual point of view—each time point is typically viewed as a real or rational number. Furthermore, the linear-programming approach by Jonsson & Bäckström [JB98] and Koubarakis [Kou96] offers a straightforward method for reasoning about

metric information; for instance, these methods can be used for extending Allen's algebra with metric constraints. The satisfiability problem for the resulting formalism is still NP-complete so the extension does not increase the computational complexity. It should also be noted that the ORD-Horn class [NB95] is still tractable after such an extension.

Adding metric constraints to partially-ordered models of time is more difficult. In this case, it is not obvious what is meant by metric constraints. The approach taken in Section 5 can fairly easily be extended to a reasonable notion of metric time; each time point is a k-vector of real numbers. We can extend this idea to ordinary partially-ordered time by considering a time point to be a function from the natural numbers to the real numbers, i.e. an infinite vector of reals. Whether this idea is to be considered reasonable or not may of course be discussed. An intuitively appealing definition of metric time in the branching time model seems to be even less obvious.

Algorithmic methods for identifying tractable disjunctive constraints have been presented earlier in the literature [BJR02, Paper II] The methods used in this paper complements this idea by providing automatic methods for proving intractability of disjunctive constraints (as an example, consider the proofs in Section 6.2). By demonstrating that sets of relations of the form $\Gamma \cup \Delta^2$ are tractable only if Δ is 2-independent of Γ, we have an automatic method for identifying NP-hard sets of relations; one can show that Δ is not 2-independent of Γ by exhaustively generating problem instances and testing for satisfiability. Even though this is not necessarily a complete method, note that many of the NP-hard sets of relations used in this article were identified using this method. Furthermore, for the domains where path-consistency decides consistency for the tractable fragments (such as $\textsc{Psat}_{po}$ and $\textsc{Psat}_{to}$), tractable disjunctive extensions can also be identified using an automatic method [BJR02, Paper II]. Together, these two methods are probably useful tools for classifying new domains. This raises the question under which conditions the methods are sufficient to provide a complete classification.

Acknowledgments

We would like to thank Víctor Dalmau and Debasis Mitra for providing helpful comments on earlier drafts of this paper. This research has been

supported by the ECSEL graduate student program, the *Swedish Research Council for the Engineering Sciences* (TFR) under grant 97-301 and by the *Swedish Research Council* (VR) under grant 221-2000-361.

References

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[AMR98] Frank D. Anger, Debasis Mitra, and Rita V. Rodriguez. Temporal constraint networks in nonlinear time. Technical report, ECAI Workshop on Temporal and Spatial Reasoning, Brighton, UK, 1998.

[Ang89] Frank D. Anger. On Lamport's interprocessor communication model. *ACM Transactions on Programming Languages Systems*, 11(3):404–417, 1989.

[AR96] Frank Anger and Rita Rodriguez. The lattice structure of temporal interval relations. *Journal of Applied Intelligence*, 6(1):29–38, 1996.

[Baa88] Sara Baase. *Computer Algorithms: Introduction to Design and Analysis*. Addison Wesley, Reading, MA, 2nd edition, 1988.

[BFR72] K. R. Baker, Peter C. Fishburn, and Fred S. Roberts. Partial orders of dimension 2. *Networks*, 2:11–28, 1972.

[BJ99] Mathias Broxvall and Peter Jonsson. Towards a complete classification of tractability in point algebras for nonlinear time. In *Proceedings of the 5th International Conference on Principles and Practice of Constraint Programming*, pages 129–143, Alexandria, VA, USA, 1999.

[BJ00] Mathias Broxvall and Peter Jonsson. Disjunctive temporal reasoning in partially ordered time structures. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 464–469, Austin, Texas, USA, 2000. AAAI Press.

[BJR02] Mathias Broxvall, Peter Jonsson, and Jochen Renz. Disjunctions, independence, refinements. *Artificial Intelligence*, 140(1–2):153–173, 2002.

[Bro01] Mathias Broxvall. The point algebra for branching time revisited. In *Proceedings of the Joint German/Austrian Conference on Artificial Intelligence (KI-2001)*, Vienna, Austria, 2001.

[CES86] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages Systems*, 8(2):244–263, 1986.

[CJJK01] David Cohen, Peter Jeavons, Peter Jonsson, and Manolis Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47(5):826–853, 2001.

[CW90] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *Symbolic Computation*, 9(3):251–280, 1990.

[DB88] Thomas Dean and Mark Boddy. Reasoning about partially ordered events. *Artificial Intelligence*, 36:375–399, 1988.

[DB99a] Thomas Drakengren and Marcus Bjäreland. Expressive reasoning about action in nondeterministic polynomial time. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 166–171, Stockholm, Sweden, 1999.

[DB99b] Thomas Drakengren and Marcus Bjäreland. Reasoning about action in polynomial time. *Artificial Intelligence*, 115(1):1–24, 1999.

[DWM99] Ivo Düntsch, Hui Wang, and Stephen McCloskey. Relations algebras in qualitative spatial reasoning. *Fundamenta Informaticae*, 39(3):229–248, 1999.

[EH86] Allen E Emerson and Joseph Y Halpern. "Sometimes" and "not never" revisited: On branching versus linear temporal logic. *Journal of the ACM*, 33(1):151–178, January 1986.

[ES88] E. Allen Emerson and Jai Srinivasan. Branching time temporal logic. In *Proceedings of REX Workshop 1988*, pages 123–172, 1988.

[ES89] E. Allen Emerson and Jai Srinivasan. Branching time temporal logic. In J. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, pages 123–172, New York, 1989. Springer-Verlag.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[GS93] Martin Charles Golumbic and Ron Shamir. Complexity and algorithms for reasoning about time: A graph-theoretic approach. *Journal of the ACM*, 40(5):1108–1133, 1993.

[Hir97] Robin Hirsch. Expressive power and complexity in algebraic logic. *Journal of Logic and Computation*, 7(3):309–351, 1997.

[JB98] Peter Jonsson and Christer Bäckström. A unifying approach to temporal constraint reasoning. *Artificial Intelligence*, 102(1):143–155, 1998.

[JDB99] Peter Jonsson, Thomas Drakengren, and Christer Bäckström. Computational complexity of relating time points with intervals. *Artificial Intelligence*, 109(1–2):273–295, 1999.

[KJJ01] Andrei Krokhin, Peter Jeavons, and Peter Jonsson. Reasoning about temporal relations: The tractable subalgebras of Allen's interval algebra. Technical Report PRG-RR-01-12, Computing Laboratory, Oxford University, 2001. Available from `web.comlab.ox.ac.uk/oucl/publications/tr/rr-01-12.html`.

[Kou96] Manolis Koubarakis. Tractable disjunctions of linear constraints. In *Proceedings of the 2nd Conference on Principles and Practice of Constraint Programming (CP'96)*, pages 297–307, Boston, MA, 1996.

[Lam78] Leslie Lamport. Time, clocks and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, 1978.

[Lam86] Leslie Lamport. The mutual exclusion problem: Part I—a theory of interprocess communication. *Journal of the ACM*, 33(2):313–326, 1986.

[McD82] Drew McDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.

[McM92] Kenneth L. McMillan. *Symbolic Model Checking. An approach to the state explosion problem.* PhD thesis, Carnegie-Mellon University, Pittsburgh, PA, USA, 1992.

[NB95] Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *Journal of the ACM*, 42(1):43–66, 1995.

[Ore62] O. Ore. *Theory of Graphs.* American Mathematical Society, Providence, RI, USA, 1962.

[RA98] Rita V. Rodriguez and Frank. D. Anger. Using constraint propagation to reason about unsynchronized clocks. *Constraints*, 2:191–202, 1998.

[Ren99] Jochen Renz. Maximal tractable fragments of the region connection calculus: A complete analysis. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 448–454, Stockholm, Sweden, 1999.

[Ren01] Jochen Renz. A spatial odyssey of the interval algebra: 1. Directed intervals. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-2001)*, pages 51–56, Seattle, WA, USA, 2001.

[RN99] 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.

[VKvB89] Marc B. Vilain, Henry A. Kautz, and Peter G. van Beek. Constraint propagation algorithms for temporal reasoning: A revised report. In Daniel S. Weld and Johan de Kleer, editors, *Readings in Qualitative Reasoning about Physical Systems*, pages 373–381. Morgan Kaufmann, San Mateo, CA, 1989.

[Win89] Glynn Winskell. An introduction to event structures. In J. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, pages 364–397, 1989.

[Yan82] Mihalis Yannakakis. The complexity of the partial order dimension problem. *SIAM J. on Alg. Disc. Meth.*, 3(3):351–358, 1982.

Paper II

Disjunctions, Independence, Refinements

Mathias Broxvall and Peter Jonsson and Jochen Renz

Abstract

An important question in constraint satisfaction is how to restrict the problem to ensure tractability (since the general problem is NP-hard). The use of disjunctions has proven to be a useful method for constructing tractable constraint classes from existing classes; the well-known 'max-closed' and 'ORD-Horn' constraints are examples of tractable classes that can be constructed this way. Three sufficient conditions (the guaranteed satisfaction property, 1-independence and 2-independence) that each ensure the tractability of constraints combined by disjunctions have been proposed in the literature. We show that these conditions are both necessary and sufficient for tractability in three different natural classes of disjunctive constraints. This suggests that deciding this kind of property is a very important task when dealing with disjunctive constraints. We provide a simple, automatic method for checking the 1-independence property—this method is applicable whenever the consistency of the constraints under consideration can be decided by path-consistency. Our method builds on a connection between independence and refinements (which is a way of reducing one constraint satisfaction problem to another.)

Contents

1 Introduction

The constraint satisfaction problem provides a natural framework for expressing many combinatorial problems in computer science. Since the general problem is NP-hard [Mac77], an important question is how to restrict the problem to ensure tractability. This research has mainly followed two different paths: restricting the scope of the constraints [Fre85, GLS99], i.e. which variables may be constrained with other variables, or restricting the constraints [DBvH99, JC95, vBD95], i.e. the allowed values for mutually constrained variables. In this paper, we will only consider problems where the constraints are restricted.

Quite a large number of tractable subclasses of the CSP problem have been identified in the literature. Due to the lack of systematicity in this search, it is of considerable interest to investigate how tractable constraint types may be combined in order to yield more general problems which are still tractable. Cohen *et al.* [CJJK01] have studied so-called 'disjunctive constraints', i.e. constraints which have the form of the disjunction of two constraints of specified types. They identified certain properties which allow for new tractable constraint classes to be constructed from existing classes. Several important classes of tractable constraints can be obtained by their method such as the Horn and Krom fragments of propositional logic, the ORD-Horn class [NB95] and the classes of max-closed and connected row-convex constraints [JC95, DBvH99].

The investigation of disjunctive constraints was continued in Broxvall & Jonsson [BJ00, Paper I] where all tractable disjunctive classes for reasoning about partially and totally ordered time were identified. Somewhat surprisingly, all of these tractable classes can be obtained by using 1-independence. This observation raised the question whether tractable disjunctive constraints can be completely characterised by these kind of properties. We partially answer this question in this paper.

We consider three different properties, known as the guaranteed satisfaction (GS) property, 1-independence and 2-independence [CJJK01]. Let Γ and Δ be two sets of relations such that the CSP problem over $\Gamma \cup \Delta$ is tractable. In short, we prove the following:

- Let the set Δ^* contain all possible finite disjunctive relations over Δ. The CSP problem for this set is tractable if and only if Δ has the GS property.

- Let the set $\Gamma^{\veebar}\Delta^*$ contain all disjunctive relations over $\Gamma \cup \Delta$ where relations in Γ are allowed to appear at most once in a disjunction (compare with the Horn fragment of propositional logic). The CSP problem for this set is tractable if and only if Δ is 1-independent of Γ.
- Consider the set $\Gamma \cup \Delta^2$ where Δ^2 contains all disjunctive relations over Δ containing at most two disjuncts (compare with the Krom fragment of propositional logic). The CSP problem for this set is tractable if and only if Δ is 2-independent of Γ.

Our results are obtained by using the definition of the disjunction combinator $^{\veebar}$ proposed in paper I [BJ00] instead of the original definition in [CJJK01]. This change makes the result cleaner since we do not have to take care of a number of pathological special cases. This issue is discussed in greater detail in the paper.

These results suggest that automatic methods for checking the GS, 1- and 2-independence properties may be very useful when working with disjunctive constraints. Also, it is hardly surprising that deciding these properties is a highly non-trivial task in many cases. For classes of binary constraints where satisfiability can be decided by checking path-consistency, we present a fairly simple method for verifying the 1-independence property. This method builds on a somewhat surprising connection between 1-independence and *refinements* [Ren99]. Loosely speaking, a refinement is a way of reducing one CSP problem to another and it has the property that if the second problem can be decided by path-consistency, then path-consistency decides the first problem, too. Refinements were successful in proving tractability of large subsets of RCC-8 as well as Allen's Interval Algebra [Ren99]. One important aspect of refinements is that their correctness can be easily checked by a computer-assisted analysis which implies that 1-independence can be automatically checked in many cases. To demonstrate the usefulness of our method, we show that all previously known independence results for the time point algebras for partially and totally ordered time [BJ00, Paper I] can be derived automatically. This raises the question whether our method is complete or not—unfortunately, we are not able to answer this question in its full generality.

The paper is organized as follows: In Section 2 we give an overview of the basic definitions concerning CSPs, disjunctions and refinements. Section 3 contains the main complexity results for combining constraints with disjunctions. In Section 4 we relate refinements and 1-independence and prove the

connection between them. We also exemplify how the method can be used for identifying tractable disjunctive constraints. Finally, the last section contains some discussions and conclusions of the results presented earlier. Some of the results in Section 4 have previously been presented in a conference paper [BJR00].

2 Preliminaries

This section consists of three parts where we define the constraint satisfaction problem, provide some background material concerning disjunctions and describe the refinement method.

2.1 The constraint satisfaction problem

Let $\mathcal{S}$ be a set of relations over some domain D (of *values*) and let V be a set of variables. The relations in $\mathcal{S}$ may be of arbitrary arity and the domain D is not necessarily finite. Let $R \in \mathcal{S}$ be a relation of arity a and $x \in V^a$ (where V^a denotes the a-fold cartesian product of V). We write $R(x)$ (a *constraint*) to denote that the variables in x are related by R. This definition allows the use of repeated variables in the scope of a constraint, e.g. $R(x, y, x)$. For any constraint $c = R(x)$, let $\mathrm{Rel}(c) = R$. The consistency problem $\textsc{CspSat}(\mathcal{S})$ is defined as follows:

Instance: A tuple (V, C) where V is a set of variables and C is a finite set of constraints over V, where for each $c \in C$, $\mathrm{Rel}(c) \in \mathcal{S}$.

Question: Is there a *satisfying instantiation* of the variables, i.e. a total function $f : V \to D$ such that for all $R(x_1, \ldots, x_a) \in C$ holds that: $(f(x_1), \ldots, f(x_a)) \in R$?

Given an instance Θ of $\textsc{CspSat}(\mathcal{S})$, let $\mathrm{Mods}(\Theta)$ denote the class of models of Θ (i.e. the satisfying instantiations) and $\mathrm{Vars}(\Theta)$ the variables appearing in Θ. Let $\bot$ denote the empty relation (of arbitrary arity).

2.2 Basics of disjunctions

We begin by introducing operators for combining relations with disjunctions.

Definition 1 Let R_1, R_2 be relations of arity i, j and define the disjunction $R_1 \vee R_2$ of arity $i + j$ as follows:

$$\begin{aligned} R_1 \vee R_2 \;\; = \;\; & \{(x_1, \ldots, x_{i+j}) \in D^{i+j} | (x_1, \ldots, x_i) \in R_1 \vee \\ & (x_{i+1}, \ldots, x_{i+j}) \in R_2\} \end{aligned}$$

Thus, the disjunction of two relations with arity i, j is the relation with arity $i + j$ satisfying either of the two relations.

To give a concrete example, let $D = \{0, 1\}$ and let the relations $\mathsf{and} = \{\langle 1, 1\rangle\}$ and $\mathsf{xor} = \{\langle 0, 1\rangle, \langle 1, 0\rangle\}$ be given. The disjunction of and and xor is given by:

$$\mathsf{and} \vee \mathsf{xor} = \left\{ \begin{array}{l} \langle 0,0,0,1\rangle, \langle 0,1,0,1\rangle, \langle 1,0,0,1\rangle, \langle 1,1,0,1\rangle, \\ \langle 0,0,1,0\rangle, \langle 0,1,1,0\rangle, \langle 1,0,1,0\rangle, \langle 1,1,1,0\rangle, \\ \langle 1,1,0,0\rangle, \langle 1,1,0,1\rangle, \langle 1,1,1,0\rangle, \langle 1,1,1,1\rangle \end{array} \right\}$$

We see that the constraint $(\mathsf{and} \vee \mathsf{xor})(x, y, x, z)$ is satisfiable when x, y and z have, for instance, been instantiated to $1, 0, 0$, respectively.

The definition of disjunction can easily be extended to sets of relations.

Definition 2 Let Γ_1, Γ_2 be sets of relations and define the disjunction $\Gamma_1 {}^{\times}\!\!\vee \Gamma_2$ as follows:

$$\Gamma_1 {}^{\times}\!\!\vee \Gamma_2 = \Gamma_1 \cup \Gamma_2 \cup \{R_1 \vee R_2 \mid R_1 \in \Gamma_1, R_2 \in \Gamma_2\}$$

The disjunction of two sets of relations $\Gamma_1 {}^{\times}\!\!\vee \Gamma_2$ is the set of disjunctions of each pair of relations in Γ_1, Γ_2 plus the sets Γ_1, Γ_2. It seems sensible to include Γ_1 and Γ_2 since one wants to have the choice of using the disjunction or not. Thus, our definition of ${}^{\times}\!\!\vee$ differs slightly from the definition given by Cohen *et al.* [CJJK01]; they define $\Gamma_1 {}^{\times}\!\!\vee \Gamma_2$ as $\{R_1 \vee R_2 \mid R_1 \in \Gamma_1, R_2 \in \Gamma_2\}$. The two definitions coincide if $\bot$ is included in both Γ_1 and Γ_2. Otherwise, the definitions are different and the implications of this are pointed out in Subsection 3.1. We will tacitly assume that $\bot$ is not a member of any set of relations that we consider. Note that CspSat(Γ) has the same complexity as CspSat($\Gamma \cup \{\bot\}$) up to polynomial-time reductions.

In many cases we shall be concerned with constraints that are specified by disjunctions of an arbitrary number of relations. Thus, we make the following definition: for any set of relations, Δ, define $\Delta^* = \bigcup_{i=0}^{\infty} \Delta^i$ where $\Delta^0 = \Delta$ and $\Delta^{i+1} = \Delta^i {}^{\times}\!\!\vee \Delta$.

For proving tractability of disjunctive constraints, a number of properties have been proposed in [CJJK01]:

Definition 3 Let Δ be a set of relations. If every instance $\textsc{CspSat}(\Delta)$ is satisfiable, then we say that Δ has the *guaranteed satisfaction* (GS) property.

$\textsc{CspSat}(\Delta^*)$ is clearly tractable if Δ has the GS property.

Definition 4 For any sets of relations Γ and Δ, define $\textsc{CspSat}_{\Delta\leq k}(\Gamma\cup\Delta)$ to be the subproblem of $\textsc{CspSat}(\Gamma\cup\Delta)$ consisting of all instances containing at most k constraints over the relations in Δ. We say that Δ is *k-independent* of Γ if the following condition holds: any set of constraints C in $\textsc{CspSat}(\Gamma\cup\Delta)$ has a solution provided every subset of C belonging to $\textsc{CspSat}_{\Delta\leq k}(\Gamma\cup\Delta)$ has a solution.

It is easy to see that if Δ is k-independent of Γ, then Δ is $k+1$-independent of Γ, too. The following result by Cohen *et al.* [CJJK01] demonstrates the usefulness of the independence property.

Theorem 5 Let Γ and Δ be sets of relations such that $\textsc{CspSat}(\Gamma\cup\Delta)$ is tractable. If Δ is 1-independent of Γ, then $\textsc{CspSat}(\Gamma^{\times}_{\vee}\Delta^*)$ is tractable. If Γ is 2-independent of $\emptyset$, then $\textsc{CspSat}(\Gamma^{\times}_{\vee}\Gamma)$ is tractable.

The notion of 1-independence can alternatively (but equivalently) be defined as follows: Let $C = \{c_1, \ldots, c_k\}$ and $D = \{d_1, \ldots, d_n\}$ be arbitrary finite sets of constraints over Γ and Δ, respectively. Then, Δ is 1-independent of Γ iff for every possible choice of C and D, the following holds: if $C \cup \{d_i\}$, $1 \leq i \leq n$, is satisfiable, then $C \cup D$ is satisfiable. Also note that Γ is 2-independent of $\emptyset$ if and only if for every constraint problem C over Γ having no solution, there exists a pair of constraints $c_i, c_j \in C$ such that $\{c_i, c_j\}$ has no solution.

2.3 Basics of the refinement method

We review the *refinement method* as introduced by Renz [Ren99] in this subsection. For proofs and additional results, see [Ren99] or its forthcoming journal version for more details [Ren].

So far the refinement method has been introduced for binary CSPs only. So, although we deal with n-ary constraints in this paper, the parts dealing with refinements apply only to binary constraints.

Let $\mathcal{A}$ be a finite set of jointly exhaustive and pairwise disjoint binary relations, also called *basic* relations, and $\mathcal{S} \subseteq 2^{\mathcal{A}}$. We denote the standard

operations composition, intersection and converse by $\circ$, $\cap$ and $\cdot^{-1}$, respectively. Furthermore, we define the unary operation $\neg$ such that $\neg R = \mathcal{A} \backslash R$ for all relations $R \subseteq \mathcal{A}$ and let eq denote the binary equality relation.

A set of constraints is path-consistent if for any consistent assignment of two variables, there exists an assignment for every third variable such that the three assignments taken together are consistent. Path-consistency can be enforced by iteratively applying the following operation to every pair of variables x_i, x_j, until a fixed point is reached (R_{ij} specifies the relation between x_i and x_j):

$$\forall k : R_{ij} := R_{ij} \cap (R_{ik} \circ R_{kj}).$$

If the empty relation occurs during this process, the set is inconsistent, otherwise the resulting set is path-consistent.

A *refinement* of a constraint xRy is a constraint $xR'y$ such that $R' \subseteq R$. A refinement of a set of constraints Θ is a set of constraints Θ' such that every constraint of Θ' is a refinement of a constraint of Θ. It is clear that if Θ' has a solution, then Θ also has a solution.

In order to handle different refinements, a *refinement matrix* is used that contains for every relation $S \in \mathcal{S}$ all specified refinements.

Definition 6 A *refinement matrix* M of $\mathcal{S}$ has $|\mathcal{S}| \times 2^{|\mathcal{A}|}$ boolean entries such that for $S \in \mathcal{S}$, $R \in 2^{\mathcal{A}}$, $M[S][R] = true$ only if $R \subseteq S$, i.e. R is a refinement of S.

Definition 7 Let $\Delta \subseteq \mathcal{S}$. M^{Δ} is the Δ-*refinement matrix* of $\mathcal{S}$ if for every $S \in \mathcal{S}$, $M^{\Delta}[S][S'] = \mathsf{true}$ iff

1. there exists a relation $R \in \Delta$ such that $S' = S \cap R$ and $S' \neq \emptyset$; or
2. $S' = S$.

The basic idea of the refinement method [Ren99] is to exploit that the path-consistency algorithm only looks at triples of constraints and that refinements of constraints are passed from triple to triple. Thus, the possible number of different triples over a set of relations $\mathcal{S}$ as well as the number of refinements of these triples is limited, although there is an infinite number of different sets of constraints Θ over $\mathcal{S}$. Therefore, it is possible to extract properties of a set of relations $\mathcal{S}$ by just analyzing a limited number of triples of constraints

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 a path-consistent triple of constraints over $\mathcal{S}$ inconsistent; succeed otherwise.

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

Figure 1: Algorithm CHECK-REFINEMENTS [Ren99]

over $\mathcal{S}$. This is done by the algorithm CHECK-REFINEMENTS (see Figure 1) which takes as input a set of relations $\mathcal{S}$ and a refinement matrix M of $\mathcal{S}$ and either succeeds or fails. A *triple* (R, S, T) of relations denotes the following CSP problem on three variables: $\{xRz, xSy, yTz\}$. Since $\mathcal{A}$ is a finite set of relations, M can be changed only a finite number of times, so the algorithm always terminates.

If CHECK-REFINEMENTS$(\mathcal{S}, M)$ returns succeed, we have checked *all* possible refinements of *every* path-consistent triple of variables as given by the refinement matrix M. Thus, applying these refinements to a path-consistent set of constraints can *never* result in an inconsistency when enforcing path-consistency. This is stated in the following theorem.

Theorem 8 (Renz [Ren99]) Let $\mathcal{S}$ be a set of relations that can be decided by path-consistency, M a refinement matrix of $\mathcal{S}$ and assume that CHECK-REFINEMENTS$(\mathcal{S}, M)$ returns succeed. For every path-consistent set Θ of constraints over $\mathcal{S}$, the following holds: for every refinement Θ' of Θ such that $x_i R' x_j \in \Theta'$ only if $x_i R x_j \in \Theta$ and $M[R][R'] = \mathsf{true}$, Θ' has a solution.

The refinement method, thus, simply consists of running the algorithm CHECK-REFINEMENTS on a set of relations $\mathcal{S}$ and a refinement matrix M. We say that $\mathcal{S}$ can be *refined* by M, if CHECK-REFINEMENTS$(\mathcal{S}, M)$ returns succeed.

Renz [Ren99] used the refinement method in a different way, namely, for showing that path-consistency decides a set of relations $\mathcal{S}$: Assume that path-consistency decides consistency for a set of relations $\mathcal{T}$. If CHECK-REFINEMENTS$(\mathcal{S}, M)$ returns succeed and if the resulting refinement matrix M' contains for each relation $S \in \mathcal{S}$ a relation $T_S \in T$, i.e. $M'[S][T_S] = true$, then path-consistency decides consistency of $\mathcal{S}$. It turned out that by using the refinement matrix $M^{\neq}$ it was possible to prove tractability for all maximal tractable subsets of RCC-8 and the Interval Algebra which contain all basic relations.

3 Tractable Disjunctions

We shall now show the close connections between tractable disjunctive constraints and the GS/independence properties. Our main results are the following: Let Γ and Δ be two sets of relations such that CSPSAT$(\Gamma \cup \Delta)$ is tractable. Then,

(1) CSPSAT(Δ^*) is tractable iff Δ has the GS property;

(2) CSPSAT$(\Gamma^{\times}\!\!\vee\Delta^*)$ is tractable iff Δ is 1-independent of Γ; and

(3) CSPSAT$(\Gamma \cup \Delta^2)$ is tractable iff Δ is 2-independent of Γ.

If these conditions are not met, then CSPSAT(Δ^*), CSPSAT$(\Gamma \vee \Delta^*)$ and/or CSPSAT$(\Gamma \cup \Delta^2)$ are NP-complete. The proofs of (1)–(3) can be found in Subsections 3.1–3.3, respectively. An interesting question is whether (3) can be strengthened to ensure tractability of CSPSAT$(\Gamma^{\times}\!\!\vee\Delta)$. We demonstrate that this does not hold in general at the end of Subsection 3.3.

The NP-completeness results are based on reductions from the following two NP-complete problems:

3-SAT
INSTANCE: Set U of variables, collection C of clauses over U such that each clause $c \in C$ has $|c| = 3$.
QUESTION: Is there a satisfying truth assignment for C?

3-COLOURABILITY
INSTANCE: Undirected graph $G = (V, E)$.
QUESTION: Does there exist a function $f : V \to \{0, 1, 2\}$ such that $f(u) \neq f(v)$ whenever $\{u, v\} \in E$?

Before we proceed, we need to prove that the problems we consider are members of NP.

Lemma 9 Assume that $\mathcal{S}$ is a tractable set of relations. For any set $\mathcal{S}'$ of relations constructed using $^{\times}_{\vee}$ and the relations in $\mathcal{S}$, CSPSAT($\mathcal{S}'$) is in NP.

Proof: Non-deterministically choose one atomic constraint from every disjunctive constraint (we assume, without loss of generality, that there exist polynomial-time computable decomposition operators for the disjunctive constraints) and show that the resulting set of constraints is satisfiable. Since CSPSAT($\mathcal{S}$) is tractable[1], CSPSAT($\mathcal{S}'$) is in NP. □

3.1 The guaranteed satisfaction property

We begin by studying the (admittedly trivial) GS property. The proof idea will, however, turn out to be very useful for proving results about the independence properties.

Theorem 10 The following statements are equivalent:

1. Δ has the GS property;
2. Δ is 1-independent of $\emptyset$;

[1]It is actually sufficient that CSPSAT($\mathcal{S}$) is in NP.

3. CspSat(Δ^*) is tractable;

4. CspSat(Δ^3) is tractable;

Otherwise, CspSat(Δ^3) and CspSat(Δ^*) are NP-complete.

Proof: We show that (1) $\Rightarrow$ (3) $\Rightarrow$ (4) $\Rightarrow$ (1) and (1) $\Leftrightarrow$ (2).

The implication (1) $\Rightarrow$ (3) is trivial and (3) $\Rightarrow$ (4) follows from the fact that $\Delta^3 \subseteq \Delta^*$. To show that (4) $\Rightarrow$ (1), we assume the opposite, i.e. CspSat(Δ^3) is tractable but Δ does not have the GS property. This implies that there exists a set of constraints $H = \{h_1, \ldots, h_n\}$ over Δ such that H is not satisfiable. Note that $|H| > 1$ since we do not allow the relation $\bot$. We choose H to be minimal; i.e. $|H|$ is as small as possible. This implies that every strict subset $H' \subset H$ is satisfiable. Finally, consider the set $\mathcal{H} = \{h_1 \vee h_2\} \cup (H - \{h_1, h_2\})$ and note that in any model of $\mathcal{H}$, either h_1 or h_2 holds, but not both.

In order to prove NP-hardness, we show that 3-Sat can be transformed to CspSat(Δ^3) in polynomial time; membership in NP follows from Lemma 9. Arbitrarily choose a 3-Sat formula $F = c_1 \wedge \ldots \wedge c_n$ over the variables $p_1, \ldots, p_m$. We incrementally construct an instance of CspSat(Δ^3) that is satisfiable iff F is satisfiable.

For each variable p_i, introduce a fresh copy of the set $\mathcal{H}$ (i.e. the copies of $\mathcal{H}$ are over disjoint sets of variables) where we denote the 'important' relations h_1 and h_2 as h_t^i and h_f^i, respectively. As we noted earlier, this will force either h_t^i or h_f^i to hold in any model but not both. We interpret h_t^i as 'p_i is true' and h_f^i as 'p_i is false'.

For each clause c_i, it is now easy to add a disjunction corresponding to the clause: for instance, $(p_i \vee \neg p_j \vee p_k)$ is translated to $h_t^i \vee h_f^j \vee h_t^k$. Obviously, the resulting set of constraints (which trivially can be computed in polynomial time) is an instance of CspSat(Δ^3) and is satisfiable iff F is satisfiable.

Finally, we show that (1) $\Leftrightarrow$ (2). The only-if direction is obvious so we prove the other direction. Assume to the contrary that there exists a set of constraints H over Δ such that H is not satisfiable. Since Δ is 1-independent of $\emptyset$, this implies that there must be a single constraint $h \in H$ that is not satisfiable—in other words, Rel(h) is the empty relation and we have a contradiction. $\square$

This result does not hold if the original definition of $\overset{\times}{\vee}$ [CJJK01] is used (see Section 2 for the exact definition). Assume that Δ has the GS property.

Then, $\bot \notin \Delta$. Assume furthermore that Γ is an arbitrary set of relations (we do not even require that CspSat(Γ) is tractable). Then, $\Gamma^{\times}_{\vee}\Delta$ is tractable! This follows from the fact that $\Gamma \not\subseteq \Gamma^{\times}_{\vee}\Delta$; every possible member of $\Gamma^{\times}_{\vee}\Delta$ is either of the form $R(x_1, \ldots, x_{arity(R)})$ where $R \in \Delta$ or $R(x_1, \ldots, x_{arity(R)}) \vee S(y_1, \ldots, y_{arity(S)})$ where $R \in \Delta$ and $S \in \Gamma$. Hence, the GS property ensures that every instance of CspSat($\Gamma^{\times}_{\vee}\Delta$) is satisfiable. It seems counter-intutitive that $\Gamma \cup \Delta$ can be a computationally harder problem than $\Gamma^{\times}_{\vee}\Delta$ which explains why we have modified the definition of $^{\times}_{\vee}$.

There are also technical reasons for defining $^{\times}_{\vee}$ the way we have done. For instance, the result in the next section would be very different otherwise. It simply states that CspSat($\Gamma^{\times}_{\vee}\Delta^*$) is tractable iff CspSat($\Gamma \cup \Delta$) is tractable and Δ is 1-independent of Γ. With the original definition of $^{\times}_{\vee}$, we would need to take care of several cases; one of them is that CspSat($\Gamma^{\times}_{\vee}\Delta^*$) is tractable if Δ has the GS property but Δ is not 1-independent of Γ (which once again is a 'strange' case where CspSat($\Gamma \cup \Delta$) may be computationally harder than CspSat($\Gamma^{\times}_{\vee}\Delta^*$).

3.2 1-Independence

The proof presented here is a slight variation of the proof of Theorem 10 so we only sketch the proof.

Theorem 11 The following statements are equivalent:

1. Δ is 1-independent of Γ;
2. CspSat($\Gamma^{\times}_{\vee}\Delta^*$) is tractable;
3. CspSat($\Gamma \cup \Delta^3$) is tractable;

Otherwise, CspSat($\Gamma \cup \Delta^3$) and CspSat($\Gamma^{\times}_{\vee}\Delta^*$) are NP-complete.

Proof: The implications (1) $\Rightarrow$ (2) follows from Theorem 5 and (2) $\Rightarrow$ (3) is trivial since $\Gamma \cup \Delta^3 \subseteq \Gamma^{\times}_{\vee}\Delta^*$. To show that (3) $\Rightarrow$ (1), we assume the opposite, i.e. CspSat($\Gamma \cup \Delta^3$) is tractable but Δ is not 1-independent of Γ. This implies that there exists a set of constraints X over Γ and a set $H = \{h_1, \ldots, h_n\}$ over Δ such that $X \cup \{h_i\}$ is satisfiable for every $1 \leq i \leq n$ but $X \cup H$ is not satisfiable. Choose X and H such that $|H|$ is as small as possible and note that $|H| \geq 2$. The existence of a set $H' \subset H$ such that $X \cup H'$

is not satisfiable contradicts the minimality of H so $X \cup H'$ is satisfiable for all $H' \subset H$. Finally, consider the set $\mathcal{X} = X \cup \{h_1 \vee h_2\} \cup (H - \{h_1, h_2\})$ and note that in any model of $\mathcal{X}$, either h_1 or h_2 holds, but not both. The result can now easily be shown by a reduction from 3-SAT that is analogous to the reduction employed in the proof of Theorem 10. □

By combining Theorems 10 and 11, we see that whenever Δ is 1-independent of Γ, Δ must have the GS property—this observation can significantly simplify the search for sets of 1-independent relations.

3.3 2-Independence

The proof of this case consists of two parts; the first part strengthens a tractability result by Cohen *et al.* [CJJK01] while the second part is a hardness result in the style of Theorems 10 and 11. The reduction is quite different, though, and is based on 3-COLOURABILITY instead of 3-SAT. In the end of this subsection (Theorem 14), we complement this positive result with a negative result showing that 2-independence is not sufficient for ensuring tractability of $\text{CSPSAT}(\Gamma^{\times}_{\vee}\Delta)$.

Theorem 12 $\text{CSPSAT}(\Gamma \cup \Delta^2)$ is tractable iff Δ is 2-independent of Γ. Otherwise, $\text{CSPSAT}(\Gamma \cup \Delta^2)$ is NP-complete.

Cohen *et al.* [CJJK01] have shown that $\text{CSPSAT}(\Delta^2)$ is tractable if Δ is 2-independent of $\emptyset$; i.e. an instance I of $\text{CSPSAT}(\Delta^2)$ has a solution if every $I' \subseteq I$ such that $|I'| = 2$ has a solution. We begin by generalising this result.

Lemma 13 If Δ is 2-independent of Γ, then $\text{CSPSAT}(\Gamma \cup \Delta^2)$ is tractable.

Proof: We show that the algorithm 2IND-SOLVABLE defined in Figure 2 succeeds when applied to C if and only if C has a solution.
only-if: Assume that 2IND-SOLVABLE returns succeed. This implies that there exists a satisfying truth assignment, μ, for $A \cup A' \cup A''$. Define the set of constraints C' as follows:

$$C' = \{c \mid \mu(q_c) = \mathsf{true}\}.$$

We first show that C' has a solution. If C' has no solution, there exist $c_1, \ldots, c_k \in C'$ such that $\text{Rel}(c_1), \ldots, \text{Rel}(c_k) \in \Delta$ and $Q_\Gamma \cup \{c_1, \ldots, c_k\}$ is

not satisfiable. We know that Q_Γ has a solution since the algorithm did not fail in line 4. Hence, the fact that μ satisfies A and A'' implies that $Q_\Gamma \cup \{c_i, c_j\}$ is satisfiable for $1 \leq i, j \leq k$ so $Q_\Gamma \cup \{c_1, \ldots, c_k\}$ is satisfiable since Δ is 2-independent of Γ. So C' does indeed have a solution.

Now, let f be a model of C'. For each disjunctive constraint in C we know that at least one of its disjuncts is a member of C', because μ satisfies the formulae in A'. We also know that every non-disjunctive constraint is a member of C' since μ satisfies the formulae in A''. Taken together, this means that C has a model.

if: Assume that C has a model f. Define the truth assignment $\mu : \{q_c \mid c \in P \cup Q_\Gamma \cup Q_\Delta\} \rightarrow \{\mathsf{true}, \mathsf{false}\}$ as follows:

$$\mu(q_c) = \mathsf{true} \text{ iff } c \text{ is satisfied by } f.$$

We show that μ is a satisfying truth assignment of $A \cup A' \cup A''$ by considering the elements of A, A' and A'' in turn.

(1) For each formula $(\neg q_{c'} \vee \neg q_{c''}) \in A$, we know that $Q_\Gamma \cup \{c', c''\}$ has no model. Hence, it cannot be the case that $\mu(c') = \mu(c'') = \mathsf{true}$, which means that $(\neg q_{c'} \vee \neg q_{c''})$ is satisfied by μ.

(2) For each formula $(q_{c'} \vee q_{c''}) \in A'$ we know that there is a constraint $c \in C$ of the form $c = c' \vee c''$. Since f is a model of C, f satisfies at least one of c' and c'' which means that $(q_{c'} \vee q_{c''})$ is satisfied by μ.

(3) For each formula $q_c \in A''$ there exists a constraint $c \in C$ that is not a disjunction. Consequently, f must satisfy c and μ satisfies q_c.

Finally, we have to show that the algorithm 2IND-SOLVABLE runs in polynomial time. This follows directly from the observation that line 5 can be computed in polynomial time (since CSPSAT($\Gamma \cup \Delta$) is tractable) and that the test in line 8 can be performed in polynomial time by using some tractable algorithm for showing the satisfiability of 2CNF formulae (such as the algorithm by Aspvall *et al.* [APT79]). □

Proof: (of Theorem 12) The if direction follows from Lemma 13. To show the other direction we assume to the contrary that CSPSAT($\Gamma \cup \Delta^2$) is tractable but Δ is not 2-independent of Γ.

This implies that there exists a set of constraints X over Γ and a set $H = \{h_1, \ldots, h_n\}$ over Δ such that $X \cup \{h_i, h_j\}$ is satisfiable for every $1 \leq i, j \leq n$ but $X \cup H$ is not satisfiable. Choose X and H such that $|H|$ is as small as possible and note that $|H| \geq 3$. The existence of a set $H' \subset H$ such that $X \cup H'$ is not satisfiable contradicts the minimality of H so $X \cup H'$ is satisfiable for all $H' \subset H$. Thus, we can define the satisfiable set $\mathcal{X} = X \cup \{h_1 \vee h_2, h_1 \vee h_3, h_2 \vee h_3, h_4, \ldots, h_n\}$ which has the following property: In every model of $\mathcal{X}$, exactly one of h_1, h_2, h_3 is not satisfied.

To prove the result, we show that 3-Colourability can be transformed to CspSat($\Gamma \cup \Delta^2$) in polynomial time. Arbitrarily choose an undirected graph $G = (V, E)$ such that $V = \{v_1, \ldots, v_k\}$. We will construct an instance of CspSat($\Gamma \cup \Delta^2$) that is satisfiable iff G is colourable with three colours.

For each vertex v_i, introduce a fresh copy of the set $\mathcal{X}$ where we denote the constraints h_1, h_2, h_3 as h_1^i, h_2^i, h_3^i, respectively. As we have already noted, this will force exactly one of h_1^i, h_2^i, h_3^i not to hold in every model. We interpret this as meaning that h_j^i does not hold as 'vertex v_i has colour j'.

For each edge $(v_i, v_j) \in E$, we add the disjunctions $h_1^i \vee h_1^j$, $h_2^i \vee h_2^j$ and $h_3^i \vee h_3^j$ which ensures that v_i and v_j are not assigned the same colour. The resulting set of constraints can be computed in polynomial time, it is an instance of CspSat($\Gamma \cup \Delta^2$) and is satisfiable iff G is 3-colourable which concludes the proof. □

Theorem 14 There exist sets of unary relations Γ, Δ such that CspSat($\Gamma \cup \Delta$) is tractable, Δ is 2-independent of Γ but CspSat($\Gamma^{\times}_{\vee}\Delta$) is NP-complete.

Proof: Consider the domain $D = \{0, 1, 2\}$. Define unary relations $\mathsf{neq}_i \subseteq D$, $0 \leq i \leq 2$, such that $\mathsf{neq}_i(x)$ holds iff $i \neq x$ and define $\mathsf{eq}_i \subseteq D$, $0 \leq i \leq 1$, such that $\mathsf{eq}_i(x)$ holds iff $i = x$. Let $\Gamma = \{\mathsf{neq}_0, \mathsf{neq}_1, \mathsf{neq}_2\}$ and $\Delta = \{\mathsf{eq}_0, \mathsf{eq}_1\}$. Proving the tractability of CspSat($\Gamma \cup \Delta$) and that Δ is 2-independent of Γ are routine verifications.

We show that CspSat($\Gamma^{\times}_{\vee}\Delta$) is NP-complete by a polynomial-time reduction from 3-Colourability. Let $G = (V, E)$ be an arbitrary undirected graph. We will construct an instance X of CspSat($\Gamma^{\times}_{\vee}\Delta$) that is satisfiable iff G can be 3-coloured.

Assume $V = \{v_1, \ldots, v_m\}$. To simplify our description of the reduction, we will only consider edges $e = (v_i, v_j)$ in E such that $i < j$. Obviously, we can do this without loss of generality since G is undirected. For each vertex

Algorithm: 2IND-SOLVABLE
Input: A finite set C of constraints over $\Gamma \cup \Delta^2$.
Output: succeed if C is satisfiable; fail otherwise.

1. $P \leftarrow \{c_1, c_2 \mid c \in C \text{ and } c = c_1 \vee c_2\}$
2. $Q_\Delta \leftarrow \{c \in C \mid c \text{ is not a disjunction and } \mathrm{Rel}(c) \in \Delta\}$
3. $Q_\Gamma \leftarrow \{c \in C \mid c \text{ is not a disjunction and } \mathrm{Rel}(c) \in \Gamma \setminus \Delta\}$
4. if Q_Γ has no solution then return fail
5. define a set of boolean variables $\{q_c \mid c \in P \cup Q_\Gamma \cup Q_\Delta\}$
6. $A \leftarrow \{(\neg q_{c'} \vee \neg q_{c''}) \mid c', c'' \in P \cup Q_\Delta \text{ and } Q_\Gamma \cup \{c', c''\} \text{ not satisfiable}\}$
7. $A' \leftarrow \{(q_{c'} \vee q_{c''}) \mid \exists c \in C \text{ such that } c = c' \vee c''\}$
8. $A'' \leftarrow \{q_c \mid c \in Q_\Gamma \cup Q_\Delta\}$
9. if $A \cup A' \cup A''$ is satisfiable
   then return succeed
   else return fail

Figure 2: Algorithm 2IND-SOLVABLE

$v \in V$, introduce a variable $\hat{v}$ and for each edge $(v, w) \in E$, introduce three variables e^i_{vw}, $0 \leq i \leq 2$. Finally, for each edge $(v, w) \in E$, add the following six constraints to X:

$$(1)\ \mathsf{neq}_0(\hat{v}) \vee \mathsf{eq}_0(e^0_{vw}) \quad (2)\ \mathsf{neq}_0(\hat{w}) \vee \mathsf{eq}_1(e^0_{vw})$$
$$(3)\ \mathsf{neq}_1(\hat{v}) \vee \mathsf{eq}_0(e^1_{vw}) \quad (4)\ \mathsf{neq}_1(\hat{w}) \vee \mathsf{eq}_1(e^1_{vw})$$
$$(5)\ \mathsf{neq}_2(\hat{v}) \vee \mathsf{eq}_0(e^2_{vw}) \quad (6)\ \mathsf{neq}_2(\hat{w}) \vee \mathsf{eq}_1(e^2_{vw})$$

The value of variables $\hat{v}$ will equal the colour of the corresponding vertex and variable e^i_{vw} is to be interpreted as follows: if $e^i_{vw} = 0$, then variable $\hat{w}$ does not have the value i; otherwise, $\hat{w}$ equals i. Now, consider constraint (1). It tells us that either $\hat{v}$ is not equal to 0 *or* the variable $\hat{w}$ is not equal to 0. Hence, the constraints (1), (3) and (5) ensure that adjacent vertices are not assigned the same colour. For this to work, it must also be true that a variable $\hat{w}$ cannot have a value i and at the same time $e^i_{vw} = 0$. This is guaranteed by constraints (2), (4) and (6).

We can now show that X is satisfiable iff G is 3-colourable.

only-if: Let M be a model of X. We show that $M(\hat{v}) \neq M(\hat{w})$ whenever there is an edge between v and w in G. Since the range of M is $\{0,1,2\}$, M can easily be modified into a three-colouring of G.

Assume to the contrary that X has a model M such that $M(\hat{v}) = M(\hat{w}) = 0$ (the other two cases are analogous) and $(v,w) \in E$. Constraints (1) and (2) implies that both $\mathsf{eq}_0(e^0_{vw})$ and $\mathsf{eq}_1(e^0_{vw})$ hold which leads to a contradiction.

if: Let $f : V \to \{0,1,2\}$ be a 3-colouring of G. Construct a model M of X as follows:

$$M(\hat{v}) = f(v);$$

$$M(e^i_{vw}) = 0 \text{ if } f(w) \neq i$$

$$M(e^i_{vw}) = 1 \text{ if } f(w) = i$$

To see that M is a model of X, arbitrarily choose a constraint c in X. Assume first that c is of the form (1) $\mathsf{neq}_0(\hat{v}) \vee \mathsf{eq}_0(e^0_{vw})$. This constraint is not satisfied iff $M(\hat{v}) = 0$ and $M(e^0_{vw}) = 1$. By the construction of M, it follows that $f(v) = 0$ and $f(w) = 0$ which contradicts the fact that f is a 3-colouring of G.

Assume c is on the form (2) $\mathsf{neq}_0(\hat{w}) \vee \mathsf{eq}_1(e^0_{vw})$ instead. This constraint is not satisfied iff $M(\hat{w}) = 0$ and $M(e^0_{vw}) = 0$. By the construction of M, it follows that $f(w) = 0$ and $f(w) \neq 0$ at the same time. □

4 1-Independence and Refinements

In the previous section we have shown that the 1-independence property is a necessary and sufficient condition for tractability of a natural class of disjunctive constraints. However, it is often quite difficult to prove that this property holds for a certain class, and this has to be proven for each class anew. Recently, Renz [Ren99] proposed a general method for proving tractability of classes of relations which is comprised by running a simple algorithm. This refinement method, which is described in Section 2.3, seems to be related to the 1-independence property in the following (simplified) way:

The 1-independence property specifies when a constraint can be added to a set of constraints without changing consistency, while by the refinement method it can be shown if a relation can be removed from a disjunctive constraint without changing consistency. Actually, removing a relation R from a disjunctive constraint xSy is the same as adding the constraint $x\neg Ry$. In Subsection 4.1, we try to elaborate this similarity and show under which conditions the 1-independence property corresponds to the refinement method and vice versa. Some successful examples for using the refinement method for proving 1-independence property are presented in Subsection 4.2. We stress once again that the results in this section are only applicable when considering binary relations.

4.1 Connections between 1-Independence and Refinements

We will now show how the refinement method can be used for proving 1-independence. Let $\mathcal{A}$ be a set of basic relations and choose $\mathcal{S} \subseteq 2^{\mathcal{A}}$ such that $\mathcal{S}$ can be decided by path-consistency. Let Δ be a subset of $\mathcal{S}$. We make the following additional assumptions about $\mathcal{S}$ and Δ:

1. $\mathsf{eq} \in \mathcal{S}$;
2. Δ is closed under intersection.

These restrictions can be imposed without loss of generality: First note that since CspSat($\mathcal{S}$) is tractable, the problem CspSat($\mathcal{S} \cup \{\mathsf{eq}\}$) is also tractable and can trivially be reduced to the first problem (by contracting any two variables related by eq to a single variable). The fact that Δ can be assumed to be closed under intersection follows from the next lemma.

Lemma 15 Let Γ, Δ be sets of relations such that Δ is 1-independent of Γ, then the closure of Δ under intersection is also 1-independent of Γ.

Proof: Let Θ be a set of constraints over Γ and $H = \{h_1, \ldots, h_n\}$ a set of constraints over $\Delta \cup \{R \cap S\}$ for some $R, S \in \Delta$. Assume that $\Theta \cup \{h_i\}$, $1 \leq i \leq n$ is satisfiable. Construct the set

$$H' = (H - \{(R \cap S)(x) \in H\}) \cup \{R(x), S(x) \mid (R \cap S)(x) \in H\}$$

and note that $\Theta \cup \{h'\}$ is satisfiable for all $h' \in H'$. The constraints in H' are all based on the relations in Δ so $\Theta \cup H'$ is satisfiable by 1-independence. It follows from the construction of H' that $\Theta \cup H$ is also satisfiable and $\Delta \cup \{R \cap S\}$ is 1-independent of Γ. □

From now on, we assume that all relations encountered are members of $\mathcal{S}$. We need a couple of lemmata before we can establish the main result.

Lemma 16 A triple (R, S, T) is satisfiable iff $R \cap (S \circ T) \neq \emptyset$.

Proof: The only-if direction is obvious. We show the other direction by choosing some basic relation $K \in R \cap (S \circ T)$ and arbitrarily picking two values a and c such that aKc. The fact that $K \in S \circ T$ implies that for all possible choices of a and c, there exists a value b such that aSb and bTc. By making the assignments $x = a$, $y = b$ and $z = c$, we have shown that (R, S, T) is satisfiable. □

Lemma 17 Assume that $\mathcal{S}$ can be refined by M^Δ, let R be a relation in $\mathcal{S}$ and $K_1, \ldots, K_n \in \Delta$. If $R \cap K_i \neq \emptyset$, $1 \leq i \leq n$, then $R \cap \bigcap_{i=1}^{n} K_i \neq \emptyset$.

Proof: Induction over n. The lemma obviously holds for $n = 1$ so we assume that it holds for $n = k$, $k \geq 1$. We show that the claim holds for $n = k + 1$. The induction hypothesis tells us that $R \cap \bigcap_{i=1}^{k} K_i \neq \emptyset$ and we know that $\bigcap_{i=1}^{k} K_i \in \Delta$ since Δ is closed under intersection. Consider the triple (R, R, eq) and note that it is path-consistent since $R = R \cap (R \circ \mathsf{eq})$ and $\mathsf{eq} = \mathsf{eq} \cap (R \circ R^{-1})$.

Since $R \cap K_{k+1} \neq \emptyset$, the fact that $\mathcal{S}$ can be refined by M^Δ implies that $(R \cap K_{k+1}, R \cap \bigcap_{i=1}^{k} K_i, \mathsf{eq})$ is satisfiable. By Lemma 16, this is equivalent with $(R \cap K_{k+1}) \cap ((R \cap \bigcap_{i=1}^{k} K_i) \circ \mathsf{eq}) \neq \emptyset$ so $(R \cap K_{k+1}) \cap (R \cap \bigcap_{i=1}^{k} K_i) = (R \cap \bigcap_{i=1}^{k+1} K_i) \neq \emptyset$ which concludes the induction. □

Theorem 18 If $\mathcal{S}$ can be refined by M^Δ, then Δ is 1-independent of $\mathcal{S}$.

Proof: Let Θ be a set of constraints over $\mathcal{S}$ and $H = \{h_1, \ldots, h_n\}$ a set of constraints over Δ. Let Θ' be the set Θ after enforcing path-consistency and assume that $\Theta \cup \{h_i\}$, $1 \leq i \leq n$, has a solution.

Arbitrarily choose i and assume that $h_i = xR'y$. Since $\Theta \cup \{h_i\}$ has a solution, $\Theta' \cup \{h_i\}$ also has a solution and there is a non-empty relation R that relates x and y in Θ'. Note that adding h_i to Θ' is the same thing as refining the relation $xRy \in \Theta'$ to $x(R \cap R')y$. Certainly, $R \cap R' \neq \emptyset$ since $\Theta' \cup \{h_i\}$ would not have a solution otherwise. Consequently, $M^{\Delta}[R][R \cap R'] = \mathsf{true}$ since $R' \in \Delta$ and by Lemma 17, we know that the relation R cannot be refined to the empty relation by adding more constraints from H. Thus, adding the constraints in H to Θ are all refinements according to M^{Δ}.

Since Check-Refinements$(\mathcal{S}, M^{\Delta})$ succeeds, Theorem 8 tells us that such refinements can be made without making Θ' inconsistent, i.e. $\Theta' \cup H$ has a solution which trivially implies that $\Theta \cup H$ has a solution. We have thus shown that Δ is 1-independent of $\mathcal{S}$ since Θ and H were arbitrarily chosen. □

This theorem gives us the possibility to prove 1-independence of Δ with respect to $\mathcal{S}$ automatically by simply running Check-Refinements$(\mathcal{S}, M^{\Delta})$. If the algorithm returns `succeed`, we know that Δ is independent of $\mathcal{S}$. In order to make use of a negative answer of the algorithm, we also have to prove the opposite direction, i.e. independence of Δ with respect to a set $\mathcal{S}$ implies that Check-Refinements$(\mathcal{S}, M^{\Delta})$ returns `succeed`. Although this is a highly desirable property, we have not been able to prove this nor did we find a counterexample. There are, however, many examples for which this conjecture holds. As we will see in Subsection 4.2, this includes all 1-independence results for the point algebras for partially and totally ordered time. We give a proof of a slightly limited version of this conjecture.

Definition 19 Let $\mathcal{S} \subseteq 2^{\mathcal{A}}$ and $R \in \mathcal{S}$. We say that *path-consistency makes R explicit* iff for every path-consistent instance Θ of CspSat$(\mathcal{S})$, the following holds: if $M(x)RM(y)$ for every $M \in \mathrm{Mods}(\Theta)$, then $xSy \in \Theta$ and $S \subseteq R$.

Theorem 20 Let $\mathcal{S} \subseteq 2^{\mathcal{A}}$ and assume that Δ is independent of $\mathcal{S}$. Then, Check-Refinements$(\mathcal{S}, M^{\Delta})$ returns `succeed` if and only if path-consistency makes $\neg R$ explicit for every $R \in \Delta$.

Proof: *only-if:* Assume to the contrary that there exists a path-consistent instance Θ of CspSat$(\mathcal{S})$, $x, y \in \mathrm{Vars}(\Theta)$ and relations $R \in \Delta$, $S \in \mathcal{S}$ such that:

1. $xSy \in \Theta$;

2. for all $M \in \text{Mods}(\Theta)$, $M(x)\neg RM(y)$; and

3. $S \cap R \neq \emptyset$.

Since $R \in \Delta$ and CHECK-REFINEMENTS$(\mathcal{S}, M^{\Delta})$ returns `succeed`, the instance

$$\Theta' = \Theta \cup \{uRv \mid uTv \in \Theta \text{ and } T \cap R \neq \emptyset\}$$

has a solution. However, $S \cap R \neq \emptyset$ so $xRy \in \Theta'$. We know that all models M of Θ have the property $M(x)\neg RM(y)$ so every model M' of Θ' must also have this property. This contradicts the fact that Θ' has a model and, consequently, $S \cap R = \emptyset$ and $S \subseteq \neg R$. We have thus shown that path-consistency makes $\neg R$ explicit.

if: Let Θ be a path-consistent instance of CSPSAT$(\mathcal{S})$ and arbitrarily choose a constraint $xSy \in \Theta$ such that $S \cap R \neq \emptyset$ for some $R \in \Delta$. The fact that path-consistency makes $\neg R$ explicit gives that $\Theta \cup \{xRy\}$ has a solution and, by independence, $\Theta' = \Theta \cup \{uRv \mid R \in \Delta,\ uTv \in \Theta \text{ and } T \cap R \neq \emptyset\}$ has a solution. However, Θ' is equivalent to Θ refined by the matrix M^{Δ} so CHECK-REFINEMENTS$(\mathcal{S}, M^{R})$ returns `succeed` by Theorem 8 □

Corollary 21 Given a set of relations $\mathcal{S} \subseteq 2^{\mathcal{A}}$ for which path-consistency computes minimal labels and a refinement matrix M^{Δ}, CHECK-REFINEMENTS-$(\mathcal{S}, M^{\mathcal{R}})$ returns `succeed` if and only if Δ is independent of $\mathcal{S}$.

Proof: If path-consistency computes minimal labels, then it makes $\neg R$ explicit for every $R \in \Delta$. □

Examples of when path-consistency computes minimal labels can, for instance, be found in Deville *et al.* [DBvH99] and Bessière *et al.* [BIL96].

4.2 Computational experience

We will demonstrate that many 1-independence results can be obtained by using the refinement method. We shall show that all independence results for the point algebras for partially and totally ordered time can be derived using refinements. This is possible since we know *every* maximal tractable

	Γ_A	Δ_A	Γ_B	Δ_B	Γ_C	Δ_C	Δ_D
$<$	•		•		•		
$\leq$	•		•				•
$<>$			•	•			
$<=>$			•	•			•
$\|$	•	•			•	•	
$\| =$	•				•	•	•
$=$	•		•		•		•
$\neq$	•	•	•	•	•	•	
$< \|$	•	•			•	•	
$\leq \|$	•				•	•	•

Table 1: Tractable classes of the point algebra for partially ordered time.

set of disjunctions of relations for partially ordered time [BJ00, Paper I] This, of course, requires a definition of a maximal tractable set of disjunctions of relations. Let B be a set of basic relations and $\Gamma \subseteq B^*$ such that CspSat(Γ) is tractable. We say that Γ is *maximal tractable* (with respect to B) iff for every $R \in B^*$ such that $R \notin \Gamma$, $\Gamma \cup \{R\}$ is not tractable.

The point algebra for partially ordered time is based on the notion of *relations* between pairs of variables interpreted over a partially-ordered set. We consider four basic relations which we denote by $<, >, =$ and $\|$. If x, y are points in a partial order $\langle T, \leq \rangle$ then we define these relations in terms of the partial ordering $\leq$ as follows:

1. $x < y$ iff $x \leq y$ and not $y \leq x$
2. $x > y$ iff $y \leq x$ and not $x \leq y$
3. $x = y$ iff $x \leq y$ and $y \leq x$
4. $x \| y$ iff neither $x \leq y$ nor $y \leq x$

The point algebra for partially ordered time has been throughly investigated earlier and a total classification with respect to tractability has been given by Broxvall and Jonsson [BJ99] and are also given in paper I of this thesis. In Broxvall and Jonsson [BJ00, Paper I] the sets of relations in Table 1 are defined and it is proven that $\Gamma_A{}^{\times}_{\vee}\Delta^*_A$, $\Gamma_B{}^{\times}_{\vee}\Delta^*_B$, $\Gamma_C{}^{\times}_{\vee}\Delta^*_C$ and Δ^*_D are the unique maximal tractable disjunctive classes of relations for partially ordered

$X\{\texttt{DR}\}Y$	iff	$X \cap Y = \emptyset$
$X\{\texttt{PO}\}Y$	iff	$\exists a, b, c : a \in X, a \notin Y, b \in X, b \in Y, c \notin X, c \in Y$
$X\{\texttt{PP}\}Y$	iff	$X \subset Y$
$X\{\texttt{PPI}\}Y$	iff	$X \supset Y$
$X\{\texttt{EQ}\}Y$	iff	$X = Y$

Table 2: The five basic relations of RCC-5.

time. The proofs of tractability for those sets relied on a series of handmade independence proofs. We will now derive these independence results using the refinement method.

To do so, we need to show that the classes $\Gamma_A, \Gamma_B, \Gamma_C$ and Δ_D are decidable by path-consistency. We begin by proving a useful connection (Lemma 22) between RCC-5 and the point algebra for partially ordered time which in turn will be needed to prove that path-consistency decides Γ_A and Γ_B.

RCC-5 [Ben94] is based on the notions of regions and binary relations on them. A region p is a regular open set of a topological space. Regions themselves do not have to be internally connected, i.e. a region may consist of different disconnected pieces.

Given two regions, their relation can be described by exactly one of the elements of the set **B** of five *basic RCC-5 relations*. The definition of these relations can be found in Table 2.

Lemma 22 Let Γ be a set of relations in the point algebra for partially ordered time and define the function σ such that

1. $\sigma(<) = \mathsf{PP}$;
2. $\sigma(>) = \mathsf{PP}^{-1}$;
3. $\sigma(=) = \mathsf{EQ}$; and
4. $\sigma(||) = (\mathsf{DR}\ \mathsf{PO})$.

Then, Γ can be decided by path-consistency if the set

$$\Gamma' = \{\bigcup_{r \in R} \sigma(r) \mid R \in \Gamma\}$$

of RCC-5 relations can be decided by path-consistency.

Proof: Let Π be an arbitrary CSP instance over the relations in Γ. Define the set Σ of RCC-5 formulae as follows: for each $x_i \ R \ x_j \in \Pi$, add the formula $x_i \ \bigcup_{r \in R} \sigma(r) \ x_j$. Note that Σ is a CSP instance over Γ' that, by assumption, can be decided by path-consistency.

We begin by comparing the composition tables for partially-ordered time and the RCC-5 relations (PP), (PP^{-1}), (EQ) and (DR PO):

	$<$	$>$	$=$	$\parallel$
$<$	$\{<\}$	$\top$	$\{<\}$	$\{\parallel <\}$
$>$	$\top$	$\{>\}$	$\{>\}$	$\{> \parallel\}$
$=$	$\{<\}$	$\{>\}$	$\{=\}$	$\{\parallel\}$
$\parallel$	$\{\parallel <\}$	$\{> \parallel\}$	$\{\parallel\}$	$\top$

	(PP)	(PP^{-1})	(EQ)	(DR PO)
(PP)	(PP)	$\top$	(PP)	(PP DR PO)
(PP^{-1})	$\top$	(PP^{-1})	(PP^{-1})	(PP^{-1} DR PO)
(EQ)	(PP)	(PP^{-1})	(EQ)	(DR PO)
(DR PO)	(PP DR PO)	(PP^{-1} DR PO)	(DR PO)	$\top$

By also noting that $<^{-1} = >$, $>^{-1} = <$, $\mathsf{PP}^{-1} = \mathsf{PPI}$, $\mathsf{PPI}^{-1} = \mathsf{PP}$ and that all other relations are invariant under $\cdot^{-1}$, it is obvious that the empty relation can be derived from Π if and only if it can be derived from Σ. Thus, we only have to show that whenever Σ has a model, Π also has a model.

Let M be a model that assigns regions to the variables $x_1, \ldots, x_n$ that appear in Σ. We define an interpretation N from the variables in Π to the partial order $\langle \{M(x_i) \mid 1 \leq i \leq n\}, \subseteq \rangle$ as follows: $N(x_i) = M(x_i)$ for $1 \leq i \leq n$. To conclude the proof, we pick an arbitrary constraint $x_i R x_j$ in Σ and show that it is satisfied by the interpretation N. Assume now, for instance, that $M(x_i)$ (PP) $M(x_j)$. By the definition of σ, we know that $\{<\} \subseteq R$ and it follows immediately that $N(x_i) < N(x_j)$ and the constraint $x_i R x_j$ is satisfied. The remaining cases can be proved analogously. □

Theorem 23 Path-consistency decides consistency for Γ_A, Γ_B, Γ_C and Δ_D.

Proof: Let $\Gamma' = \{\bigcup_{r \in R} \sigma(r) \mid R \in \Gamma_A\}$ (where σ is defined as in Lemma 22) and note that $\Gamma' \subseteq R_5^{28}$ [JD97]. Since R_5^{28} can be decided by path-consistency [RN99], Lemma 22 implies that path-consistency decides Γ_A.

Similarly, we can verify that path-consistency decides Γ_B; in this case, $\Gamma'' = \{\bigcup_{r \in R} \sigma(r) \mid R \in \Gamma_B\} \subseteq R_5^{14}$ [JD97]. Showing that R_5^{14} is decided by path-consistency is straightforward and left as an exercise (hint: compare R_5^{14} and the point algebra for totally ordered time and recall that the latter is decided by path consistency).

For Γ_C the result follows from the fact that it is a subset of Γ_A and Δ_D is trivially decided by path-consistency. □

By using the algorithm CHECK-REFINEMENTS, we can automatically verify that $\Delta_A, \Delta_B, \Delta_C$ and Δ_D are valid refinements of $\Gamma_A, \Gamma_B, \Gamma_C$ and Δ_D, respectively. Theorem 23 now gives that $\Delta_A, \Delta_B, \Delta_C$ and Δ_D are independent of $\Gamma_A, \Gamma_B, \Gamma_C$ and Δ_D, respectively, so we have proven tractability of all maximal tractable sets of disjunctions of relations for the point algebra for partially ordered time.

In Broxvall and Jonsson [BJ00, Paper I] the point algebra for totally ordered time is also investigated and the following two classes are defined:

$$\mathcal{X}_1 = \{(<), (<=), (<>), (=)\}^{\times}_{\vee}\{(<>)\}$$

$$\mathcal{X}_2 = \{(<=), (=)\}^*$$

These two classes are the only two maximal tractable sets of disjunctions of relations. It is well-known that path-consistency decides the point algebra for totally ordered time and the independence result can easily be verified using the refinement algorithm.

5 Conclusions and Open Questions

We have studied the complexity of reasoning with disjunctive constraints. We have shown that three previously presented properties are necessary and sufficient for tractability of CSPSAT(Δ^*), CSPSAT($\Gamma^{\times}_{\vee}\Delta^*$) and CSPSAT($\Gamma \cup \Delta^2$). There is at least one interesting case that is not covered by our results so we pose the following problem:

Open question 1. Assume CSPSAT($\Gamma \cup \Delta$) is tractable. What is a necessary and sufficient condition for tractability of CSPSAT($\Gamma^{\times}_{\vee}\Delta$)?

Ideas taken from Cohen *et al.* [CJG00] can probably be used for answering this question if we restrict Γ and Δ to be relations over disjoint domains.

We have provided a method for automatically deciding the 1-independence property based on refinements. The only requirement for applying this method is the sufficiency of path-consistency for deciding consistency in the class of constraints under consideration. In many cases this can, however, also be shown by using refinement techniques. We have demonstrated that this method is complete in two cases (the point algebras for totally-ordered and partially-ordered time) but we have not been able to prove this in general. We ask the following:

Open question 2. Assume path-consistency decides CspSat(Γ). Is it true that $\Delta \subseteq \Gamma$ is 1-independent of Γ if and only if Check-Refinements(Γ, M^{Δ}) returns succeed?

Even if it turns out that the answer to the previous question is 'yes', there is still room for improving the refinement method since (1) it is restricted to binary relations only; and (2) path-consistency must decide the underlying CspSat problem.

Open question 3. Given arbitrary sets Γ,Δ of relations, is there an algorithm for deciding whether Δ is 1-independent of Γ or not?

The previous questions naturally suggest our final question.

Open question 4. Given arbitrary sets Γ,Δ of relations, is there an algorithm for deciding whether Δ is 2-independent of Γ or not?

Acknowledgements

We would like to thank Andrei Bulatov, David Cohen, Peter Jeavons, Andrei Krokhin and the anonymous reviewers for valuable comments. Mathias Broxvall has been supported by the ECSEL graduate student program. Peter Jonsson has been supported by the Swedish Research Council for the Engineering Sciences (TFR) under grant 97-301 and the Swedish Research Counncil (VR) under grant 221-2000-361. Jochen Renz has been partially supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the

project FAST-QUAL-SPACE which is part of the DFG special research effort on 'Spatial Cognition', by the Wallenberg foundation and by a Marie Curie Fellowship of the European Community programme "Improving Human Potential" under contract number HPMF-CT-2000-00667.

References

[APT79] Bengt Aspvall, Michael F. Plass, and Robert Endre Tarjan. A linear time algorithm for testing the truth of certain quantified Boolean formulas. *Information Processing Letters*, 8:121–123, 1979.

[Ben94] Brandon Bennett. Spatial reasoning with propositional logics. In *Proc. 4th Int'l Conf. on Principles of Knowledge Repr. and Reasoning (KR-94)*, pages 165–176, Bonn, Germany, 1994.

[BIL96] Cristian Bessière, Amar Isli, and Gérard Ligozat. Global consistency in interval algebra networks: tractable subclasses. In *Proc. 12th Eur. Conf. on Artif. Intell. (ECAI-96)*, pages 3–7, Budapest, Hungary, 1996.

[BJ99] Mathias Broxvall and Peter Jonsson. Towards a complete classification of tractability in point algebras for nonlinear time. In *Proceedings of the 5th International Conference on Principles and Practice of Constraint Programming*, pages 129–143, Alexandria, VA, USA, 1999.

[BJ00] Mathias Broxvall and Peter Jonsson. Disjunctive temporal reasoning in partially ordered time structures. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 464–469, Austin, Texas, USA, 2000. AAAI Press.

[BJR00] Mathias Broxvall, Peter Jonsson, and Jochen Renz. Refinements and independence: A simple method for identifying tractable disjunctive constraints. In *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming*, pages 114–127, Singapore, 2000.

[CJG00] David Cohen, Peter Jeavons, and Richard Gault. New tractable classes from old. In *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming*, pages 160–171, Singapore, 2000.

[CJJK01] David Cohen, Peter Jeavons, Peter Jonsson, and Manolis Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47(5):826–853, 2001.

[DBvH99] Yves Deville, Olivier Barette, and Pascal van Hentenryck. Constraint satisfaction over connected row convex constraints. *Artificial Intelligence*, 109(1–2):243–271, 1999.

[Fre85] Eugene C. Freuder. A sufficient condition for backtrack-bounded search. *Journal of the ACM*, 32:755–761, 1985.

[GLS99] Georg Gottlob, Nicola Leone, and Francesco Scarcello. A comparison of structural CSP decomposition methods. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 394–399, Stockholm, Sweden, 1999.

[JC95] Peter Jeavons and Martin C. Cooper. Tractable constraints on ordered domains. *Artificial Intelligence*, 79:327–339, 1995.

[JD97] Peter Jonsson and Thomas Drakengren. A complete classification of tractability in RCC-5. *Journal of Artificial Intelligence Research*, 6:211–221, 1997.

[Mac77] Alan K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.

[NB95] Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *Journal of the ACM*, 42(1):43–66, 1995.

[Ren] Jochen Renz. On the complexity of binary relations. Technical report, Technische Universität Wien, forthcoming.

[Ren99] Jochen Renz. Maximal tractable fragments of the region connection calculus: A complete analysis. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 448–454, Stockholm, Sweden, 1999.

[RN99] 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.

[vBD95] Peter van Beek and Rina Dechter. On the minimality and decomposability of row-convex constraint networks. *Journal of the ACM*, 42(3):543–561, 1995.

Paper III

Constraint Satisfaction on Infinite Domains: Composing Domains and Decomposing Constraints

Mathias Broxvall

Abstract

We investigate the constraint satisfiability problem for infinite domains and demonstrate how tractable disjunctive constraints from disjunct domains can be combined to yield new tractable sets of relations. Inspired by these results, we also present a method that can be used to decompose problems into subproblems which can be solved independently. This method can be used to enhance any backtracking based constraint solver and we test the efficiency of this method on different ensembles of realistic problem instances.

Contents

1 Introduction

Many problems in computer science and elsewhere can be naturally expressed as constraint satisfaction problems. Although the constraint satisfaction problem is NP-hard in general, many special cases that can be solved in polynomial time have been identified (cf. [KJJ01]) and much effort has been put into discovering new tractable cases and methods to construct new tractable cases from old ones, see for instance [CJG00]. In this paper we investigate the constraint satisfaction problem for primarily infinite domains such as Allen's interval algebra [All83].

Previously, a method for dealing with disjunctions by identifying the independence property has been investigated [CJJK00]. We demonstrate that the independence property is preserved when building constraints from disjunct domains using the multiple relational union [CJG00]. For many tractable constraint satisfaction problems, satisfiability is determined by path consistency. (For instance, all previously identified tractable cases of RCC-8 can be determined by path consistency [Ren99].) We demonstrate that decidability by path consistency is preserved under the multiple relational union operator.

Inspired by the methods of ensuring tractability by restricting the allowed constraint relations, we also examine structural properties that can be exploited to solve problem instances more efficiently. More specifically, we identify a property among relations which can be used to decompose problem instances on a structural level to simpler problems which can be solved independently. Many other structural decomposition methods have been examined previously [GLS01, GJC94] and although they all guarantee tractability of problem instances of a certain structure their major weakness is that they fail when used on problem instances which are not decomposable and where backtracking is necessary. Although the method we suggest does not ensure tractability other than for very specialized structures, it can be used to enhance the efficiency of almost any backtracking based solver for constraint satisfaction problems over infinite domains. We also propose an automatic method for checking this property. Furthermore, we test this method on realistic problems and note that it can yield significantly improved performance for certain domains and has a very small overhead cost otherwise.

We will continue this paper with a brief introduction to the constraint satisfaction problem. Section 3 contains new methods for identifying cases

where problem instances over multiple domains can be solved in polynomial time. In Section 4 we give a decomposition method inspired by our earlier results and study it for a couple of example domains. This is followed in the last section by some tests on the practical speedup gained by utilizing the decomposition method on realistic problems.

2 Preliminaries

We begin by recalling the general constraint satisfaction problem CSPSAT. The consistency problem CSPSAT($\mathcal{S}$) for sets $\mathcal{S} = \{R_i | R_i \subseteq D^{n_i}\}$ over a domain D is defined as follows:

Instance: A set V of variables over a domain D and a finite set Θ of constraints $\langle R, x_1, \ldots, x_n \rangle$, where $R \in \mathcal{S}$ is a constraint of arity n.

Question: Is there a total function $f : V \to D$ such that for each constraint $\langle R, x_1, \ldots, x_n \rangle \in \Theta$ the following holds: $\langle f(x_1), \ldots, f(x_n) \rangle \in R$.

The general CSPSAT problem is NP-hard, but many sets of relations have been found such that deciding satisfiability of CSPSAT problem instances containing only those relations is tractable. By using the multiple relational union of tractable sets of relations in different domains, Cohen *et al.* [CJG00] have constructed new tractable sets of relations. Sometimes we refer to the constraint graph of a given problem instance. This is the graph consisting of a node for each variable in the instance and an edge (or hyper-edge) for each constraint connecting the variables in the constraint.

Definition 1 Let Γ_1 and Γ_2 be sets of relations over the non-empty sets D_1 and D_2 respectively. Then the *multiple relational union* of Γ_1 and Γ_2, written as $\Gamma_1 \dot{\bowtie} \Gamma_2$, is the following set of relations over $D_1 \cup D_2$:

$$\Gamma_1 \dot{\bowtie} \Gamma_2 = \left\{ R_1 \cup R_2 : \begin{array}{c} R_1 \in \Gamma_1, R_2 \in \Gamma_2 \text{ and} \\ \text{arity}(R_1) = \text{arity}(R_2) \end{array} \right\}$$

We also define two projection operators ρ_1, ρ_2 taking a composite relation $r \in \Gamma_1 \dot{\bowtie} \Gamma_2$ to its parts as follows:

$$\rho_1(r) = r \cap D_1^{\text{arity}(r)}$$

$$\rho_2(r) = r \cap D_2^{\text{arity}(r)}$$

These operators can also be applied to problem instances $\Pi = \langle V, C\rangle$ as follows:

$$\rho_1(\Pi) = \langle V, \{\langle \rho_1(R), x, y\rangle | \langle R, x, y\rangle \in C\}\rangle$$
$$\rho_2(\Pi) = \langle V, \{\langle \rho_2(R), x, y\rangle | \langle R, x, y\rangle \in C\}\rangle$$

In order to express disjunctive constraints we need to define the disjunction of relations:

Definition 2 Let R_1, R_2 be relations of arity i, j and define the disjunction $R_1 \vee R_2$ of arity $i + j$ as follows:

$$R_1 \vee R_2 = \left\{ (x_1, \ldots, x_{i+j}) \in D^{i+j} \;\middle|\; \begin{array}{c} (x_1, \ldots, x_i) \in R_1 \vee \\ (x_{i+1}, \ldots, x_{i+j}) \in R_2 \end{array} \right\}$$

Thus, the disjunction of two relations with arity i, j is the relation with arity $i + j$ satisfying either of the two relations. The disjunction of two sets of relations $\Gamma_1 {}^{\times}_{\vee} \Gamma_2$ is the set of disjunctions of each pair of relations in Γ_1, Γ_2 plus the sets Γ_1, Γ_2. Furthermore: for any set of relations, Δ, define $\Delta^* = \bigcup_{i=0}^{\infty} \Delta^i$ where $\Delta^0 = \{\bot\}$ and $\Delta^{i+1} = \Delta^i {}^{\times}_{\vee} \Delta$. The empty relation which is always unsatisfied is here written as $\bot$.

To prove tractability of disjunctive CSP languages, the independence property [CJJK00] is a useful tool.

Definition 3 For any sets of relations Γ and Δ, we say that Δ is *independent* with respect to Γ if for any set of constraints C in CSPSAT($\Gamma \cup \Delta$), C has a solution whenever every $C' \subseteq C$, which contains at most one constraint belonging to Δ, has a solution.

By identifying this property we can now solve certain classes of disjunctive constraints in polynomial time by using the following result given by Cohen *et al.* [CJJK00].

Theorem 4 For any sets of relations Γ and Δ, if CSPSAT($\Gamma \cup \Delta$) is tractable and Δ is independent with respect to Γ, then CSPSAT($\Gamma {}^{\times}_{\vee} \Delta^*$) is tractable.

We will demonstrate how the multiple relational union and independence can be combined, giving us the possibility to reason tractably over problem instances with constraints from different domains and containing certain types of disjunctions. The following domains will be used as examples in this paper:

- Allen's interval algebra where the variables represent intervals on the real line. See [All83] for definitions.
- The point algebra for totally ordered time where we reason about totally ordered points and have the primitive relations $<, =$ and $>$ defined in the obvious way. See also [BJ00, Paper I].
- The previous algebra can be extended to the point algebra for partially ordered time in which we allow the points to be partially ordered. The primitive relation then also contains the relation *parallel* ($\|$) which holds iff two time points are unordered. See also [BJ00, Paper I].
- Another point algebra for nonlinear time is the point algebra for branching time in which the points are ordered in a tree structure. It contains the relations $<, =, >$ and $\|$ as described above. For further information see [Bro01] also contained in paper I.
- The spatial algebras RCC-5 and RCC-8. A good description can be found in Renz and Nebel [RN99].

3 Composing domains

In this section we will study the complexity of the constraint satisfaction problem over multiple domains. We show that decidability by path consistency is preserved under the multiple relational union. Furthermore, we demonstrate that in order to be able to handle disjunctive constraints from disjoint domains we need to restrict ourselves to *well-typed* problem instances, that is, problem instances where the domain of variables is unambiguously determined by the constraints. For instance, the problem instance containing only the following constraints from RCC-8 and the algebra for totally ordered time is not well-typed since it is not possible to derive whether z, w, k represents regions (belong to the domain of RCC-8) or time points.

$$x < y \vee z \text{ NEQ } w$$

$$z > \text{NEQ } k$$

We write the multiple relational union of the timepoint relation $>$ and the RCC-8 relation NEQ as $>$NEQ here. Formally, restricting our instances to be well-typed can be described as a requirement that for each variable v in a

problem instance over domains $D_1, \ldots, D_n$ the constraint $\langle =_{D_i}, v, v\rangle$ should be included in the constraints where $=_{D_i}$ is the equality relation over some domain D_i. Note that the relation $=_{D_i}$ is a much more restrictive constraint than the general equality constraint = which does not say anything about the domain of the variables. By only considering well-typed problem instances, we can solve certain disjunctive constraints in polynomial time by noting that the independence property is preserved under the multiple relational union. We also demonstrate that without this restriction, the satisfiability problem is NP-hard as soon as even the simplest form of disjunctions is allowed. We begin with a quick note on how the composition operator behaves for relations over disjunctive domains.

Lemma 5 $(\gamma_1 \cup \gamma_2)\circ(\gamma_1' \cup \gamma_2') = (\gamma_1 \circ \gamma_1') \cup (\gamma_2 \circ \gamma_2')$ where γ_1, γ_1' and γ_2, γ_2' are relations over two disjunctive domains D_1 and D_2.

Proof sketch: Note that all the variables involved in the constraints either are all assigned values from D_1 or all from D_2. If the variables are assigned values from D_i the composition evaluates to $\gamma_i \circ \gamma_i'$, to preserve the choice of domain both versions must be maintained, thus $(\gamma_1 \circ \gamma_1') \cup (\gamma_2 \circ \gamma_2')$ □

Theorem 6 Let Γ_1 and Γ_2 be two tractable sets of relations over disjoint domains D_1 and D_2 such that path consistency implies global consistency, then consistency for $\Gamma_1 \dot{\bowtie} \Gamma_2$ is decided by path consistency.

Proof: Let $\Pi = \langle V, C\rangle$ be an arbitrary path consistent problem instance over the relations $\Gamma_1 \dot{\bowtie} \Gamma_2$ and define Π_1, Π_2 as follows:

$$\Pi_1 = \langle V, \{\langle \rho_1(R), x, y\rangle | \langle R, x, y\rangle \in C \wedge \rho_1(R) \neq \emptyset\}\rangle$$

$$\Pi_2 = \langle V, \{\langle \rho_2(R), x, y\rangle | \langle R, x, y\rangle \in C \wedge \rho_2(R) \neq \emptyset\}\rangle$$

We begin by showing that Π_1 and Π_2 are path consistent. Consider each triple of constraints $x\ \rho_i(R_1)\ y$,$x\ \rho_i(R_2)\ z$, $z\ \rho_i(R_3)\ y$ in Π_i. Since Π is path consistent and by using Lemma 5 we see that:

$$\rho_1(R_1) \cup \rho_2(R_1) \subseteq (\rho_1(R_2) \cup \rho_2(R_2)) \circ (\rho_1(R_3) \cup \rho_2(R_3)) =$$

$$(\rho_1(R_2) \circ \rho_1(R_3)) \cup (\rho_2(R_2) \circ \rho_2(R_3))$$

By noting that the relations in the given domains are closed under composition we see that $\rho_1(R_1) \subseteq \rho_1(R_2) \circ \rho_1(R_3)$ and $\rho_2(R_1) \subseteq \rho_2(R_2) \circ \rho_2(R_3)$.

Thus, each such triple of constraints is path consistent and hence Π_1, Π_2 are consistent.

Next, we will show that Π is consistent. Let f_1, f_2 be models of Π_1 and Π_2 respectively. We construct an interpretation of Π as follows:

$$f(x) = \begin{cases} f_1(x) & \text{if there exists } \langle R, x, y\rangle \text{ or } \langle R, y, x\rangle \\ & \text{in } C \text{ such that } \rho_1(R) \neq \emptyset \\ f_2(x) & \text{otherwise} \end{cases}$$

Now consider an arbitrary constraint $c = \langle R, x, y\rangle \in C$. We consider two cases; If $\rho_1(R) \neq \emptyset$ then c is satisfied by f_1 and hence by f. Otherwise, since $\rho_1(R) = \emptyset$ we have that for all constraints $c' = \langle R', x, \_\rangle$ and $c'' = \langle R'', \_, x\rangle$ holds that $\rho_1(R') = \rho_1(R'') = \emptyset$ because of the path consistency of Π. Hence, $f(x), f(y) \in D_2$ and c is satisfied by f_2. Thus, all constraints of Π are satisfied by f and consistency for $\Gamma_1 \dot{\bowtie} \Gamma_2$ is decided by path consistency. □

By noting that decidability by path consistency is preserved by the multiple relational union, we are not only able to solve problem instances using standardized algorithms but also have a necessary precondition for employing automated methods to identify the independence property [BJR00, Paper II] and partitioning relations (Theorem 14).

Only allowing binary constraints is (in practice) a severe limitation when modeling problems. Theoretically it is of course possible to model problems involving non-binary relations by using binary constraints over more complex domains. The limitation of having only binary constraints is especially apparent in the context of combining several domains since a problem instance constructed only by binary constraints can essentially be seen as a number of disjoint, simpler problems, where the solutions of different subproblems have no influence on each other. For instance, given a problem instance over domains $D_1, \ldots, D_n$ with k sets of variables $V_1, \ldots, V_k$ such that the constraints occur only within each V_i, satisfiability of the the problem is simply asking whether for every V_i there exists some D_j such that $\rho_j(V_i)$ is satisfiable. This limits the expressibility since constraints such as "event A occurred before event B or region X is not the same as region Y" cannot be modeled. We continue by demonstrating why we need to restrict our problem instances to be well-typed in order to be able to decide satisfiability for these kind of constraints in polynomial time. For this purpose we need the definition of the NP-complete[GJ79] problem MONOTONE 3SAT:

Instance: Set U of variables, collection C of clauses over U such that each $c \in C$ has $|c| = 3$ and contains either only positive or negative literals.

Question: Is there a truth assignment for U satisfying all clauses in C?

We can now by means of a reduction from this problem prove that deciding satisfiability is an NP-hard problem when we allow disjunctions of the multiple relational union of relations from two disjoint domains. This means that in order to be able to decide satisfiability of problem instances involving disjunctions and multiple domains we must restrict ourselves to well-typed problem instances.

Theorem 7 Let γ_1, γ_2 be two binary relations over disjoint domains D_1, D_2 respectively. Then CSPSAT$((\{\gamma_1\} \dot{\bowtie} \{\gamma_2\})^2)$ is NP-hard.

Proof: We make a reduction from the NP-complete problem MONOTONE 3SAT. Let $P = \langle U, C \rangle$ be an arbitrary MONOTONE 3SAT problem instance. We construct a problem instance Π of CSPSAT$((\{\gamma_1\} \dot{\bowtie} \{\gamma_2\})^2)$ as follows and show that Π is satisfiable iff P is satisfiable:

1. For each variable $v \in U$ let x_v, y_v be two fresh variables and add the constraint:
$$x_v(\gamma_1 \cup \gamma_2)y_v$$

2. For each positive clause $c = \langle i, j, k \rangle \in C$ add two fresh variables t_c, s_c and the constraints:
$$x_i\gamma_1 y_i \vee t_c\gamma_1 s_c$$
$$x_j(\gamma_1 \cup \gamma_2)s_c \vee x_k(\gamma_1 \cup \gamma_2)s_c$$
and for each negative clause:
$$x_i\gamma_2 y_i \vee t_c\gamma_2 s_c$$
$$x_j(\gamma_1 \cup \gamma_2)s_c \vee x_k(\gamma_1 \cup \gamma_2)s_c$$

Next, we prove that Π is satisfiable iff P is satisfiable. For the if-direction let $\mathcal{I}$ be a satisfying truth assignment for P and we construct an interpretation f of Π as follows.
$$f(x_i) = \begin{cases} d_1 & \text{if } \mathcal{I}(i) \text{ is true} \\ d_2 & \text{otherwise} \end{cases}$$

$$f(y_i) = \begin{cases} d_1' & \text{if } \mathcal{I}(i) \text{ is true} \\ d_2' & \text{otherwise} \end{cases}$$

$$f(t_{c^+}) = \begin{cases} f(x_j) & \text{if } \mathcal{I}(i) \text{ is true} \\ d_1 & \text{otherwise} \end{cases}$$

$$f(s_{c^+}) = \begin{cases} f(y_j) & \text{if } \mathcal{I}(i) \text{ is true} \\ d_1' & \text{otherwise} \end{cases}$$

where d_i, d_i' are two arbitrary values in D_i such that $d_i \gamma_i d_i'$ and c^+ denotes arbitrary positive tuples $\langle i, j, k \rangle$. Assignment of negative tuples if handled analogously as positive tuples by f. It is easy to see that each constraint of Π is satisfied by f and, hence, f is a model of Π. Thus Π is satisfiable whenever P is satisfiable.

For the only-if direction, let f be a model of Π and $\mathcal{I}$ the truth assignment assigning a the value true iff $f(x_a) \in D_1$. Obviously, each clause $\langle i, j, k \rangle$ is satisfied by $\mathcal{I}$ since either $x_i \in D_i$ or $s \in D_i$. If $s \in D_i$ then either x_j or x_k is also in D_i. Thus we have proven that Π is satisfiable iff P is satisfiable and, CSPSAT$((\Gamma_1 \dot{\bowtie} \Gamma_2)^2)$ is NP-complete. □

Next, we show that by restricting the allowed problem instances to be well-typed we can solve some disjunctive constraints in polynomial time by using the independence property. For these kinds of problem instances it is pointless to use the $\dot{\bowtie}$ operator to combine relations since each such constraint in a problem instance can immediately be refined to one of its projections. Hence, we instead use the $\cup$ operator on sets of relations from disjoint domains.

Theorem 8 Let Γ_1 and Γ_2 be two sets of relations over disjoint domains D_1 and D_2 and $\Delta_1 \subseteq \Gamma_1$,$\Delta_2 \subseteq \Gamma_2$ two sets of relations independent of Γ_1,Γ_2 respectively. Then, $\Delta_1 \cup \Delta_2$ is independent of $\Gamma_1 \cup \Gamma_2$, for all well-typed problem instances.

Proof: Let $\Pi = \langle V, C \rangle$ be a problem instance of CSPSAT$(\Gamma_1 \cup \Gamma_2)$, such that for every subset $C' \subseteq C$ containing at most one constraint from $\Delta_1 \cup \Delta_2$, $\langle V, C' \rangle$ is satisfiable.

By showing that Π is satisfiable we prove that $\Delta_1 \dot{\bowtie} \Delta_2$ is independent of $\Gamma_1 \dot{\bowtie} \Gamma_2$. First, assume that the constraint graph of Π is connected. This implies that all variables in Π have been defined to be of the domain D_i so we consider $\rho_i(\Pi)$ which is a subproblem of Π. We note that all subproblems of

$\rho_i(\Pi)$ containing at most one relation from Δ_i are also a subproblem of some set of constraints C' as described initially. Hence, each such subproblem of $\rho_i(\Pi)$ is satisfiable and since Δ_i is independent of Γ_i we have $\rho_i(\Pi)$ satisfiable. Thus, Π is satisfiable whenever the constraint graph of Π is connected.

For the case when the constraint graph is disconnected we simply apply the same reasoning on each component of constraints. □

The applicability of this result is obvious when we consider problems involving more than one domain. For instance, since inequality is independent of the point relations and since the set $\Delta_{\hat{\mathcal{H}}_8}$ is independent of the tractable $\hat{\mathcal{H}}_8$ fragment of RCC-8 [BJR00] we see that we can in polynomial time solve problem instances over time points and regions containing disjunctions of both the point relation disequality and the $\Delta_{\hat{\mathcal{H}}_8}$ relations. The possibilities of combining different domains with previous independence and tractability results is thus very promising.

4 The decomposition method

In this section, we are concerned with situations where we cannot solve our problem instances in polynomial time and need methods to speed up the solving process. We introduce the notion of partitioning relations and show how these can be used to decompose complex problem instances into several smaller and simpler instances that can be solved independently.

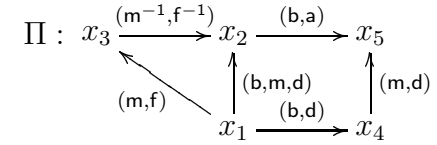
Definition 9 Let Γ be a set of relations and $\Pi = \langle V, C\rangle$ be a problem instance over Γ containing only binary constraints. A *partitioning* of Π is a problem instance $\pi = \langle\{\pi_1, \ldots, \pi_n\}, c\rangle$ with variables representing problem instances $\pi_1 = \langle v_1, c_1\rangle, \ldots, \pi_n = \langle v_n, c_n\rangle$ such that:

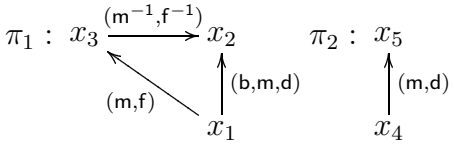
1. all $c_i \subseteq C$
2. $V = \bigcup_{i=1}^{n} v_i$ and $v_1, \ldots, v_n$ are disjoint.
3. For all binary constraints $x\ R\ y \in C$ holds $x\ R\ y \in c_i$ for some c_i or $x \in v_i$,$y \in v_j$ and $\exists R' \subseteq R : \pi_i\ R'\ \pi_j \in c$.
4. π is satisfiable.

It is important here to distinguish between relations and constraints. Since a relation is only a set of tuples from a domain, there is no problem in condition three above to constraint the relation R' to be included in the relation R.

The concept of partitioning is further explained by an example below where we take a problem instance of five variables and split it into two partitions containing three and two variables respectively. The constraints within each of those partitions are preserved as they are. From the constraints between the two partitions we get a partitioning which is a problem instance of two variables π_1, π_2 and just one constraint which is trivially satisfiable.

Example 10 We can partition the Allen interval algebra problem instance Π below into the partitioning $\pi = \langle\{\pi_1, \pi_2\}, \{\pi_1\ (\mathsf{b})\ \pi_2\}\rangle$ with π_1, π_2 as given below. As we will see later this allows us to solve π_1, π_2 independently of each other.





Next, we present a property of relations that, in a way, resembles the independence property.

Definition 11 Let Γ be a set of binary relation over a domain D and $\varphi \subseteq \Gamma$ a set of relations such that: For every problem instance Π from Γ and every partitioning $\pi, \pi_1, \ldots, \pi_n$ of Π such that π contains only relations from φ, Π is satisfiable iff $\pi_1, \ldots, \pi_n$ is satisfiable. We say that φ *partition* Γ.

By identifying relations which *partitions* other relations we can simplify the search problem of deciding satisfiability for problem instances by identifying partitionings consisting of these relations. Similar to the results in the previous section we show that the partitioning property of relations is preserved when combining domains.

Theorem 12 Let Γ_1, Γ_2 be sets of relations over the disjoint domains D_1, D_2. If φ_1, φ_2 are two subsets of Γ_1, Γ_2 such that φ_i partitions Γ_i then $\varphi_1 \cup \varphi_2$ partitions $\Gamma_1 \cup \Gamma_2$.

Proof: Let $\Pi = \langle V, C \rangle$ be a problem instance of CSPSAT($\Gamma_1 \cup \Gamma_2$) and $\pi, \pi_1, \ldots, \pi_n$ a satisfiable partitioning of Π. We prove that $\varphi_1 \cup \varphi_2$ partitions $\Gamma_1 \cup \Gamma_2$ by showing that Π is satisfiable iff $\pi_1, \ldots, \pi_n$ is satisfiable. The only-if direction is trivial; for the if direction assume that $\pi_1, \ldots, \pi_n$ is satisfiable.

We can safely assume that the constraint graph for each partition π_i is connected. First, assume that the constraint graph of π is connected and that π contains only constraints over the domain D_i. It must then hold that each partition π_j contains only constraints over the domain D_i since the variables in π_j are connected to the variables in π. Then, each $\rho_i(\pi_j)$ and $\rho_i(\pi)$ is satisfiable and since φ_i partitions Γ_i we see that $\rho_i(\Pi)$ is satisfiable. Since the constraint graph for π and each π_j is connected, we see that the constraint graph for Π is connected. This mean that $\Pi = \rho_i(\Pi)$ and Π is satisfiable.

When the constraint graph of π is not connected, we can simply consider each component of the graph. □

As the next result demonstrates we can combine partitioning sets of relations easily. This gives us a more convenient description of sets of partitioning relations by only giving their minimal elements, since we implicitly know that their unions are also partitioning.

Theorem 13 Let φ be a set of relations partitioning Γ, then the set $\varphi' = \{\gamma_1 \cup \gamma_2 | \gamma_1, \gamma_2 \in \varphi\}$ partitions Γ.

Proof: Let Π be an arbitrary problem instance of Γ and $\pi', \pi'_1, \ldots, \pi'_n$ a satisfiable φ' partitioning of Π. Since π' is satisfiable it can be reduced to some $\pi \subset \pi'$ containing only relations from φ and hence, $\pi, \pi'_1, \ldots, \pi'_n$ is a satisfiable φ problem instance. Hence, Π is satisfiable iff $\pi'_1, \ldots, \pi'_n$ is satisfiable. Thus, we have proven that φ' partitions Γ. □

By using the following result we have an automatic method for deriving sets of partitioning relations.

Theorem 14 Let Γ be a set of relations over a domain D, $\psi \subseteq \Gamma$ such that consistency for ψ is determined by path-consistency and:

$$\forall r \in \Gamma : \exists r_1, \ldots, r_n \in \psi : r = r_1 \cup \cdots \cup r_n$$

For all sets of relations $\varphi \subset \psi$, we have that φ partitions Γ iff the constraints $\{x\ \gamma_1\ y, y\ \gamma_1^{-1}\ z, z\ \gamma_2\ x\}$ are consistent for all $\gamma_1 \in \varphi, \gamma_2 \in \psi$.

Proof: For the if-direction, let Π be an arbitrary problem instance of Γ and $\pi, \pi_1, \ldots, \pi_n$ a satisfiable partitioning of Π. We show that Π is satisfiable whenever $\pi_1, \ldots, \pi_n$ is satisfiable. Let $f, f_1, \ldots, f_n$ be a model of $\pi, \pi_1, \ldots, \pi_n$ respectively. We construct a new problem instance Π' containing all the variables of Π and the following constraints:

1. For each $x, y \in \pi_i$ add the constraint $x\ R\ y$ where R is the primitive relation containing $\langle f_i(x), f_i(y)\rangle$.

2. For each $x \in \pi_i, y \in \pi_j$ add the constraint $x\ R\ y$ where R is a relation in φ containing $\langle f(\pi_i), f(\pi_j)\rangle$.

Consider each triple of constraints $x\ R_1\ y, y\ R_2\ z, x\ R_3\ z$. We examine three cases. If x, y, z are all in one partition π_i, this triple of constraints must clearly be path consistent since they originate from the model f_i. Also, if they all belong to different partitions π_i, π_j, π_k they must also be path consistent since they are related only by the path consistent relations from f. For the remaining case when one of the variables comes from one partition and the two others from some other partition we see that $R_i \in \varphi, R_j = R_i^{-1}$ and $R_k \in \Gamma$ and hence, they are consistent. Thus, Π' is path consistent and, since it only contains primitive relations, consistent. By noting that Π is a relaxation of Π' we see that Π is consistent and hence the if-direction of the theorem holds.

Next, we prove the only-if direction. We show that the constraints $\Pi : x\ \gamma_1\ y, y\ \gamma_1^{-1}\ z, z\ \gamma_2\ x$ are consistent for all $\gamma_1 \in \varphi, \gamma_2 \in \Gamma$ by constructing a partitioning of Π assigning x, z to one partition and y to another. Obviously this partitioning is satisfiable and both partitions are satisfiable. Hence, Π is satisfiable. □

By using the previous theorem, it is now trivial to construct partitioning sets of relations for many relational algebras. We give only the primitive relations of the partitioning sets of relations here since inclusion of all their disjunctions follows from Theorem 13. The following sets of relations partitions the full set of relations for their respective domains.

- $\{<, >\}$ of the point algebra for linear time.

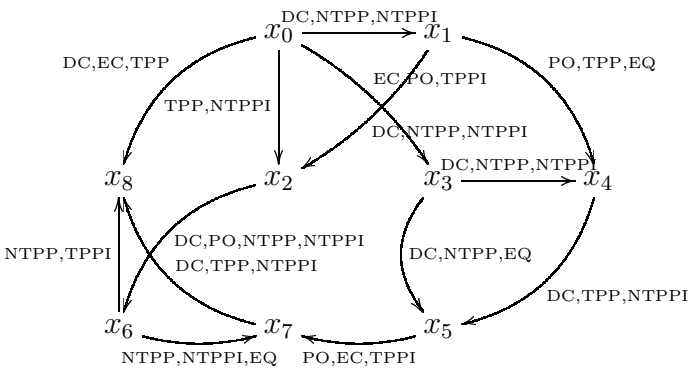


Figure 1: A problem instance for the domain RCC-8

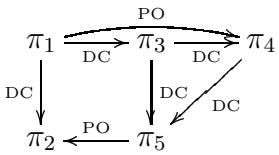
- $\{<, \|, >\}$ of the point algebra for partially ordered time.
- The relation $\|$ of the point algebra for branching time.
- $\{(\mathsf{b}), (\mathsf{a})\}$ of Allen's interval algebra.
- $\{DC, PO\}$ of RCC-8 and $\{DR, PO\}$ of RCC-5.

Having identified partitioning relations for some domains we now give another example of partitioning a problem instance and see how this can be used to speed up the satisfiability calculations for this instance. Consider the RCC-8 problem instance Π given in Figure 1 which contains ten constraints not present in any (previously identified) tractable set of relations for RCC-8. If we were to solve this problem as a standard backtracking search problem [RN01] and non-deterministically split these constraints into constraints in some tractable set of relations we would in the worst case end up backtracking over a complete search tree of depth 10 containing several thousand nodes. However, by a simple preprocessing step we can partition the problem as follows:

$$\pi_1 = \{x_0, x_1, x_2\} \;\; \pi_2 = \{x_6, x_7, x_8\}$$

$$\pi_3 = \{x_3\} \;\; \pi_4 = \{x_4\} \;\; \pi_5 = \{x_5\}$$

The constraints within each partition are left unchanged, and the constraints between partitions π are as given below:



Note that π is path-consistent and only contains relations present in $\hat{\mathcal{H}}_8$ which is a set of relations for which path-consistency decides consistency [RN99]. Also, all relations in π partitions the full set of relations for RCC-8. Hence, $\pi, \pi_1, \ldots, \pi_5$ is a satisfiable partitioning of Π. Thus, we can decide consistency of Π by checking the consistency for the partitions $\pi_1, \ldots, \pi_5$ independently of each other which can be done by visiting very few nodes. Ideally this method yields a significant speed up for problem instances where a satisfiable partitioning can be found. In practice the original problem would be solved much more efficiently than visiting a complete search tree of depth 10 by employing better techniques to direct search and prune the search tree and therefore the actual speed up will not be as large as given in this example. However, as we will see in the test runs, the actual speed up is for most randomly generated problem instances is enough to motivate the use of the partitioning method.

Next, we define partitionings on problem instances containing disjunctions and demonstrate how partitioning relations can be used there as well. Furthermore, when using disjunctions the notion of *partial* partitionings also makes sense.

Definition 15 Let $\Pi = \langle V, C \rangle$ be a problem instance containing constraints from a set of binary relations Γ and disjunctions thereof. A *partitioning with disjunctions* of Π is a problem instance $\pi = \langle \{\pi_1, \ldots, \pi_n\}, c \rangle$ with variables representing problem instances $\pi_1 = \langle v_1, c_1 \rangle, \ldots, \pi_n = \langle v_n, c_n \rangle$ such that:

1. all $c_i \subseteq C$
2. $V = \bigcup_{i=1}^{n} v_i$ and $v_1, \ldots, v_n$ are disjoint.
3. For all binary constraints $x\ R\ y$ in C or part of a disjunction in C holds $x\ R\ y \in c_i$ for some c_i or $x \in v_i$,$y \in v_j$ and $\exists R' \subseteq R : \pi_i\ R'\ \pi_j \in c$.

4. For each disjunction $d : x_1\ R_1\ y_1\ \vee\ \ldots\ \vee\ x_m\ R_n\ y_m$ in Π it holds that either $x_i \in v_j, y_i \in v_k$ and $\exists R' \subset R_i : \pi_j\ R'\ \pi_k \in c$ for some i, j, k or $x_1, \ldots, x_m, y_1, \ldots, y_m \in v_i$ and $d \in c_i$ for some i.

5. π is satisfiable.

A *partial* partitioning (with disjunctions) is a partitioning with disjunctions not satisfying condition 4 above.

Example 16 If we add the constraint $x_1\ (\mathsf{f})\ x_3 \vee x_1\ (\mathsf{b})\ x_5$ to Π in Example 10 we get a partitioning with disjunctions. If we furthermore add the constraint $x_1\ (\mathsf{a}, \mathsf{d})\ x_2 \vee x_4\ (\mathsf{b}, \mathsf{m}^{-1})\ x_5$ to Π we only get a partial partitioning.

Our next result shows us that we can use partitioning relations to decompose problem instances containing disjunctions.

Theorem 17 Let Π be a problem instance containing binary and disjunctive constraints over Γ and let φ be a set of relations partitioning Γ. Furthermore, let $\pi, \pi_1, \ldots, \pi_n$ be a partitioning with disjunctions of Π such that π contains only relations from φ. Then Π is satisfiable iff $\pi_1, \ldots, \pi_n$ is satisfiable.

Proof: Proof by induction over the number of disjunctive constraints. The base case with zero disjunctions is trivial since $\pi, \pi_1, \ldots, \pi_n$ then is a satisfiable partitioning (without disjunctions) of Π. For the inductive step, let $d : x_1\ R_1\ y_1\ \vee\ \ldots\ \vee\ x_m\ R_n\ y_m$ be a constraint of Π We have two cases: First assume that $x_i \in v_j, y_i \in v_k$ and $\exists R' \subset R_i : \pi_j\ R'\ \pi_k \in c$ hold for some i, j, k. If Π is satisfiable, then so is trivially $\pi, \pi_1, \ldots, \pi_n$. For the other direction, let Π' be Π with d refined as $x_i R_i y_i$; if Π' is satisfiable then so is Π. By noting that $\pi, \pi_1, \ldots, \pi_n$ is a partial partitioning of Π' also the induction hypothesis gives that Π' and Π is satisfiable. Hence, Π is satisfiable iff $\pi, \pi_1, \ldots, \pi_n$.

Otherwise, we have $x_1, \ldots, x_m, y_1, \ldots, y_m \in v_i$ and $d \in c_i$ for some i. If some π_j is unsatisfiable then so is Π since π_j is a subproblem of Π. Otherwise, if each π_j is satisfiable we form π_i' as π_i with d substituted for one of its binary constraints $x_a\ R\ y_a$ which is satisfied by some model of π_i. Evidently π_i' is satisfiable. We also form Π' as Π with d substituted by $x_a\ R\ y_a$ and note that Π is satisfiable if Π' is satisfiable. Since $\pi, \pi_1, \ldots, \pi_{i-1}, \pi_i', \pi_{i+1}, \ldots, \pi_n$ is a satisfiable partitioning with disjunctions of Π' the induction hypothesis gives that Π' is satisfiable and hence Π is satisfiable. □

no partitions	16	78	19	75	81	83	90	163	547	18046
with partitions	19	24	19	25	27	28	33	155	546	6028

Table 1: Number of nodes visited when solving a set of twelve problem instances for partially ordered time containing 40 variables and 150 disjunctions of binary constraints each.

Even in the case when we fail to identify a partitioning with disjunctions, we can use a partial partitioning to remove some of the (disjunctive) constraints in the problem thus making the problem easier to solve since fewer constraints need to be considered during backtracking.

Theorem 18 Let $\Pi = \langle V, C\rangle$ be a problem instance and $\pi = \langle v, c\rangle, \pi_1 = \langle v_1, c_1\rangle, \ldots, \pi_n = \langle v_n, c_n\rangle$ a partial partitioning of Π such that π only contains relations from a set of partitioning relations φ. Furthermore, form the set of disjunctions $M \subseteq C$ with a disjunct in different partitions:

$$M = \left\{ \begin{array}{c} d : x_{d_1} R_{d_1} y_{d_1} \vee \ldots x_{d_m} R_{d_m} y_{d_m} \in C \text{ such that} \\ \text{for some } d_i, d_j, d_k : x_{d_i} \in v_{d_j}, y_{d_i} \in v_{d_k} \text{ and} \\ \exists R' \subset R_{d_i} : \pi_{d_j}\ R'\ \pi_{d_k} \in c \wedge j \neq k \end{array} \right\}$$

and let Π' be Π with each constraint c such that $c \in M$ substituted for $x_{d_i} R_{d_i} y_{d_i}$. Then Π is satisfiable iff Π' is satisfiable.

Proof: If Π' is satisfiable then trivially so is Π. For the other direction, assume that Π is satisfiable. We form Π'' from Π by replacing each disjunctive constraint not in D by one of its binary constraints. Obviously, there exists at least one way of doing this so that Π'' is satisfiable. Next, note that $\pi, \pi_1, \ldots, \pi_n$ is a partitioning of Π'' and Theorem 17 gives that Π'' with each disjunction d in D replaced by $x_{d_i} R_{d_i} y_{d_i}$ is satisfiable. Since this is Π' with some disjunctions (those not in D) constrained to binary relations, we find that Π' is satisfiable. □

We describe a method for identifying a partitioning for arbitrary problem instances. Let φ be a partitioning set of relations and $\Pi = \langle \{v_1, \ldots, v_n\}, C\rangle$ a problem instance. We can now create a partitioning of Π as follows:

1. For each v_i create the partition $\pi_i = \langle \{v_i\}, \emptyset\rangle$. Let $\top^\varphi$ be the union of all relations in φ and set $\pi = \langle \{x_1, \ldots, x_n\}, \{\langle \top^\varphi, x_i, y_j\rangle | 1 \leq i, j \leq n\}\rangle$

2. For each binary constraint $\langle R, x, y\rangle \in C$ and each disjunction containing such a constraint: Let R' be the relation such that $\langle R', x, y\rangle$ in π and refine this constraint to $\langle R' \cap R, x, y\rangle$.

3. Contract each pair of partitions π_i, π_j such that $\langle \emptyset, x_i, x_j\rangle \in \pi$. That is, replace each occurrence of x_j with x_i in π and set $\pi_i = \pi_i \cup \pi_j$. Calculate path consistency for π and repeat until π is path consistent.

4. Contract, in the same manner as in the previous step, two arbitrarily chosen partitions in π until π is consistent.

5. For each disjunctive constraint $d : x_1\ R_1\ y_1\ \vee\ \ldots\ \vee\ x_m\ R_n\ y_m$ in Π such that $x_i \in v_j, y_i \in v_k$ and $\exists R' \subset R_i : \pi_j\ R'\ \pi_k \in c$ does not hold for any i, j, k, contract all partitions containing one of $x_1, \ldots, x_m$ or $y_1, \ldots, y_m$.

6. For each constraint $c = \langle R, x_1, \ldots, x_n\rangle$ in C such that $x_1, \ldots, x_n \in \pi_i$ for some i, add c to the constraints of π_i.

Clearly, the algorithm above identifies only valid partitionings. In most cases φ is decided by path consistency and π will already be consistent in step four of the algorithm. In the cases where φ is not decided by path consistency, a domain-dependent heuristic for choosing partitions to contract can be used to increase the chance of finding partitionings.

This method of identifying partitionings can be exploited by a solver in several ways. First, it can be used as a simple preprocessing step on instances containing only binary relations by splitting them into simpler instances and solving them independently. For problem instances containing disjunctions, the possibilities of identifying partitionings increase as disjunctions are replaced by binary relations during search. It is therefore worthwhile during search to continuously make new attempts to identify partitionings. Furthermore, the possibilities in which we can use the method is not limited to identifying partitionings with disjunctions and solve them independently. If we halt the algorithm before step five we get a partial partitioning which together with Theorem 18 gives us the opportunity of eliminating some of the constraints in order to speed up search.

Constraints	Part.	random			powerlaw			smallworld		
binary only	no	101	111	119	59	67	75	63	71	80
	yes	74	88	95	41	49	65	5.5	3.5	2.7
nearby disj.	no	83	120	76k	83	115	74k	115	1445	151k
	yes	34	92	72k	31	70	49k	46	1100	155k
random disj.	no	61	229	80k	87	184	58k	116	1064	124k
	yes	30	206	80k	36	125	37k	35	1378	116k

Table 2: Average number of nodes visited when solving problem instances from the point algebra for partially ordered time.

5 Results

It is easy to construct problem instances which can be solved by the method described in the previous section thousands of times faster than when the method is not used. This can be done by for instance taking the disjoint union of several hard problem instances. In this section we test the efficiency of using this method on slightly more realistic problems by running it on different ensembles of randomly generated problem instances in different domains. For this purpose we have implemented a system that solves constraint satisfaction problems for arbitrary domains using backtracking and path-consistency algorithms. Our system also includes options to control the use of different heuristics such as forward checking and the partitioning method described here. The system as well as the problem instances on which it was tested is available for downloading from the author's web-site at:

`http://www.ida.liu.se/~matbr`

Many different methods of generating constraint graphs for empirically evaluating constraint solving algorithms have been suggested in the literature [BA99, WS98, Hog96]. The perhaps simplest method of generating random problem instances is that of generating purely random constraint graphs. However, such graphs rarely occur in real life [Wal01]. Rather than using purely random instances, constraint graphs with a *smallworld* [WS98] or *powerlaw* [BA99] structure have been suggested since they occur frequently in real world problems. We have therefore run our tests using both random, powerlaw and smallworld graphs.

We have run our system on both problem instances consisting purely of binary constraints as well as on problem instances containing disjunctions of binary constraints. Since the partitioning method can be combined with other heuristics such as forward checking and fail-first, there is a forward checking component in our system similar to that described by Stergiou and Koubarakis [SK00]. This is done by simply adding code that attempts to identify partitionings every nth time the forward-checking algorithm visits a new node in the search tree. If a partitioning is found, then we solve the problem by recursively calling the forward checking algorithm on each partition.

A fairly typical example of the number of nodes visited when solving a set of problem instances can be found in Table 1. As can be seen from this example the time needed to solve problem instances varies greatly even though the problem size is kept constant. Thus, just a few of the problem instances will dominate the average of the solution time which would therefore produce a poor measurement of the efficiency of the algorithms. Furthermore, simply comparing the median execution times of the algorithms is also a bad idea as this tells us nothing about the performance of the easiest and the hardest problems. We therefore choose to present our test data by sorting the problem instances by difficulty and give the average of the lowermost, middle and topmost third. We also choose to present the number of nodes visited during search because it is easier to compare with other systems and it gives a more accurate and reproducible measurement compared to measuring runtimes. Comparing the total number of nodes visited when using the partitioning method and when not using it gives a fair measurement since the execution time of computing partitionings is small compared to the total execution time.

We begin the investigations by comparing the efficiency of using our method by running tests on problem instances containing only binary relations. Furthermore, we have chosen to run the solver on only problem instances which are initially path consistent. We do this since problem instances that are not path consistent can rather be regarded as malformed problem instances than normal unsatisfiable problems since the cost of detecting and removing these problem instances is relatively small compared to the cost of solving problem instances for which we need to use backtracking. If we had chosen not to filter out the non-path-consistent problem instances, the experiment would be completely dominated by the cost of performing the initial path consistency check since this would be enough to rule out

Constraints	Part.	random			powerlaw			smallworld		
binary only	no	25	86	1754	30	95	838	4.6	9.6	694
	yes	24	82	1773	32	131	854	4.8	8.8	674
nearby disj.	no	610	5652	94k	48	1149	67k	1120	4582	118k
	yes	514	4621	90k	44	1138	66k	1002	4153	116k
random disj.	no	734	4889	90k	532	13k	158k	583	7120	78k
	yes	733	5080	113k	530	15k	141k	582	7099	78k

Table 3: Average number of nodes visited when solving problem instances from the Allen algebra.

most problem instances. Hence, the actual performance of the backtracking algorithms would not have been measured if we were using such problem instances.

Since we also want to test the efficiency of using the partitioning method on problem instances containing disjunctions, we need a method to generate such problem instances. The method we chose was to first generate a graph with either a random, powerlaw or smallworld structure just as for the binary problem instances. Edges from this graph were in turn picked in pairs to become the binary constraints in a disjunction of two binary constraints. We tested the difference between two different methods of picking these edges, either purely randomly or with a weighted probability where edges close to each other in the graph were more likely to be picked together.

We began by measuring the number of nodes visited when solving problem instances for the point algebra for partially ordered time. By varying the number of constraints and variables in the different ensembles we generated problem instances in the phase transition region. For the binary problem instances the constraint graphs contained 300 variables and had an average degree of between 5 and 15 depending on the ensemble. For the disjunctive constraints we had between 25 and 50 variables and 175 to 220 constraints. As we will see, the gains of using the partitioning method on partially ordered time is sufficiently large for problem instances containing disjunctions to motivate its use. The success in this case can be attributed to the fact that as many as three of the four primitive relations can be present between partitions. Table 2 contains the average number of nodes visited when solving problem instances from these ensembles.

When testing the algorithms on Allen's interval algebra we choose problem ensembles containing between 40 and 100 variables and with an average constraint graph degree of 11 to 15 for the binary problem instances. For the problem instances containing disjunctions we had between 12 and 15 variables and 95 to 120 constraints. Using only between 40 and 100 variables for the binary problem instances may sound little. However, solving Allen algebra problem instances in the phase transition region is a hard computational problem, see e.g. the experimental results of Nebel [Neb97], and analyzing the algorithm on larger problem instances would be impractical.

As can be seen in the results, the gains from using the partitioning method on this domain are not as large as for other domains. This can best be understood by noting that only two primitive relations, *before* and *after*, of the thirteen possible primitive relations can be present between partitions in problem instances for this algebra. The likelihood of partitions of relations occurring in randomly generated problem instances for this algebra is thus not as big as for the point algebra for partially ordered time and this is reflected in the number of visited nodes in Table 3. Note however that the poor results of this method on randomly generated problem instances does not necessarily imply that the method should not be used for Allen's interval algebra. Even though it does not yield much speed up for most problem instances, it can achieve high gains for some highly structured problem instances and although it can give a slight performance loss in some cases, it more often than not provides a performance boost even on randomly generated problem instances.

We have also performed tests on several other domains such as the point algebra for linear time, RCC-5 and RCC-8. The improvements for these domains lie between that for Allen's interval algebra and that for the point algebra for partially ordered time. We note that the method is more successful for problem instances from domains with a larger proportion of partitioning relations and with more structure than random graphs.

It can reasonably be expected that the success of using the partitioning method on problem instances from different domains is dependent on the proportion of partitioning relations in that domain. For instance, for partially ordered time it is quite successful since 75% of the primitive relations partition the rest of the relations. For the Allen algebra where only 15% of the primitive relations partition the rest, the results of the test runs are not as promising. If this holds in the general case, it should be fairly efficient to use this method for the point algebra for linear time where 67% of

Disj.	Structure	Part.	lin			rcc-5			rcc-8		
none	random	no	1			156	171	179	292	398	2731
		yes	1			138	148	159	294	396	2731
	powerlaw	no	1			22	28	32	150	162	175
		yes	1			11	8.0	5.9	77	79	79
nearby	random	no	62	75	6986	60	222	62k	162	3403	97k
		yes	26	43	6616	43	236	64k	137	3376	74k
	powerlaw	no	38	71	3573	62	110	26k	54	109	8205
		yes	18	31	727	44	100	22k	43	95	8124
random	random	no	81	95	22k	61	94	20k	136	2020	89k
		yes	34	100	18k	41	80	21k	141	2284	90k
	powerlaw	no	60	97	13k	46	82	38k	74	1280	71k
		yes	22	59	7652	31	63	37k	67	1309	67k

Table 4: Average number of nodes visited for the point algebra for linear time, RCC-5 and RCC-8.

the primitive relations can be used in partitions and though less efficiently still useful for RCC-5 and RCC-8 where 40% and 25% respectively, of the relations can be used. Some test runs for these domains can be found in Table 4. Note that we do not provide data for binary problem instances for linear time since the full set of binary relations for linear time can be solved by path consistency so only one node needs to be visited to solve these kinds of problem instances.

6 Conclusions

In this paper we have examined the constraint satisfaction problem and presentd new methods for deriving tractable classes containing disjunctions for problem instances combining disjoint domains. We have investigated how to efficiently solve the constraint satisfaction problem for intractable cases and identified a property which enables us to split problem instances into smaller and simpler instances which can be solved independently. Furthermore, we also provide an automatic method to test for this property. By implementing a general system for solving such problems, we have also been able to test the efficiency of using the methods described on realistic problem instances. As was expected, the results vary greatly depending on the

domain and the type of problem instances generated. The benefit of using this method is much greater on domains where a larger proportion of the relations can be used in partitionings.

References

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[BA99] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[BJR00] Mathias Broxvall, Peter Jonsson, and Jochen Renz. Refinements and independence: A simple method for identifying tractable disjunctive constraints. In *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming*, pages 114–127, Singapore, 2000.

[Bro01] Mathias Broxvall. The point algebra for branching time revisited. In *Proceedings of the Joint German/Austrian Conference on Artificial Intelligence (KI-2001)*, Vienna, Austria, 2001.

[CJG00] David Cohen, Peter Jeavons, and Richard Gault. New tractable classes from old. In *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming*, pages 160–171, Singapore, 2000.

[CJJK00] David Cohen, Peter Jeavons, Peter Jonsson, and Manolis Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47(5):826–853, 2000.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[GJC94] Marc Gyssens, Peter G. Jeavons, and David A. Cohen. Decomposing constraint satisfaction problems using database techniques. *Artificial Intelligence*, 66:57–89, 1994.

[GLS01] Georg Gottlob, Nicola Leone, and Francesco Scarcello. A comparison of structural CSP decomposition methods. *Artificial Intelligence*, 124:243–282, 2001.

[Hog96] Tad Hogg. Refining the phase transition in combinatorial search. *Artificial Intelligence*, 81(1-2):127–154, 1996.

[KJJ01] Andrei Krokhin, Peter Jeavons, and Peter Jonsson. Reasoning about temporal relations: The tractable subalgebras of Allen's interval algebra. Technical Report PRG-RR-02-12, Oxford University, 2001.

[Neb97] Bernhard Nebel. Solving hard qualitative temporal reasoning problems: Evaluating the efficiency of using the ORD-Horn class. *Constraints*, 1(3):175–190, 1997.

[Ren99] Jochen Renz. Maximal tractable fragments of the region connection calculus: A complete analysis. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 448–454, Stockholm, Sweden, 1999.

[RN99] 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.

[RN01] Jochen Renz and Bernhard Nebel. Efficient methods for qualitative spatial reasoning. *Journal of Artificial Intelligence Research (JAIR)*, 15:289–318, 2001.

[SK00] Kostas Stergiou and Manolis Koubarakis. Backtracking algorithms for disjunctions of temporal constraints. *Artificial Intelligence*, pages 81–117, 2000.

[Wal01] Toby Walsh. Search on high degree graphs. In *Proceedings of the 17th International Joint Conferance on Artificial Intelligence (IJCAI-2001)*, pages 266–271, Seattle, WA, USA, 2001.

[WS98] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

Paper IV

A Method for Metric Temporal Reasoning

Mathias Broxvall

Abstract

Several methods for temporal reasoning with metric time have been suggested — for instance, Horn Disjunctive Linear Relations (Horn DLRs). However, it has been noted that implementing this algorithm is non-trivial since it builds on fairly complicated polynomial-time algorithms for linear programming. Instead, an alternative approach which augments Allen's interval algebra with a Simple Temporal Problem (STP) has been suggested [Con00]. In this paper, we present a new point-based approach STP* for reasoning about metric temporal constraints. STP* subsumes the tractable preconvex fragment of the augmented interval algebra and can be viewed as a slightly restricted version of Horn DLRs. We give an easily implementable algorithm for deciding satisfiability of STP* and demonstrate experimentally its efficiency. We also present a method for finding solutions to consistent STP* problem instances.

Contents

1 Introduction

Temporal reasoning in various forms has long been an important task in many areas of AI. Examples of tasks that have been studied include deciding consistency or finding a solution of a set of constraints over temporal variables. Typically these variables represent time points or time intervals over a linear time domain. The constraints imposed on these variables can either be of a purely qualitative nature, stating relations such as precedence or equality, or of a more quantitative nature. Examples of the later case would be to state that a certain time point occurs at least a given amount of time units before another or that an interval is of a certain length.

The time complexity for qualitative reasoning is fairly well understood. Reasoning about intervals is commonly done using Allen's interval algebra [All83] for which the satisfiability problem is NP-complete in the general case but for which several tractable fragments have been identified. One of the more useful tractable fragments is the ORD-Horn fragment [NB95] which consists of the 868 interval relations which can be expressed by conjunctions of constraints of the form $x\ R\ y \vee z \neq w$ where x, y, z, w denote endpoints of the intervals and $R \in \{\leq, \neq\}$. Note that x, y or z, w may denote the same endpoints for non-disjunctive constraints such as the interval constraint "before or meets" which can be written as $x^+ \leq y^- \vee x^+ \neq x^+$. The ORD-Horn fragment is also known as the set of preconvex interval algebra relations. Qualitative reasoning on time points is done using the point algebra and satisfiability can easily be decided using path consistency.

A number of methods have been proposed for reasoning about metric time. One of these is *Horn Disjunctive Linear Relations* [JB98, Kou01], Horn DLRs for short. This is a temporal constraint formalism for which satisfiability can be checked in polynomial time. The expressibility of Horn DLRs subsumes the ORD-Horn algebra and most of the other tractable formalisms for temporal reasoning and satisfiability is decided using a linear programming approach. As a consequence, the algorithm for deciding satisfiability is not trivial to implement and special considerations need to be taken to the numerical precision during calculations. These problems have been pointed out in a number of papers, cf. [Con00, PS99]. For a good presentation of the polynomial time algorithms for linear programming and the numerical issues involved see e.g. [PS82]. Another slightly more efficient algorithm solving the linear programming problem has been presented by Vaidya [Vai87].

A method proposed by Condotta [Con00] handles metric time by aug-

menting Allen's interval algebra and the rectangle algebra [Güs89] with quantitative STP constraints on the endpoints. Although this method yields an algorithm which is simpler to implement than the Horn DLR approach, it lacks the expressibility of Horn DLRs and has the rather high time complexity of $O(n^5)$.

In this paper we present yet another approach for handling metric time called STP* which has an expressibility subsuming that of the tractable ORD-Horn fragment of the augmented interval (and rectangle) algebra but slightly more restricted than Horn DLRs. By restricting ourselves to using this method we get a very simple algorithm for deciding satisfiability that has a low practical time complexity and a good expressibility. We recall that the Simple Temporal Problem (STP) allows metric constraints of the form $x+r \leq y$ where x, y represent point variables and r is a real constant. The satisfiability problem for STPs and STPs augmented with disequality constraint (STP$^{\neq}$) relations has been well investigated previously [DMP91, GC97] and efficient algorithms have been identified for deciding satisfiability. By basing our approach on STPs and allowing disjunctions of disequality we get an expressive formalism that retains most of the simplicity and efficiency of the satisfiability algorithms for STP and STP$^{\neq}$. As we will see later in this paper, the practical cost of adding metric information is relatively small when we use this method.

We continue this paper by recalling the definitions of STP$^{\neq}$, presenting STP* and an algorithm for solving the satisfiability problem in the next section. In the following section we investigate the expressibility of STP* and show that the tractable preconvex fragment of the augmented interval algebra can be expressed in terms of STP* constraints and that certain rectangle algebra relations can also be expressed in STP*. Subsequently we continue with some studies and comparisons of an actual implementation of this algorithm to other approaches. Finally, in the last section we present some concluding remarks and open questions.

2 Simple Temporal Problem

We begin this section by giving the definition of STP$^{\neq}$ introduced by Gerevini and Cristani [GC97] and recall some of their previous results. This is in turn followed by the necessary definitions for extending STP$^{\neq}$ with disjunctions of disequality, yielding STP*, and an algorithm for deciding satisfiability of

STP* problem instances.

Definition 1 An STP$^{\neq}$ problem instance is a tuple $\langle V, C\rangle$ of a set of variables V and constraints C such that all constraints in C are of the form $x + r \leq y$ or $x + r \neq y$ for $x, y \in V$ and $r \in R$. The satisfiability problem for STP$^{\neq}$ is that of finding a mapping from the variables onto R satisfying all the constraints C.

We say that an STP$^{\neq}$ problem instance $\mathcal{T}$ entails $x + d = y$ iff there exists no solution for $\mathcal{T}$ such that $x + d \neq y$. An STP$^{\neq}$ problem instance can also be considered as a labeled graph over its variables with an edge labeled r between x, y for each constraint $x - r \leq y$. This graph is called the distance graph of the instance. For STP$^{\neq}$ the following result have been proven by Gerevini and Cristani [GC97] which is helpful for deciding satisfiability of STP$^{\neq}$ problem instances. This will later be used in the proofs for the algorithm deciding satisfiability of STP* instances.

Lemma 2 An STP$^{\neq}$ problem instance $\mathcal{T}$ is consistent iff $\mathcal{T}$ does not have negative cycles in its distance graph, and it does not entail $w + d = v$ for any inequation $w + d \neq v$ in $\mathcal{T}$.

We are now ready to define the STP* point algebra and the satisfiability problem for it. As we will see, STP* and STP$^{\neq}$ are closely related and the algorithm for deciding satisfiability of STP* problem instances resembles that for deciding satisfiability of STP$^{\neq}$ instances given by Gerevini and Cristani [GC97].

Definition 3 An STP* problem instance is a tuple $\langle V, C\rangle$ of a set of variables V and constraints C such that all constraints in C is of the form $x_1 + r_1 \leq y_1 \vee x_2 + r_2 \neq y_2 \vee \cdots \vee x_n + r_n \neq y_n$ for $x, y \in V, r \in R$ and $n \geq 1$. The satisfiability problem for STP* is that of finding a mapping from the variables onto R satisfying all the constraints C.

The maximum clause size of a problem instance is the largest n for all the constraints in it. Note that the definition allows simple $x + r \leq y$ constraints by having $n = 1$. Since STP* is a restriction of the Horn-DLR framework satisfiability could in principle be decided using the independence algorithm [CJJK00] and a solver for STP$^{\neq}$. Indeed, the algorithm we present here is largely based on the independence algorithm but by using Lemma 2 we have

1 Algorithm: STP-1
2 Input: A set C of STP^* clauses over the set of variables $1 \ldots n$.
3 Let M be the $n \times n$ matrix such that $M[k][l] = 0$ if $k = l$ and ∞ otherwise .
4 **repeat**
5 $changed \leftarrow false$
6 update M to contain the shortest path between every pair of variables.
7 **if** $\exists x : M[x][x] < 0$ **then reject**
8 **for** $c = x_1 + r_1 \leq y_1 \vee x_2 + r_2 \neq y_2 \vee \cdots \vee x_m + r_m \neq y_m$ in C **do**
9 **if** $r_1 < M[x_1][y_1] \wedge \neg\exists i > 1 : M[x_i][y_i] \neq r_i \vee M[y_i][x_i] \neq r_i$ **then**
10 $M[x_1][y_1] \leftarrow r_1$
11 $changed \leftarrow true$
12 remove c from C
13 **end**
14 **until** $changed = false$
15 **accept**

Figure 1: The algorithm for solving STP* problem instances

enhanced the algorithm so that a solution is created incrementally and a complete call to an underlying STP$^{\neq}$ solver can be avoided in each iteration which makes the algorithm more efficient when a shortest path algorithm which handles incremental changes is used in step 6 of the algorithm. In our implementation a modified Floyd-Warshall [CLR97] that keep track of modified variables is used for this purpose. We can now prove that the algorithm given in Figure 1 only accepts satisfiable problem instances.

Theorem 4 Algorithm STP-1 accepts only satisfiable STP* problem instances

Proof: Let Π be an STP* problem instance that is accepted by STP-1 and let M, C be the adjacency matrix and the set of clauses left when the algorithm accepts. We construct an STP$^{\neq}$ problem instance Π' from Π by having the constraints $x - M[x][y] \leq y$ for all x, y such that $M[x][y] < \infty$. For each clause $c = x_1 + r_1 \leq y_1 \vee x_2 + r_2 \neq y_2 \vee \cdots \vee x_m + r_m \neq y_m$ in C we add the constraint $x_i + r_i \neq y_i$ for some i such that $M[x_i][y_i] \neq r_i$ or $M[y_i][x_i] \neq r_i$.

Obviously, Π is satisfiable if Π' is satisfiable. We note that Π' contains no negative cycles and that for each constraint $x + r \neq y$ we have that Π does

not entail $x + r = y$. By lemma 2 we see that Π' is satisfiable and thus Π is also satisfiable. □

We continue by proving that the algorithm only rejects unsatisfiable STP* problem instances.

Theorem 5 Algorithm STP-1 rejects only unsatisfiable STP* problem instances.

Proof: Let Π be an STP* problem instance that is rejected by the algorithm and let M, C be the adjacency matrix and the set of clauses left when the algorithm rejects. Assume that Π is satisfiable. It is then possible to construct an STP$^{\neq}$ problem instance Π' by selecting one term of each clause in Π such that Π' is satisfiable. From lemma 2 we know that Π' contains no negative cycles and that for each $x + r \neq y$ constraint Π' does not entail $x + r = y$.

We say that M contain a constraint $x - r \leq y$ if $M[x][y] = r$ and $r < \infty$ and we show that M only contains constraints that are present in Π' by induction on the number of constraints in M. The base case of zero constraints is trivially true. For the inductive case assume that this holds for the first n constraints added to M by the algorithm and let $c = x_1 + r_1 \leq y_1 \vee x_2 + r_2 \neq y_2 \vee \cdots \vee x_m + r_m \neq y_m$ be the clause containing the $(n+1)$th constraint $x_1 + r_1 \leq y_1$ added. Since we have $\neg\exists i > 2 : M[x_i][y_i] \neq r_i \vee M[y_i][x_i] \neq r_i$ and all constraints in M are also present in Π' we see that $x_i + r_i = y_i$ is entailed by Π' for all $i > 2$. Thus, Π' must also contain the constraint $x_1 + r_1 \leq y_1$ added by the algorithm to M.

Now, since M only contains constraints in Π' and the algorithm rejects, we see that M and Π' have a negative cycle, which contradicts lemma 2. Thus, Π is unsatisfiable. □

Next, we note that the algorithm runs in $O(|C|(|V|^3 + |C|t))$ time for problem instances $\langle V, C\rangle$ with maximum clause size t since the shortest path problem can be solved in $O(n^3)$ using e.g. the Floyd-Warshall algorithm [CLR97]. For converted interval and rectangle problem instances our algorithm thus takes $O(n^5)$ time where n is the number of variables in the original problem instance. Thus, our algorithm has the same asymptotical time complexity as Condotta's algorithm for the preconvex fragment of the augmented interval algebra but with with a greater expressibility.

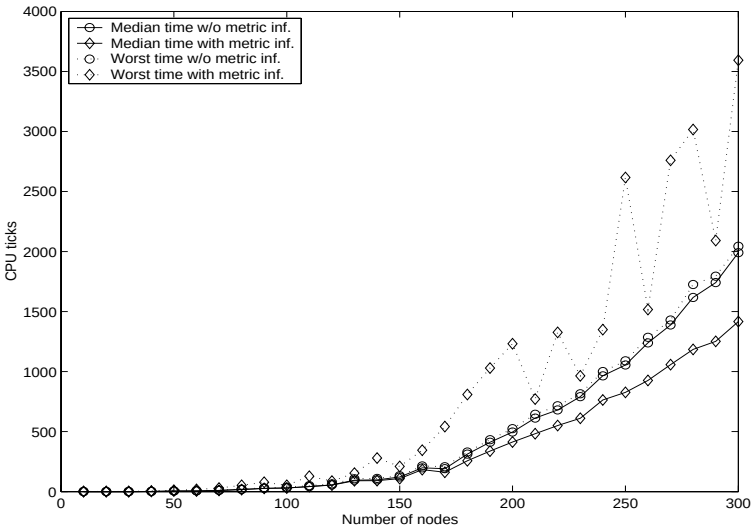


Figure 2: Running time of the STP* solver for converted interval algebra problem instances with and without constraints imposed on the lengths of intervals. For each sample, 500 problem instances were generated and solved.

For some applications it is not sufficient to show that a problem instance is satisfiable but an actual solution is also sought. Finding the solution to a given consistent STP* problem instance can be done by constructing the equivalent STP$^{\neq}$ problem instance implied by the algorithm STP-1 and using for instance the approach found in [GC97] to find the solution to the STP$^{\neq}$ problem instance. The conversion to STP$^{\neq}$ is done by converting each disjunctive clause $c = x_1 + r_1 \leq y_1 \vee x_2 + r_2 \neq y_2 \vee \cdots \vee x_n + r_n \neq y_n$ that is removed at step 12 of the algorithm to the STP$^{\neq}$ constraints $x_1 + r_1 \leq y_1$ and for each clause remaining in C when the algorithm accepts convert it to the STP$^{\neq}$ constraint $x_i + r_i \neq y_i$ where $i > 2$ and $M[x_i][y_i] \neq r_i \vee M[y_i][x_i] \neq r_i$.

3 Expressing Interval and Rectangle Algebra Problems in STP*

In this section we discuss how interval and rectangle algebra problems can be expressed in terms of STP* problems. We demonstrate that the full set of preconvex interval algebra relations (i.e. the ORD-Horn algebra) can be expressed using STP* and that the expressibility of STP* subsumes that of

the preconvex fragment of the augmented interval algebra [Con00].

The interval algebra was introduced by Allen [All83] and is a relational algebra consisting of 13 atomic relations and their disjunctions. The relations of the interval algebra operate over intervals in a real valued domain are the following: *before (b), after (a), meets (m), met-by (mi), overlaps (o), overlapped-by (oi), during (d), includes (di), starts (s), started-by (si), finishes (f), finished-by (fi)* and *equals (eq)*. An interval algebra network is a set of variables and a set of binary constraints over the variables where each constraint is a disjunction of the atomic relations.

In the rectangle algebra the variables represents rectangles and the constraints disjunctions of tuples of interval constraints over the rectangles projected onto two orthogonal axes. For instance, the constraint $x(b, m)y$ signifies that the rectangle x should come strictly before y on the first axis and should meet y along the second axis.

In order to be able to compare the STP* formalism with the augmented interval algebra we need it's definition as well. The following definition come from Condotta [Con00].

Definition 6 An augmented interval network $\mathcal{M}$ is a pair $\langle \mathcal{N}, \mathcal{S} \rangle$ where $\mathcal{N} = \langle V, C \rangle$ is an interval network which represents the qualitative constraints on the intervals in V and $\mathcal{S} = \langle Points(V), C' \rangle$ is an STP instance representing the quantitative constraints on the distances between the bounds of the intervals in V.

Condotta [Con00] has proven that deciding satisfiability for augmented interval networks containing only preconvex relations in its qualitative part can be done in $O(n^5)$ time.

Theorem 7 The satisfiability problem for the augmented interval algebra restricted to problem instances containing only preconvex relations can be reduced to the satisfiability problem for STP* in linear time.

Proof: Let $\mathcal{M} = \langle \mathcal{N}, \mathcal{S} \rangle$ be an augmented interval network such that $\mathcal{N}$ contains only preconvex (ORD-Horn) relations. By definition we know that there exists a constant number of equivalent STP* constraints for each preconvex relation. We construct an STP* problem instance by introducing two STP* variables x^+ and x^- and the constraint $x^- \leq x^+$ for each interval x in $\mathcal{N}$ and translate each qualitative $\mathcal{IA}$ constraint in $\mathcal{N}$ into a constant

number of STP* constraints. By also including the STP constraints $\mathcal{S}$ we have an an STP* problem instance which is satisfiable iff $\mathcal{M}$ is satisfiable. □

Furthermore, STP* is able to express constraints which cannot be expressed with only the preconvex relations of the interval algebra. For instance, let $x\ R\ y$ be an interval algebra constraint with a preconvex relation R and C the corresponding set of STP* constraints. If we let C' be C with the term $z^+ \neq w^+$ added to each clause and C'' be C with the term $z^- \neq w^-$ added we can express for instance the non-binary constraints $x\ R\ y \vee z\ neq\ w$ as $C' \cup C''$ and $x\ R\ y \vee z \neg(f \vee fi)w$ as C'. We continue by recalling the definition of the augmented rectangle algebra as given by Condotta [Con00] and will see that STP* manages to express some of the constraints in this algebra too.

Definition 8 An augmented rectangle network $\mathcal{M}$ is a triple $\langle \mathcal{N}, \mathcal{S}_1, \mathcal{S}_2 \rangle$ where $\mathcal{N} = \langle V, C \rangle$ is a rectangle network which represent the qualitative constraints on the rectangles in V and where $\mathcal{S}_1 = \langle Points(V'), C' \rangle$ and $\mathcal{S}_2 = \langle Points(V''), C'' \rangle$ are two STPs representing the quantitative constraints on the distances between the bounds of the interval of V' and V'' respectively, where V', V'' represents the set of the projections of the rectangles in V onto the first and second axis respectively.

In this case, we represent each rectangle with four STP* variables representing the endpoints of the rectangles projected onto each axis. For instance, the rectangle algebra constraint $x(b, mi)y$ representing the fact that x lies left of and meets y from above can be expressed by the STP* constraints $x_1^+ \leq y_1^-$, $x_1^+ \neq y_1^-$, $x_2^- \leq y_2^+$ and $y_2^+ \leq x_2^-$. Obviously, since deciding satisfiability for the full rectangle algebra is an NP-complete problem, STP* cannot express all the relations in the full rectangle algebra. Note however that there exist large tractable fragments of the rectangle algebra such as the set of strongly preconvex rectangle algebra relations [BCF99]. The full extent of which rectangle algebra relations can be expressed in terms of STP* constraints remains an open question. Note that all the basic relations in the rectangle algebra can be expressed using STP* constraints.

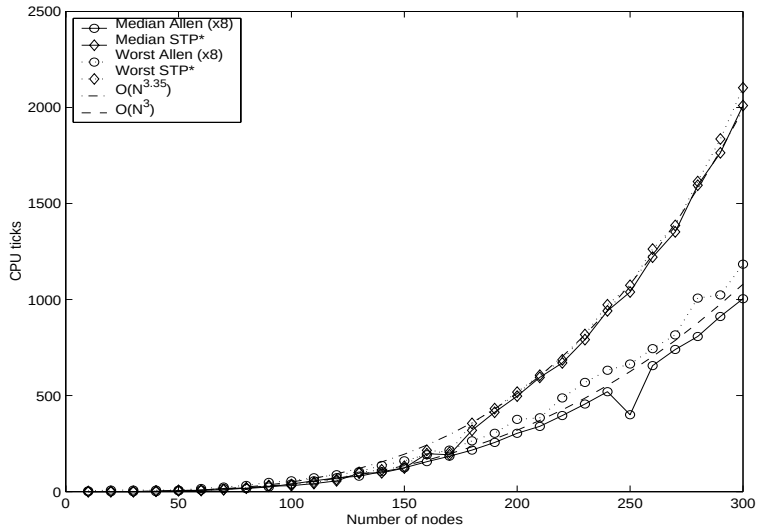


Figure 3: Measured running times for the solvers varying the number of nodes in problem instances. For each sample 100 problem instances were generated and solved. Note that the cpu time for the Allen algebra based solver has been multiplied by a factor of eight to better illustrate the asymptotical behavior of the solvers.

4 Evaluating STP*

As was shown in the previous section the expressibility of STP* subsumes that of the preconvex fragment of the augmented interval algebra while still retaining a fairly simple algorithm for deciding satisfiability. In this section we investigate the actual runtime properties of this algorithm. We note that algorithm STP-1 is both simple to implement and gives a practical and efficient method for reasoning about points and intervals with metric information. Although the theoretical time complexity of the algorithm is $O(|C|(|V|^3 + |C|t))$ for problem instances $\langle V, C\rangle$ with maximum clause size t (for most applications t is a fixed constant and can be disregarded), it runs in almost cubic time with regard to the number of variables in practice.

In order to test the running time of the algorithm we have first generated random sets of problem instances (without metric information) of the interval algebra. We choose a random graph structure for these problems and with a constant factor chosen from the phase transition region [CKT91] as the average constraintedness. We tested the algorithm on these data and

noted that it ran in approximately $O(n^{3.35})$ time for this specific problem ensemble. A reasonable guess considering the behavior of the algorithm would be that adding metric information (in this case, constraints on the lengths of intervals) will not affect the practical time complexity of the algorithm considerably. As can be seen in Figure 2 adding metric information to our problem instances gives a slight performance increase in the median case, since more problem instances can be found unsatisfiable at an earlier stage, and a small constant performance decrease in the worst case for each problem instance ensemble.

We continue by comparing the running time of this algorithm by that of another solver working directly on the intervals of a problem instance that was used for evaluating the efficiency of using the ORD-Horn fragment [Neb97]. For this we generate a number of instances having only qualitative preconvex interval constraints from Allen's interval algebra and solve these instances using both an efficient solver [Neb97] deciding path consistency on the interval constraints directly and using our solver. The conversion of the interval algebra constraints is done by a simple translation procedure containing a case for each possible interval relation, the cost of translating the constraints is thus linear with the number of interval constraints which is quadratic with respect to the number of variables. In the graphs we have omitted the cost of translating the constraints since it is assumed that in a real-world application the translation should either be done in the modeling of the problem or with a more efficient in-memory approach.

As can be seen, the cost of using our solver is relatively low. Since the size of problem instances are doubled when working on point relations, using STP* takes a constant factor of eight times longer. In order to more easily compare the asymptotical behavior of the two different approaches this constant factor has been removed in the figures but it is important to remember this extra constant cost induced by the extended expressibility of STP* compared to the interval algebra when solving real-world problems. Furthermore, the addition of metric constraints gives a higher time complexity, but in practice not as high as the theoretical $O(n^5)$. Overall, the cost of allowing metric information in problem instances seems to be small and is mostly dominated by the constant factor of eight for problem instances of reasonable size. The running times for the two different approaches using the same problem instances can be found in Figure 3. It would here also have been interesting to make a comparison of the STP* solver and a solver for Horn DLRs. However, as far as we know there exists no publicly available

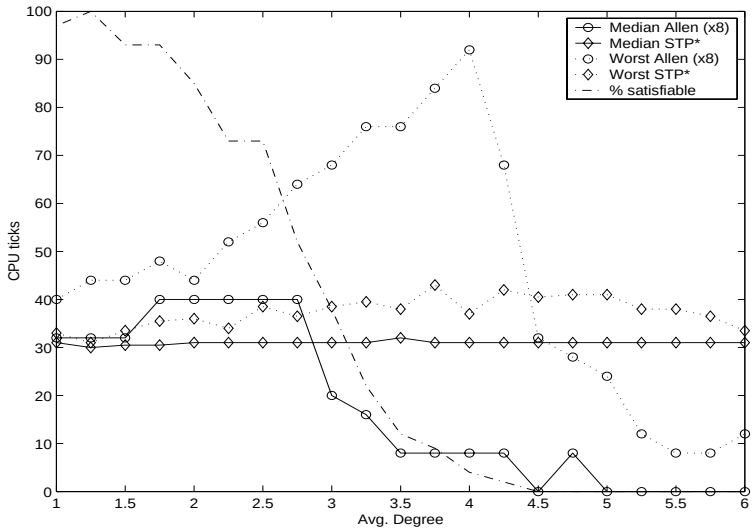


Figure 4: Measured running times for the solvers using 100 problem instances of 100 nodes each and varying the average degree. Note that the cpu time for the Allen algebra based solver has been multiplied by a factor of eight to simplify comparisons.

implementation of such a solver.

Of further interest is also to study the two different approaches with respect to the probability of satisfiability for the problem instances. This is done in Figure 4 by varying the average degree of problem instances. Again, the running times of the approach based on Allen's interval algebra have been scaled by a factor eight to ease comparisons. As we can see the effect of varying the constraintedness of the problem instances is much larger for the $\mathcal{IA}$ approach than for STP* where the running time has very little correlation with the constraintedness of problem instances. This can perhaps best be explained by noting that information is lost in the conversion from interval algebra problems to the point algebra approach used in STP*.

5 Concluding Remarks

In this paper we have introduced a new extension STP* of the Simple Temporal Problem (STP) which allows disjunctions of disequalities in the constraints and demonstrates how this algebra can be used efficiently for tem-

poral reasoning with metric time. We have given a simple and efficient algorithm for deciding satisfiability of STP* and shown how a solution to a given STP* instance can be found by means of a reduction to STP$^{\neq}$.

The expressibility of STP* extends that of the set of preconvex relations for the augmented interval algebra, which is the only tractable set of relations for this algebra which contains all the basic relations. Furthermore, this algebra can be used not only for temporal reasoning but can also handle, for instance, fractions of the rectangle algebra.

For future work, it would be interesting to enhance the algorithm so that intervals involving only purely qualitative constraints would be handled as they are, without being converted into two STP* points. This would remove some of the efficiency loss induced by having twice as large problem instances for converted $\mathcal{IA}$ instances. Another open question is that of which rectangle algebra relations can be expressed in terms of STP* constraints. Is it possible to express the full set of strongly-preconvex $\mathcal{RA}$ relations using only the point constraints allowed in STP*?

References

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[BCF99] Philippe Balbiani, Jean-Francois Condotta, and Luis Fariñas del Cerro. A new tractable subclass of the rectangle algebra. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 442–447, Stockholm, Sweden, 1999. Morgan Kaufmann Publishers.

[CJJK00] David Cohen, Peter Jeavons, Peter Jonsson, and Manolis Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47(5):826–853, 2000.

[CKT91] Peter Cheeseman, Bob Kanefsky, and William M. Taylor. Where the really hard problems are. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 331–337, Chambéry, France, 1991. Morgan Kaufmann.

[CLR97] Thomas H Cormen, Charles E Leiserson, and Ronald L Rivest. *Introduction to Algorithms*. MIT press, 1997.

[Con00] Jean-Francois Condotta. The augmented interval and rectangle networks. In *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*, pages 571–579, Trento, Italy, 2000. Morgan Kaufmann Publishers.

[DMP91] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial Intelligence*, 49(1-3):61–95, 1991.

[GC97] Alfonso Gerevini and Matteo Cristani. On finding a solution in temporal constraint satisfaction problems. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI 97)*, pages 1460–1465, Nagoya, Japan, 1997.

[Güs89] Hans W. Güsgen. Spatial reasoning based on Allen's temporal logic. Technical Report ICSI TR89-049, International Computer Science Institute, 1989.

[JB98] Peter Jonsson and Christer Bäckström. A unifying approach to temporal constraint reasoning. *Artificial Intelligence*, 102(1):143–155, 1998.

[Kou01] Manolis Koubarakis. Tractable disjunctions of linear constraints: basic results and applications to temporal reasoning. *Theoretical Computer Science*, 266(1-2):311–339, 2001.

[NB95] Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *Journal of the ACM*, 42(1):43–66, 1995.

[Neb97] Bernhard Nebel. Solving hard qualitative temporal reasoning problems: Evaluating the efficiency of using the ord-horn class. *Constraints*, 1(3):175–190, 1997. Archive of C-programs: **ftp://ftp.informatik.uni-freiburg.de/documents/papers/ki/allen-csp-solving.programs.tar.gz**.

[PS82] C H Papadimitriou and K Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, New Jersey, 1982.

[PS99] Arun K. Pujari and Abdul Sattar. A new framework for reasoning about points, intervals and durations. In *Proceedings of the*

Sixteenth International Joint Conference on Artificial Intelligence (IJCAI 99), pages 1259–1267, Stockholm, Sweden, 1999.

[Vai87] Pravin M. Vaidya. An algorithm for linear programming which requires $O(((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 29–38, New York City, 1987.