Appears in: Proceedings of KR'94, pages 379-390.

Complexity Results for First-Order Theories of Temporal Constraints

Manolis Koubarakis[*]
Computer Science Division
Dept. of Electrical and Computer Engineering
National Technical University of Athens
Zographou 157 73
Athens GREECE

Abstract

We study the complexity of quantifier elimination and decision in first-order theories of temporal constraints. With the exception of Ladkin, AI researchers have largely ignored this problem. We consider the first-order theories of point and interval constraints over two time structures: the integers and the rationals. We show that in all cases quantifier-elimination can be done in PSPACE. We also show that the decision problem for arbitrarily quantified sentences is PSPACE-complete while for $\exists_k$ sentences it is Σ_k^p-complete. Our results must be of interest to researchers working on temporal constraints, computational complexity of logical theories, constraint databases and constraint logic programming.

1 INTRODUCTION

The study of temporal constraints has recently received much attention from the AI community [All83, LM88, Lad88, VKvB89, vBC90, DMP91, KL91, Mei91, vB92, Kou92, GS93, SD93]. Much of this work draws upon concepts and techniques from the literature of general constraint satisfaction [Mon74, Mac77, DP88]. With the exception of [Lad88], all previous research has concentrated on temporal constraints which are quantifier-free formulas in some first-order theory of time. The problems studied include deciding whether a set of constraints is consistent, computing the minimal network corresponding to a set of constraints, finding one solution, performing variable elimination and enforcing global consistency.

[Lad88] studied the first-order theory of qualitative interval-to-interval relations introduced by Allen [All83] and gave a quantifier-elimination (and decision) algorithm. This algorithm works by reducing quantifier-elimination in the given theory to quantifier-elimination in the first-order theory of rational order. The only deficiency of Ladkin's work is that he does not discuss the complexity of his algorithm. He simply assumes that quantifier-elimination in the theory of rational order will be done using standard methods and cites [CK73] as a reference. However [CK73] (as well as any other standard logic text-book) does not take complexity issues very seriously; thus their algorithms are usually inefficient.

In this paper we follow the line of research initiated by [Lad88] and extend it by considering more expressive languages of temporal constraints. Our contributions can be summarized as follows: DIFFERENCE LOGIC

1. We study the first-order theory of (qualitative and quantitative) point constraints over the integers. The atomic formulas (i.e., constraints) in this theory are of the form $t \sim c$ or $t_i - t_j \sim c$ where $\sim$ can be $<$, $>$ or $=$ and c is an integer. We show that elimination of quantifiers can be achieved in PSPACE (theorem 4.5) and that the decision problem is PSPACE-complete (theorem 4.1). For $\exists_k$ sentences of this theory the complexity of the decision problem is Σ_k^p-complete (theorem 4.3).
2. We also study the first-order theory of (qualitative and quantitative) point constraints over the rationals. The atomic formulas in this theory are of the form $t \sim c$ or $t_i - t_j \sim c$ where $\sim$ can be $<$, $>$ or $=$ and c is a rational. The complexity results for this theory are similar: elimination of quantifiers can be achieved in PSPACE (theorem 5.5) and the decision problem is PSPACE-complete (theorem 5.1). For $\exists_k$ sentences of this theory the complexity of the decision problem is also Σ_k^p-complete (theorem 5.4).
3. Finally, we briefly turn to the first-order theories

[*]Current Address: IC-Parc, Dept. of Computing, Imperial College, London SW7 2BZ, U.K. Email: msk@doc.ic.ac.uk

of point and interval constraints over the integers (or rationals). As a direct collorary of the above results we point out that the complexity of quantifier-elimination and decision for these theories does not change due to the introduction of interval constraints (section 6).

The above results will be important for researchers in constraint databases [KKR90] and constraint logic programming [CMT92]. They should also be interesting to the theoretical computer science community in general. This is so because the first-order theory of point constraints over the integers is a subtheory of Presburger arithmetic while the first-order theory of point constraints over the rationals is a subtheory of real addition with order.

The rest of this paper is organized as follows. In section 2 we introduce the theories of point constraints which we will consider. In section 3 we present standard quantifier elimination algorithms for these theories. In sections 4 and 5 we use these standard algorithms as a basis for the development of more sophisticated quantifier elimination algorithms. In section 6 we extend our results to theories of point and interval constraints. Finally, section 7 presents our conclusions.

2 FIRST-ORDER THEORIES OF POINT CONSTRAINTS

In this paper we study the first-order theories of point and interval constraints over two linear and unbounded time structures: the set of integers $\mathbb{Z}$ and the set of rationals $\mathbb{Q}$. In this section we consider *points* to be our only time entities. In the first theory that we consider, points will be identified with the integers. The language *diPCL* (*di*screte *P*oint *C*onstraint *L*anguage) will allow us to make stamements about points in time. diPCL is a first order language with equality defined as follows. The non-logical symbols of diPCL include a countably infinite set of (point or integer) constants, function symbol $-$ of arity 2 and predicate symbol $<$ of arity 2.

The set of *terms* of diPCL are defined as follows:

- Constants and variables are terms.
- If t_1, t_2 are variables or constants then $t_1 - t_2$ is a term.

An *atomic formula* of diPCL is a formula of the form $t \sim c$ or $c \sim t$ where $\sim$ is $<$ or $=$, and t is a term. The set of *well-formed formulas* of diPCL are defined as usual (i.e., they are built up from the atomic formulas using quantifiers and connectives). As usual, we will write $t_1 \leq t_2$ instead of $t_1 = t_2 \;\vee\; t_1 < t_2$, $t_2 > t_1$ instead of $t_1 < t_2$, and $t_1 \geq t_2$ instead of $t_1 = t_2 \vee t_1 > t_2$.

The symbols of diPCL are interpreted with respect to the fixed structure **Z** which captures our assumptions. The domain of **Z** is $\mathbb{Z}$. To each constant symbol, **Z** assigns an element of $\mathbb{Z}$. To function symbol $-$, **Z** assigns the function $-_{\mathbb{Z}}$ which is the subtraction operation for integers. To predicate symbol $<$, **Z** assigns the relation $<_{\mathbb{Z}}$ ("less than") over the integers.

We will take the theory of structure **Z** to be *the* theory of point constraints in linear, unbounded and discrete time. This theory will be denoted by *diPC*.[1]

Example 2.1 The following is a diPCL sentence:

$$(\forall x)(\forall y)(x - y \leq 5 \;\supset\; (\exists z)(x - z \leq 3 \;\wedge\; z \leq 2)).$$

Let us now develop a theory of point constraints which is exactly like diPC except that the time structure is the set of rational numbers. *Points* will again be our only time entities. This time they will be identified with the rationals. The language dePCL (*de*nse *P*oint *C*onstraint *L*anguage) will allow us to make stamements about points in time. dePCL is a first order language with equality defined as follows. The non-logical symbols of dePCL include a countably infinite set of (point or rational) constants, function symbol $-$ of arity 2 and predicate symbol $<$ of arity 2.

The set of *terms* of dePCL are defined exactly as for diPCL. An *atomic formula* of dePCL is a formula of the form $t \sim c$ or $c \sim t$ where $\sim$ is $<$ or $=$, and t is a term. The set of *well-formed formulas* of dePCL are defined as usual.

The symbols of dePCL are interpreted with respect to the fixed structure **Q** which captures our assumptions for dense time. The domain of **Q** is $\mathbb{Q}$. To each constant symbol, **Q** assigns an element of $\mathbb{Q}$. To function symbol $-$, **Q** assigns the function $-_{\mathbb{Q}}$ which is the subtraction operation over the rationals. To predicate symbol $<$, **Q** assigns the relation $<_{\mathbb{Q}}$ over the rationals.

We will take the theory of structure **Q** to be *the* theory of point constraints in linear, unbounded and dense time. This theory will be denoted by *dePC*.

3 NAIVE QUANTIFIER ELIMINATION ALGORITHMS

Let us first define our terminology. The following definition is standard [End72, CK73].

Definition 3.1 *A theory Th admits elimination of quantifiers iff for every formula ϕ there is a quantifier free formula ϕ' such that $Th \models \phi \equiv \phi'$.*

Proposition 3.1 *The theories diPC and dePC admit elimination of quantifiers.*

[1] We adhere to the following convention. If a theory is denoted by Th, the language of Th will be denoted by ThL.

Proof: Let us assume that ϕ is a formula of diPCL or dePCL. We can find a quantifier-free formula ϕ' equivalent to ϕ in the following way:

1. Compute the PNF $(Qt_1)\cdots(Qt_m)\psi(t_1,\ldots,t_m)$ of ϕ. elimination inside-out.
2. Eliminate quantifier (Qt_m) as follows. Let Q be $\exists$ and $\theta_1 \vee \cdots \vee \theta_k$ be the DNF of $\psi(t_1,\ldots,t_m)$. In the case of a dePCL formula, eliminate variable t_m from each θ_i to compute θ_i' using Fourier's algorithm.[2]
   In the case of a diPCL formula, first transform every strict inequality into a weak one and then apply Fourier's algorithm. Fourier's algorithm does not give correct results for arbitrary linear inequalities over the integers but *does work* correctly for diPCL inequalities. The resulting expression is $\theta_1' \vee \cdots \vee \theta_k'$.
   Now let us assume Q is $\forall$ and $\theta_1 \vee \cdots \vee \theta_k$ is the DNF of $\neg\psi(t_1,\ldots,t_m)$. Eliminate variable t_m from each θ_i to compute θ_i' as above. The resulting expression is $\neg(\theta_i' \vee \cdots \vee \theta_k')$.
3. Repeat step 2 to eliminate quantifiers $(Qt_{m-1}),\ldots,(Qt_1)$ to obtain a quantifier-free formula in DNF.
4. Simplify the above DNF formula as follows. Until no other change is possible, substitute $true \wedge \alpha$ by α, $true \vee \alpha$ by $true$, $false \wedge \alpha$ by $false$ and $false \vee \alpha$ by α.

The correctness of the above procedure can be verified easily. For diPC a similar method appears in [Rab77]. ■

Notation 3.1 If ϕ is a formula then $|\phi|$ denotes the length of ϕ.

Theorem 3.1 *Let ϕ be a diPCL or dePCL formula with K variables and M quantifiers. The quantifier elimination algorithm of proposition 3.1 computes a quantifier-free DNF formula equivalent to ϕ in $O(MK^2\,|\phi|^{(K(K+1))^M})$ time.*

Proof: There can be at most two types of constraints involving a single variable x or a difference $x_i - x_j$: one giving an upper bound and/or one giving a lower bound. Thus if we have K variables, we can have $K(K+1)$ types of constraints. The formula $\psi(t_1,\ldots,t_m)$ can have at most $|\psi|$ constraints of each type. Therefore the DNF form of ψ can contain $O(|\psi|^{K(K+1)})$ disjuncts and each disjunct can contain at most $K(K+1)$ constraints. Finally, after M quantifier eliminations the resulting formula can have $O(|\phi|^{(K(K+1))^M})$ disjuncts. The time to eliminate a variable from a disjunct is $O(K^2)$. The result follows easily. ■

[2] Fourier's algorithm was originally proposed for linear inequalities over the reals [Sch86]. Eliminating a variable x from a conjunction of inequalities using this algorithm can be done as follows. First all inequalities involving x are written in the form $L \le x$ or $x \le U$ where L and U do not contain x. Then for every pair of inequalities $L \le x$ and $x \le U$, x is eliminated and the resulting inequality $L \le U$ is returned. Finally, all inequalities which do not involve x are also returned.

You need to argue that the obtained q.f. formula is also in di/dePCL.

The above quantifier elimination algorithms are simple but inefficient. The problem with these algorithms is the transformation into DNF; this creates an exponential blow-up in time and space complexity. We will now turn to ~~better algorithms which avoid transformations into DNF.~~

4 AN IMPROVED QUANTIFIER ELIMINATION ALGORITHM FOR diPC

First, we present a decision procedure for sentences of diPCL. Then we use this procedure for developing a quantifier elimination algorithm for arbitrary diPCL formulas. Ferrante & Rackoff

Our starting point will be the quantifier elimination technique of [FR75]. The main point of this technique is that "given a particular theory, one gives an elimination of quantifiers procedure, analyzes it to see how *large* constants can grow, and then uses this analysis (...) to limit quantifiers to range over finite sets instead of an infinite domain" [FR75].

We start with a definition from [FR75].

Definition 4.1 Let i be an integer. We will say that i is limited by the positive integer L, denoted by $i \preceq L$, iff $|i| \le L$.

As usual, we will also write $i \succeq j$ when $j \preceq i$. The following definitions introduce additional notation.

Definition 4.2 Let ϕ be diPCL formula. Then $(\forall z \preceq L)\phi$ is a shorthand for the formula

$$(\forall z)(z \ge -L \wedge z \le L \supset \phi)$$

and $(\exists z \preceq L)\phi$ is a shorthand for

$$(\exists z)(z \ge -L \wedge z \le L \wedge \phi).$$

Definition 4.3 Let ϕ be a diPCL formula. The expression $maxabs(\phi)$ will denote the maximum absolute value of the integers appearing in ϕ.

The following lemmas are at the heart of the elimination of quantifiers.

Lemma 4.1 *Let us assume that $\bar{t}$ is a vector of variables, y is a variable, $\phi(\bar{t}, y)$ is a diPCL formula with*

SMALL WITNESS PROPERTY

K quantifiers and $\overline{\tau}$ is a vector of elements of $\mathbb{Z}$ limited by W. Then $\mathbf{Z} \models (Qy)\phi(\overline{\tau}, y)$ iff

$$\mathbf{Z} \models \boxed{(Qy \preceq 2^K(maxabs(\phi) + 1) + W + 2)}\phi(\overline{\tau}, y).$$

If $\overline{t} = ()$, the lemma holds with $W = 0$.

Proof: Let us eliminate quantifiers from ϕ using the procedure of proposition 3.1. After the first quantifier is eliminated, the integer constants in the resulting formula are limited by $2maxabs(\phi) + 1$. With a simple inductive argument, we can show that after all K quantifiers are eliminated the integer constants in the resulting formula $\phi'(\overline{t}, y)$ are limited by $2^K maxabs(\phi) + 2^K - 1$ or (more compactly) by $2^K(maxabs(\phi) + 1)$.

Now let us consider the formula $\phi'(\overline{\tau}, y)$. Every point constraint in this formula can be written in the form $y \sim c$ where $\sim$ is $<, =$ or $>$ and c is limited by $2^K(maxabs(\phi) + 1) + W + 1$. The truth value of $\phi'(\overline{\tau}, y)$ is the same for all y such that $y \succeq 2^K(maxabs(\phi)+1)+W+2$. Therefore the truth value of $(Qy)\phi'(\overline{\alpha}, \overline{\tau}, y)$ can be determined by simply determining the truth value of $\phi'(\overline{\tau}, y)$ for every integer limited by $2^K(maxabs(\phi) + 1) + W + 2$. ■

Lemma 4.2 *Let us assume that ϕ is a diPCL sentence. Let $(Q_1 t_1), \ldots, (Q_K t_K)$ be all the quantifiers of ϕ in left to right order of appearance. Then $\mathbf{Z} \models \phi$ iff $\mathbf{Z} \models \phi'$ where ϕ' is the same as ϕ except that $(Q_i t_i)$ is substituted by $(Q_i t_i \preceq 2^{K+i-2}(maxabs(\phi)+1)+i+1)$ for all $i = 1, \ldots, K$.*

Proof: The sentence ϕ can be written as $(Q_1 t_1)\psi_1(t_1)$ where ψ_1 is a formula with $K - 1$ quantifiers. Thus lemma 4.1 implies that $\mathbf{Z} \models (Q_1 t_1)\psi_1(t_1)$ iff

$$\mathbf{Z} \models (Q_1 t_1 \preceq 2^{K-1}(maxabs(\phi) + 1) + 2)\psi_1(t_1).$$

A similar inductive argument gives the result. ■

Remark 4.1 The models of computation used in the rest of this paper are deterministic and non-deterministic Turing machines with a read-only input tape, a fixed number of read/write work tapes and a write-only output tape where the head can never move left. The *time* of a computation is its length. The *space* of a computation is the number of cells visited on the *work tapes*. Precise definitions of these notions and the associated time and space complexity classes can be found in the standard literature [Joh90].

If $i > 1$ then $\log i$ will denote $\lceil \log_2 i \rceil$. By convention we assume that $\log 1 = 1$.

Formulas will be represented by the standard binary encoding. Integers are written in binary. Rationals are written as fractions of two integers. Variables are chosen from v_0, v_1, v_{10} etc. (i.e., subscripts are written in binary). Finally, the size of all integer constants in a formula ϕ is assumed to be less than $\log |\phi|$.

The following theorem gives the exact complexity of the decision problem for diPC.

Theorem 4.1 *Let ϕ be a diPCL sentence. The problem of deciding whether $\mathbf{Z} \models \phi$ is PSPACE-complete.*

Proof: *Lower bound.* PSPACE-hardness follows from a straightfoward reduction from Quantified Boolean Formulas [Sto77]. $\rightarrow$ or APTIME

Upper bound. It easy to write a recursive algorithm diPC-Eval which can be used to decide whether $\mathbf{Z} \models \phi$ using lemma 4.2. diPC-Eval can be implemented by a deterministic Turing machine which uses a stack for storing the activation record of each recursive call.

Whenever diPC-Eval is called with argument a quantified subformula ϕ' of ϕ, it determines the truth value of ϕ' by cycling through integers limited by $2^{2K-2}(maxabs(\phi) + 1) + K + 1$. Such integers can be written using $O(|\phi|)$ space. Therefore diPC-Eval needs $O(|\phi|^2)$ space for keeping track of the current assignment to the quantified variables of ϕ. No more space is required for bookkeeping. Thus deciding if $\mathbf{Z} \models \phi$ can be done in $O(|\phi|^2)$ space. ■

The reader is invited to compare the above theorem with the following result. $\rightarrow$ Berman

Theorem 4.2 *[Ber80] The problem of deciding a sentence of length n of Presburger Arithmetic is complete for the class $\bigcup_{k>0} TA[2^{2^{n^k}}, n]$.*

The class $TA[t(n), a(n)]$ is the set of all problems solvable by alternating Turing machines using at most $t(n)$ time and $a(n)$ alternations on inputs of length n [Ber80, Joh90].

Let us now consider diPCL sentences with a fixed number of alternations of quantifiers.

Definition 4.4 Let $\mathcal{L}$ be a first-order language and k be a fixed natural number. A $\exists_k$ (resp. $\forall_k$) formula of $\mathcal{L}$ is a formula in prenex normal form with k alternations of quantifiers beginning with an existential (resp. universal) quantifier.

Theorem 4.3 *Let ϕ be a $\exists_k$ sentence of diPCL. The problem of deciding whether $\mathbf{Z} \models \phi$ is Σ_k^p-complete.*

A corresponding Π_k^p bound can be established for $\forall_k$ sentences of diPCL.

It might be interesting to compare the above result with the following result of [RL78].

Theorem 4.4 *There exist constants $d, e > 0$ such that a deterministic Turing machine can decide a sentence with length $n > 4$ and at most K alternations of quantifiers, in the first order theory of integer addition with order, in $O(2^{dn^{K+4}})$ space and $O(2^{2^{en^{K+4}}})$ time.*

4.1 OPEN FORMULAS

We will now present a quantifier elimination algorithm for open formulas of diPCL. Let us assume we have a formula $\phi(\overline{t})$ of diPCL such that $\overline{t}$ is the vector of all the free variables of ϕ. A quantifier free formula ϕ' equivalent to ϕ can be found in the following way. At first, using the techniques presented above, we estimate how large integer constants can grow in the constraints of a quantifier-free formula equivalent to ϕ. Then we use this information to construct a finite partition of the space $\mathbb{Z}^{|\overline{t}|}$ into *regions* with the following properties:

1. Every region can be represented by a conjunction of atomic diPCL formulas.
2. The truth value of the sentence $\phi(\overline{\tau})$ is the same for all points $\overline{\tau}$ in the same region.

Therefore we can check whether *all* points in a region satisfy $\phi(\overline{t})$ by picking a *single* point in the region and checking whether $\mathbf{Z} \models \phi(\overline{\tau})$ is true. The latter check can be done using the algorithm diPC-Eval of theorem 4.1. Every conjunction of atomic formulas which represents a region for which this check succeeds becomes a disjunct in the DNF form of ϕ'. Similar techniques have been used in the context of query processing in constraint databases by [KKR90].

The technical tools which we will introduce immediately come from the temporal constraint literature [DMP89]. Similar tools have also been used under various names in [Rev90, CM90] for studying the complexity of query evaluation for Datalog with integer gap-order constraints. Let C be a set (i.e., conjunction) of diPCL inequalities in variables $x_1, \ldots, x_n$. The *binary inequality constraint network* (BICN) associated with C is a labeled directed graph $G = (V, E)$ where $V = \{1, \ldots, n\}$. Node i represents variable x_i and edge (i, j) represents the binary constraints involving x_i and x_j. Let us assume that the constraints on $x_j - x_i$ are $x_j - x_i \leq d_{ij}$ and $x_j - x_i \geq -d_{ji}$ where $-d_{ji} < d_{ij}$. Then the corresponding BICN N will have an edge $i \rightarrow j$ labeled by the *convex* interval $N_{ij} = [-d_{ji}, d_{ij}]$. Unary constraints are represented with the introduction of a special variable $x_0 = 0$. The notions of solution set, consistency and minimality are defined as usual [DMP91]. We will also use a function $Constraints(N)$ which gives the set of binary constraints represented by BICN N. The formal definition of $Constraints$ is omitted.

An alternative graph representation will also be useful. Let C be a set of diPCL inequalities in variables $x_1, \ldots, x_n$. The *distance graph* associated with C is a directed labelled graph $G = (V, E)$ where $V = \{1, \ldots, n\}$. Node i represents variable x_i and edge (i, j) represents the binary constraints involving x_i and x_j. If there is a constraint $x_j - x_i \leq d_{ij}$ in C then edge $i \rightarrow j$ of the associated distance graph will be labelled by d_{ij}. The concept of *minimal distance graph* can be defined similarly with the concept of minimal network. Given the minimal network associated with a set of weak inequality constraints, it is trivial to construct the associated minimal distance graph (and vice versa). The set of constraints $Constraints(G)$ represented by a distance graph G is defined as for BICN.

Definition 4.5 Let $Z \subseteq \mathbb{Z}$ and $Z \neq \emptyset$. An *integer BICN with bounds from the set* Z is a consistent BICN with all edges labeled with $[c, c], (c, \infty)$ or $(-\infty, c)$ where $c \in Z$.

If an integer BICN has $n+1$ nodes (including the 0-th one which corresponds to variable $x_0 = 0$), we will say that it is of *size n*.

Let us now define the concept of a formula corresponding to a BICN N.

Definition 4.6 Let N be a BICN of size n. The formula with free variables $\overline{x} = (x_1, \ldots, x_n)$ corresponding to N is the diPCL formula $\bigwedge_{\theta \in Constraints(M)} \theta$ where M is the minimal network equivalent to N. This formula will be denoted by $\Phi(N)$.

Definition 4.7 If ϕ is a diPCL formula with K quantifiers and vector of free variables $\overline{t}$ then Z_ϕ will denote the set of integers

$$\{i : \, i \in \mathbb{Z} \text{ and } |i| \leq \left|\overline{t}\right| 2^K (maxabs(\phi) + 1)\}.$$

The following lemma tells us how to partition the answer space into regions.

Lemma 4.3 *Let $\phi(\overline{t})$ be a diPCL formula. Then there is a quantifier-free formula $\phi'(\overline{t})$ equivalent to ϕ with the following properties:*

1. *ϕ' is in DNF*
2. *Every disjunct of ϕ' is the diPCL formula corresponding to an integer BICN N of size $\left|\overline{t}\right|$ with bounds from the set Z_ϕ.*

Proof: Let $\phi_1(\overline{t})$ be the quantifier-free formula

$$\psi_1(\overline{t}) \,\vee\, \cdots \,\vee\, \psi_m(\overline{t})$$

equivalent to ϕ obtained by the algorithm of proposition 3.1. Let us also assume that the only relation symbol appearing in ψ_1 is $\leq$. As in the proof of lemma 4.1, integer constants in ϕ_1 are limited by $2^K(maxabs(\phi) + 1)$.

Let $N_1, \ldots, N_{m_1}$, $(m_1 \leq m)$ be the minimal BICN corresponding to the consistent ψ_i's $(i = 1, \ldots, m)$. Let us create the formula

$$\phi_2(\overline{t}) \equiv \psi'_1(\overline{t}) \,\vee\, \cdots \,\vee\, \psi'_{m_1}(\overline{t})$$

where

$$\psi'_i = \bigwedge_{\eta \in Constraints(N_i)} \eta, \quad \text{for all } i = 1, \ldots, m_1.$$

Every integer r, such that $t_k - t_l \leq r$ (or $t_k - t_l = r$) is a conjunct in some ψ'_i, is the length of the shortest (*simple*) path connecting nodes k and l in the minimal distance graph for N_i [DMP91]. The number of edges in a simple path is $|\bar{t}|$ and the label of each edge is limited by $2^K(maxabs(\phi)+1)$. Therefore every integer in ϕ_2 is limited by $|\bar{t}|\, 2^K(maxabs(\phi)+1)$.

We can now obtain a formula equivalent to ϕ_2 in the desired form as follows:

1. From every ψ'_i construct a conjuction of disjunctions ψ''_i in the following way.
   (a) For every pair of variables t_k, t_l such that $t_k - t_l = r_1$ is in ψ'_i, add $t_k - t_l = r_1$ to ψ''_i.
   (b) For every pair of variables t_k, t_l such that $t_k - t_l \leq r_1$ and $t_k - t_l \geq r_2$ is in ψ'_i, add
   $$\bigvee_{r_1 \leq r \leq r_2} t_k - t_l = r$$
   to ψ''_i.
   (c) For every pair of variables such that only $t_k - t_l \leq r_1$ is in ψ'_i, add conjunct
   $$t_k - t_l < -\Lambda \;\vee \bigvee_{-\Lambda \leq r \leq r_1} t_k - t_l = r$$
   to ψ''_i where $\Lambda = |\bar{t}|\, 2^K(maxabs(\phi)+1)$.
   (d) For every pair of variables such that only $t_k - t_l \geq r_2$ is in ψ'_i, add conjunct
   $$\bigvee_{r_2 \leq r \leq \Lambda} t_k - t_l = r \;\vee\; t_k - t_l > \Lambda$$
   to ψ''_i where Λ is as above.
   (e) For every pair of variables t_k, t_l such that ψ'_i contains no constraint involving t_k and t_l, add conjunct
   $$t_k - t_l < -\Lambda \vee \bigvee_{r_1 \leq r \leq r_2} t_k - t_l = r \vee t_k - t_l > \Lambda$$
   to ψ''_i where Λ is as above.
2. Transform every $\psi''_i(\bar{t})$ in DNF. Let
   $$\theta_1(\bar{t}) \;\vee\; \cdots \;\vee\; \theta_n(\bar{t})$$
   be the resulting formula.
3. Let $M_1, \ldots, M_{n_1}$, $(n_1 \leq n)$ be the minimal BICN corresponding to consistent θ_i's $(i = 1, \ldots, n)$. The wanted formula $\phi'(\bar{t})$ is
   $$\theta'_1(\bar{t}) \;\vee\; \cdots \;\vee\; \theta'_{n_1}(\bar{t})$$
   where
   $$\theta'_i = \bigwedge_{\eta \in Constraints(M_i)} \eta, \quad \text{for all } i = 1, \ldots, n_1.$$

■

The following lemmas give some important properties of integer BICN.

Lemma 4.4 *Let $\overline{\tau}$ be an element of $\mathbb{Z}^n$. For any set $Z \subseteq \mathbb{Z}$, there exists a unique integer BICN N with bounds from Z such that $\mathbf{Z} \models \Phi(N)(\overline{\tau})$.*

Proof: Existence is obvious. For the uniqueness part, we simply observe that $N_1 \neq N_2$ implies that $\Phi(N_1) \wedge \Phi(N_2)$ is inconsistent. ■

The following lemma allows us to verify the truth of a diPCL formula over a region of the answer space by verifying its truth over a *point* in this region.

Lemma 4.5 *Let $\phi(\bar{t})$ be a diPCL formula and N be an integer BICN with bounds from the set Z_ϕ. Then $\mathbf{Z} \models \Phi(N) \supset \phi$ iff $\mathbf{Z} \models \phi(\overline{\tau})$ is true for an arbitrary $\overline{\tau}$ such that $\mathbf{Z} \models \Phi(N)(\overline{\tau})$.*

Proof: The "only if" part is trivial so we consider the "if" part. Let us assume that there is $\overline{\tau} \in \mathbb{Z}^{|\bar{t}|}$ such that $\mathbf{Z} \models \Phi(N)(\overline{\tau})$ and $\mathbf{Z} \models \psi(\overline{\tau})$. Let

$$\phi'(\bar{t}) \equiv \theta'_1(\bar{t})) \;\vee\; \cdots \;\vee\; \theta'_n(\bar{t})$$

be the diPCL formula equivalent to ϕ computed as in lemma 4.3. Then there exists a single disjunct $\theta_i(\bar{t})$ of ϕ' such that $\mathbf{Z} \models \theta_i(\overline{\tau})$. But then θ_i must be $\Phi(N)$ from lemma 4.4. Therefore $\mathbf{Z} \models \forall(\Phi(N) \supset \phi)$ since $\Phi(N)$ is a disjunct of ϕ'. ■

Now we are ready to demonstrate the main result of this section.

Theorem 4.5 *Let ϕ be a diPCL formula. A quantifier-free formula equivalent to ϕ in DNF can be computed in PSPACE.*

Proof: Let us assume that $\bar{t}$ is the vector of all free variables of ϕ. We can compute a quantifier-free formula equivalent to ϕ in DNF as follows. We generate one by one all integer BICN of size $|\bar{t}|$ with bounds from Z_ϕ. For every such BICN N, we find a solution $\overline{\tau}$ of N and check whether $\mathbf{Z} \models \phi(\overline{\tau})$ using the algorithm DIPC-EVAL of theorem 4.1. If this check succeeds, the diPCL formula corresponding to N becomes a disjunct of the resulting quantifier-free formula. The correctness of this procedure follows from lemmas 4.3 and 4.5.

Finding a solution of a BICN can be done in the following way. We first compute the minimal network M equivalent to N. Then we find a solution of M using *backtrack-free* search as follows [DMP91]. We initially assign the value 0 to t_0. Then we assign to t_1 any value which satisfies the constraints involving t_1 and t_0. We proceed in the same fashion with t_2, t_3 and so on.

It is not difficult to show that the above procedure can be implemented by a deterministic Turing machine in

PSPACE. A detailed proof can be found in [Kou94a]. ■

5 AN IMPROVED QUANTIFIER ELIMINATION ALGORITHM FOR dePC

Let us now turn to quantifier elimination and decision for dePCL. We will first develop a decision procedure for sentences of dePCL. Then we will use this procedure for developing a quantifier elimination algorithm for arbitrary dePCL formulas.

At first we show that we can confine our attention to formulas of dePCL involving *only integer* constants. A similar result appears in [Alu91] in the context of checking emptiness of the language of a timed automaton.

Definition 5.1 Let ϕ be a formula of dePCL and $r \in \mathbb{Q}$. Then ϕ_r will denote the formula which is obtained from ϕ by replacing each rational constant c by $r \cdot c$.

Lemma 5.1 *Let ϕ be a formula of dePCL with free variables $\overline{t}$. If $r \in \mathbb{Q}$, $r > 0$ and $\overline{\tau} \in \mathbb{Q}^n$ then*

$$\mathbf{Q} \models \phi[\overline{t} \leftarrow \overline{\tau}] \text{ iff } \mathbf{Q} \models \phi_r[\overline{t} \leftarrow r \cdot \overline{\tau}].$$

Proof: Use induction on the structure of ϕ. ■

The following definition will be used heavily in the subsequent discussion.

Definition 5.2 A quantifier Qz is limited by the fraction A/B, denoted by $Qz \preceq A/B$, if, instead of ranging over all rational numbers, it ranges over all numbers whose numerator is limited by A and whose denominator *is* B.

Lemma 5.2 *Let ϕ be the following dePCL sentence*

$$(Q_1 t_1) \cdots (Q_K t_K) \psi(t_1, \ldots, t_K).$$

Let us also assume that only integer constants are involved in ϕ. Then $\mathbf{Q} \models \phi$ iff $\mathbf{Q} \models \phi'$ where ϕ' is the same as ϕ except that $(Q_i t_i)$ is substituted by

$$(Q_i \preceq \tfrac{2^{K+3i-2}(maxabs(\phi)+1)}{2^i})$$

for all $i = 1, \ldots, K$.

Proof: We will use induction on the order of appearance of the quantifier in ϕ. We assume that ϕ is in PNF. Note that the proof still goes through when this assumption is dropped.

Base case, $i = 1$. Let us eliminate quantifiers $(Q_2 t_2), \ldots, (Q_K t_K)$ from ϕ using the procedure of proposition 3.1. The integer constants in the resulting formula $(Q_1 t_1)\phi'(t_1)$ are limited by $2^{K-1}(maxabs(\phi) + 1)$ (this can be shown as in lemma 4.1). These integers partition $\mathbb{Q}$ into a set S of $4 \cdot 2^{K-1}(maxabs(\phi)+1)+3$ intervals of the form $(-\infty, r)$ or $[r, r]$ or $(r, r+1)$ or (r, ∞) where $r \preceq 2^{K-1}(maxabs(\phi) + 1)$. The truth value of $\phi'(t_1)$ remains the same for all t_1 in the same interval of S. Therefore we can determine the truth value of $(Qt_1)\phi'(t_1)$ in $\mathbf{Q}$ by determining the truth value of $\phi'(t_1)$ while t_1 ranges over a finite set of representatives, one for each interval of S. The set

$$S_{rep} = \{ (t_1 + t_2)/2,\ (t_1 + t_2)/2 + 1 :\ t_1, t_2 \in \mathbb{Z} \text{ and } t_1, t_2 \preceq 2^{K-1}(maxabs(\phi) + 1) \}$$

is an appropriate set of such representatives. The elements of S_{rep} are rationals limited by $2^{K+1}(maxabs(\phi) + 1)/2$.

Inductive step. Let us now assume that the lemma holds for quantifiers $(Q_1 t_1), \ldots, (Q_i t_i)$. We will show that the lemma holds for $(Q_{i+1} t_{i+1})$ as well. From the induction hypothesis we have $\mathbf{Q} \models \phi$ if and only if $\mathbf{Q} \models \phi_1$ where ϕ_1 is

$$(Q_1 t_1 \preceq \tfrac{2^{K+1}(maxabs(\phi)+1)}{2}) \cdots$$
$$(Q_i t_i \preceq \tfrac{2^{K+3i-2}(maxabs(\phi)+1)}{2^i})$$
$$(Q_{i+1} t_{i+1}) \cdots (Q_K t_K) \psi(t_1, \ldots, t_K).$$

Let us eliminate quantifiers $(Q_{i+2} t_{i+2}), \cdots, (Q_K t_K)$ from ϕ_1 using the procedure of proposition 3.1 to arrive at the following formula ϕ_2:

$$(Q_1 t_1 \preceq \tfrac{2^{K+1}(maxabs(\phi)+1)}{2}) \cdots$$
$$(Q_i t_i \preceq \tfrac{2^{K+3i-2}(maxabs(\phi)+1)}{2^i})$$
$$(Q_{i+1} t_{i+1}) \psi_1(t_1, \ldots, t_{i+1}).$$

The truth value of ϕ_2 depends on the truth values of all formulas

$$(Q_{i+1} t_{i+1}) \psi_1(\tau_1, \ldots, \tau_i, t_{i+1})$$

where

$$\tau_j \leq \frac{2^{K+3j-2}(maxabs(\phi) + 1)}{2^j}, \quad j = 1, \ldots, i.$$

The atomic formulas of $(Q_{i+1} t_{i+1}) \psi_1(\tau_1, \ldots, \tau_i, t_{i+1})$ are of the form $t_{i+1} \sim r$ or $t_{i+1} \sim \tau_j + r$ where $1 \leq j \leq i$, $\sim$ is $<, \leq$ or $=$, τ_j is a rational limited as above and r is an integer limited by $2^{K-(i+1)}(maxabs(\phi) + 1)$.

Let us now observe that every rational limited by $2^{K+3j-2}(maxabs(\phi) + 1)/2^j$, where $1 \leq j \leq i - 1$, is included in the set of rationals limited by $2^{K+3i-2}(maxabs(\phi) + 1)/2^i$. Therefore $\tau_j + r$ is a rational limited by $2^{K+3i-1}(maxabs(\phi) + 1)/2^i$.

Using an argument similar to the one given for the base case, we can see that the quantifier $(Q_{i+1} t_{i+1})$ can be limited to range over rationals that are the average of two rationals limited by $2^{K+3i-1}(maxabs(\phi) + 1)/2^i$, or are one smaller or one larger than all such averages. As a result, $(Q_{i+1} t_{i+1})$ can be limited by

$2^{K+3(i+1)-2}(maxabs(\phi)+1)/2^{i+1}$. The result follows. ■

Definition 5.3 If ϕ is a dePCL formula then $maxabs(\phi)$ will denote the maximum absolute value of the integers which appear in ϕ as numerators or denominators. If ϕ involves only integer constants then $maxabs(\phi)$ will denote the maximum absolute value of the integers which appear in ϕ.

We are now ready to prove the basic result of this section.

Theorem 5.1 *Let ϕ be a dePCL sentence. The problem of deciding whether $\mathbf{Q} \models \phi$ is PSPACE-complete.*

Proof: *Lower bound.* PSPACE-hardness follows from a straightfoward reduction from QBF [Sto77].

Upper bound. An algorithm for this problem can proceed as follows. First, we transform ϕ into formula ϕ' which involves only integer constants. This can be done by multiplying every fraction p/q of ϕ by the product of all denominators of fractions in ϕ. Therefore every integer in ϕ' will be limited by $maxabs(\phi)^{|\phi|+1}$. We can conclude that

$$|\phi'| \leq |\phi|\,(|\phi|+1)\log(maxabs(\phi)) \leq \\ 2\,|\phi|^2\log(maxabs(\phi)) \leq 2\,|\phi|^2\log|\phi|$$

and $maxabs(\phi') \leq maxabs(\phi)^{|\phi|+1}$.

Then we can devise a recursive algorithm dePC-Eval, similar to diPC-Eval, which decides ϕ'. This algorithm will make use of lemma 5.2 to limit quantifiers over finite sets of rationals. Every such rational has a numerator limited by $2^{4K-2}(maxabs(\phi')+1)$ and a denominator limited by 2^K where K is the number of quantifiers in ϕ'. Storing the numerator of each one of these rationals requires

$$4K-2+\log(maxabs(\phi')+1) \leq \\ 4K-1+\log(maxabs(\phi)^{|\phi|+1}) \leq \\ 4K-1+(|\phi|+1)\log(maxabs(\phi)) \leq O(|\phi|\log|\phi|)$$

bits while storing the denominator requires $O(|\phi|)$ bits. Thus dePC-Eval needs $O(|\phi|^2\log|\phi|)$ space for keeping track of the current assignment to the quantified variables of ϕ'. No more space is required for bookkeeping. Therefore the total space requirement of dePC-Eval is $O(|\phi|^2\log|\phi|)$. ■

It might be interesting to compare the above result with the following theorems. The first one considers a theory which is less expressive than dePC. The second deals with the full first-order theory of real addition with order.

Ferrante & Geiser

Theorem 5.2 *[FG77] Deciding a sentence of length n in the first order theory of rational order can be done in deterministic space $O(n\log n)$.*

Theorem 5.3 *[Ber80, BM80] The problem of deciding a sentence of length n in the theory of real addition with order is complete for the class $\bigcup_{k>0} TA[2^{n^k}, n]$.*

The next theorem follows easily from the above discussion. It is also a consequence of the fact that deciding a formula of the same form in the first-order theory of real addition with order is also Σ_k^p-complete [Son85].

Theorem 5.4 *Let ϕ be $\exists_k$ sentence of dePCL. The problem of deciding whether $\mathbf{Q} \models \phi$ is Σ_k^p-complete.*

A corresponding Π_k^p bound can be established for $\forall_k$ sentences of dePCL.

5.1 Open Formulas

We will now present a quantifier elimination algorithm for open formulas of dePCL. We will proceed as in the case of diPCL formulas. The following lemma will allow us to concentrate on dePCL formulas involving only integer constants.

Lemma 5.3 *Let ϕ be a formula of dePCL and ϕ' be a quantifier-free formula equivalent to ϕ. If $r \in \mathbb{Q}$ and $r > 0$ then $\phi' \equiv \psi_{r^{-1}}$ where ψ is a quantifier-free formula equivalent to ϕ_r.*

Proof: Let $\overline{x}$ be the vector of variables in ϕ'. If $\overline{\chi} \in \mathbb{Q}^{|\overline{x}|}$ then the following equivalences prove the lemma (with help from lemma 5.1):

$$\mathbf{Q} \models \phi'[\overline{x} \leftarrow \overline{\chi}] \text{ iff } \mathbf{Q} \models \phi[\overline{x} \leftarrow \overline{\chi}] \text{ iff} \\ \mathbf{Q} \models \phi_r[\overline{x} \leftarrow r\cdot\overline{\chi}] \text{ iff } \mathbf{Q} \models \psi[\overline{x} \leftarrow r\cdot\overline{\chi}] \text{ iff} \\ \mathbf{Q} \models \psi_{r^{-1}}[\overline{x} \leftarrow \overline{\chi}].$$

■

Now assume we are given a formula ϕ of dePCL. We can find a quantifier-free formula equivalent to ϕ as follows. First we transform ϕ into a formula ϕ_r which has only integer constants by multiplying every fraction by an appropriate integer r. Then we find a quantifier-free formula ψ equivalent to ϕ_r. Finally, we compute $\psi_{r^{-1}}$ which is a quantifier-free formula equivalent to ϕ.

Let us then assume that $\phi(\overline{t})$ is a formula of dePCL which involves only integer constants, and $\overline{t}$ are all the free variables of ϕ. A quantifier free formula ϕ' equivalent to ϕ can be found in the following way. At first, we estimate how large integer constants can grow in the constraints of the answer. Then we use this information to construct a finite partition of the space $\mathbb{Q}^{|\overline{t}|}$ into *regions* with the following properties:

1. Every region can be represented by a conjunction of dePCL-constraints involving only integer constants.
2. The truth value of the sentence $\phi(\overline{\tau})$ is the same for all points $\overline{\tau}$ in the same region.

Therefore we can check whether *all* points in a region satisfy $\phi(\overline{t})$ by picking a *single* point $\overline{\tau}$ in the region and checking whether $\mathbf{Q} \models \phi(\overline{\tau})$ is true. The latter check can be done using the above algorithm DEPC-EVAL. Every conjunction of constraints representing a region for which this check succeeds, becomes a disjunct in the DNF form of ϕ'.

Let us now introduce the machinery required for presenting our method. We first define rational BICN.

Definition 5.4 A *rational BICN with bounds from the set* $Z \subseteq \mathbb{Z}$ is a consistent BICN with all edges labeled with $[c, c], (c, d), (c, \infty)$ or $(-\infty, c)$ where $c < d$ and $c, d \in Z$.

The *size* of a rational BICN is defined as for integer BICN.

Let us now define the concept of a formula corresponding to a rational BICN N.

Definition 5.5 Let N be a rational BICN of size n. The formula with free variables $\overline{x} = (x_1, \ldots, x_n)$ corresponding to N is the dePCL formula $\bigwedge_{\theta \in Constraints(M)} \theta$ where M is the minimal network equivalent to N. This formula will be denoted by $\Phi(N)$.

Definition 5.6 If ϕ is a dePCL formula with K quantifiers and vector of free variables $\overline{t}$ then Z_ϕ will denote the set of integers

$$\{i : \ i \in \mathbb{Z} \text{ and } |i| \leq |\overline{t}|\, 2^K maxabs(\phi)\}.$$

The following lemmas help to establish the main result. The reader can easily notice the similarity with the development of section 4.1. Lemma 5.4 tells us how to partition the answer space into regions. Subsequent lemmas give properties of rational BICN. The proofs of some of the lemmas are omitted.

Lemma 5.4 *Let $\phi(\overline{t})$ be a dePCL formula involving only integer constants. Then there is a quantifier-free formula $\phi'(\overline{t})$ equivalent to ϕ with the following properties:*

1. *ϕ' is in DNF*
2. *Every disjunct of ϕ' is the dePCL formula corresponding to a rational BICN N of size $|\overline{t}|$ with bounds from the set Z_ϕ.*

Proof: First, we transform ϕ into an equivalent quantifier-free formula ϕ_1 using the algorithm of proposition 3.1. Integer constants in ϕ_1 will be limited by $2^K maxabs(\phi)$. Then we proceed as in lemma 4.3. ■

Lemma 5.5 *Let $\overline{\tau}$ be an element of $\mathbb{Q}^n$. For any set $Z \subseteq \mathbb{Z}$, there exists a unique rational BICN N with bounds from Z such that $\mathbf{Q} \models \Phi(N)(\overline{\tau})$.*

Lemma 5.6 *Let $\phi(\overline{t})$ be a dePCL formula involving only integer constants. If N is a rational BICN with bounds from the set Z_ϕ then $\mathbf{Q} \models \Phi(N) \supset \phi$ iff $\mathbf{Q} \models \phi(\overline{\tau})$ for an arbitrary $\overline{\tau}$ such that $\mathbf{Q} \models \Phi(N)(\overline{\tau})$.*

The following theorem gives the main result of this section.

Theorem 5.5 *Let ϕ be a dePCL formula. A quantifier-free formula equivalent to ϕ in DNF can be computed in PSPACE.*

Proof: Let us assume that ϕ has K quantifiers and $\overline{t}$ is the vector of all its free variables. We can find a quantifier-free formula equivalent to ϕ as follows. First, we transform ϕ into a formula ϕ' which involves only integer constants. This can be done as in theorem 5.1: we multiply every fraction of ϕ by the product P of all denominators of fractions in ϕ.

Secondly, we generate, one by one, all rational BICN of size $|\overline{t}|$ with bounds from

$$Z_{\phi'} = \{i : \ i \in \mathbb{Z} \text{ and } |i| \leq |\overline{t}|\, 2^K maxabs(\phi)^{|\phi|+1}\}$$

(since $maxabs(\phi') \leq maxabs(\phi)^{|\phi|+1}$). For each BICN N, we find a solution $\overline{\tau}$ of N (using the minimal network M) and check whether $\mathbf{Q} \models \phi'(\overline{\tau})$ using algorithm DEPC-EVAL of theorem 5.1. If this check succeeds then we divide each constant of $\Phi(N)$ by P and use the result to form a disjunct of the returned formula. The correctness of this procedure follows from the previous lemmas. It is not difficult to see that the above algorithm can be implemented by a deterministic Turing machine in PSPACE [Kou94a]. ■

6 THEORIES OF POINT AND INTERVAL CONSTRAINTS

We will now extend the theory diPC to take interval constraints into account. We define the language diTCL (*di*screte *T*emporal *C*onstraint *L*anguage) which is an extension of diPCL. The time entities in this language are points and intervals. Points are identified with the integers while intervals are considered to be pairs of points. diTCL has two sorts: $\mathcal{Z}$ (for points or integers) and $\mathcal{I}_\mathcal{Z}$ (for integer intervals). The non-logical symbols of diTCL include a countably infinite set of constant symbols of sort $\mathcal{Z}$ (the point or integer constants), function symbols L and R of sort $(\mathcal{I}_\mathcal{Z}, \ \mathcal{Z})$, function symbol $-$ of sort $(\mathcal{Z}, \ \mathcal{Z}, \ \mathcal{Z})$ and predicate symbol $<$ of sort $(\mathcal{Z}, \ \mathcal{Z})$.

The set of *simple terms* of diTCL are defined by the following rules:

- Constants are simple terms.
- A variable of sort $\mathcal{Z}$ is a simple term.
- If i is a variable of sort $\mathcal{I}_\mathcal{Z}$ then i_L and i_R are simple terms.

The set of *terms* of diTCL can now be defined as follows:

- Simple terms are terms.
- If t_1 and t_2 are simple terms then $t_1 - t_2$ is a term.

An *atomic formula* of diTCL is a formula of the form $t \sim c$ or $c \sim t$ where $\sim$ is $<$ or $=$ and t is a term. The set of *well-formed formulas* is defined as usual.

The symbols of diTCL are interpreted with respect to the fixed structure $\mathbf{ZI_Z}$ which captures our assumptions. $\mathbf{ZI_Z}$ assigns to the sort $\mathcal{Z}$ the set of integers $\mathbb{Z}$, and to the sort $\mathcal{I}_{\mathcal{Z}}$ the set of *integer intervals* $Int(\mathbb{Z}) = \{(a,b) : \ a, b \in \mathbb{Z} \text{ and } a <_{\mathbb{Z}} b\}$. To each constant symbol of sort $\mathcal{Z}$, $\mathbf{ZI_Z}$ assigns an element of $\mathbb{Z}$. To function symbols L and R, $\mathbf{ZI_Z}$ assigns the functions $l, r : \ Int(\mathbb{Z}) \rightarrow \mathbb{Z}$ such that $l((a,b)) = a$ and $r((a,b)) = b$. These functions map each interval to its left and right endpoint respectively. To function symbol $-$, $\mathbf{ZI_Z}$ assigns the function $-_{\mathbb{Z}}$ which is the subtraction operation over the integers. To predicate symbol $<$, $\mathbf{ZI_Z}$ assigns the relation $<_{\mathbb{Z}}$ over the integers.

We will take the theory of structure $\mathbf{ZI_Z}$ to be *the* theory of point and interval constraints in linear, unbounded and discrete time. This theory will be denoted by *diTC*.

In a simililar way we can extend dePC to account for time intervals. The corresponding theory will be called *deTC* and its language *deTCL*. Detailed definitions can be found in [Kou94a].

Let us observe that we can introduce all the interval-to-interval relations and point-to-interval relations of [Mei91] as defined relations in these theories.

We can now use the results of sections 4 and 5 and a translation from diTCL to diPCL (as in [Lad88]) to achieve the following results. Details can be found in [Kou94a].

Theorem 6.1 *The decision problem for arbitrary diTCL or deTCL sentences is PSPACE-complete. The decision problem for $\exists_k$ sentences of diTCL or deTCL is Σ_k^p-complete.*

Theorem 6.2 *Let ϕ be a diTCL or deTCL formula. A quantifier-free formula equivalent to ϕ in DNF can be computed in PSPACE.*

7 CONCLUSIONS

In this paper we discussed the complexity of quantifier elimination and decision algorithms for the theory of point constraints over the integers (diPC) and over the rationals (dePC). The theory diPC is a special case of the theory of Presburger arithmetic while dePC is a special case of the theory of real addition with order. We have shown that in both cases quantifier-elimination can be done in PSPACE. We have also shown that the decision problem for arbitrarily quantified sentences is PSPACE-complete while for $\exists_k$ sentences it is Σ_k^p-complete. The bounds for the two theories of point constraints do not change if intervals and interval constraints are introduced.

Our results will be interesting to researchers in constraint databases [KKR90] but also to the theoretical computer science community in general. In [Kou94a, Kou94b] the results of this paper are used to study the complexity of query evaluation in indefinite temporal constraint databases. In related work [Rev90] considered Datalog with integer gap-order constraints ($Datalog^{<_{\mathbb{Z}}}$). A *gap-order constraint* is a constraint of the form:

$$x = c, \ x = y, \ x < c, \ c < x \text{ or } x - y < g$$

where x, y are variables ranging over $\mathbb{Z}$, $c \in \mathbb{Z}$, $g \in \mathbb{Z}$ and $g \geq 0$. Revesz showed that $Datalog^{<_{\mathbb{Z}}}$ queries over integer gap-order databases can be evaluated in closed form. In addition recognizing whether a certain tuple is in the answer to a query can be done with PTIME *data complexity*.[3] Independently the above problem has been solved in [CM90] who have also considered gap-order constraints over a dense domain. The only difference with [Rev90] is that [CM90] do not consider data complexity thus the complexity of their query evaluation procedure is EXPTIME.

[3]The notion of data complexity was originally introduced in [Var82] in the context of query evaluation for relational databases. When we consider *data complexity*, we measure the complexity of evaluating a query over a database as a function of the database size only; the query program and the database schema are considered *fixed*.

References

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

[Alu91] R. Alur. *Techniques for Automatic Verification of Real-Time Systems*. PhD thesis, Dept. of Computer Science, Stanford University, 1991.

[Ber80] L. Berman. The Complexity of Logical Theories. *Theoretical Computer Science*, 11:71–78, 1980.

[BM80] A.R. Bruss and A.R. Meyer. On Time-Space Classes and their Relation to the Theory of Real Addition. *Theoretical Computer Science*, 11:59–69, 1980.

[CK73] C.C. Chang and H.J. Keisler. *Model Theory*. North Holland, 1973.

[CM90] J. Cox and K. McAloon. Decision Procedures for Constraint Based Extensions of Datalog. Technical Report 90-09, Dept. of Computer and Information Sciences, Brooklyn College of C.U.N.Y., 1990.

[CMT92] J. Cox, K. McAloon, and C. Tretkoff. Computational Complexity and Constraint Logic Programing Languages. *Annals of Mathematics and Artificial Intelligence*, 5:163–189, 1992.

[DMP89] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal Constraint Networks. In R. Brachman, H. Levesque, and R. Reiter, editors, *Proceedings of 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 83–93, Toronto, Ontario, 1989.

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

[DP88] Rina Dechter and Judea Pearl. Network-Based Heuristics for Constraint Satisfaction Problems. *Artificial Intelligence*, 34(1):1–38, 1988.

[End72] H.B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 1972.

[FG77] J. Ferrante and J.R. Geiser. An Efficient Decision Procedure for the Theory of Rational Order. *Theoretical Computer Science*, 4(2):227–233, 1977.

[FR75] J. Ferrante and C. Rackoff. A Decision Procedure for the First Order Theory of Real Addition with Order. *SIAM Journal on Computing*, 4(1):69–76, 1975.

[GS93] A. Gerevini and L. Schubert. Efficient Temporal Reasoning Through Timegraphs. In *Proceedings of IJCAI-93*, pages 648–654, 1993.

[Joh90] D.S. Johnson. A Catalog of Complexity Classes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A, chapter 2. North-Holland, 1990.

[KKR90] Paris C. Kanellakis, Gabriel M. Kuper, and Peter Z. Revesz. Constraint Query Languages. In *Proceedings of the 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 299–313, 1990. Long version to appear in Journal of Computer and System Sciences.

[KL91] H. Kautz and P. Ladkin. Integrating Metric and Qualitative Temporal Reasoning. In *Proceedings of AAAI-91*, pages 241–246, 1991.

[Kou92] Manolis 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. *Foundations of Temporal Constraint Databases*. PhD thesis, Computer Science Division, Dept. of Electrical and Computer Engineering, National Technical University of Athens, February 1994.

[Kou94b] Manolis Koubarakis. The Complexity of Query Evaluation in Indefinite Temporal Constraint Databases, February 1994. Unpublished paper.

[Lad88] Peter Ladkin. Satisfying First-Order Constraints About Time Intervals. In *Proceedings of AAAI-88*, pages 512–517, 1988.

[LM88] Peter Ladkin and Roger Maddux. On Binary Constraint Networks. Technical Report KES.U.88.8, Kestrel Institute Technical Report, 1988.

[Mac77] A.K. Mackworth. Consistency in Networks of Relations. *Artificial Intelligence*, 8(1):99–118, 1977.

[Mei91] I. Meiri. Combining Qualitative and Quantitative Constraints in Temporal Reasoning. In *Proceedings of AAAI-91*, pages 260–267, 1991.

[Mon74] U. Montanari. Networks of Constraints: Fundamental Properties and Applications to Picture Processing. *Information Sciences*, 7:95–132, 1974.

[Rab77] M.O. Rabin. Decidable theories. In *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations* *of Mathematics*, pages 595–629. North-Holland, 1977.

[Rev90] Peter Z. Revesz. A Closed Form for Datalog Queries with Integer Order. In *Proceedings of the 3rd International Conference on Database Theory*, pages 187–201, 1990. Long version to appear in Theoretical Computer Science.

[RL78] C.R. Reddy and D.W. Loveland. Presburger Arithmetic with Bounded Quantifier Alternation. In *Proc. of ACM Symposium on the Theory of Computing*, pages 320–325, 1978.

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

[SD93] E. Schwalb and R. Dechter. Coping with Disjunctions in Temporal Constraint Satisfaction Problems. In *Proceedings of AAAI-93*, 1993.

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

[Sto77] L.J. Stockmeyer. The Polynomial-Time Hierarchy. *Theoretical Computer Science*, 3:1–22, 1977.

[Var82] Moshe Vardi. The Complexity of Relational Query Languages. In *Proceedings of ACM SIGACT/SIGMOD Symposium on Principles of Database Systems*, pages 137–146, 1982.

[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.