Integrating Metric and Qualitative Temporal Reasoning

Henry A. Kautz
AT&T Bell Laboratories
Murray Hill, NJ 07974
kautz@research.att.com

Peter B. Ladkin
International Computer Science Institute
Berkeley, CA 94704
ladkin@icsi.berkeley.edu

abstract

Research in Artificial Intelligence on constraint-based representations for temporal reasoning has largely concentrated on two kinds of formalisms: systems of simple linear inequalities to encode metric relations between time points, and systems of binary constraints in Allen's temporal calculus to encode qualitative relations between time intervals. Each formalism has certain advantages. Linear inequalities can represent dates, durations, and other quantitive information; Allen's qualitative calculus can express relations between time intervals, such as disjointedness, that are useful for constraint-based approaches to planning.

In this paper we demonstrate how metric and Allen-style constraint networks be integrated in a constraint-based reasoning system. The highlights of the work include a simple but powerful logical language for expressing both quantitative and qualitative information; translations algorithms between the metric and Allen sublanguages that entail a minimal loss of information; and a constraint-propagation procedure for problems expressed in a combination of metric and Allen constraints.

Topic: General Knowledge Representation, Temporal Reasoning.
Secondary Topics: Automated Reasoning, Constraint Satisfaction.

This paper will also be presented at the AAAI Spring Symposium on Constraint-Based Reasoning.

1 Introduction

Research in Artificial Intelligence on constraint-based representations for temporal reasoning has largely concentrated on two kinds of formalisms: systems of simple linear inequalities [Malik and T.O., 1983, Valdes-Perez, 1986, Dechter *et al.*, 1989] to encode metric relations between time points, and systems of binary constraints in Allen's temporal calculus [Allen, 1983, Vilain *et al.*, 1989, Ladkin and Madux, 1987, van Beek and Cohen, 1989] to encode qualitative relations between time intervals. Each formalism has certain advantages. Linear inequalities can represent dates, durations, and other quantitive information that appears in real-world planning and scheduling problems. Allen's qualitative calculus can express certain crucial relations between time intervals, such as *disjointedness*, that *cannot* be expressed by any collection of simple linear inequalities. Such disjointedness constraints form the basis for constraint-based approaches to planning [Allen, 1991].

In this paper we demonstrate how metric and qualitative knowledge can be integrated in a constraint-based reasoning system. One approach to this problem (as used, for example, in the "time map" system of Dean and McDermott [87]) is to directly attach rules that enforce disjointedness constraints to a network of linear inequalities. One limitations of such an approach is that some natural qualitative inferences are not performed: for example, the facts that interval i is during j and j is disjoint from k are not combined to reach the conclusion i is disjoint from k. Another disadvantage is that it is often more convenient for the user to enter assertions in a qualitative language, even if they can be represented numerically.

Instead of try to augment a single reasoning system, we will take an approach briefly suggested by Dechter, Meiri, and Pearl [89] (henceforth "DMP"), and combine a metric reasoning system with a full Allen-style constraint network. The contributions of our research include the following:

1. A simple but powerful logical language $\mathcal{L}$ for expressing both quantitative and qualitative information. The language subsumes both networks of two-variable difference inequalities (called $\mathcal{L}_M$) and networks of binary Allen constraints (called $\mathcal{L}_A$), but is much more powerful than either. The axioms of Allen's temporal calculus are *theorems* of $\mathcal{L}$.

2. An extension of DMP's algorithms for networks of non-strict inequalities to handle both the strict and the non-strict inequalities that appear in $\mathcal{L}_M$.

3. optimal translations between $\mathcal{L}_M$ and $\mathcal{L}_A$. As we noted, the two formalisms have orthogonal expressive power, so an exact translation is impossible; we say that a translation is *optimal* when it entails a minimal loss of information. Formally, $f : \mathcal{L}_1 \to \mathcal{L}_2$ is optimal iff for any $\alpha \in L_1$ and $\beta \in L_2$, then $\alpha \models \beta$ iff $f(\alpha) \models \beta$.

4. A constraint-propagation procedure for the combined constraint language $\mathcal{L}_M \cup \mathcal{L}_A$, which is based on the translation algorithms. The user of the system is able to enter information in terms of point difference inequalities or qualitative interval constraints, whichever is necessary or most convenient.

The system we describe in this paper is completely implemented in Common Lisp, and is available from the first author.

2 A Universal Temporal Language

Consider the following model of time: time is linear, and time points can be identified with the rationals under the usual ordering $<$. The difference of any two time points is likewise a rational number. An interval is a pair of points $\langle n, m \rangle$, where $n < m$. Two intervals stand in a particular qualitative relationship such as "overlaps" just when their endpoints stand in a particular configuration — in this case, when the starting point of the first falls before the starting point of the second, and the final point of the first falls between the two points of the second.

The following language $\mathcal{L}$ lets us say everything we'd like to about this model. It is typed predicate calculus with the following types and symbols:

types are Rational and Interval.
functions are

L, R : Interval $\Longrightarrow$ Rational

Intuitively, i_L is the starting (left) endpoint of i, and i_R is the final (right) endpoint.

$-$ (subtraction): Rational $\times$ Rational $\Longrightarrow$Rational

Rational numerals.

predicates are

$<, \leq, =$: Rational $\times$ Rational

Allen Predicates : Interval $\times$ Interval

P(recedes), M(eets), O(verlaps), S(tarts), D(uring), F(inishes), =, and the inverses P$^\smile$, M$^\smile$, O$^\smile$, S$^\smile$, D$^\smile$, F$^\smile$.

The language does not include constants to name specific intervals; instead, we use unbound variables to name intervals, with the understanding that any particular model provides an interpretation for free variables.

It is useful to distinguish two special syntactic forms. Formulas of the form

$$i(r_1)j \vee \cdots \vee i(r_n)j$$

where the r_i are Allen predicates are called *simple Allen constraints*, and are abbreviated as

$$i(r_1 + \cdots + r_n)j$$

The sublanguage of such formulas is called $\mathcal{L}_A$. Formulas of the following two forms

$$i_F - j_G < n \qquad i_F - j_G \leq n$$

where $F, G \in \{L, R\}$ and n is a numeral are called *simple metric constraints.* The sublanguage of such formulas is called $\mathcal{L}_M$. Note, however, that $\mathcal{L}$ is much richer than the union of $\mathcal{L}_M$ and $\mathcal{L}_A$. For example, the formulas in Table 1 are part of $\mathcal{L}$, but appear in neither $\mathcal{L}_A$ nor $\mathcal{L}_M$.

The following axioms capture the intended model of time.

- Arithmetic axioms for $-$ (subtraction), $<$, $\leq$, and numerals.
- $\forall i \,.\, i_L < i_R$
- Meaning postulates for each Allen predicate. The axioms for the non-inverted predicates appear in Table 1.

We write $C \models_{\mathcal{L}} D$ to mean that D holds in all of models of C that satisfy these axioms.

The original presentation of the Allen calculus described the predicates by a set of *transitivity axioms* such as

$$\forall i, j, k \,.\, i(M)j \wedge j(D)k \supset i(D + S + O)k$$

All of these formulas are *theorems* of $\mathcal{L}$, rather than axioms [Kautz and Ladkin, 1991].

Since $\mathcal{L}$ is just first-order logic, we could solve problems that involve both metric and Allen assertions by employing a complete and general inference method, such as resolution. This is almost certain to be impractically slow. On the other hand, it appears that we do not need the full power of $\mathcal{L}$ to

$\forall x, y\ .$	$i = j$	$\equiv$	$i_L - j_L \leq 0$	$\wedge$	$j_L - i_L \leq 0$	$\wedge$	$i_R - j_R \leq 0$
						$\wedge$	$j_R - i_R \leq 0$
$\forall x, y\ .$	$i(P)j$	$\equiv$	$i_R - j_L < 0$				
$\forall x, y\ .$	$i(M)j$	$\equiv$	$i_R - j_L \leq 0$	$\wedge$	$j_l - i_R \leq 0$		
$\forall x, y\ .$	$i(O)j$	$\equiv$	$i_L - j_L < 0$	$\wedge$	$j_L - i_R < 0$	$\wedge$	$i_R - j_R < 0$
$\forall x, y\ .$	$i(S)j$	$\equiv$	$i_L - j_L \leq 0$	$\wedge$	$j_L - i_L \leq 0$	$\wedge$	$i_R - j_R < 0$
$\forall x, y\ .$	$i(D)j$	$\equiv$	$j_L - i_L < 0$	$\wedge$	$i_R - j_R < 0$		
$\forall x, y\ .$	$i(F)j$	$\equiv$	$j_L - i_L < 0$	$\wedge$	$i_R - j_R \leq 0$	$\wedge$	$j_R - i_R \leq 0$

Table 1: Meaning postulates for Allen predicates.

express many interesting temporal reasoning problems. The sublanguage $\mathcal{L}_M$ can express constraints on the duration of an interval (e.g., $i_L - i_R < -3$); on the elapsed time between intervals (e.g., $i_R - j_L < 5$); and between an interval and an absolute date, which we handle by introducing a "dummy" interval which is taken to begin at time number 0 (e.g., $i_L - \mathit{day0}_L < -14$). But $\mathcal{L}_M$ by itself is not adequate for many problems. For example, in the sublanguage $\mathcal{L}_A$ one can assert that intervals i and j are disjoint by the formula $i(P + M + M^{\smile} + P^{\smile})j$, but there is no equivalent formula in $\mathcal{L}_M$. Such a disjointedness constraint is useful in planning; for example, if i is a time during which a robot holds a block, and j is a time during which the robot's hand is empty, a planning system might want to make the assertion that $i(P+M+M^{\smile}+P^{\smile})j$. Another useful expression in $\mathcal{L}_A$ is $i(S+F)j$, which means that interval i starts or finishes j; for example, in scheduling a conference, you might want to assert that a certain talk begins or ends the conference.

So $\mathcal{L}_M \cup \mathcal{L}_A$ appears to be a good candidate for a practical temporal language. In order to develop an inference procedure for this language, let us examine the constraint satisfaction procedures that are known for $\mathcal{L}_M$ and $\mathcal{L}_A$ individually.

3 Constraint Networks

$\mathcal{L}_M$ and $\mathcal{L}_A$ can each express certain *binary constraint satisfaction problems* (CSP) [Montanari, 74]. A binary CSP is simply a set (also called a *network*) of quantifier-free assertions in some language, each containing two variables. One possible task is to find a particular assignment of values to the variables that simultaneously satisfies all the constraints in the network: that is, to find a *model* of the network. (Henceforth in this paper we will always talk in terms of models rather than variable assignments.) Another important task is to compute the *minimal network representation* of the problem, which is defined as follows:

Definition: Minimal Network Representation
Suppose G is a consistent network of binary constraints in some language. Then the binary constraint network G' in that language is a minimal network representation of G iff the following all hold:

1. G' is logically equivalent to G.

2. For every pair of variables in G there is a constraint containing those variables in G'.

3. For any model of a single constraint in G', there is a model for all of G' which agrees with the former on the interpretation of the variables that appear in that constraint.

Hence from the minimal network representation one can "read off" the possible values that can be assigned to any variable.

$\mathcal{L}_M$ is very similar to what DMP called *simple temporal constraint satisfaction problems* (STCSP). They considered sets (or *networks*) of formulas of the form

$$n \leq x - y \leq m$$

where x and y are variables and n and m are numbers. Their representation differs from $\mathcal{L}_M$ in the following ways: (1) They abbreviated two inequalities in one formula, which is, of course, unimportant. (2) They use simple variables like x for time points, where $\mathcal{L}_M$ uses terms like i_L and i_R. Again this difference is not significant, because the interpretation of an interval i is simply the pair consisting of the interpretations of i_L and i_R. So we can treat i_L and i_R as "variables" in the CSP formulation. (3) Formulas in $\mathcal{L}_M$ include strict ($<$) as well as non-strict ($\leq$) inequalities.

DMP proved that an all-pairs shortest-path algorithm [Aho *et al.*, 1976, page 198] can compute the minimal network representation of a STCSP. One can modify the algorithm to handle the two kinds of inequalities as follows. We represent a formula M from $\mathcal{L}_M$ by a graph, where the nodes are the terms that appear in M (that is, i_L and i_R for each interval variable i), and the directed arc from i_F to j_G is labeled with the pair $\langle n, 1 \rangle$ if

$$i_F - j_G \leq n$$

appears in M, and labeled $\langle n, 0\rangle$ if

$$i_F - j_G < n$$

appears in M. Next we add the constraints from $\mathcal{L}$ that state that the left point of an interval is before its right point; that is, we add an arc $i_L\langle 0, 0\rangle i_R$ for each i. Finally we compute the shortest distance between all nodes in the graph using the following definitions for comparison and addition:

$$\langle m, x\rangle < \langle n, y\rangle \equiv m < n \vee (m = n \wedge x < y)$$

$$\langle m, x\rangle + \langle n, y\rangle = \langle m + n, \min(x, y)\rangle$$

The result is the minimal network representation of M. An arc appears between every pair of nodes in the graph, and the inequalities corresponding to the arcs are the strongest such inequalities implied by M. This procedure takes $O(n^3)$ time.

Binary CSP's based on the qualitative language $\mathcal{L}_A$ have been studied extensively [Allen, 1983, Ladkin and Madux, 1987, Vilain *et al.*, 1989, van Beek and Cohen, 1989]. Computing the minimal network representation of a set of such constraints is NP-Hard. In practice, however, one can approximate the minimal network by a weaker notion, called *n-consistency*. While we do not have space here to discuss the details of n-consistency, we note that the original presentation of $\mathcal{L}_A$ by Allen [83] included an algorithm that computes "3-consistency" in $O(n^3)$ time, and VanBeek [89] studied the improvements to the approximation likely to be found by computing higher degrees of consistency. For any *fixed* n, n-consistency can be computed in polynomial time.

Thus we have an efficient and complete algorithm for inference in $\mathcal{L}_M$, and a number of efficient approximation algorithms for $\mathcal{L}_A$. Figure 3 presents a constraint satisfaction algorithm for the union of the two languages. The method is to separately compute the minimal network representation of the metric and Allen constraints; derive new Allen constraints from the metric network and add these to the Allen network; derive new metric constraints from the Allen network and add these to the metric network; and repeat this process until no new statements can be derived. The system answers any query in $\mathcal{L}_M \cup \mathcal{L}_A$ by examining the appropriate network. The procedure is clearly correct; but now we must see how to translate $\mathcal{L}_M$ to $\mathcal{L}_A$ and vice-versa.

function combined-metric-Allen$(M, A) =$
input: simple metric network M and simple Allen network A
output: minimal networks M', A' implied by $M \cup A$

    repeat
        $A' :=$ **metric-to-Allen**$(M) \cup A$
        $M' :=$ **Allen-to-metric**$(A') \cup M$
        $M := M'; \quad A := A'$
    until $A = A'$ and $M = M'$
    return M', A'

end combined-metric-Allen

Figure 1: Inference procedure for $\mathcal{L}_M \cup \mathcal{L}_A$.

4 Translating and Combining Metric and Allen Constraints

This section presents the optimal translations between the metric and Allen constraint languages, and a complexity analysis of the combined inference algorithm. We begin with the translation from $\mathcal{L}_M$ to $\mathcal{L}_A$. At first impression, one might think that it is sufficient to convert each metric constraint to the Allen constraint it implies. For example, from the meaning postulates one can deduce that

$$i_L - j_L < 0 \supset i(P + M + O + F^{\smile} + D^{\smile})j$$

So, if the metric network M contains $i_L - j_L < -3$ (which implies the antecedent of the formula), the translation includes $i(P + M + O + F^{\smile} + D^{\smile})j$. This approach is correct, but fails to capture all implications in $\mathcal{L}_M$. For example, suppose M is the following network:

$$\begin{aligned} i_L - i_R &< -3 \\ j_R - j_L &< 2 \end{aligned}$$

The minimal network representation of M has only trivial constraints between i and j (such as $i_L - j_R < \infty$), so the approach just outlined fails to infer that i *cannot* be during j, because i has longer duration than j.

Therefore an optimal translation must consider several metric constraints at a time; but how many? One might imagine that the problem required an exponential procedure that checked consistency of every possible Allen constraint between two intervals with all of M. Fortunately, this is not necessary:

function metric-to-Allen(M) =
input: a simple metric constraint network M.
output: the strongest set of simple Allen constraints implied by M.

```
let M' be the minimal network representation of M
if M' is inconsistent then return any inconsistent Allen network
A_M := ∅
for each pair of intervals i, j do
    let S be the {i_L, i_R, j_L, j_R} subnet of M'
    R := ∅
    for each primitive Allen relation r do
        S' := S ∪ { m | m is a simple metric constraint
                        in the meaning postulate for i(r)j }
        if S' is consistent then R := R ∪ {r}
    end do
    A_M := A_M ∪ {i(R)j}
enddo
return A_M
end metric-to-Allen
```

Figure 2: Converting simple metric constraints to simple Allen constraints.

we can compute the strongest set of implied Allen constraints by considering constraints between just four points at a time. The algorithm **metric-to-Allen** appears in Figure 2, and the following theorem formally states that it is optimal.

Theorem 1 *The algorithm* **metric-to-Allen** *is correct and entails minimal loss of information: For any $M \in L_M$ and $A \in L_A$, it's the case that $M \models_{\mathcal{L}} A$ iff* **metric-to-Allen**$(M) \models_{\mathcal{L}} A$. *The algorithm runs in $O(n^2)$ time, where n is the number of intervals.*

Proof: By theorem 2 of [Dechter *et al.*, 1989], any consistent and minimal simple metric network is decomposable. This means that any assignment of values to a set of terms that satisfies the subnet containing those terms can be extended to a satisfying assignment for the entire net. Another way of saying this is that if such a subnet has a model, then the net has a model that agrees with the subnet's model on the interpretation of the terms in the subnet.

Note that if two models agree on the interpretations of the terms i_L, i_R, j_L, j_R then they assign the same truth value to the expression $i(r)j$ where r is any primitive Allen relation. From the construction of S' it is therefore the case that S' is consistent iff S' has a model iff S has a model in which $i(r)j$ holds iff M' has a model in which $i(r)j$ holds. Since M and M' are logically equivalent, we see that for any pair of intervals i and j, $i(R)j \in$ **metric-to-Allen**(M) iff for all $r \in R$ and no $r \notin R$, M has some model in which $i(r)j$ holds.

To show that the algorithm is correct, suppose that $i(R)j \in$ **metric-to-Allen**(M). If this clause were not implied by M, then there would be some model of M in which i and j stand in an Allen relation not in R. But that is impossible, as stated above. So $M \models_{\mathcal{L}}$ **metric-to-Allen**(M), and **metric-to-Allen**$(M) \models_{\mathcal{L}} A$ implies $M \models_{\mathcal{L}} A$.

To show that the algorithm entails minimal loss of information, suppose that $M \models_{\mathcal{L}} A$. Because A is a conjunction of statements of the form $i(R)j$, we can assume without loss of generality that it is a single such statement. From the operation of the algorithm we see that there is some R' such that $i(R')j \in$ **metric-to-Allen**(M). We claim that $R' \subset R$. Suppose not; then there would be an $r \in R'$ such that $r \notin R$. But the former means that there is a model of M in which $i(r)j$ holds, and the latter means that there is no such model, since in any particular model only a single Allen relation holds between a pair of intervals. So since $R' \subset R$ means that $i(R')j$ implies $i(R)j$, it follows that **metric-to-Allen**$(M) \models_{\mathcal{L}} i(R)j$.

The complexity $O(n^2)$ follows immediately from the iteration of the outer loop; everything inside takes constant time. ∎

Next we consider the translation from $\mathcal{L}_A$ to $\mathcal{L}_M$. It is not sufficient to simply replace each Allen predicate with its definition according to the meaning postulates, because the resulting formula is not necessarily in $\mathcal{L}_M$. Indeed, we can show that the problem is inherently intractable:

Theorem 2 *Computing the strongest set of simple metric constraints equivalent to a set of simple Allen constraints is NP-Hard.*

Proof: Checking the consistency of a set of formulas in $\mathcal{L}_A$ is NP-Complete, but checking consistency of formulas in $\mathcal{L}_M$ is polynomial. Since the best translation must preserve consistency, the translation itself must be NP-Hard. ∎

Suppose, however, we wish to compute the minimal network representation of a set of simple Allen constraints for other reasons. We can then quickly compute the strongest set of simple metric constraints implied by that network, by computing the metric constraints one Allen constraint at a time. Figure 4 presents the algorithm **Allen-to-metric** that performs this calculation; the following theorem states that this algorithm is optimal.

Theorem 3 *The algorithm* **Allen-to-metric** *is correct and entails minimal loss of information: For any $A \in L_A$, $M \in L_M$, it's the case that $A \models_{\mathcal{L}} M$ iff* **Allen-to-metric**$(A) \models_{\mathcal{L}} M$. *The algorithm runs in $O(e + n^2)$ time, where e is the time needed to compute the minimal network representation of the input, and n is the number of intervals.*

function Allen-to-metric(A) =
input: a simple Allen constraint network A
output: the strongest set of simple metric constraints
implied by A.

let A' be the minimal network representation of A
if A' is inconsistent **then return** any inconsistent metric network
$M_A := \emptyset$
for each pair of intervals i, j **do**
let R be the (complex) Allen relation such that $i(R)j$ appears in A'
$S := \{\ m \mid m$ is of the form $x - y < 0$ or $x - y \leq 0$
and $x, y \in \{i_L, i_R, j_L, j_R\}\ \}$
for each primitive Allen relation r in R **do**
$S := S \cap \{\ m \mid m$ is a simple metric constraint
implied by $i(r)j\ \}$
end do
$M_A := M_A \cup S$
end do
returnM_A
end Allen-to-metric

Figure 3: Converting simple Allen constraints to simple metric constraints.

Proof: At the end of the inner loop, it is clear that $m \in S$ iff m is a metric constraint implied by each $i(r)j$ for each $r \in R$; that is, $m \in S$ iff m is implied by $i(R)j$. Since $A \models_{\mathcal{L}} A' \models_{\mathcal{L}} i(R)j$ for each such $i(R)j$ that appears in M, it follows that $A \models_{\mathcal{L}}$ **Allen-to-metric**(A). Therefore the algorithm is correct: if **Allen-to-metric**$(A) \models_{\mathcal{L}} M$, then $A \models_{\mathcal{L}} M$.

To show that the algorithm entails minimal loss of information, suppose that $A \models_{\mathcal{L}} M$. Because M is conjunction of simple metric constraints, without loss of generality we can assume it is a single such constraint: $x - y < n$ or $x - y \leq n$, where $x, y \in \{i_L, i_R, j_L, j_R\}$. Furthermore, because A is equivalent (using the meaning postulates) to a boolean combination of difference inequalities containing only the number 0, it is plain that n cannot be negative; and furthermore, if n is positive, A must also imply the constraint $x - y \leq 0$. So without loss of generality we can also assume that M is of the form $x - y < 0$ or $x - y \leq 0$.

At the start of the loop in which the algorithm selects the pair of intervals $\langle i, j \rangle$ the variable S contains M, and we claim that S must still contain M at the conclusion of the inner loop. Suppose not; then there is some $r \in R$ such that $i(r)j$ has a model $\mathcal{M}$ in which M does not hold. But because $i(R)j \in A'$ and A' is the minimal network representation of A, it must be the case the A has a model that agrees with $\mathcal{M}$ on the interpretations of i and j. Therefore A has a model that falsifies M, so A cannot imply M after all.

The complexity $O(e + n^2)$ follows immediately from the iteration of the outer loop;

everything inside takes constant time. ■

Finally we turn to an analysis of the algorithm **combined-metric-Allen**. What is its computational complexity? The answer depends on how many times the algorithm iterates between the two networks. Because each iteration must strengthen at least one simple Allen constraint (which can only be done 13 times per constraint), in the worst case the number is linear in the maximum size of the Allen network (or $O(n^2)$ in the number of intervals). In fact, this is its lower bound, as well: we have discovered a class of temporal reasoning problems that shows that the maximum number of iterations does indeed grow with the size of the constraint set [Kautz and Ladkin, 1991].

Theorem 4 *The algorithm* **combined-metric-Allen** *is correct:* $M \cup A \models_{\mathcal{L}}$ **combined-metric-Allen**(M, A). *The algorithm terminates in* $O(n^2(e + n^3))$ *time, where* n *is the number of intervals that appear in* $M \cup A$*, and* e *is the time required to compute the minimum network representation of* A*.*

The question of whether **combined-metric-Allen** is a complete inference procedure for the language $\mathcal{L}_M \cup \mathcal{L}_A$ remains open. We are currently investigating whether the algorithm detects all inconsistent networks, and whether it always computes the minimal network representation in $\mathcal{L}_M \cup \mathcal{L}_A$.

5 Conclusions

The framework presented in this paper unifies the great body of research in AI on metric and qualitative temporal reasoning. We demonstrated that both STCSP's and Allen's temporal calculus can be viewed as sublanguages of a simple yet powerful temporal logic. We provided algorithms that translate between the languages with a minimal loss of information. Along the way we generalized known techniques for dealing with non-strict linear inequalities to handle strict inequalities as well. Finally, we showed how the translations can be used to combine two well-understood constraint-satisfaction procedures into one for the union of the two languages.

References

[Aho *et al.*, 1976] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley Publishing Co., Reading, MA, 1976.

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

[Allen, 1991] James Allen. Planning as temporal reasoning. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR-89)*, Cambridge, MA, 1991.

[Dean and McDermott, 87] T. L. Dean and D. V. McDermott. Temporal data base management. *Artificial Intelligence*, 32:1–55, 87.

[Dechter *et al.*, 1989] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. In Ronald J. Brachman, Hector J. Levesque, and Raymond Reiter, editors, *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR '89)*, page 83, San Mateo, CA, May 1989. Morgan Kaufmann Publishers, Inc.

[Kautz and Ladkin, 1991] Henry Kautz and Peter Ladkin. Temporal constraint propagation: an exercise in hybrid reasoning. In preparation, 1991.

[Ladkin and Madux, 1987] Peter Ladkin and Roger Madux. The algebra of convex time intervals, 1987.

[Malik and T.O., 1983] J. Malik and Binford T.O. Reasoning in time and space. In *Proceedings of 8th IJCAI 1983*, pages 343–345. IJCAI, 1983.

[Montanari, 74] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7:95–132, 74.

[Valdes-Perez, 1986] Raul E. Valdes-Perez. Spatio-temporal reasoning and linear inequalities. A.I. Memo No. 875, M.I.T. Artificial Intelligence Laboratory, Cambridge, MA, February 1986.

[van Beek and Cohen, 1989] Peter van Beek and Robin Cohen. Approximation algorithms for temporal reasoning. Research Report CS-89-12, University of Waterloo, Waterloo, Ontario, Canada, 1989.

[Vilain *et al.*, 1989] Marc Vilain, Henry Kautz, and Peter van Beek. Constraint propagation algorithms for temporal reasoning: a revised report. In Johan deKleer and Dan Weld, editors, *Readings in Qualitative Reasoning About Physical Systems*. Morgan Kaufmann, Los Altos, CA, 1989.