TECHNICAL REPORT
R-151 July, 1990

Faster Constraint Satisfaction Algorithms for Temporal Reasoning[*]

Itay Meiri and Judea Pearl

Cognitive Systems Laboratory
Computer Science Department
University of California
Los Angeles, CA 90024

1. Introduction

Vilain and Kautz [1986] presented a ~~time~~ point algebra for temporal reasoning. The elements of the algebra are the possible relations that can exist between points of time; they correspond to ~~all~~ the subsets of $\{<,>,=\}$, as shown in Figure 1. The singleton subsets of $\{<,=,>\}$ represent **basic relations** – *precedes*, *same* and *follows*. The elements of the algebra are disjunctions of these basic relations. For example, $x \neq y$ represents the disjunction *x (precedes or follows) y*, or $x\ \{<,>\}\ y$ in a set notation[1].

relation	set	symbol
x () y	$\{\}$	$\varnothing$
x (precedes) y	$\{<\}$	$<$
x (same) y	$\{=\}$	$=$
x (follows) y	$\{>\}$	$>$
x (precedes or same) y	$\{<,=\}$	$\leq$
x (follows or same) y	$\{>,=\}$	$\geq$
x (precedes or follows) y	$\{<,>\}$	$\neq$
x (precedes or same or follows) y	$\{<,=,>\}$	?

Figure 1. Elements of the point algebra.

A (binary) **constraint network** over the point algebra involves a set of variables, $\{X_1, \ldots, X_n\}$, each representing a time point, and a set of constraints expressed in the point algebra. Each constraint is of the form $X_i\ R\ X_j$, where R is a relation of the algebra. A constraint network is associated with a *directed* **constraint graph**, where node i represents variable X_i, and

* This work was supported in part by the National Science Foundation Grant # IRI 8815522 and an Air Force Grant # AFOSR 880177

1. We shall use symbol and set notation interchangeably.

an arc $i \longrightarrow j$ represents a constraint, C_{ij}, between X_i and X_j; the arc is labeled by the corresponding point algebra element[2].

Example 1. A constraint network is shown in Figure 2.

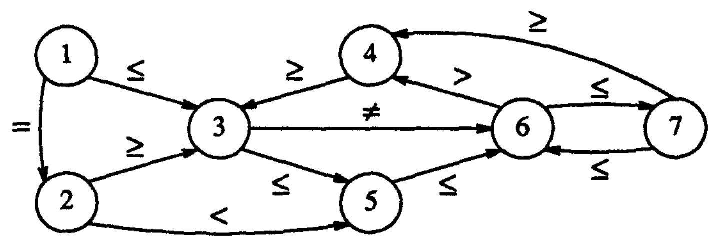

Figure 2. A constraint network over the point algebra.

A tuple which satisfies all the constraints is called a **solution.** The network is **consistent** if at least on solution exists. A network whose arcs are labeled only by basic relations is called a **singleton labeling.** A **consistent label** for arc $i \longrightarrow j$ is a basic relation that labels $i \longrightarrow j$ in some *consistent* singleton labeling. The **minimal constraint** for arc $i \longrightarrow j$ is the set of all consistent labels for $i \longrightarrow j$. The **minimal network** of a given network consists of the minimal constraints for all arcs.

Given a network G, we consider the following tasks:

1. Determining consistency.

2. Determining whether a given label is consistent for a given arc.

3. Determining consistency of a subset of arc labels; namely, whether they can be extended to a consistent singleton labeling.

4. Computing the minimal constraint for a given arc.

2. The convention is that arcs labeled by ? (the universal constraint) are omitted. Also, we assume that no arc is labeled by ∅ (in such case the network is trivially inconsistent).

5. Computing the minimal network.

Some of these tasks were addressed by Vilain and Kautz [1986] and vanBeek [1989]. In particular, vanBeek [1989] presented an $O(n^4)$ algorithm (three and four consistency) which computes the minimal network and, as a by product, determines consistency. He also devised an $O(n^2)$ algorithm to compute minimal arc constraints; however, this algorithm is exact only for a subset of the point algebra that excludes "$\neq$".

This paper presents efficient algorithms for the above tasks. The main results are that tasks 1–4 can be solved in $O(|E|)$ time, and task 5 – computing the minimal network – can be carried out in $O(|E||V|^2)$ time. Improvements with respect to existing algorithms are the following:

1. We can determine consistency in $O(|E|)$ time, instead of $O(|V|^4)$ time using four consistency.

2. We can compute minimal arc constraints in the *full* point algebra in $O(|E|)$ time.

3. We can compute the minimal network in $O(|E||V|^2)$ time, instead of $O(|V|^4)$ using four consistency. This improvement is significant mainly in sparse networks.

The proposed algorithms are easy to implement, as they merely involve a simple graph-based criterion.

2. Solving Reasoning Tasks in the Point Algebra

We now present an algorithm which determines consistency of a given network in $O(|E|)$ time. The algorithm makes use of a **precedence graph**, which displays precedence relations between variables. It has the same node set as the input network, and an arc $i \rightarrow j$ means that X_i precedes or is the same as X_j in any solution to G. Given a network $G = (V,E)$, the precedence graph,

$G_< = (V, E_<)$, is defined by the construction procedure shown in Figure 3. The precedence graph of Example 1 is depicted in Figure 4.

Construction of precedence graph

```
1. $E_< \leftarrow \emptyset$;
2. for each arc $i \rightarrow j$ in $E$ do
3.     if $C_{ij}$ is "<" or "≤"
4.         then add $i \rightarrow j$ to $E_<$;
5.     if $C_{ij}$ is ">" or "≥"
6.         then add $j \rightarrow i$ to $E_<$;
7.     if $C_{ij}$ is "="
8.         then add both $i \rightarrow j$ and $j \rightarrow i$ to $E_<$;
9. end;
```

Figure 3. A Precedence graph construction algorithm.

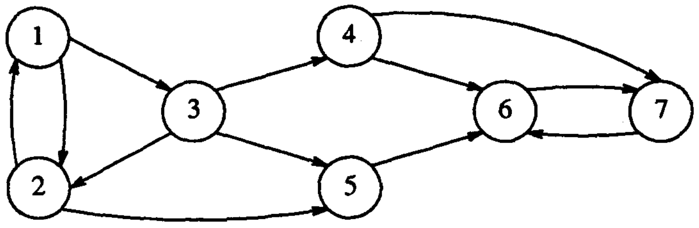

Figure 4. Precedence graph of Example 1.

The following properties of the precedence graph can be easily verified.

Proposition 1. Let $G_<$ be the precedence graph of a given network G. Then,

i. If there exists a directed path from i to j in $G_<$, then $X_i \leq X_j$ in any solution to G.

ii. If i and j belong to the same strongly-connected component[3] in $G_<$, then $X_i = X_j$ in any solution to G. If G is consistent, then the minimal constraint for arc $i \rightarrow j$ is $\{=\}$.

3. Nodes i and j belong to the same strongly-connected component, if there exist directed paths from i to j and from j to i.

The next theorem provides a necessary and sufficient condition for consistency in the point algebra.

Theorem 1. let $G = (V,E)$ be a constraint network over the point algebra, and let $G_< = (V,E_<)$ be its precedence graph. G is consistent if and only if $\{=\} \subseteq C_{ij}$, for any pair of nodes, i and j, that belong to the same strongly-connected component in $G_<$.

Proof. The only if part follows from Proposition 1; we now prove the if part. Assume that $\{=\} \subseteq C_{ij}$ for any pair of nodes, i and j, that belong to the same strongly-connected component in $G_<$. We shall show that G is consistent by constructing a solution.

Consider the **reduced graph**, G_r, for $G_<$ (called the superstructure of G in [Even 1979]); it represents the interconnections among the strongly-connected components of $G_<$. The nodes of G_r are the strongly-connected components, and a directed arc $C_i \rightarrow C_j$ implies that there exists an edge $u \rightarrow v$ in $G_<$, where $u \in C_i$ and $v \in C_j$. It is well-known that G_r forms a DAG (directed acyclic graph). Therefore, its nodes can be topologically ordered; namely, they can be given distinct weights w, such that if there exists an arc $i \rightarrow j$, then $w_i < w_j$. We next show that assigning value w_i to all nodes in component C_i yields the desired solution.

We must show that the assignment is permitted by all constraints. Consider an arbitrary constraint, C_{ij}, in G. There are two cases depending on i and j.

Case 1: Nodes i and j belong to the same component in $G_<$. Therefore, $w_i = w_j$. Since $\{=\} \subseteq C_{ij}$, the assignment is permitted by C_{ij}.

Case 2: i and j belong to two distinct components in $G_<$, C_i and C_j, respectively. There are four cases depending on C_{ij}.

Case 2a: C_{ij} is "=". By the construction of $G_<$, i and j belong to the same strongly-connected component; contradiction.

Case 2b: C_{ij} is "$\neq$" or ?. Since $w_i \neq w_j$, the assignment is permitted by C_{ij}.

Case 2c: C_{ij} is "<" or "$\leq$". By the construction of $G_<$, there exists an arc $i \rightarrow j$ in $G_<$, and thus arc $C_i \rightarrow C_j$ in G_r. Hence, $w_i < w_j$, and the assignment is consistent.

Case 2d: C_{ij} is ">" or "$\geq$". By the construction of $G_<$, there exists an arc $j \rightarrow i$ in $G_<$, and thus arc $C_j \rightarrow C_i$ in G_r. Hence, $w_j < w_i$, and the assignment is consistent.

Since the assignment satisfies all the constraints it constitutes a solution, and thus G is consistent. □

Consider the precedence graph of Example 1. There are four strongly-connected components – $C_1 = \{1,2,3\}$, $C_2 = \{4\}$, $C_3 = \{5\}$ and $C_4 = \{6,7\}$. Clearly, the condition of Theorem 1 is satisfied, and thus the network is consistent. To illustrate the construction used in Theorem 1, consider the reduced graph shown in Figure 5, and the ordering (C_1, C_2, C_3, C_4). The resulting solution is:

$$\{X_1 = X_2 = X_3 = 1, X_4 = 2, X_5 = 3, X_6 = X_7 = 4\}.$$

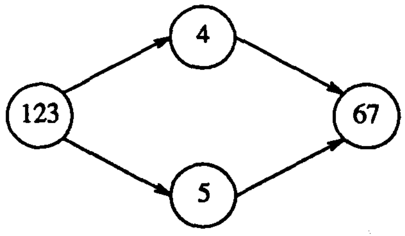

Figure 5. Reduced graph of Example 1.

Theorem 1 provides an effective test for determining consistency. The test is implemented in algorithm CONS, shown in Figure 6.

Theorem 2. Algorithm CONS determines consistency of a given network, $G = (V,E)$, in $O(|E|)$ time.

]

Determining consistency in the point algebra

```
1. $Consistent \leftarrow True$;
2. construct the precedence graph $G_<$;
3. find the strongly-connected components;
4. for each arc $i \rightarrow j$ in $E$ do
5.     if $i$ and $j$ belong to the same strongly-connected component then
6.         if $(C_{ij} \cap \{=\}) = \emptyset$ then
7.             $Consistent \leftarrow False$ and exit;
8. end;
```

Figure 6. CONS – an algorithm to determine consistency in the point algebra.

Proof. The correctness of CONS follows immediately from Theorem 1. The construction of the precedence graph (Step 1) takes time proportional to the number of edges; finding the strongly-connected components (Step 3) can be done in time $O(|E|)$ [Even 1979]; Steps 4-8 also take $O(|E|)$ time. Hence, the total running time is $O(|E|)$. □

Having determined consistency, we can now solve the remaining tasks.

1. To test whether a given relation, R, is consistent for arc $i \rightarrow j$, we apply CONS to the network with arc $i \rightarrow j$ labeled by R. Likewise, to determine whether a given partial labeling can be extended to a consistent labeling, we apply CONS to the network labeled with the given relations. Clearly, these tasks require $O(|E|)$ time.

2. To compute the minimal constraint for arc $i \rightarrow j$, we simply find which of the three basic relations are consistent for $i \rightarrow j$. Again, this task requires $O(|E|)$ time.

3. To compute the minimal network, we compute the minimal arc constraints for all pairs of variables. This task can be carried out in $O(|E| |V|^2)$ time.

Algorithm CONS and its derivatives are easy to implement. Compared to four consistency, they are conceptually meaningful, as they make use of the precedence graph which portrays temporal precedence relations; the operations of four consistency, on the other hand, are not given any temporal meaning.

3. Summary

We presented efficient algorithms for solving reasoning tasks in constraint networks over Vilain and Kautz's point algebra. We reduced the time complexity of determining consistency, finding minimal arc constraints and testing consistency of arc labels to $O(|E|)$, and reduced the complexity of computing the minimal network to $O(|E||V|^2)$ time. The algorithms are conceptually simple, being based on a simple graph inspection criterion.

References

1. Even S., Graph Algorithms, Computer Science Press, Rockville, Md, 1979.

2. vanBeek P., Approximation Algorithms for Temporal Reasoning, Proc. of IJCAI-89, Detroit, MI, 1291-1296, 1989.

3. Vilain M. and Kautz H., Constraint Propagation Algorithms for Temporal Reasoning, Proc. of AAAI-86, 377-382, 1986.