Tractable Disjunctions of Linear Constraints*

Manolis Koubarakis

Dept. of Computation
UMIST
P.O. Box 88
Manchester M60 1QD
United Kingdom
manolis@co.umist.ac.uk
http://www.co.umist.ac.uk/~manolis

Abstract

We study the problems of deciding consistency and performing variable elimination for disjunctions of linear inequalities and inequations with *at most one* inequality per disjunction. This new class of constraints extends the class of generalized linear constraints originally studied by Lassez and McAloon. We show that deciding consistency of a set of constraints in this class can be done in polynomial time. We also present a variable elimination algorithm which is similar to Fourier's algorithm for linear inequalities. Finally, we use these results to improve the best known temporal reasoning algorithms for the Ord-Horn subclass of Allen's interval formalism. This answers an open question posed by Nebel and Bürckert.

1 Introduction

Linear constraints over the reals have recently been studied in depth by researchers in constraint logic programming (CLP) and constraint databases (CDB) [JM94, KKR95, Kou94c]. Two very important operations in CLP and CDB systems are deciding consistency of a set of constraints, and performing variable elimination. Subclasses of linear constraints over the reals have also been considered in temporal reasoning [DMP91, Kou92, Kou94a, Kou95, NB95]. Important operations in temporal reasoning applications are the following: (i) deciding consistency of a set of binary temporal constraints, (ii) performing variable elimination, and (iii) computing the strongest feasible constraints between every pair of variables.

Disjunctions of linear constraints over the reals are important in many applications [JM94, DMP91, Kou92, Kou94a, Kou94b, Kou95, NB95]. The problem of deciding consistency for an arbitrary set of disjunctions of linear constraints is NP-complete [Son85]. It is therefore interesting to discover classes of disjunctions of linear constraints for which consistency can be decided in PTIME. In [LM89a], Lassez and McAloon have studied the class of *generalized linear constraints* which

*This is a revised and extended version of a paper with the same title which appears in the Proceedings of the 2nd International Conference on Principles and Practice of Constraint Programming (CP96), Cambridge, MA, August 19-22, 1996. Lecture Notes in Computer Science, Vol. 1118, pages 297-307.

includes linear inequalities (e.g., $2x_1 + 3x_2 - 5x_3 \leq 6$) and disjunctions of linear inequations[1] (e.g., $2x_1 + 3x_2 - 4x_3 \neq 4 \ \vee \ x_2 + x_3 + x_5 \neq 7$). Among other things, they have shown that the consistency problem for this class can be solved in PTIME.

[Kou92, IvH93, Imb93, Imb94] have studied the problem of variable elimination for generalized linear constraints. The basic algorithm for variable elimination has been discovered independently in [Kou92] and [Imb93], but [Kou92] has used the result only in the context of temporal constraints. The basic algorithm is essentially an extension of Fourier's elimination algorithm [Sch86] to deal with disjunctions of inequations. If S is a set of constraints, let $|S|$ denote its cardinality. Let $C = I \cup D_n$ be a set of generalized linear constraints, where I is a set of inequalities and D_n is a set of disjunctions of inequations. If we eliminate m variables from C using the basic algorithm proposed by Koubarakis and Imbert then the resulting set contains $O(|I|^{2^m})$ inequalities and $O(|D_n||I|^{2^{m+1}})$ disjunctions of inequations. A lot of these constraints are redundant. Imbert has studied this problem in more detail and presented more advanced algorithms that eliminate redundant constraints [Imb93, Imb94].

In this paper we go beyond the above work on generalized linear constraints. Our contributions can be summarized as follows:

- We extend the class of generalized linear constraints to include disjunctions with an unlimited number of inequations and *at most one* inequality per disjunction. For example:

  $$3x_1 + x_5 - 4x_3 \leq 7 \ \vee \ 2x_1 + 3x_2 - 4x_3 \neq 4 \ \vee \ x_2 + x_3 + x_5 \neq 7$$

  The resulting class will be called the class of *Horn* constraints since there seems to be some analogy with Horn clauses. We show that deciding consistency can still be done in PTIME for this class (Theorem 3.4). This result has also been obtained independently by Jonsson and Bäckström [Jon96]. We also extend the basic variable elimination algorithm of [Kou92, Imb93] for this new class of constraints.

- We study a special class of Horn constraints, called *Ord-Horn constraints*, originally introduced in [NB95]. This class is important for temporal reasoning based on the *Ord-Horn class* of interval relations expressible in Allen's formalism [All83, NB95]. Our results allow us to improve the best known algorithms for *consistency checking* and *computing the strongest feasible constraints* for this class. This answers an open problem of [NB95].

The paper is organized as follows. Section 2 introduces the basic concepts needed for the developments of this paper. Section 3 presents the algorithm for deciding consistency. Section 4 presents the algorithm for variable elimination. Section 5 presents our results for the class of Ord-Horn constraints. Finally, Section 6 discusses future research.

2 Preliminaries

We consider the n-dimensional Euclidean space $\mathcal{R}^n$. A *linear constraint* over $\mathcal{R}^n$ is an expression $a_1x_1 + \cdots + a_nx_n \sim b$ where $a_1, \ldots, a_n, b$ are integers, $x_1, \ldots, x_n$ are variables ranging over the real numbers, and $\sim$ is $\leq, =$ or $\neq$. Depending on what $\sim$ is, we will distinguish linear constraints into *inequalities* (e.g. $2x_1 + 3x_2 - 5x_3 \leq 6$), *equations* (e.g., $2x_1 + 3x_2 - 5x_3 = 6$) and *inequations* (e.g., $2x_1 + 3x_2 - 5x_3 \neq 6$).

Let us now define the class of constraints that we will consider.

[1] Some people prefer the term disequations [Imb94].

Definition 2.1 A *Horn constraint* is a disjunction $d_1 \vee \cdots \vee d_n$ where each $d_i,\ i = 1, \ldots, n$ is a weak linear inequality or a linear inequation, and the number of inequalities among $d_1, \ldots, d_n$ does not exceed one. If there are no inequalities then a Horn constraint will be called *negative*. Otherwise it will be called *positive*. Horn constraints of the form $d_1 \vee \cdots \vee d_n$ with $n \geq 2$ will be called *disjunctive*.

Example 2.1 The following are examples of Horn constraints:

$$3x_1 + x_5 - 3x_4 \leq 10,\ x_1 + x_3 + x_5 \neq 7,$$
$$3x_1 + x_5 - 4x_3 \leq 7 \vee 2x_1 + 3x_2 - 4x_3 \neq 4 \vee x_2 + x_3 + x_5 \neq 7,$$
$$4x_1 + x_3 \neq 3 \vee 5x_2 - 3x_5 + x_4 \neq 6$$

The first and the third constraint are positive while the second and the fourth are negative. The third and fourth constraint are disjunctive.

According to the above definition weak inequalities are positive Horn constraints. Sometimes we will find it more convenient to consider inequalities separately from positive disjunctive Horn constraints. If d is a positive disjunctive Horn constraint then

$$d \equiv \neg(E \wedge i)$$

where E is a conjunction of equations and i is an inequality. We will often use this notation for positive Horn constraints.

Notice that we do not need to introduce strict inequalities in the above definition. A strict inequality like $x_1 + x_2 + x_3 < 5$ can be equivalently written as follows:

$$x_1 + x_2 + x_3 \leq 5,\ x_1 + x_2 + x_3 \neq 5$$

Similarly, the constraint $x_1 + x_2 + x_3 < 5 \vee \phi$ where ϕ is a disjunction of inequations can be equivalently written as the conjunction of the following constraints:

$$x_1 + x_2 + x_3 \leq 5 \vee \phi,\ x_1 + x_2 + x_3 \neq 5 \vee \phi$$

A similar observation is made in [NB95].

Negative Horn constraints have been considered before in [LM89a, LM89b, Kou92, IvH93, Imb93, Imb94, Kou95]. Nebel and Bürckert have studied the class of *Ord-Horn constraints* in the context of qualitative interval reasoning [NB95]. Ord-Horn constraints form a proper subclass of Horn constraints, and will be considered in detail in Section 5. Horn constraints have also been studied by Jonsson and Bäckström [Jon96] who discovered independently the result discussed in Section 3.

We will now present some standard definitions.

Definition 2.2 Let C be a set of constraints in variables $x_1, \ldots, x_n$. The *solution set* of C, denoted by $Sol(C)$, is:

$$\{(r_1, \ldots, r_n) \in \mathcal{R}^n : \text{ for every } c \in C,\ (r_1, \ldots, r_n) \text{ satisfies } c\}$$

Each member of $Sol(C)$ is called a *solution* of C. A set of constraints is called *consistent* if its solution set is non-empty. We will use the same notation, $Sol(\cdot)$, for the solution set of a single constraint.

Remark 2.1 In the rest of the paper we will usually consider one or more sets of constraints e.g., $C_1, \ldots, C_m$ in variables $x_1, \ldots, x_n$. In this case we will always regard $Sol(C_i)$ a subset of $\mathcal{R}^n$ even though C_i might contain less than n variables. For example, if we happen to deal with

$$C_1 = \{x_1 \leq x_2,\ x_2 \leq x_3,\ x_3 \neq x_4\} \text{ and } C_2 = \{x_1 \leq x_2,\ x_2 \leq x_3\}$$

we may write $Sol(C_1) \subset Sol(C_2)$ where $Sol(C_1), Sol(C_2) \subset \mathcal{R}^4$. Similarly, we may write $Sol(C_2) \subset Sol(x_1 \leq x_2)$.

We will also use the alternative notation $Sol^*(\cdot)$. If C is a set of constraints, $Sol^*(C)$ will always be regarded a subset of $\mathcal{R}^k$ where k is the number of variables of C (independently of any other constraint set considered at the same time). This notation will come handy in Section 4 where we study variable elimination.

Definition 2.3 Let C_1 and C_2 be sets of constraints in the same set of variables. C_1 will be called *equivalent* to C_2 if $Sol(C_1) = Sol(C_2)$. A constraint c *logically follows* from a set of constraints C, denoted by $C \models c$, if every solution of C satisfies c.

We will now present some concepts of convex geometry [Sch86, Gru67] that will enable us to study the geometric aspects of the constraints considered. We will take our definitions from [LM89a]. If V is a subspace of the n-dimensional Euclidean space $\mathcal{R}^n$ and p a vector in $\mathcal{R}^n$ then the translation $p + V$ is called an *affine space*. The intersection of all affine spaces that contain a set S is an affine space, called the *affine closure* of S and denoted by $Aff(S)$. If e is a linear equation then the solutions set of e is called a *hyperplane*. In $\mathcal{R}^3$ the hyperplanes are the planes. In $\mathcal{R}^2$ the hyperplanes are the straight lines. A hyperplane is an affine space and every affine space is the intersection of a finite number of hyperplanes. If E is a set of equalities then $Sol(E)$ is an affine space. If i is a linear inequality then the solution set of i is called a *half-space*. If I is a set of inequalities then $Sol(I)$ is the intersection of a finite number of half-spaces, and is called a *polyhedral* set.

A set $S \subseteq \mathcal{R}^n$ is called *convex* if the line segment joining any pair of points in S is included in S. Affine subspaces of $\mathcal{R}^n$ are convex. Half-spaces are convex. Also, polyhedral sets are convex.

If d is a negative Horn constraint then the solution set of d is $Sol(d) = \mathcal{R}^n \setminus Sol(\neg d)$. The constraint $\neg d$ is a conjunction of equations thus $Sol(\neg d)$ is an affine space. If $\neg d$ is inconsistent then d is equivalent to $true$ (e.g., $x \neq 2 \vee x \neq 3$). In the rest of the paper we will ignore negative Horn constraints that are equivalent to $true$.

If d is a positive disjunctive Horn constraint of the form $\neg(E \wedge i)$ then $Sol(d) = \mathcal{R}^n \setminus Sol(\neg d)$. The constraint $\neg d$ is a conjunction $E \wedge i$ where E is a conjunction of equations and i is a strict inequality. If $E \equiv true$ then d is essentially a weak inequality $\neg i$ (e.g., $x \neq x \vee y \leq 5 \equiv y \leq 5$). If $E \wedge i$ is inconsistent then its corresponding Horn constraint d is equivalent to $true$ (e.g., $x \neq 2 \vee x \leq 3 \equiv true$). If $E \wedge i$ is consistent and $Sol(E) \subseteq Sol(i)$ then $d \equiv \neg E$, so d is actually a negative Horn constraint (e.g., $x \neq 2 \vee y \neq 2 \vee x \geq 3 \equiv x \neq 2 \vee y \neq 2$). If $E \wedge i$ is consistent and $Sol(E) \not\subseteq Sol(i)$ then its solution set will be called a *half affine space*. In $\mathcal{R}^3$ the half affine spaces are half-lines or half-planes. For example, $z = 2 \wedge x > 0$ is a half plane. In the rest of the paper we will ignore positive disjunctive Horn constraints equivalent to a weak inequality, a negative Horn constraint or $true$.

3 Deciding Consistency

[LM89a] showed that negative Horn constraints can be treated independently of one another for the purposes of deciding consistency. The following is one of their basic results.

Theorem 3.1 Let $C = I \cup D_n$ be a set of constraints where I is a set of linear inequalities and D_n is a set of negative Horn constraints. Then C is consistent if and only if I is consistent, and for each $d \in D_n$ the set $I \cup \{d\}$ is consistent.

Whether a set of inequalities is consistent or not, can be decided in PTIME using Kachian's linear programming algorithm [Sch86]. We can also detect in PTIME whether $I \cup \{d\}$ is consistent by simply running Kachian's algorithm $2n$ times to decide whether I implies every equality e in the conjunction of n equalities $\neg d$. In other words, deciding consistency in the presence of negative Horn contraints can be done in PTIME.[2]

Is it possible to extend this result to the case of positive disjunctive Horn constraints? In what follows, we will answer this question affirmatively. Let us start by pointing out that the independence property of negative Horn constraints does *not* carry over to positive ones.

Example 3.1 Let $I = \{x \geq 1, x \leq 5, y = 3\}$. The constraint sets

$$I \cup \{\neg(y = 3 \ \wedge \ x > 1)\} \ \text{ and } \ I \cup \{\neg(y = 3 \ \wedge \ x = 1)\}$$

are consistent. But the set $I \cup \{\neg(y = 3 \ \wedge \ x > 1), \ \neg(y = 3 \ \wedge \ x = 1)\}$ is inconsistent.

Fortunately, there is still enough structure available in our problem which we can exploit to come up with a PTIME consistecy checking algorithm. Let $C = I \cup D_p \cup D_n$ be a set of constraints where I is a set of inequalities, D_p is a set of positive *disjunctive* Horn constraints, and D_n is a set of negative Horn constraints. Intuitively, the solution set of C is empty only if the polyhedral set defined by I is *covered* by the affine spaces and half affine spaces defined by the Horn constraints.

The algorithm Consistency shown in Figure 1 proceeds as follows. Initially, we check whether I is consistent. If this is the case, then we proceed to examine whether $Sol(I)$ can be covered by $Sol(\{\neg d : d \in D_p \cup D_n\})$. To verify this, we make successive passes over $D_p \cup D_n$. In each pass, we carry out two checks. The *first check* discovers whether there is any positive Horn constraint $d \equiv \neg(E \ \wedge \ i)$ such that $Sol(I)$ is included in the affine space defined by E. If this is the case then d is discarded and I is updated to reflect the part possibly "cut off" by d. The resulting solution set $Sol(I)$ is still a polyhedral set. An inconsistency can arise if $Sol(I)$ is reduced to $\emptyset$ by successive "cuts". In each pass we also check whether there is an affine space (represented by the negation of a negative Horn constraint) which covers $Sol(I)$. In this case there is an inconsistency as well. The algorithm stops when there are no more affine spaces or half affine spaces that pass the two checks. In this case C is consistent.

Let us now prove the correctness of algorithm Consistency. First, we will need a few technical lemmas. The first two lemmas show that the sets resulting from successive "cuts" inflicted on $Sol(I)$ by positive Horn constraints passing the first check of the algorithm are indeed polyhedral. The lemmas also give a way to compute the inequalities defining these sets.

Lemma 3.1 Let I be a set of inequalities and $\neg(E \ \wedge \ i)$ be a positive disjunctive Horn constraint such that $Sol(I) \subseteq Sol(E)$. Then $Sol(I \wedge \neg(E \wedge i)) = Sol(I \wedge \neg i)$.

Proof: Let $\overline{x} \in Sol(I \ \wedge \ \neg(E \ \wedge \ i))$. Then $\overline{x} \in Sol(I)$ and $\overline{x} \in Sol(\neg(E \ \wedge \ i))$. If $\overline{x} \in Sol(\neg(E \ \wedge \ i))$ then $\overline{x} \in Sol(\neg E)$ or $\overline{x} \in Sol(\neg i)$. But $Sol(I) \cap Sol(\neg E) = \emptyset$ because $Sol(I) \subseteq Sol(E)$. Therefore, $\overline{x} \in Sol(I)$ and $\overline{x} \in Sol(\neg i)$. Equivalently, $\overline{x} \in Sol(I \ \wedge \ \neg i)$. The other direction of the proof is trivial. ■

[2] The exact algorithm that Lassez and McAloon give in [LM89a] is different but this is not significant for the purposes of this paper.

Algorithm CONSISTENCY
Input: A set of constraints $C = I \cup D_p \cup D_n$.
Output: "consistent" if C is consistent. Otherwise "inconsistent".
Method:

If I is inconsistent **then return** "inconsistent"

Repeat
  $Done \leftarrow true$
  For each $d \in D_p \cup D_n$ **do**
    If $d \equiv \neg(E \wedge i) \in D_p$ and $Sol(I) \subseteq Sol(E)$ **then**
      $I \leftarrow I \wedge \neg i$
      If I is inconsistent **then return** "inconsistent"
      $Done \leftarrow false$
      Remove d from D_p
    Else If $d \in D_n$ and $Sol(I) \subseteq Sol(\neg d)$ **then**
      Return "inconsistent"
    End If
  End For
Until $Done$

Return "consistent"

Figure 1: Deciding consistency of a set of Horn constraints

Lemma 3.2 Let I be a set of inequalities and $d_k \equiv \neg(E_k \wedge i_k),\ k = 1, \ldots, m$ be a set of positive disjunctive Horn constraints such that $Sol(I) \subseteq Sol(E_1)$ and

$$Sol(I \wedge \bigwedge_{k=1}^{l} \neg i_k) \subseteq Sol(E_{l+1}) \quad \text{for} \quad l = 1, \ldots, m-1.$$

Then

$$Sol(I \wedge \bigwedge_{k=1}^{m} d_k) = Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k).$$

Proof: The proof is by induction on m. The base case $m = 1$ follows from Lemma 3.1. For the inductive step, let us assume that the lemma holds for $m-1$ Horn constraints. Then

$$Sol(I \wedge \bigwedge_{k=1}^{m} d_k) = Sol(I \wedge \bigwedge_{k=1}^{m-1} d_k) \cap Sol(d_m) =$$

$$Sol(I \wedge \bigwedge_{k=1}^{m-1} \neg i_k) \cap Sol(d_m) = Sol((I \wedge \bigwedge_{k=1}^{m-1} \neg i_k) \wedge d_m)$$

using the inductive hypothesis.
The assumptions of this lemma and Lemma 3.1 imply that

$$Sol((I \wedge \bigwedge_{k=1}^{m-1} \neg i_k) \wedge d_m) = Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k).$$

Thus

$$Sol(I \wedge \bigwedge_{k=1}^{m} d_k) = Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k).$$

■

The following lemmas show that if there are Horn constraints that do not pass the two checks of algorithm CONSISTENCY then the affine spaces or half affine spaces corresponding to their negations cannot cover the polyhedral set defined by the inequalities.

Lemma 3.3 Let S be a convex set of dimension d and suppose that $S_1, \ldots S_n$ are convex sets of dimension $d_i < d,\ i = 1, \ldots, n$. Then $S \not\subseteq \bigcup_{i=1}^{n} S_i$.

Proof: See Lemma 2 of [LM89a]. ■

Lemma 3.4 Let I be a consistent set of inequalities and $d_k \equiv \neg(E_k \ \wedge \ i_k),\ k = 1, \ldots, m$ be a set of Horn constraints such that $Sol(I) \not\subseteq Sol(E_k)$ and $Sol(I) \cap Sol(E_k) \neq \emptyset$ for all $k = 1, \ldots, m$. Then $Sol(I) \not\subseteq \bigcup_{k=1}^{m} Sol(\neg d_k)$.

Proof: The proof is very similar to the proof of Theorem 1 of [LM89a].

Since $Sol(I) \not\subseteq Sol(E_k)$ then $Aff(Sol(I)) \not\subseteq Sol(E_k)$. This means that $Sol(E_k) \cap Aff(Sol(I))$ is an affine space of strictly lower dimension than $Aff(Sol(I))$. Then $Sol(E_k) \cap Sol(I)$ is of strictly lower dimension than $Sol(I)$ since the dimension of $Sol(I)$ is equal to that of $Aff(Sol(I))$. Thus from Lemma 3.3, $Sol(I) \not\subseteq \bigcup_{k=1}^{m} Sol(E_k)$. Notice now that $Sol(\neg d_k) \subseteq Sol(E_k)$ for all $k = 1, \ldots, m$. Therefore $\bigcup_{k=1}^{m} Sol(\neg d_k) \subseteq \bigcup_{k=1}^{m} Sol(E_k)$. We can now conclude that $Sol(I) \not\subseteq \bigcup_{k=1}^{m} Sol(\neg d_k)$.
■

The following theorems demonstrate that the algorithm CONSISTENCY is correct and can be implemented in PTIME.

Theorem 3.2 If algorithm CONSISTENCY returns "inconsistent" then its input C is inconsistent.

Proof: If the algorithm returns "inconsistent" in its first line then I, and therefore C, is inconsistent.

If the algorithm returns "inconsistent" in the third if-statement then there are positive Horn constraints $d_k \equiv \neg(E_k \ \wedge \ i_k),\ k = 1, \ldots, m \leq |D_p|$ such that the assumptions of Lemma 3.2 hold for I and $d_1, \ldots, d_m$. Therefore

$$Sol(I \ \wedge \ \bigwedge_{k=1}^{m} d_k) = Sol(I \ \wedge \ \bigwedge_{k=1}^{m} \neg i_k) = \emptyset.$$

Consequently, $Sol(C) = \emptyset$ because $Sol(C) \subseteq Sol(I \ \wedge \ \bigwedge_{k=1}^{m} d_k)$.

If the algorithm returns "inconsistent" in the fourth if-statement then there are positive Horn constraints $d_1, \ldots, d_n \in D_p$ and negative constraint $d_{m+1} \in D_n$ such that the assumptions of Lemma 2 hold for I and $d_1, \ldots, d_m$, and

$$Sol(I \ \wedge \ \bigwedge_{k=1}^{m} d_k) \subseteq Sol(\neg d_{m+1}).$$

But then

$$Sol(C) \subseteq Sol(I \ \wedge \ \bigwedge_{k=1}^{m+1} d_k) = Sol(I \ \wedge \ \bigwedge_{k=1}^{m} d_k) \cap Sol(d_{m+1}) = \emptyset.$$

■

Theorem 3.3 If algorithm CONSISTENCY returns "consistent" then its input C is consistent.

Proof: If the algorithm returns "consistent" then I is consistent. Let $d_1, \ldots, d_m$ be the positive Horn constraints removed from $D_p \cup D_n$ by the algorithm, and $d_{m+1}, \ldots, d_n$ be the remaining Horn constraints. Then

$$Sol(C) = Sol(I \wedge \bigwedge_{k=1}^{n} d_k) = Sol(I \wedge \bigwedge_{k=1}^{m} d_k) \setminus \bigcup_{k=m+1}^{n} Sol(\neg d_k) =$$

$$Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k) \setminus \bigcup_{k=m+1}^{n} Sol(\neg d_k).$$

Notice that $Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k) \neq \emptyset$ otherwise the algorithm outputs "inconsistent" in Step 2. Also, $Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k) \not\subseteq Sol(E_k)$ for all $k = m+1, \ldots, n$ otherwise the algorithm would have removed d_k from $D_p \cup D_n$.

Without any loss of generality we can also assume that

$$Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k) \cap Sol(E_k) \neq \emptyset$$

for all $k = m+1, \ldots, n$ (if this does not hold for constraint d_k, this constraint can be discarded without changing $Sol(C)$). From Lemma 3.4 we can now conclude that $Sol(I \wedge \bigwedge_{k=1}^{m} \neg i_k) \not\subseteq \bigcup_{k=m+1}^{n} Sol(\neg d_k)$. Therefore $Sol(C) \neq \emptyset$. ■

Theorem 3.4 The algorithm Consistency can be implemented in PTIME.

Proof: It is not difficult to see that the algorithm can be implemented in PTIME. The consistency of I can be checked in PTIME using Khachiyan's algorithm for linear programming [Sch86]. The test $Sol(I) \subseteq Sol(E)$ can be verified by checking whether every equation e in the conjunction E is implied by I. This can be done in PTIME using Khachiyan's algorithm $2n$ times where n is the number of equations in E. In a similar way one can implement the test $Sol(I) \subseteq Sol(\neg d)$ in PTIME when d is a negative Horn constraint. ■

We have just showed that the consistency of a set of Horn constraints can be determined in PTIME. This is an important result with potential applications in any CLP or CDB system dealing with linear constraints [JM94, KKR95, Kou94c]. We will now turn our attention to the problem of eliminating one or more variables from a given set of Horn constraints.

4 Variable Elimination

In this section we study the problem of variable elimination for sets of Horn constraints. The algorithm VarElimination, shown in Figure 2, eliminates a given variable x from a set of Horn constraints C. More variables can be eliminated by successive applications of VarElimination.

The first step of VarElimination rewrites every constraint containing variable x in a convenient form: each constraint involving x is rewritten as $x \neq A \vee \phi$ or $x \leq U \vee \phi$ or $L \leq x \vee \phi$ where U, L, A and ϕ do *not* contain x. Then the algorithm proceeds to eliminate x using a procedure similar to the one presented in [Kou92, Imb93] for the case when we only have inequalities and negative Horn constraints. Notice that VarElimination does *not* consider inequalities separately from positive disjunctive Horn constraints (as algorithm Consistency did in Section 3).

Theorem 4.1 The algorithm VarElimination is correct.

Algorithm VarElimination
Input: A set of Horn constraints C in variables X, and a variable $x \in X$ to be eliminated from C.
Output: A set of Horn constraints C' in variables $X \setminus \{x\}$ such that $Sol^*(C') = Projection_{X\setminus\{x\}}(Sol^*(C))$.
Method:

Step 1:
For each constraint $c \in C$ **do**
  If c contains a disequation involving x **then**
    Rewrite c as $x \neq A \vee \phi$ such that ϕ and A do not contain x
  ElseIf c contains an inequality involving x **then**
    Rewrite c as $x \leq U \vee \phi$ or $L \leq x \vee \phi$ such that U, L and ϕ do not contain x
  EndIf
EndFor

Step 2:
$C' := \emptyset$
For each pair of positive Horn constraints $x \leq U \vee \phi_1$ and $L \leq x \vee \phi_2$ **do**
  Add $L \leq U \vee \phi_1 \vee \phi_2$ to C'
End For

Step 3:
For each pair of positive Horn constraints $x \leq U \vee \phi_1$ and $L \leq x \vee \phi_2$ **do**
  For each Horn constraint $x \neq A \vee \phi$ **do**
    Add $A \neq L \vee A \neq U \vee \phi \vee \phi_1 \vee \phi_2$ to C'
  End For
End For

Step 4:
Add each constraint not containing x to C'

Return C'

Figure 2: A variable elimination algorithm

Proof: Let the variables of C be $X = \{x, x_2, \ldots, x_n\}$. If $(x^0, x_2^0, \ldots, x_n^0) \in \mathcal{R}^n$ is an element of $Sol^*(C)$ then it can be easily seen that it is also an element of $Sol^*(C')$.

Conversely, take $(x_2^0, \ldots, x_n^0) \in \mathcal{R}^{n-1} \cap Sol^*(C')$ and consider the set $C(x, x_2^0, \ldots, x_n^0)$. If this set is simplified by removing constraints equivalent to true, parts of disjunctions equivalent to false, and redundant constraints then

$$C(x, x_2^0, \ldots, x_n^0) = \{l^0 \leq x, x \leq u^0\} \cup \{x \neq a_k^0,\ k = 1, \ldots, \lambda\}.$$

Let us now assume (by contradiction) that there is no value $x^0 \in \mathcal{R}^n$ such that $(x^0, x_2^0, \ldots, x_n^0) \in Sol^*(C)$. This can happen only under the following cases:

1. $u^0 < l^0$. In this case inequalities $x \leq u^0$ and $l^0 \leq x$ come from positive Horn constraints rewritten as $x \leq u \vee \phi_1$ and $l \leq x \vee \phi_2$ in Step 1 of the algorithm. Thus
$$\phi_1(x_2^0, \ldots, x_n^0) \equiv \phi_1(x_2^0, \ldots, x_n^0) \equiv false$$
otherwise these constraints would have been discarded from $C(x, x_2^0, \ldots, x_n^0)$ during its simplification. But because $l \leq u \vee \phi_1 \vee \phi_2 \in C'$ and $(x_2^0, \ldots, x_n^0) \in Sol^*(C')$ then $l^0 \leq u^0$. Contradiction!

2. $l^0 = u^0 = a_j^0,\ 1 \leq j \leq \lambda$. With reasoning similar to the above, we can show that this case is also impossible.

Finally, we can conclude that there exists a value $x^0 \in \mathcal{R}$ such that

$$(x, x_2^0, \ldots, x_n^0) \in Sol^*(C).$$

■

Let C be a set of Horn constraints. Eliminating m variables from C with repeated applications of the above algorithm will result in a set with $O(|C|^{2^m})$ Horn constraints. Many of these contraints will be redundant; it is therefore important to extend this work with efficient redundancy elimination algorithms that can be used together with VarElimination.

This section concludes our study of the basic reasoning problems concerning Horn constraints. We will now turn our attention to a suclass of Horn constraints with important applications to temporal reasoning.

5 Reasoning with Ord-Horn Constraints

This section studies a special class of Horn constraints, called Ord-Horn constraints, originally introduced in [NB95]. This class is important in interval based temporal reasoning [All83] as we will immediately show below.

Definition 5.1 An *Ord-Horn constraint* is a Horn constraint $d_1 \vee \cdots \vee d_n$ where each $d_i,\ i = 1, \ldots, n$ is an inequality $x \leq y$ or an inequation $x \neq y$ and x and y are variables ranging over the real numbers.

Example 5.1 The following are examples of Ord-Horn constraints:

$$x_1 \leq x_2,\ x_1 \neq x_3,\ x_1 \neq x_3 \vee x_2 \neq x_4,$$
$$x_1 \leq x_2 \vee x_3 \neq x_4 \vee x_5 \neq x_6$$

The first and the last constraint are positive while the second and the third are negative.

Basic Interval Relation	Symbol	Pictorial Representation	Endpoint Constraints
$i\ before\ j$	b	iiiiiiii jjjjjjjj	$i^- < j^-,\ i^- < j^+,$ $i^+ < j^-,\ i^+ < j^+$
$j\ after\ i$	bi		
$i\ meets\ j$	m	iiiiiiii jjjjjjjjjj	$i^- < j^-,\ i^- < j^+,$ $i^+ = j^-,\ i^+ < j^+$
$j\ met\text{-}by\ i$	mi		
$i\ overlaps\ j$	o	iiiiiiii jjjjjjjjjj	$i^- < j^-,\ i^- < j^+,$ $i^+ > j^-,\ i^+ < j^+$
$j\ overlapped\text{-}by\ i$	oi		
$i\ during\ j$	d	iiiiiiii jjjjjjjjjjjjjjjj	$i^- > j^-,\ i^- < j^+,$ $i^+ > j^-,\ i^+ < j^+$
$j\ includes\ i$	di		
$i\ starts\ j$	s	iiiiiiiii jjjjjjjjjjjjjjj	$i^- = j^-,\ i^- < j^+,$ $i^+ > j^-,\ i^+ < j^+$
$j\ started\text{-}by\ i$	si		
$i\ finishes\ j$	f	iiiiiiiiii jjjjjjjjjjjjjjjj	$i^- > j^-,\ i^- < j^+,$ $i^+ > j^-,\ i^+ = j^+$
$j\ finished\text{-}by\ i$	fi		
$i\ equals\ j$	eq	iiiiiiiiiii jjjjjjjjjj	$i^- = j^-,\ i^- < j^+,$ $i^+ > j^-,\ i^+ = j^+$

Figure 3: The thirteen basic relations of Allen

In [All83], Allen introduced a calculus for reasoning about intervals in time. An *interval* is an element of the following set $\mathcal{I}$:

$$\mathcal{I} = \{(b, e) :\ b, e \in \mathcal{R} \text{ and } b < e\}$$

If i is an interval variable, i^- and i^+ will denote the endpoints of i. Allen's calculus is based on thirteen mutually exclusive binary relations which can capture all the possible ways two time intervals can be related. These *basic* relations are

$$before,\ meets,\ overlaps,\ during,\ starts,\ finishes,\ equals$$

and their inverses (*equals* is its own inverse). Figure 3 gives a pictorial representation of these relations. For reasons of brevity, we will use the symbols b, m, o, d, s, f and e to refer to the basic relations in Allen's formalism. The inverse of each relation will be denoted by the name of the relation with the suffix i (for example, the inverse of b will be denoted by bi).

Allen's calculus has received a lot of attention and has been the formalism of choice for representing qualitative interval information. Whenever the interval information to be represented is indefinite, a disjunction of some of the thirteen basic relations can be used to represent what is known. There are 2^{13} such disjunctions representing qualitative relations between two intervals. Each one of these relations will be denoted by the set of its constituent basic relations e.g., $\{b, bi, d, m\}$. The *empty* relation will be denoted by $\emptyset$, and the *universal* relation will be denoted by $\perp$. The set of all 2^{13} relations expressible in Allen's formalism will be denoted by $\mathcal{A}$ [NB95].

The following definition will be useful below.

Definition 5.2 Let $\mathcal{S}$ be a subset of $\mathcal{A}$, i and j be variables representing intervals, and $R \in \mathcal{S}$. An *$\mathcal{S}$-constraint* is any expression of the form $i\ R\ j$.

Example 5.2 If interval i denotes the time that John reads his morning newspaper and j denotes the time that he has breakfast, and we know that John never reads a newspaper while he is eating, then the $\mathcal{A}$-constraint

$$i\ \{b, bi, m, mi\}\ j$$

characterizes i and j according to the information given.

Definition 5.3 Let C be a set of $\mathcal{S}$-*constraints*. The solution set of C is:

$$Sol(C) = \{((b_1, e_1), \ldots, (b_n, e_n)) \in \mathcal{I}^n : \text{ for every } c \in C, ((b_1, e_1), \ldots, (b_n, e_n)) \text{ satisfies } c\}$$

Unfortunately, all interesting reasoning problems associated with Allen's interval calculus are NP-hard [VKvB89] therefore it is interesting to consider subsets of Allen's formalism, in the form of subsets of $\mathcal{A}$, that have better computational properties.[3] Three such subsets of $\mathcal{A}$ have received more attention:

- The set $\mathcal{C}$ which consists of all relations $R \in \mathcal{A}$ which satisfy the following condition. If i and j are intervals, $i\ R\ j$ can be equivalently expressed as a conjunction of inequalities $p_1 < p_2$ or $p_1 \leq p_2$ where p_1 and p_2 are endpoints of i and j.

  The set $\mathcal{C}$ is called the *continuous endpoint subclass* of $\mathcal{A}$ [VKvB89].

- The set $\mathcal{P}$ which consists of all interval relations $R \in \mathcal{A}$ which satisfy the following condition. If i and j are intervals, $i\ R\ j$ can be equivalently expressed as a conjunction of inequalities $p_1 < p_2$, $p_1 \leq p_2$ or inequation $p_1 \neq p_2$ where p_1 and p_2 are endpoints of i and j.

  The set $\mathcal{P}$ is called the *pointisable subclass* of $\mathcal{A}$ [VKvB89, vBC90, vB92]. Because $\mathcal{C} \subset \mathcal{P}$ the pointisable subclass is more expressive than the continuous endpoint subclass.

- The set $\mathcal{H}$ which consists of all interval relations $R \in \mathcal{A}$ which satisfy the following condition. If i and j are intervals, $i\ R\ j$ can be equivalently expressed as a conjunction of Ord-Horn constraints on the endpoints of i and j. The disjunctive Ord-Horn constraints involved in this equivalence are not arbitrary. There are at most three of them, and each one consists of *two* disjuncts of the form $i^- \ op_1 \ j^-$ and $i^+ \ op_2 \ j^+$ where op_1, op_2 is $\leq$ or $\neq$.

  The set $\mathcal{H}$ was introduced by Nebel and Bürckert and named the *Ord-Horn subclass* [NB95]. Because $\mathcal{P} \subset \mathcal{H}$ the Ord-Horn subclass is more expressive than the pointisable subclass. It consists of 868 relations i.e., it covers more than 10% of $\mathcal{A}$.

Example 5.3 The following are $\mathcal{P}$-constraints:

$$i_1\ \{d, o\}\ i_2, \quad i_3\ \{d, s\}\ i_4$$

Their equivalent endpoint constraints are:

$$i_1^- < i_1^+,\ i_2^- < i_2^+,\ i_1^- \neq i_2^-,\ i_1^- < i_2^+,\ i_1^+ > i_2^-,\ i_1^+ < i_2^+$$

$$i_3^- < i_3^+,\ i_4^- < i_4^+,\ i_3^- \leq i_4^-,\ i_3^- < i_4^+,\ i_3^+ > i_4^-,\ i_3^+ < i_4^+$$

The second $\mathcal{P}$-constraint is also a C-constraint while the first one is not. For an enumeration of $\mathcal{C}$ and $\mathcal{P}$, see [vBC90].

Example 5.4 The $\mathcal{A}$-constraint $i\ \{b, bi\}\ j$ is not an $\mathcal{H}$-constraint. The constraint

$$i_1\ \{b, d, di, o, oi, si\}\ i_2$$

is an $\mathcal{H}$-constraint which is not a $\mathcal{P}$-constraint. Its equivalent endpoint constraints are:

$$i_1^- < i_i^+,\ i_2^- < i_2^+,\ i_1^- < i_2^+,\ i_1^+ \neq i_2^-,\ i_1^+ \neq i_2^+,$$

[3] At the expense of being less expressive; no free meal here!

$$i_1^- \neq i_2^- \ \vee \ i_1^+ \geq i_2^+, \quad i_1^- \neq i_2^- \ \vee \ i_1^+ \neq i_2^+$$

An enumeration of $\mathcal{H}$ together with several related C programs has been provided by Nebel and Burckert. See [NB95] for details.

The following reasoning problems have been studied for the above subclasses [VKvB89, vBC90, vB92, LM94, NB95].

- Given a set C of $\mathcal{S}$-constraints, decide whether C is consistent.
- Given a set C of $\mathcal{S}$-constraints, determine the strongest feasible relation between each pair of interval variables i and j. The *strongest feasible relation* between two interval variables i and j is the smallest set R such that $C \models i \ R \ j$. This is the same as computing the *minimal network* corresponding to the given set of constraints.[4]

In this section we will show how our results can be used to improve the best known algorithms for the above reasoning problems in the case of the Ord-Horn subclass. We start with two theorems from [NB95].

Theorem 5.1 *Let C be a set of $\mathcal{H}$-constraints. Deciding whether C is consistent can be done in $O(n^3)$ time where n is the number of variables in C.*

Theorem 5.2 *Let C be a set of $\mathcal{H}$-constraints. Computing the feasible relations between all pairs of variables can be done in $O(n^5)$ time where n is the number of variables in C.*

We will now use the results of Section 3 to improve the complexity bounds of the above theorems.

Theorem 5.3 *Let C be a set of $\mathcal{H}$-constraints. Let n be the number of variables in C, and h be the number of constraints $(i \ R \ j) \in C$ such that $R \in \mathcal{H} \setminus \mathcal{C}$. Deciding whether C is consistent can be done in $O(\max(n^2, hn))$ time.*

Proof: First we translate C into a set of Ord-Horn constraints C'. Since $|C| = O(n^2)$, this translation can be achieved in $O(n^2)$ time. Let $C' = I \cup D_p \cup D_n$ where I is a set of inequalities, D_p is a set of positive disjunctive Horn constraints and D_n a set of negative Horn constraints. The translation of Nebel and Bürckert shows that C' contains $n_1 = 2n$ point variables and $|D_p \cup D_n| = O(h)$ [NB95].

We will use algorithm Consistency from Figure 1 to decide C'. In this case Consistency can be made to work in $O(\max({n_1}^2, |D_p \cup D_n| \, n_1))$ time as follows. Checking the consistency of I can be done in $O({n_1}^2)$ time by constructing a directed graph G corresponding to I and examining its strongly connected components [vB92]. Now notice that the statement If-Else-End-If in algorithm Consistency is executed $O(|D_p \cup D_n|)$ times. Each execution of this statement takes $O(n_1)$ time. Let us see why. If the examined constraint d is in D_p, the test $Sol(I) \subseteq Sol(E)$ amounts to checking whether the single inequality E is implied by I. This can be done in $O(n_1)$ time by examining the strongly connected components of G. Similarly, if d is in D_n, the test $Sol(I) \subseteq Sol(\neg d)$ can be done in $O(n_1)$ time. Therefore deciding whether C' is consistent can be done in $O(\max({n_1}^2, |D_p \cup D_n| \, n_1))$ time. Thus deciding whether C is consistent can be done in $O(\max(n^2, hn))$ time. ■

Theorem 5.4 *Let C be a set of $\mathcal{H}$-constraints. Let n be the number of variables in C, and h be the number of constraints $(i \ R \ j) \in C$ such that $R \in \mathcal{H} \setminus \mathcal{C}$. Computing the feasible relations between all pairs of variables can be done in $O(\max(n^4, hn^3))$ time.*

[4] We will not define minimal networks formally here. The interested reader can consult [vB92] (or many other temporal reasoning papers).

Proof: As in [NB95], we will consider all $O(n^2)$ pairs of variables in turn. For each pair we check whether each of the thirteen basic relations is consistent with the given constraints. The basic relations that satisfy this criterion form the strongest feasible relation between the pair. Using the algorithm of Theorem 5.3, each check can be done in $O(\max(n^2, hn))$ time. The bound of the theorem follows immediately. ■

In the worst case the parameter h, as specified in the above theorems, can be $O(n^2)$. However in practical applications, we expect h to be significantly less than $O(n^2)$ thus the above theorems are *real* improvements over [NB95].

6 Future Research

In future research we would like to study more advanced variable elimination algorithms for Horn constraints. The results of [Imb93, Imb94] that apply to negative Horn constraints only, should be a good starting point in this direction.

Another interesting problem, which occupies us currently, is to extend the results of [Kou95] to the pointizable subclass $\mathcal{P}$ and the Ord-Horn subclass $\mathcal{H}$ of $\mathcal{A}$. [Kou95] studied the problem of enforcing *global consistency* for sets of quantitative temporal constraints over the real numbers. In a *globally consistent* constraint set all interesting constraints are explicitly represented and the projection of the solution set on any subset of the variables can be computed by simply collecting the constraints involving these variables. An important consequence of this property is that a solution can be found by *backtrack-free* search [Fre82]. Enforcing global consistency can take an exponential amount of time in the worst case [Fre78, Coo90]. As a result it is very important to identify cases in which *local consistency*, which presumably can be enforced in polynomial time, implies global consistency [Dec92].

The class of temporal constraints considered in [Kou95] includes equalities of the form $x - y = r$, inequalities of the form $x - y \leq r$ and inequations of the form $x - y \neq r$ where x, y are variables ranging over the real numbers and r is a real constant. [Kou95] shows that *strong 5-consistency* is necessary and sufficient for achieving global consistency for this class of constraints. It also gives an algorithm which achieves global consistency in $O(Hn^4)$ where n is the number of variables and H is the number of inequations. The details of this algorithm demonstrate that there are situations where it is impossible to enforce global consistency without introducing disjunctions of inequations e.g.,

$$x_1 - y_1 \neq 5 \ \vee \ x_2 - y_2 \neq 7.$$

The results of [Kou95] can provide the basis for efficient global consistency algorithms for the pointizable subclass $\mathcal{P}$. The open question is whether one can use the results of this paper and [Kou95] to find efficient algorithms for global consistency for the ORD-Horn subclass.

References

[All83] J.F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, November 1983.

[Coo90] M.C. Cooper. An optimal k-consistency algorithm. *Artificial Intelligence*, 41(1):89–95, 1990.

[Dec92] R. Dechter. From local to global consistency. *Artificial Intelligence*, 55:87–107, 1992.

[DMP91] R. Dechter, I. Meiri, and J. Pearl. Temporal Constraint Networks. *Artificial Intelligence*, 49(1-3):61–95, 1991. Special Volume on Knowledge Representation.

[Fre78] E. Freuder. Synthesizing Constraint Expressions. *Communications of ACM*, 21(11):958–966, November 1978.

[Fre82] E. Freuder. A Sufficient Condition For Backtrack-Free Search. *Journal of ACM*, 29(1):24–32, 1982.

[Gru67] B. Grunbaum. *Convex Polytopes*. John Wiley and Sons, 1967.

[Imb93] J.-L. Imbert. Variable Elimination for Generalized Linear Constraints. In *Proceedings of the 10th International Conference on Logic Programming*, 1993.

[Imb94] J.-L. Imbert. Redundancy, Variable Elimination and Linear Disequations. In *Proceedings of the International Symposium on Logic Programming*, pages 139–153, 1994.

[IvH93] J.-L. Imbert and P. van Hentenryck. On the Handling of Disequations in CLP over Linear Rational Arithmetic. In F. Benhamou and A. Colmerauer, editors, *Constraint Logic Programming: Selected Research*, Logic Programming Series, pages 49–71. MIT Press, 1993.

[JM94] J. Jaffar and M.J. Maher. Constraint Logic Programming: A Survey. *Journal of Logic Programming*, 19-20:503–581, 1994.

[Jon96] Jonsson, P. and Bäckström, C. A Linear Programming Approach to Temporal Reasoning. In *Proceedings of AAAI-96*, 1996.

[KKR95] P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint Query Languages. *Journal of Computer and System Sciences*, 51:26–52, 1995.

[Kou92] M. Koubarakis. Dense Time and Temporal Constraints with $\neq$. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR'92)*, pages 24–35. Morgan Kaufmann, San Mateo, CA, October 1992.

[Kou94a] M. Koubarakis. Complexity Results for First-Order Theories of Temporal Constraints. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR'94)*, pages 379–390. Morgan Kaufmann, San Francisco, CA, May 1994.

[Kou94b] M. Koubarakis. Database Models for Infinite and Indefinite Temporal Information. *Information Systems*, 19(2):141–173, March 1994.

[Kou94c] M. Koubarakis. Foundations of Indefinite Constraint Databases. In A. Borning, editor, *Proceedings of the 2nd International Workshop on the Principles and Practice of Constraint Programming (PPCP'94)*, volume 874 of *Lecture Notes in Computer Science*, pages 266–280. Springer Verlag, 1994.

[Kou95] M. Koubarakis. From Local to Global Consistency in Temporal Constraint Networks. In *Proceedings of the 1st International Conference on Principles and Practice of Constraint Programming (CP'95)*, volume 976 of *LNCS*, pages 53–69, Cassis, France, September 1995. Long version to appear in Theoretical Computer Science (invited submission to a special issue dedicated to CP'95, Editors: U. Montanari and F. Rossi).

[LM89a] Jean-Louis Lassez and Ken McAloon. A Canonical Form for Generalized Linear Constraints. Technical Report RC15004 (#67009), IBM Research Division, T.J. Watson Research Center, 1989.

[LM89b] Jean-Louis Lassez and Ken McAloon. A Canonical Form for Generalized Linear Costraints. In *TAPSOFT '89, Advanced Seminar on Foundations of Innovative Software Development, Lecture Notes in Computer Science 351*, pages 19–27. Springer Verlag, 1989.

[LM94] P. Ladkin and R. Maddux. On binary constraint problems. *Journal of the ACM*, 41(3):435–469, May 1994.

[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, January 1995.

[Sch86] A. Schrijver, editor. *Theory of Integer and Linear Programming.* Wiley, 1986.

[Son85] E. Sontag. Real Addition and the Polynomial Time Hierarchy. *Information Processing Letters*, 20:115–120, 1985.

[vB92] Peter van Beek. Reasoning About Qualitative Temporal Information. *Artificial Intelligence*, 58:297–326, 1992.

[vBC90] Peter van Beek and Robin Cohen. Exact and Approximate Reasoning about Temporal Relations. *Computational Intelligence*, 6:132–144, 1990.

[VKvB89] Marc Vilain, Henry Kautz, and Peter van Beek. Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report. In D.S. Weld and J. de Kleer, editors, *Readings in Qualitative Reasoning about Physical Systems*, pages 373–381. Morgan Kaufmann, 1989.