From Local to Global Consistency in Temporal Constraint Networks

Manolis Koubarakis*

Dept. of Computation
UMIST
P.O. Box 88
Manchester M60 1QD
U.K.
manolis@sna.co.umist.ac.uk

Abstract. We study the problem of global consistency for several classes of quantitative temporal constraints which include inequalities, inequations and disjunctions of inequations. In all cases that we consider we identify the level of local consistency that is necessary and sufficient for achieving global consistency and present an algorithm which achieves this level. As a byproduct of our analysis, we also develop an interesting minimal network algorithm.

1 Introduction

One of the most important notions found in the constraint satisfaction literature is global consistency [Fre78]. 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].

In this paper we study the problem of enforcing global consistency for sets of quantitative temporal constraints over the rational (or real) numbers. The class of constraints that we consider includes equalities (e.g., $x_i - x_j = 5$), inequalities (e.g., $x_i - x_j \leq 5$), inequations (e.g, $x_i - x_j \neq 5$) and disjunctions of inequations ($x_i - x_j \neq 5 \vee x_k - x_l \neq 8$). For the representation of equalities, inequalities and inequations, we utilize *binary temporal constraint networks*. Disjunctions of inequations are represented separately.

Disjunctions of inequations have been introduced in [Kou92] following the observation that in the process of eliminating variables from a set of temporal

* Most of this work was performed while the author was at the National Technical University of Athens and at Imperial College, London. At Imperial College financial support was received from project CHRONOS funded by EPSRC and DTI.

constraints, an inequation can give rise to a disjunction of inequations.[2] In related temporal reasoning research [VK86, vB90, GS93, GSS93] have considered inequations of the form $t_1 \neq t_2$ in the context of PA networks. Also, [Mei91a] has studied inequations of the form $t \neq r$ (r a real constant) in the context of point networks with *almost-single-interval domains*. In a more general context, researchers in constraint logic programming (originally [LM89] and later [IvH93, Imb93, Imb94]) have studied disjunctions of arbitrary linear inequations (e.g., $2x_1 + 3x_2 - 4x_3 \neq 4 \vee x_2 + x_3 + x_5 \neq 7$). [LM89, IvH93] concentrate on deciding consistency and computing canonical forms while [Imb93, Imb94] deal mostly with variable elimination. It is interesting to notice that the basic algorithm for variable elimination in this case has been discovered independently in [Kou92] and [Imb93] although [Kou92] has used the result only in the context of temporal constraints.

The contributions of this paper can be summarized as follows.

1. We show that strong 5-consistency is necessary and sufficient for achieving global consistency in temporal constraint networks for inequalities and inequations (Corollary 4).[3] This result (and all subsequent ones) rely heavily on an observation of [LM89, Kou92, Imb93]: *(disjunctions of) inequations can be treated* independently *of one another for the purposes of deciding consistency or performing variable elimination.*
   We give an algorithm which achieves global consistency in $O(Hn^4)$ where n is the number of nodes in the network and H is the number of inequations (Theorems 3 and 5). The analysis of this algorithm demonstrates that there are situations where it is *impossible* to enforce global consistency without introducing disjunctions of inequations.
   A detailed analysis of the global consistency algorithm also gives us an algorithm for computing the minimal temporal constraint network in this case. The complexity of this algorithm is $O(\max(Hn^2, n^3))$ (Theorem 7).
2. We also consider global consistency of point algebra networks [VK86]. In this case strong 5-consistency is also necessary and sufficient for achieving global consistency (Theorem 9). This result, which answers an open problem of [vB90], also follows from [Kou92] but the bounds of the algorithms given there were not the tightest possible.
3. Finally we consider global consistency when disjunctions of inequations are also allowed in the given constraint set. This case is mostly of theoretical interest and is presented here for completeness. In this case, strong $(2V + 1)$-consistency is necessary and sufficient for achieving global consistency (Corollary 11). The parameter V is the maximum number of variables in any disjunction of inequations.

Most of the above results come from the author's Ph.D. thesis [Kou94a] or are refinements of ideas presented there.

[2] Elimination of variables is a very important operation in temporal constraint databases [Kou94a, Kou94b, Kou94c].

[3] As shown in [DMP91] if only inequalities are considered path consistency is necessary and sufficient for achieving global consistency.

The paper is organized as follows. The next section presents definitions and preliminaries. Section 3 discusses global consistency of temporal constraint networks while Section 4 presents an algorithm for computing the minimal network. Section 5 considers the case of point algebra networks. Section 6 considers the case of arbitrary temporal constraints. Finally Section 7 summarizes our results. Most proofs are omitted and can be found in the longer version of this paper.

2 Definition and Preliminaries

We consider time to be linear, dense and unbounded. *Points* will be our only time entities. Points are identified with the rational numbers but our results still hold if points are identified with the reals. The set of rational numbers will be denoted by $\mathcal{Q}$.

A *temporal constraint* is a formula $t - t' \leq r,\ t - t' < r,\ t - t' = r$ or

$$t_1 - t'_1 \neq r_1 \ \vee \ \cdots \ \vee \ t_n - t'_n \neq r_n$$

where $t, t', t_1, \cdots, t_n, t'_1, \cdots, t'_n$ are variables and $r, r_1, \cdots, r_n$ are constants. The rationale for introducing disjunctions of inequations into the picture has been given in [Kou92]. We assume that the reader is familiar with the notions of solution and consistency for a set of temporal constraints [Kou92]. If C is a set of temporal constraints then $Sol(C)$ will denote its set of solutions.

If c is a disjunction of inequations then $\overline{c}$ denotes the *complement* of c i.e., the conjunction of equations obtained by negating c. If C is a set of equalities in n variables, the solution set of C is an affine subset of $\mathcal{Q}^n$. If C is a set of inequalities in n variables, the solution set of C is a convex polyhedron in $\mathcal{Q}^n$. If C is a set of disjunctions of inequations, the solution set of C is $\mathcal{Q}^n \setminus Sol(\{\overline{c} : \ c \in C\})$.

Let C be a set of temporal constraints in variables $x_1, \ldots, x_n$ which contains only equations, inequalities and inequations (but not disjunctions of inequations). The *temporal constraint network* (TCN) 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. The set of constraints associated with a TCN N will be denoted by $Constraints(N)$.

Let us assume that the set of constraints c_{ij} on $x_j - x_i$ is

$$\{x_j - x_i \leq d_{ij},\ x_j - x_i \geq -d_{ji},\ x_j - x_i \neq r^1_{ji}, \ldots, x_j - x_i \neq r^{h_{ji}}_{ji}\}$$

where $-d_{ji} < r^1_{ji} < \cdots < r^{h_{ji}}_{ji} < d_{ij}$. Then the corresponding TCN N will have an edge $i \rightarrow j$ labeled by the *almost-convex interval*

$$N_{ij} = [-d_{ji}, r^1_{ji}) \cup (r^1_{ji}, r^2_{ji}) \cup \cdots \cup (r^{h_{ji}-1}_{ji}, r^{h_{ji}}_{ji}) \cup (r^{h_{ji}}_{ji}, d_{ij}].$$

Interval N_{ij} can also be open from the right or left depending on the form of the inequalities in c_{ij}. The h_{ij} values $r^1_{ji}, \ldots, r^{h_{ji}}_{ji}$ will be called the *"holes"* of interval N_{ij}. We define a function *holes* such that, for each interval N_{ij},

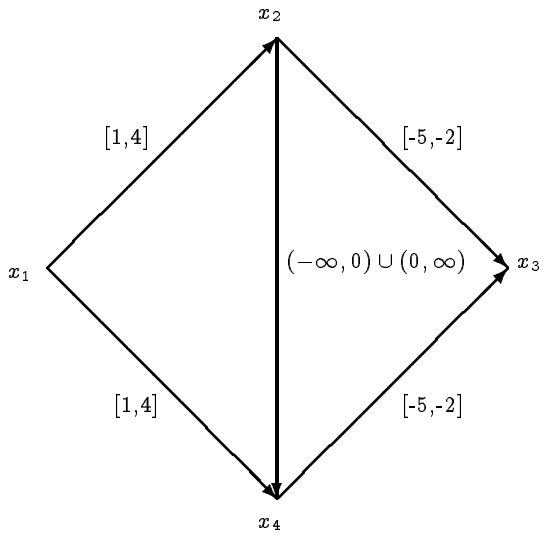

Fig. 1. A temporal constraint network

$holes(N_{ij}) = \{r_{ji}^1, \ldots, r_{ji}^{h_{ji}}\}$. As usual unary constraints will be represented as binary constraints with the introduction of a special variable $x_0 = 0$.

The TCN of Figure 1 represents the constraints

$$1 \leq x_2 - x_1 \leq 4,\ 2 \leq x_2 - x_3 \leq 5,\ 1 \leq x_4 - x_1 \leq 4,\ 2 \leq x_4 - x_3 \leq 5,\ x_4 - x_2 \neq 0.$$

Given an interval I, $conv(I)$ will denote the *convex hull* of I i.e., the minimal (in the set-theoretic sense) convex interval which includes I. Formally,

$$conv([a, r_1) \cup (r_1, r_2) \cup \cdots \cup (r_{k-1}, r_k) \cup (r_k, b]) = [a, b]$$

and $conv(I) = I$ if I is convex. If N is a TCN then $conv(N)$ denotes the TCN which is obtained from N by substituting each interval N_{ij} by $conv(N_{ij})$.

For the case of TCN, the operations of composition and intersection of almost-convex intervals are defined as usual [Mei91b]. The following proposition is straightforward.

Proposition 1. The class of *almost-convex intervals* over $\mathcal{Q}$ is closed under composition and intersection.

The notions of solution, consistency, equivalence and minimality for TCN are defined as usual [Mei91b].

3 Global Consistency of a TCN

We will first consider enforcing global consistency in a TCN.

Notation 3.1 Let C be a set of constraints in variables $x_1, \ldots, x_n$. For any i such that $1 \leq i \leq n$, $C(x_1, \ldots, x_i)$ will denote the set of constraints in C involving only variables $x_1, \ldots, x_i$.

The following definition is from [Dec92].

Definition 2. Let C be a set of constraints in variables $x_1, \ldots, x_n$ and $1 \leq i \leq n$. C is called *i-consistent* iff for every $i-1$ distinct variables $x_1, \ldots, x_{i-1}$, every valuation $u = \{x_1 \leftarrow x_1^0, \ldots, x_{i-1} \leftarrow x_{i-1}^0\}$ such that u satisfies the constraints $C(x_1, \ldots, x_{i-1})$ and every variable x_i different from $x_1, \ldots, x_{i-1}$, there exists a rational number x_i^0 such that u can be extended to a valuation $u' = u \cup \{x_i \leftarrow x_i^0\}$ which satisfies the constraints $C(x_1, \ldots, x_{i-1}, x_i)$. C is called *strong i-consistent* if it is j-consistent for every j, $1 \leq j \leq i$. C is called *globally consistent* iff it is i-consistent for every i, $1 \leq i \leq n$.

Let us present some examples illustrating the above definitions.

Example 1. The constraint set $C = \{x_2 - x_1 \leq 5,\ x_1 - x_3 \leq 2,\ x_5 - x_4 \leq 1,\ x_4 - x_6 \leq 3\}$ is 1- and 2-consistent but not 3-consistent. For example, the valuation $v = \{x_2 \leftarrow 10,\ x_3 \leftarrow 2\}$ satisfies $C(x_2, x_3) = \emptyset$ but it cannot be extended to a valuation which satisfies C.

We can enforce 3-consistency by adding the constraints $x_2 - x_3 \leq 7$ and $x_5 - x_6 \leq 4$ to C. The resulting set is 3-consistent and also globally consistent.

Example 2. The constraint set $C = \{x_2 - x_1 = 5,\ x_1 - x_4 \neq 1\}$ is 1- and 2-consistent but not 3-consistent. For example, the valuation $v = \{x_2 \leftarrow 6,\ x_4 \leftarrow 0\}$ satisfies $C(x_2, x_4) = \emptyset$ but it cannot be extended to a valuation which satisfies C.

We can enforce 3-consistency by adding the constraint $x_2 - x_4 \neq 6$ to C. The resulting set is 3-consistent and also globally consistent.

Example 3. The constraint set $C = \{x_2 - x_1 \leq 5,\ x_1 - x_3 \leq 2,\ x_2 - x_3 \leq 7,\ x_1 - x_4 \neq 1\}$ is strong 3-consistent but not 4-consistent. For example, the valuation $v = \{x_2 \leftarrow 7,\ x_3 \leftarrow 0,\ x_4 \leftarrow 1\}$ satisfies $C(x_2, x_3, x_4) = \{x_2 - x_3 \leq 7\}$ but it cannot be extended to a valuation which satisfies C.

Enforcing 4-consistency amounts to adding the *disjunction*

$$x_2 - x_4 \neq 6 \ \vee \ x_3 - x_4 \neq -1.$$

The resulting set is 4-consistent and also globally consistent.

Example 4. The constraint set $C = \{x_2 - x_1 \leq 5,\ x_1 - x_3 \leq 2,\ x_2 - x_3 \leq 7,\ x_5 - x_4 \leq 1,\ x_4 - x_6 \leq 3,\ x_5 - x_6 \leq 4,\ x_1 - x_4 \neq 1\}$ is strong 3-consistent but not 4-consistent. Adding the constraint $x_2 - x_4 \neq 6 \ \vee \ x_3 - x_4 \neq -1$ (as in the previous example) is not enough. For example, the valuation $v = \{x_5 \leftarrow 2,\ x_6 \leftarrow -2,\ x_1 \leftarrow 2\}$ satisfies $C(x_5, x_6, x_1) = \{x_5 - x_6 \leq 4\}$ but it cannot be extended to a valuation which satisfies $C(x_5, x_6, x_1, x_4)$.

We can enforce 4-consistency by also adding the constraint $x_5 - x_1 \neq 0 \vee x_6 - x_1 \neq -4$ to C. Let the resulting set be C'. C' is strong 4-consistent but not 5-consistent. For example, the valuation $v = \{x_2 \leftarrow 7,\ x_3 \leftarrow 0,\ x_5 \leftarrow 2,\ x_6 \leftarrow -2\}$ satisfies $C(x_2, x_3, x_5, x_6) = \{x_2 - x_3 \leq 7,\ x_5 - x_6 \leq 4\}$ but it cannot be extended to a valuation which satisfies $C(x_2, x_3, x_5, x_6, x_1)$ (or $C(x_2, x_3, x_5, x_6, x_4)$).

We can enforce 5-consistency by adding the constraint

$$x_2 - x_3 \neq 7 \ \vee \ x_5 - x_6 \neq 4 \ \vee \ x_2 - x_5 \neq 5$$

to C'. The resulting constraint set is strong 5-consistent and also globally consistent.

Figure 2 presents algorithm TCN-GCONSISTENCY which enforces global consistency on its input TCN. TCN-GCONSISTENCY takes as input a TCN and returns an equivalent set of temporal constraints which is globally consistent. TCN-GCONSISTENCY's output is not a TCN because, as the above examples indicate, enforcing global consistency might result in the introduction of disjunctions of inequations which cannot be represented by a TCN. TCN-GCONSISTENCY takes advantage of an observation of [Kou92, Imb93]: inequations can be treated *independently* of one another for performing variable elimination.

The algorithm TCN-GCONSISTENCY essentially enforces strong 5-consistency on its input network N. As we will show shortly, this level of local consistency is enough for achieving global consistency. In step 1, TCN-GCONSISTENCY enforces strong 3-consistency on $conv(N)$. This is achieved by running the modified Floyd-Warshall algorithm of [DMP91, KL91] on $conv(N)$. Let N' denote the resulting TCN and $A' = Constraints(N')$. Then $conv(N')$ is minimal and globally consistent [DMP91, KL91].

In step 2, TCN-GCONSISTENCY completes its job. For each $r^g_{ik} \in holes(N_{ki})$ or equivalently for each inequation $x_i - x_k \neq r^g_{ik}$ of $A = Constraints(N)$, TCN-GCONSISTENCY explores the inequalities of A involving x_i and x_k in the following systematic way. Figure 3 illustrates the structure of the subnetworks of N explored in this step. Edges labeled with $\neq$ denote non-convex intervals.

1. If there are inequalities $x_m - x_i \leq d_{im}$ and $x_i - x_l \leq d_{li}$ then step 2.1 ensures that any valuation $v = \{x_l \leftarrow x^0_l, x_m \leftarrow x^0_m, x_k \leftarrow x^0_k\}$, which satisfies $A(x_l, x_m, x_k)$, can be extended to a valuation $v' = v \cup \{x_i \leftarrow x^0_i\}$ which satisfies $A(x_l, x_m, x_k, x_i)$. This is achieved with the introduction of the inequation constraint

   $$x_m - x_k \neq d_{im} + r^g_{ik} \ \vee \ x_l - x_k \neq -d_{li} + r^g_{ik}.$$

   If there are inequalities $x_s - x_k \leq d_{ks}$ and $x_k - x_t \leq d_{tk}$ then step 2.1 also ensures that any valuation $v = \{x_s \leftarrow x^0_s, x_t \leftarrow x^0_t, x_i \leftarrow x^0_i\}$, which satisfies $A(x_s, x_t, x_i)$, can be extended to a valuation $v' = v \cup \{x_k \leftarrow x^0_k\}$ which satisfies $A(x_s, x_t, x_i, x_k)$. This is achieved with the introduction of the inequation constraint

   $$x_s - x_i \neq d_{ks} - r^g_{ik} \ \vee \ x_t - x_i \neq -d_{tk} - r^g_{ik}.$$

Algorithm TCN-GConsistency
Input: A consistent TCN N.
Output: A globally consistent set of constraints equivalent to N.

Method:
1. Step 1: Enforce path consistency on conv(N).
2. For $k, i, j = 1$ to n do
3. $\quad N_{ij} := N_{ij} \oplus (conv(N_{ik}) \otimes conv(N_{kj}))$
4. EndFor

5. Step 2: Enforce global consistency.
6. $C := \emptyset$
7. For $i, k = 1$ to n do
8. $\quad$ For $g = 1$ to h_{ik} do

9. $\quad\quad$ Step 2.1
10. $\quad\quad$ For $m, l = 1$ to n do
11. $\quad\quad\quad$ If N_{im}, N_{li} are closed from the right then
12. $\quad\quad\quad\quad C := C \cup \{x_m - x_k \neq d_{im} + r^g_{ik} \ \vee \ x_l - x_k \neq -d_{li} + r^g_{ik}\}$
13. $\quad\quad\quad$ EndIf
14. $\quad\quad$ EndFor

15. $\quad\quad$ Step 2.2
16. $\quad\quad$ For $m, l, s, t = 1$ to n do
17. $\quad\quad\quad$ If $N_{im}, N_{li}, N_{ks}, N_{tk}$ are closed from the right then
18. $\quad\quad\quad\quad C := C \cup \{x_m - x_l \neq d_{im} + d_{li} \ \vee \ x_s - x_t \neq d_{ks} + d_{tk} \vee$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad x_m - x_s \neq r^g_{ik} + d_{im} - d_{ks}\}$
19. $\quad\quad\quad$ Endif
20. $\quad\quad$ EndFor

21. $\quad$ EndFor
22. EndFor

23. Return $Constraints(N) \cup C$

Fig. 2. Enforcing global consistency

2. If there are inequalities $x_m - x_i \leq d_{im}$, $x_i - x_l \leq d_{li}$, $x_s - x_k \leq d_{ks}$ and $x_k - x_t \leq d_{tk}$ then step 2.2 ensures that any valuation $v = \{x_l \leftarrow x^0_l, x_m \leftarrow x^0_m, x_s \leftarrow x^0_s, x_t \leftarrow x^0_t\}$, which satisfies $A(x_l, x_m, x_s, x_t)$, can be extended to a valuation $v' = v \cup \{x_i \leftarrow x^0_i, x_k \leftarrow x^0_k\}$ which satisfies $A(x_l, x_m, x_s, x_t, x_i, x_k)$. This is achieved with the introduction of the inequation constraint

$$x_m - x_l \neq d_{im} + d_{li} \ \vee \ x_s - x_t \neq d_{ks} + d_{tk} \ \vee \ x_m - x_s \neq r^g_{ik} + d_{im} - d_{ks}.$$

Discussion. It is possible that step 2 of algorithm TCN-GConsistency introduces constraints that are not strictly necessary for enforcing global consistency. This happens when a generated constraint is equivalent to *true* or

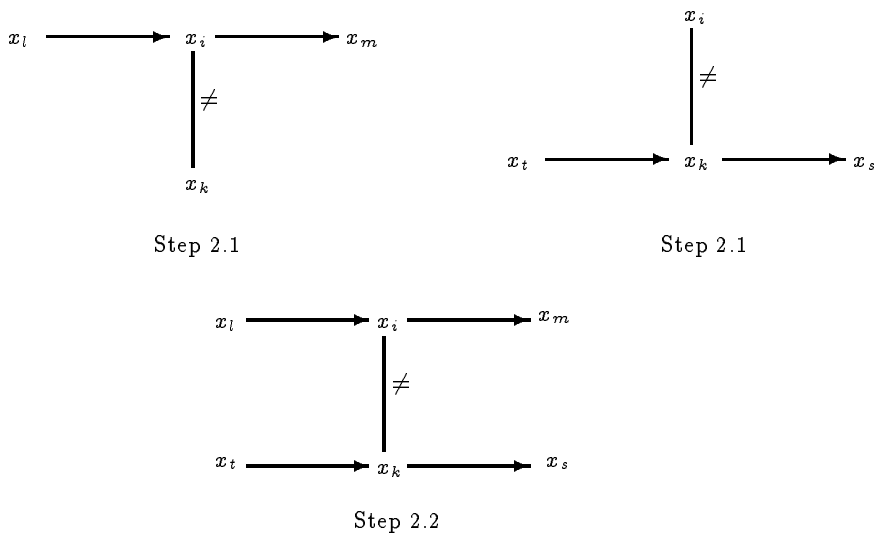

Fig. 3. The subnetworks examined by step 2 of TCN-GCONSISTENCY

when it is implied by another constraint. TCN-GCONSISTENCY can also introduce disjunctions of inequations that are equivalent to inequations (e.g., $x_1 - x_5 \neq 2 \vee x_1 - x_5 \neq 2$). We tolerate this inefficiency because it allow us to present our ideas clearly and minimizes the case analysis in the forthcoming proofs. The reader can consult [Kou94a] for an improved but complicated version of TCN-GCONSISTENCY.

The following theorem demonstrates the correctness of algorithm TCN-GCONSISTENCY.

Theorem 3. *The algorithm* TCN-GCONSISTENCY *is correct i.e., it returns a globally consistent set of constraints equivalent to the input network.*

Corollary 4. *Strong 5-consistency is necessary and sufficient for achieving global consistency of a TCN.*

Proof. Example 4 shows the necessity of achieving strong 5-consistency. The sufficiency follows from the previous theorem; the algorithm TCN-GCONSISTENCY essentially achieves strong 5-consistency.

The following theorem gives the complexity of TCN-GCONSISTENCY.

Theorem 5. *The running time of* TCN-GCONSISTENCY *is* $O(Hn^4)$ *where* H *is the number of inequations and* n *is the number of variables in the input TCN.*

Proof. Step 1 takes $O(n^3)$ time, step 2.1 takes $O(Hn^2)$ time and step 2.2 takes $O(Hn^4)$ time.

4 Computing Minimal TCN

In this section we present an algorithm for computing the minimal network equivalent to a given TCN. The algorithm TCN-MINIMAL shown in Figure 4 is essentially a by-product of algorithm TCN-GCONSISTENCY. Notice that the constraints in the minimal TCN will be only inequalities and inequations. Therefore an algorithm for computing the minimal TCN can be constructed if we start with TCN-GCONSISTENCY and omit any part that generates a disjunction of inequations. This can be achieved by a detailed analysis of Step 2 of TCN-GCONSISTENCY.

Before proceeding with the algorithm let us clarify our notion of minimality. We will use the traditional definition of a minimal network which is as follows [DMP91, Mei91b]. A binary constraint network N is called *minimal* if there is no tighter network equivalent to it.[4] However this definition is not the only one available in the literature. [Dec92] defines a binary constraint network to be minimal if every subnetwork of size two is consistent relative to the full network (i.e, if any instantiation of two variables which is locally consistent can be extended to a solution of the full network). In our case these two definitions are not equivalent. Consider the constraint set $C = \{x_1 \leq x_2,\ x_2 \leq 5,\ x_2 \neq x_3\}$. If we adopt the first definition the minimal TCN N for C has $N_{13} = (-\infty, +\infty)$. But the valuation $v = \{x_1 \leftarrow 5,\ x_3 \leftarrow 5\}$ which satisfies the constraint express by N_{13} cannot be extended to a solution of C (C implies $x_3 \neq x_1 \ \vee \ x_3 \neq 5$ but this cannot be represented by N).

TCN-MINIMAL computes the minimal TCN in four steps. In the first step, we enforce path-consistency on the convex part $conv(N)$ of the input network N. In Step 2, TCN-MINIMAL performs constraint propagation involving equalities from $conv(N)$ and inequations from L. More precisely, for every inequation $x_i - x_j \neq r \in L$ and every equality $x_k - x_i = d_{ki} \in conv(N)$ Step 2.1 adds inequation $x_k - x_j \neq r + d_{ki}$ to N. Similarly, for every inequation $x_i - x_j \neq r \in L$ and every equality $x_j - x_k = d_{jk} \in conv(N)$ Step 2.2 adds inequation $x_i - x_k \neq r + d_{jk}$ to N.

In Step 3, TCN-MINIMAL considers subnetworks of N like the ones considered by Step 2.2 of TCN-GCONSISTENCY (see Figure 3) when $l = t$ and $m = s$.[5] In this case the constraint generated by TCN-GCONSISTENCY is equivalent to a binary inequation thus it should be reflected in the minimal TCN. This can be shown as follows. If $l = t$ and $m = s$ then Step 2.2 of TCN-GCONSISTENCY examines the constraint set

$$\{x_m - x_i \leq d_{im},\ x_i - x_l \leq d_{li},\ x_m - x_k \leq d_{km},\ x_k - x_l \leq d_{lk},\ x_i - x_k \neq r\}$$

and generates the constraint

$$x_m - x_l \neq d_{im} + d_{li} \ \vee \ x_m - x_l \neq d_{km} + d_{lk} \ \vee \ 0 \neq r + d_{im} - d_{km}.$$

[4] A binary constraint network M is *tighter* than N if for every i, j, $M_{ij} \subseteq N_{ij}$.

[5] The case where $l = s$ and $m = t$ does not need to be considered because it leads to disjunctions of inequations that are equivalent to *true*.

Algorithm TCN-MINIMAL
Input: A consistent TCN N.
Output: A minimal TCN equivalent to N.

Method:
Step 1: Enforce path consistency on $conv(N)$ (as in Step 1 of TCN-GCONSISTENCY).

Step 2:
Let L be the list of inequations in N.
For every (i, j, r) in L do
  For $k = 1$ to n do
    If $-d_{ik} = d_{ki}$ then
      $N_{kj} := N_{kj} \oplus ((-\infty, r + d_{ki}) \cup (r + d_{ki}, \infty))$
    EndIf
  EndFor

  For $k = 1$ to n do
    If $-d_{kj} = d_{jk}$ then
      $N_{ik} := N_{ik} \oplus ((-\infty, r + d_{jk}) \cup (r + d_{jk}, \infty))$
    EndIf
  EndFor
EndFor

Step 3:
For every (i, k, r) in L do
  For $m, l = 1$ to n do
    If $N_{im}, N_{li}, N_{km}, N_{lk}$ are closed from the right and $m \neq l$ and
      $r + d_{im} - d_{km} = 0$ and $d_{li} + d_{im} = d_{lk} + d_{km}$ then
        $N_{lm} := N_{lm} \oplus ((-\infty, d_{li} + d_{im}) \cup (d_{li} + d_{im}, \infty))$
    EndIf
  EndFor
EndFor

Step 4:
For every (i, k, r) in L do
  For $m, t = 1$ to n do
    If $-d_{mi} = d_{im}$ and $-d_{kt} = d_{tk}$ then
      $N_{tm} := N_{tm} \oplus ((-\infty, d_{im} + d_{tk} + r) \cup (d_{im} + d_{tk} + r, \infty))$
    EndIf
  EndFor
EndFor

Return N

Fig. 4. A minimal TCN algorithm

If $r + d_{im} - d_{km} = 0$ and $d_{im} + d_{li} = d_{km} + d_{lk}$ then the above constraint becomes $x_m - x_l \neq d_{im} + d_{li}$ otherwise it evaluates to *true*.

Finally, in Step 4 TCN-MINIMAL considers subnetworks of N like the ones considered by Step 2.2 of TCN-GCONSISTENCY when $l = m$ and $t = s$. In this case the constraint generated by TCN-GCONSISTENCY is also equivalent to a binary inequation. This can be shown as follows. If $l = m$ and $t = s$ then Step 2.2 of TCN-GCONSISTENCY considers the constraint set

$$\{x_m - x_i \leq d_{im},\ x_i - x_m \leq d_{mi},\ x_t - x_k \leq d_{kt},\ x_k - x_t \leq d_{kt},\ x_i - x_k \neq r\}$$

and generates the constraint

$$-d_{im} \neq d_{mi} \ \vee \ -d_{kt} \neq d_{tk} \ \vee \ x_m - x_t \neq d_{im} + d_{tk} + r.$$

If $-d_{im} = d_{mi}$ and $-d_{kt} = d_{tk}$ then this constraint becomes $x_m - x_t \neq d_{im} + d_{tk} + r$ otherwise it evaluates to *true*.

The following lemma summarizes the most important property of algorithm TCN-MINIMAL.

Lemma 6. *If* TCN-GCONSISTENCY *computes a binary inequation c and N is the output of* TCN-MINIMAL *then $c \in Constraints(N)$.*

Proof. By case analysis of Step 2 of algorithm TCN-GCONSISTENCY.

The following theorem shows that the algorithm TCN-MINIMAL is correct and gives its complexity.

Theorem 7. *The algorithm* TCN-MINIMAL *computes the minimal TCN equivalent to its input in $O(\max(Hn^2, n^3))$ time where H is the number of inequations and n is the number of variables.*

Proof. The correctness part follows from the previous lemma. The complexity bound is achieved by either maintaining L explicitly or by having an adjacency list recording the inequations for every node of N.

An algorithm with the same complexity has also been discovered independently by Gerevini and Cristani [GC95]. without prior analysis of the global consistency problem. A careful comparison of the two algorithms shows that Step 2 of TCN-MINIMAL computes *3-path implicit inequations*, Step 3 deals with *forbidden subgraphs* and Step 4 deals with *4-path implicit inequations* (this new terminology comes from [GC95] and the reader is referred there for more details).

Independently, Isli has studied a subclass of the class of temporal constraints that we consider in this section [Isl94]. Isli does not consider inequations of the form $x - y \neq r$ where $r \neq 0$, and achieves the same complexity bound for computing the minimal network.

5 Global Consistency of Point Algebra Networks

We will now turn our attention to an important subset of TCN: the point algebra (PA) networks introduced in [VKvB89]. A *point algebra network (PAN)* is a labeled directed graph where nodes represent variables and edges represent PA constraints. The labels of the edges are chosen from the set of relations $\{<, \leq, > , \geq, =, \neq, ?\}$. The symbol ? is used to label an edge $i \rightarrow j$ whenever there is no constraint between variables x_i and x_j.

Van Beek and Cohen have studied PAN in detail [vBC90, vB92]. Theorem 7 and the following results of [vB92] show that the complexity of computing the minimal network does not change when we go from PAN to TCN.

Theorem 8. *The minimal network equivalent to a PAN can be computed in* $O(\max(Hn^2, n^3))$ *time where* H *is the number of edges labeled with* $\neq$ *and* n *is the number of nodes.*

In [vBC90] the minimal network is computed by algorithm AAC. However, in the proof of correctness of AAC (Theorem 4 of [vBC90]), Van Beek and Cohen suggest that the algorithm for computing the minimal network of a given PAN also achieves global consistency. This is not true and has been corrected in [vB90]. As the following example demonstrates, the introduction of disjunctions of inequations is necessary for achieving global consistency in this case. But algorithm AAC of [vBC90] does not introduce such disjunctions so it cannot achieve global consistency.

Example 5. For the PAN with constraints

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

AAC will also introduce constraints $x_1 \leq x_3,\ x_4 \leq x_6$. The resulting PAN is strong 3-consistent but *not* globally consistent. This can be demonstrated via an argument similar to the one for Example 4. If we enforce strong 5-consistency with the addition of constraints $x_1 \neq x_5 \ \vee \ x_3 \neq x_5,\ x_4 \neq x_2 \ \vee \ x_6 \neq x_2$ and $x_1 \neq x_3 \ \vee \ x_1 \neq x_4 \ \vee \ x_1 \neq x_6$, then the resulting set is globally consistent.

Global consistency of PAN can be enforced by TCN-GConsistency if PAN are represented by their equivalent TCN. Global consistency of PAN has also been discussed (under the name decomposability) in Section 5 of [Kou92] and algorithm Decompose has been proposed for achieving this task. The algorithm is correct but it adopts a representation which is rather inappropriate for the task at hand and leads to a complexity bound which is not the tightest. The following theorem summarizes the result of Section 3 as it applies to PAN.

Theorem 9. *Strong 5-consistency is necessary and sufficient for achieving global consistency in PAN. Strong 5-consistency can be enforced in* $O(Hn^4)$ *time where* H *is the number of edges labeled with* $\neq$ *and* n *is the number of nodes.*

Let us now comment on some observations of Dechter [Dec92] on the problem of enforcing global consistency in PAN. [Dec92] discusses global consistency in general constraint networks with finite variable domains. The most important result of [Dec92] is the following. If N is a constraint network with constraints of arity r or less and domains of size k or less which is strongly $(k(r-1)+1)$-consistent, then N is globally consistent.

The above result can be applied to PAN if PAN are redefined as "traditional" constraint networks where variables represent relations between two points and constraints are defined by the transitivity table of [VKvB89]. This representation yields a constraint network with $k = 3$ and $r = 3$. Dechter's result now gives us the following. If strong 7-consistency in PAN can be enforced with ternary constraints then strong 7-consistency implies global consistency. Dechter uses the aforementioned incorrect assertion of [vBC90] to conclude that strong 7-consistency in the traditional formulation of PAN can be enforced with ternary constraints. Thus she also concludes that in the traditional formulation strong 7-consistency implies global consistency [Dec92, page 100]. In the light of Theorem 9, Dechter's conclusion remains unjustified.

6 The General Case

Let us now consider enforcing global consistency when disjunctions of inequations are allowed in the given constraint set.

Example 6. The constraint set

$$C = \{x_5 \leq x_1,\ x_1 \leq x_6,\ x_5 \leq x_6,\ x_7 \leq x_3,\ x_3 \leq x_8,\ x_7 \leq x_8,\ x_9 \leq x_2,\\ x_2 \leq x_{10},\ x_9 \leq x_{10},\ x_1 \neq y \vee x_2 \neq z \vee x_3 \neq w\}$$

is strong 7-consistent but not 8-consistent. For example, the valuation

$$v = \{y \leftarrow 0,\ z \leftarrow 0,\ w \leftarrow 0,\ x_2 \leftarrow 0,\ x_3 \leftarrow 0,\ x_5 \leftarrow 0,\ x_6 \leftarrow 0\}$$

satisfies $C(y, z, w, x_2, x_3, x_5, x_6) = \{x_5 \leq x_6\}$ but it cannot be extended to a valuation which satisfies $C(y, z, w, x_2, x_3, x_5, x_6, x_1)$. We can enforce 8-consistency by adding the constraints

$$x_5 \neq y \vee x_6 \neq y \vee x_2 \neq z \vee x_3 \neq w$$
$$x_1 \neq y \vee x_9 \neq z \vee x_{10} \neq z \vee x_3 \neq w$$
$$x_1 \neq y \vee x_2 \neq z \vee x_7 \neq w \vee x_8 \neq w.$$

The resulting set is strong 8-consistent but not 9-consistent. We can enforce 9-consistency by adding the constraints

$$x_5 \neq y \vee x_6 \neq y \vee x_9 \neq z \vee x_{10} \neq z \vee x_3 \neq w$$
$$x_1 \neq y \vee x_9 \neq z \vee x_{10} \neq z \vee x_7 \neq w \vee x_8 \neq w$$
$$x_5 \neq y \vee x_6 \neq y \vee x_2 \neq z \vee x_7 \neq w \vee x_8 \neq w.$$

Algorithm GCONSISTENCY
Input: A set of temporal constraints $C = C_i \cup C_d$ where C_i is a set of inequalities and C_d is a set of disjunctions of inequations.
Output: A globally consistent set of constraints equivalent to C.

Method:
Step 1: Enforce strong 3-consistency on C_i.
Let N be the TCN corresponding to C_i.
For $k, i, j = 1$ to n do
    $N_{ij} := N_{ij} \oplus (N_{ik} \otimes N_{kj})$
EndFor

Step 2: Enforce global consistency
$C'_d := \emptyset$
For each $c \in C_d$ do
    For all subsets $\{k_1, \ldots, k_i\}$ of the set of variables of c do
        For $m_1, \ldots, m_i, l_1, \ldots, l_i = 1$ to n do
            If $N_{k_1 m_1}, \ldots, N_{k_i m_i}, N_{l_1 k_1}, \ldots, N_{l_i k_i}$ are closed from the right then
                Eliminate variables $x_{k_1}, \ldots, x_{k_i}$ from
                $\overline{c},\ x_{m_1} - x_{k_1} = d_{k_1 m_1},\ x_{k_1} - x_{l_1} = d_{l_1 k_1}, \ldots, x_{m_i} - x_{k_i} = d_{i m_i},$
                $x_{k_i} - x_{l_i} = d_{l_i k_i}$
                to obtain c'
                $C'_d := C'_d \cup \{\overline{c'}\}$
            Endif
        EndFor
    EndFor
EndFor

Return $Constraints(N) \cup C_d \cup C'_d$

Fig. 5. Enforcing global consistency

The resulting set is strong 9-consistent but not 10-consistent. We can enforce 10-consistency by adding the constraint

$$x_5 \neq y \ \vee\ x_6 \neq y \ \vee\ x_9 \neq z \ \vee\ x_{10} \neq z \ \vee\ x_7 \neq w \ \vee\ x_8 \neq w.$$

The resulting set is strong 10-consistent and also globally consistent.

Figure 5 presents algorithm GCONSISTENCY which enforces global consistency on its input constraint set. The reader should have no problem understanding the details of GCONSISTENCY since it is a straightforward generalization of algorithm TCN-GCONSISTENCY.

The following theorem demonstrates the correctness of GCONSISTENCY.

Theorem 10. *The algorithm* GCONSISTENCY *is correct i.e., it returns a globally consistent set of constraints equivalent to the input one.*

In essence, algorithm GCONSISTENCY achieves strong $2V + 1$-consistency where V is the maximum number of variables in any disjunction of inequations. Thus we have the following corollary.

Corollary 11. *Let C be a set of temporal constraints. If C is $2V+1$-consistent, where V is the maximum number of variables in any disjunction of inequations, then C is globally consistent.*

The time complexity of GConsistency is exponential in V. However, if V is *fixed* then the time complexity of GConsistency is polynomial in the number of variables and the number of constraints in C. This has an interesting consequence for variable elimination due to its relation to global consistency.

Corollary 12. *Let C be a set of temporal constraints such that the number of variables in every disjunction of inequations is fixed. Eliminating any number of variables from C can be done in time polynomial in the number of variables and the number of constraints. In addition, the resulting constraint set has size polynomial in the same parameters.*

Proof. Let $x_1, \ldots, x_n$ be all the variables of C. When V is fixed, the size of the constraint set generated by algorithm GConsistency is polynomial in the number of variables and the number of constraints. If C is globally consistent then for any i such that $1 \leq i \leq n$, $C(x_1, \ldots, x_i)$ is the projection of $Sol(C)$ on $\{x_1, \ldots, x_i\}$. Thus we can eliminate variables $x_1, \ldots, x_i$ from C by running GConsistency on C and returning $C(x_{i+1}, \ldots, x_n)$. This algorithm takes time polynomial in the number of variables and the number of constraints.

The above corollary complements Theorem 4.4 of [Kou92] which states that variable elimination can result in constraint sets with an exponential number of disjunctions of inequations.

7 Conclusions

We discussed the problem of enforcing global consistency in sets of quantitative temporal constraints which include inequalities, inequations and disjunctions of inequations. It would be interesting to combine our results with the results of [Mei91b] in order to identify classes of qualitative and quantitative point/interval constraints where global consistency is tractable.

8 Acknowledgements

I would like to thank Peter van Beek, Amar Isli, Alfonso Gerevini and the reviewers of CP95 for interesting discussions concerning the topics of this paper. Gerevini pointed out to us that an earlier version of Lemma 6 had been stated inaccurately.

References

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

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

[DMP91] Rina Dechter, Itay Meiri, and Judea 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.

[GC95] A. Gerevini and M. Cristani. Reasoning with Inequations in Temporal Constraint Networks. Technical report, IRST - Instituto per la Ricerca Scientifica e Tecnologica, Povo TN, Italy, 1995. A shorter version will be presented at the IJCAI-95 Workshop on Spatial and Temporal Reasoning, August 1995, Montreal, Canada.

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

[GSS93] A. Gerevini, L. Schubert, and S. Schaeffer. Temporal Reasoning in Timegraph I-II. *SIGART Bulletin*, 4(3):21–25, 1993.

[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.

[Isl94] A. Isli. Constraint-Based Temporal Reasoning: a Tractable Point Algebra Combining Qualitative, Metric and Holed Constraints. Technical Report 94-06, LIPN-CNRS URA 1507, Inst. Galilée, Université Paris-Nord, 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.

[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. Available by anonymous ftp from host passion.doc.ic.ac.uk, file IC-Parc/Papers/M.Koubarakis/phd-thesis.ps.Z.

[Kou94b] Manolis 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.

[Kou94c] Manolis 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.

[LM89] Jean-Louis Lassez and Ken McAloon. A Canonical Form for Generalized Linear Costraints. Technical Report RC15004 (#67009), IBM Research Division, T.J. Watson Research Center, 1989.

[Mei91a] I. Meiri. Combining Qualitative and Quantitative Constraints in Temporal Reasoning. Technical Report R-160, Cognitive Systems Laboratory, University of California, Los Angeles, 1991.

[Mei91b] I. Meiri. Combining Qualitative and Quantitative Constraints in Temporal Reasoning. In *Proceedings of AAAI-91*, pages 260–267, 1991.

[vB90] Peter van Beek. Exact and Approximate Reasoning About Qualitative Temporal Relations. Technical Report TR 90-29, Department of Computing Science, University of Alberta, August 1990.

[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.

[VK86] Marc Vilain and Henry Kautz. Constraint Propagation Algorithms for Temporal Reasoning. In *Proceedings of AAAI-86*, pages 377–382, 1986.

[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.

This article was processed using the LaTeX macro package with LLNCS style