Approximation Algorithms for Temporal Reasoning

Peter van Beek
Department of Computer Science
University of Waterloo
Waterloo, Ontario
CANADA N2L 3G1

Abstract

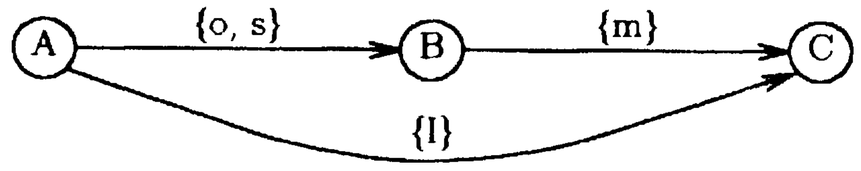
We consider a representation for temporal relations between intervals introduced by James Allen, and its associated computational or reasoning problem: given possibly indefinite knowledge of the relations between some intervals, how do we compute the strongest possible assertions about the relations between some or all intervals. Determining exact solutions to this problem has been shown to be (almost assuredly) intractable. Allen gives an approximation algorithm based on constraint propagation. We give new approximation algorithms, examine their effectiveness, and determine under what conditions the algorithms are exact.

1. Introduction

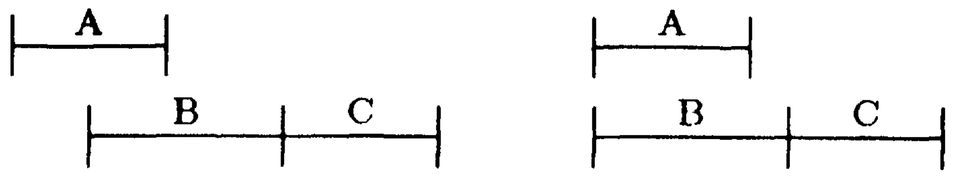
Allen (1983) gives an algebra for representing and reasoning about temporal relations between events represented as intervals. Possible application areas of the algebra include natural language processing (Allen, 1984; Song and Cohen, 1988), planning (Allen and Koomen, 1983), and a part of a knowledge representation language (Koubarakis et al., 1989). This algebra has been cited by others for its simplicity and ease of implementation with constraint propagation algorithms. The elements of the algebra are sets of the seven basic relations that can hold between two intervals, and their converses.

relation	symbol	converse	meaning
x before y	b	bi	xxx yyy
x meets y	m	mi	xxxyyy
x overlaps y	o	oi	xxx yyy
x during y	d	di	xxx yyyyy
x starts y	s	si	xxx yyyyy
x finishes y	f	fi	xxx yyyyy
x equal y	eq	eq	xxx yyy

There is a natural graphical notation where the vertices represent intervals and the directed edges are labeled with elements from the algebra representing the set of possible relations between the two intervals. Here is an example.



When the relationship between two intervals is ambiguous or indefinite we label the edge with the set of all the possible relations. So in our example, interval A either overlaps or starts interval B (but not both since the thirteen basic relations are mutually exclusive). Let {I} be the set of all basic relations, {b, bi, m, mi, o, oi, d, di, s. si, f, fi, eq}. The set of all possible labels on edges is $2^{\{I\}}$, the power set of {I}. Any edge for which we have no direct knowledge of the relationship is labeled with {I}; hence, the graphs are complete. Inference is done in this scheme through composition of relations: given a relation between A and B and between B and C we can compute a constraint on the relation between A and C. Doing this for our example we determine that our knowledge of the relationship between A and C can be strengthened to {b}. To see that this is true we show the two possible arrangements of the intervals along an imaginary time line.



A overlaps B in the diagram on the left, A starts B in the one on the right, and B meets C in both. We see that in both diagrams A is before C. Hence the result.

1.1. Statement of the Problem

Suppose we are given a set of events, represented as the intervals they occur over, and knowledge of the relationships between some of the intervals. The problem is to make explicit the strongest possible assertions about the relationships between intervals. We now make this somewhat more formal. Given is a directed graph with labels on the edges from the set of elements of the interval algebra. A *consistent singleton labeling* of the graph is a labeling where it is possible

to map the intervals to a time line and have the single relations between intervals hold (as in the example above). The *minimal label* corresponding to a label consists of only the elements of that label capable of being part of a consistent singleton labeling of the graph. The problem then is to determine the minimal labels, removing only those elements from the labels that could not be part of a consistent singleton labeling. Call this the *minimal labeling problem (MLP).* Vilain and Kautz (1986) show that determining an exact solution to the MLP is NP-hard. This strongly suggests that no polynomial time algorithm exists.

Supposing that we still wish to solve instances of the problem, several alternatives present themselves:

- Exponential algorithms: Solve the problem exactly but devise efficient exponential algorithms. These may still be practical even though their worst case is exponential. Valdes-Perez (1987) gives a dependency-directed backtrack algorithm, but it only finds one consistent singleton labeling of the graph or reports unsatisfiability.
- Easy special cases: Interesting special cases of an NP-Hard problem may be solvable in polynomial time. This alternative often takes the form of limiting the expressive power of the representation language.
- Approximation algorithms: Solve the problem approximately using an algorithm that is guaranteed polynomial. That is, design algorithms that do not behave badly—in terms of the quality of the produced solution—too often, assuming some probabilistic distribution of the instances of the problem. Allen's (1983) $0(n^3)$ algorithm is just such an approximation algorithm.

1.2. Overview

In this paper we explore the latter two alternatives: algorithms for computing approximations to the minimal labels between intervals and some special cases where approximation algorithms are exact. We consider two versions of the problem: an all-to-all version where we compute the minimal labels between every pair of intervals, and a one-to-all version where we determine the minimal labels between one interval and every other interval. Allen gives an approximation algorithm for the all-to-all problem based on constraint propagation. We present an algorithm for computing better approximations for the all-to-all version of the problem and an algorithm for the one-to-all version of the problem. For both versions of the problem we identify easy (polynomial time) special cases where the algorithms are exact, give the results of some computational experiments, and propose a test for predicting when the approximation algorithms are useful. For the all-to-all problem, identifying easy cases involves first giving a counter example to a result in the literature.

An extended version of this paper that includes proofs is available (van Beek, 1989).

2. The All-to-All Problem

The minimal labeling problem (MLP) is related to the constraint satisfaction problem (CSP) (Montanari, 1974, Mackworth, 1977). Tsang (1987) and Ladkin (1988) discuss how an MLP can be viewed as a CSP. The algorithms and results developed for the CSP can then also be applied to the MLP. Mackworth (1977) discusses approximation algorithms for the CSP, called consistency algorithms, that remove local inconsistencies that could never be part of a global solution. One, two, and three consistency are generally referred to as node, arc, and path consistency, respectively. Freuder (1978, 1982) generalizes this to k-consistency and defines strong k-consistency as j-consistent for all $j < k$. It can be shown that for the interval algebra, strong Ar-consistency is equivalent to ensuring that, for every choice of k of the n vertices, every element of the associated labels is capable of being part of a consistent singleton labeling of the subgraph of k vertices. Strong n-consistency then ensures the labeling is minimal.

2.1. The Path Consistency Algorithm

Allen's algorithm is a special case of the path consistency algorithm for constraint satisfaction. Descriptions of the path consistency algorithm can be found in (Allen, 1983 and Mackworth, 1977).

To use the path consistency algorithm we need to define the operations of intersection and composition of two labels. Intersection is just set intersection. Let C_{ij} be the label on the edge between interval i and interval j. Given that labels on edges can represent a disjunction of possible relations between two intervals, Allen defines the composition of two labels as the pair-wise multiplication of the elements,

$$C_{ik} \,.\, C_{kj} \;<=>\; \{\, a \;X\; b \mid a \in C_{ik},\; b \in C_{kj} \,\} \qquad (2.1)$$

where X is defined over the seven basic relations and their converses and is easily implemented as a table lookup (see Allen, 1983 for the complete table).

2.2. Improving the Approximation

In general, Allen's algorithm, being an approximation algorithm, will not always compute the minimal label between two intervals. In this section we explore better (and, unfortunately, more expensive) approximation algorithms. We develop an $0(n^4)$ consistency algorithm. The labels computed by the algorithm, as with Allen's algorithm, will always be a superset (not necessarily proper) of the minimal labels. But the algorithm computes a better approximation in that fewer disjuncts remain that could not be part of a consistent singleton labeling of the graph.

Recall the definition of a minimal label: every element of the label is capable of being part of a singleton labeling of the entire graph that can be consistently mapped to a time line. It can be shown that for the interval algebra the path consistency algorithm, as an approximation, ensures the labels are minimal with respect to all subgraphs of size three.

Inputs A matrix *C* where Cij is the label on edge (i, *j).*
Output: A path consistency approximation to the minimal labels for C_{ij} *i, j* = 1,...,n.

procedure ALL4
begin
  $Q \leftarrow \bigcup_{1 \le i < j \le n}$ RELATED PATHS (i, j)
  while Q is not empty **do begin**
    select and delete a 4-tuple (i, k, l, j) from Q
    $t \leftarrow C_{ij} \cap \Delta_{ikl} \cdot \Delta_{klj}$
    if $(t \neq C_{ij})$ **then begin**
      $C_{ij} \leftarrow t$
      $C_{ji} \leftarrow$ INVERSE (t)
      $Q \leftarrow Q \cup$ RELATED PATHS (i, j)
    end
  end
end

procedure RELATED PATHS (i, j)
return
$\{(k, i, j, l) \mid 1 \le k < l \le n,\ k, l, i, j$ distinct $\} \cup$
$\{(i, j, l, k),$
$(k, l, i, j) \mid 1 \le k, l \le n,\ k, l, i, j$ distinct $\}$

Figure 1 Three and Four Consistency Algorithm

Another way to view the action of the path consistency algorithm together with the definition of composition of labels (equation 2.1) is that entry C_{ij}, the label on the edge (t, *j),* gets updated to be the set of elements of the old label that can be part of a consistent singleton labeling of the triangle (i, *k, j).* That is, we ensure that the labels are minimal with respect to all triangles (or 3-cliques) and composition is defined over labels on edges that share a vertex. The simple idea for improving the approximation is then to ensure that the labels are minimal with respect to all subgraphs of four vertices (or 4-cliques) and define composition over labels of triangles that share an edge.

$$\Delta_{ikl} \cdot \Delta_{klj} \Leftrightarrow \{(a \times d) \cap (b \times e) \mid a \in C_{ik}, c \in C_{kl}, e \in C_{lj}, b \in (a \times c) \cap C_{il}, d \in (c \times e) \cap C_{kj}\} \quad (2.2)$$

The modified path consistency algorithm is given in Figure 1. The algorithm with the new definition of composition (equation 2.2) ensures that C_{ij} becomes the set of elements of the old label that can be part of a consistent singleton labeling of the subgraph of four vertices. If a label on an edge changes it may in turn constrain other labels so procedure RELATED PATHS returns all structures of four vertices in which the edge participates, taking into account symmetries to prevent redundant computation.

Theorem 1. *The algorithm of Figure 1, achieves three and four consistency and requires* $0(n^4)$ *time.*

The idea for developing the initial better approximation algorithm can be generalized to develop successively more expensive algorithms that compute progressively better approximations. But higher orders of consistency quickly become impractical for all but the smallest problems.

2.3. Easy (Polynomial Time) Special Cases

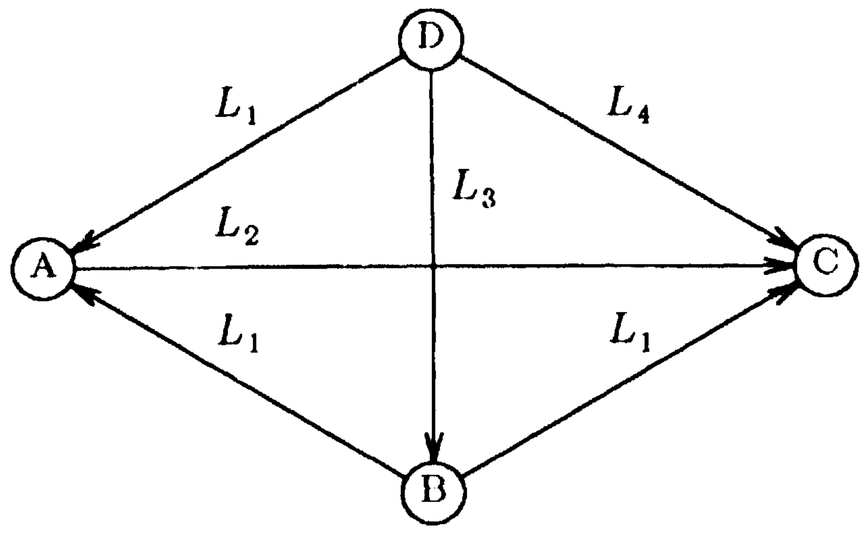
In this section we explore how far we must restrict the expressive power of the representation language to guarantee that we can compute exact solutions in polynomial time. ValdeVPerez (1987) shows that graphs that are not labeled with disjunctions can be solved exactly in $0(n^3)$ time using Allen's algorithm.

Vilain and Kautz (1986) claim something stronger. They define a time point algebra for representing and reasoning about the possible relations between points, as opposed to intervals. Let PA^* denote Vilain and Kautz's point algebra. $PA^\#$ is the algebraic structure with underlying set $\{<, \le, =, >, \ge, \neq, ?\}$ and binary operators intersection and composition. Note that $\le$, for example, is an abbreviation of $\{<, =\}$ and *?* means there is no constraint between two points, $\{<, =, >\}$ Intersection is then set intersection. Composition is defined as in the interval algebra (equation 2.1) except that multiplication is now given by,

×	<	=	>
<	<	<	?
=	<	=	>
>	?	>	>

Vilain and Kautz show that a subset of the interval algebra can be translated into this time point algebra. Let $SP^\#$ be the set of labels in the interval algebra that can be translated into relations between the endpoints of the intervals using the underlying set of *PA#.*

Vilain and Kautz assert (Theorem 4, p. 380) that the path consistency algorithm (Allen's algorithm) is exact for computing the minimal labels between points. The consequences for the interval algebra are the following. If their claim is true we can solve the subset *SP#* of the interval algebra exactly by first translating into the point algebra. However, their claim is false. Here we present a counter-example demonstrating that the path consistency algorithm is not exact for Vilain and Kautz's point algebra. The counter-example also shows that path consistency is not exact for *SP** if, instead of first translating into the point algebra, we use the interval algebra representation directly. Below is the interval algebra representation of the example with labels chosen from *SP*.*



where $L_1 = \{d, di, o, oi, m, f, fi\}$
$L_2 = \{eq, b, di, o, s, si, fi\}$
$L_3 = \{b, d, o, f, fi\}$
$L_4 = \{b, d, o, s\}$

The translation into the point representation is the following (where A- and A^+ represent the start and end points of interval A, respectively)

	A^-	A^+	B^-	B^+	C^-	C^+	D^-	D^+
A^-	=	<	≠	≤	≤	<	≠	≤
A^+	>	=	>	?	≠	?	>	?
B^-	≠	<	=	<	≠	<	≠	≠
B^+	≥	?	>	=	≥	?	>	≥
C^-	≥	≠	≠	≤	=	<	?	≠
C^+	>	?	>	?	>	=	>	>
D^-	≠	<	≠	<	?	<	=	<
D^+	≥	?	≠	≤	≠	<	>	=

Applying a path consistency algorithm results in no changes; the relations between points are all considered to be minimal. However, the relation A- < B is not minimal, thus demonstrating that the algorithm is not exact for the point algebra. The minimal relation is A- < B^+. This change is also reflected in the original interval algebra representation: the minimal label between vertex A and vertex B is {d, di, o, oi, f, fi}, with the meets relation having been dropped because it could not participate in any consistent singleton labeling of the graph. Interestingly, Allen's algorithm is also not exact when applied to the interval algebra representation of this example, whereas the algorithm we proposed in the previous section computes the minimal labeling. To reiterate, the counter-example shows that the path consistency algorithm is not exact for *SP** and *PA*.* We have been informed that Ladkin (1989) shows that for *PA** path consistency does guarantee that, if the minimal labels on edges is the empty set, this will be detected.

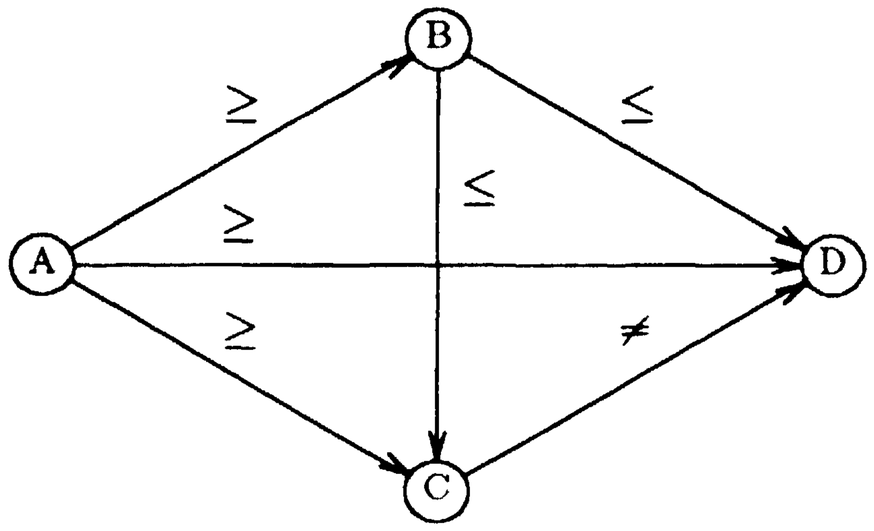
Define a new point algebra, *PA,* with the same binary operators and underlying set as *PA** with the exception that ^ is excluded from the underlying set. Let *SP* be the set of labels in the interval algebra that can be translated into relations between the endpoints of the intervals using the underlying set of *PA.* Path consistency is exact for *SP* and for *PA.* The following lemma on the intersection of convex sets is useful in the proof of exactness.

Lemma 1 (Kelly's theorem). *Let F be a finite family of at least n-hl convex sets in R^n such that every n+1 sets in F have a point in common. Then all the sets in F have a point in common.*

Theorem 2. *The path consistency algorithm is exact if all labels are chosen from SP. It is also exact for PA, the point algebra that excludes the ^ relation.*

Theorem 2 is proved by showing by induction that if all labels are from *SP* or *PA* and there is path consistency then the graph is strongly Ar-consistent for all *k* < n. Hence the labels are exact. The inductive step relies on the fact that the relations between vertices (the labels) form convex sets allowing the application of Lemma 1. The proof is constructive and gives an algorithm for finding consistent singleton labelings.

In the remainder of this section we show that the four consistency algorithm developed in section 2.2 is exact for *SP** and *PA** (recall *PA** includes #, *PA* doesn't). If 5* is permitted in the language of the point algebra, the relations no longer form convex sets and path consistency is no longer sufficient. Here is the smallest counter-example to the exactness of path consistency for *PA** and, up to isomorphism, is the only counter-example of four vertices.



The graph is path consistent. But this is not the minimal labeling since not every element is capable of being part of a consistent singleton labeling. To see this, choose the singleton labeling for the triangle (A, B, C) such that A = B = C. We now have D < A, B < D, and D ^ C. Using standard interval notation and substitution of equals, D must be in the three intervals (—00, a], [a, +00), and (—00, *a)* U (a, +00). It is easily seen that these intervals are pair-wise consistent but together are inconsistent. If the graph was also made four consistent, say by applying algorithm A114 (Figure 1), the label between A and B would be > and this counter-example could not occur.

The counter-example then is unique for n = 4 and cannot occur if the graph is three and four consistent. But can we find a counter-example for n > 4? The answer is, no. It can be shown (see the longer version of the paper) that any larger counter-example must have a subgraph of 4 vertices isomorphic to the example above. But this is ruled out by four consistency.

Theorem 3. *Hie three and four consistency algorithm of Figure 1 is exact if all labels are chosen from SP*. It is also exact for PA*, the time point algebra that includes the* 5* *relation.*

We characterized the subsets of the interval algebra for which the path consistency algorithm and the four-consistency algorithm are exact. Unfortunately these subsets are small. We must quite severely restrict our representation language to guarantee efficient and exact solutions.

3. The One-to-All Problem

The algorithms given in the previous section compute approximations to the minimal labels between every interval and every other interval (the all-to-all version of the problem). If we are only interested in the relationships between one interval and every other interval or between two particular intervals then, in computing the relationships between all intervals, we may be doing too much work. In this section we present an efficient algorithm for the one-to-all version of the problem and show that the algorithm is exact for a useful subset of the interval and point algebras.

3.1. A One-to-All Approximation Algorithm

Our solution to the one-to-all version of the problem is an adaptation of Dijkstra's (1959) algorithm for computing the shortest path from a single source vertex to every other vertex. His algorithm, which can be categorized as a label-setting algorithm, produces poor quality solutions when applied to the interval algebra. In the algorithm of Figure 2, we allow a label to change after it has been tentatively fixed and perhaps further constrain other labels. This change turns the algorithm into a label-correcting algorithm where no labels are considered final until the procedure halts. This change to Dijkstra's algorithm also appears in Edmonds and Karp (1972) in the context of finding shortest paths where negative arc lengths are allowed. Johnson (1973) showed that, if the labels are integers, this change makes the algorithm exponential in the worst case. In this context, though, the algorithm is $0(n^2)$.

Theorem 4. *The one-to-all algorithm of Figure 2 requires $0(n^2)$ time.*

3.2. Easy (Polynomial Time) Special Cases

Here we explore how far we must restrict the expressive power of the representation language to guarantee that our one-to-all approximation algorithm of Figure 2 (One3) is exact. Note that the all-to-all algorithms compute approximations to all the minimal labels, but even the labels we are not interested in help us by further constraining the labels we are interested in. One3 does not do this; it uses less information to compute its approximations. Hence, in general its approximations are poorer than those of the all-to-all algorithms. Surprisingly though, One3 is exact for the same subset of the interval algebra for which the path consistency (Allen's algorithm) is exact.

Theorem 5. *The label-correcting algorithm of Figure 2 is exact for SP and PA, provided the minimal label on an edge is not the empty set or null relation.*

The proof of the theorem uses the property that composition distributes over intersection. This property is true for *SP* and *PA* only if we can guarantee that the intersection of two labels will never result in the empty set. It is easy to show that the theorem is false if the empty set is the minimal label on an edge.

Input: A source vertex *s* and a matrix *C* where element C_{ij}, is the label on edge (i, *j)*
Output: An approximation to the minimal labels for C_{sj}, $j = 1, . ,n$

```
procedure ONE3
begin
    L ← V − { s }
    while L is not empty do begin
        select a vertex v from L
        L ← L − { v }
        for each t in V do begin
            l ← C_st ∩ C_sv · C_vt
            if l ≠ C_st then begin
                C_st ← l
                L ← L ∪ { t }
            end
        end
    end
end
```

Figure 2 A One-to-All Algorithm

4. Experimental Results

In this section we present the results of some computational experiments comparing the quality of the solutions produced by Allen's and our approximation algorithms. The experiments give a partial answer to the question: with what degree of confidence can we rely on the less expensive approximate solutions? We also present a simple test for predicting when the approximation algorithms will and will not produce good quality approximations.

For each problem of size *n* we randomly generated a consistent singleton labeling and then added uncertainty in the form of additional disjuncts on the possible relationships between two intervals. We then applied the three approximation algorithms, chose a particular edge, determined the minimal or exact label on that edge using an exact backtracking algorithm, and recorded whether the less expensive approximate solutions differed from the exact solution. Table 1 summarizes the results for two distributions and three algorithms: One3 (Figure 2), A113 (the path consistency algorithm), and A114 (Figure 1). Distribution one was chosen to roughly approximate instances that may arise in a planning application (as estimated from a block-stacking example in Allen and Koomen, 1983). Fortunately, for this class of problems the results suggest that for a reassuringly large percentage of the time we can use the path consistency algorithm with near impunity: the outcome is the same as that of using an exact algorithm. With a different distribution, however, up to two-thirds of the labels on average were not minimal.

Let *SP* be the subset of the interval algebra discussed earlier that can be solved exactly using the path consistency algorithm. Computational evidence shows a strong correlation between the percentage of the total labels that are from *SP* and how well the One3, A113, and A114 algorithms approximate the exact solution. Recall that Theorem 2 (Theorem 5) states

Table 1

Average percentage differences between the approximation algorithms and an exact algorithm for various problem sizes. *Distribution 1:* About 75% of the time the uncertainty added is {1} and the remaining time consists of from 0 to 3 of the basic relations. *Distribution 2:* All labels are equally likely to be added as uncertainty. 150 tests were performed for each problem size, n

	Distribution 1			Distribution 2		
n	One3	All3	All4	One3	All3	All4
20	6.0	0.0	0.0	72.7	66.0	36.7
30	10.7	0.0	0.0	88.7	41.3	9.3
40	18.0	1.3	0.7	95.3	12.0	3.3
50	12.7	0.0	0.0	90.7	4.0	2.0
60	18.0	0.7	0.0	84.0	0.0	0.0

that A113 (One3) is exact when all the labels are from *SP* so we cannot improve on that. But, as the percentage of the total labels that are from *SP* nears zero, an increasing number of the labels (on average) assigned by the approximation algorithms are not minimal. Thus we have an effective test for predicting whether it would be useful to apply a more expensive algorithm.

5. Conclusions

We considered a popular representation for temporal relationships between intervals introduced by James Allen and its associated computational or reasoning problem of, given possibly indefinite knowledge of the relations between some intervals, computing the strongest possible assertion about the relations between some or all intervals. Allen gives an approximation algorithm based on constraint propagation. We presented an algorithm for computing better approximations for the all-to-all version of the problem and a test for predicting when this more expensive algorithm is useful. We presented an algorithm for the one-to-all version of the problem and a test for predicting when this less expensive algorithm is useful. We gave a counter example to a result in the literature and identified easy (polynomial time) special cases of both versions of the problem.

Acknowledgements

I thank Robin Cohen, Bruce Spencer, Fei Song, Fahiem Bacchus, Andre Trudel, and Paul van Arragon for help, advice, and encouragement. Financial support was gratefully received from the Natural Sciences and Engineering Research Council of Canada, ITRC, and the University of Waterloo.

References

Allen, J. F. 1983. Maintaining Knowledge about Temporal Intervals. *Comm. ACM* 26, 832-843.

Allen, J. F. 1984. Towards a General Theory of Action and Time. *Artificial Intelligence* 23, 123-154.

Allen, J. F., and J. A. Koomen. 1983. Planning Using a Temporal World Model. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI),* Karlsruhe, W. Germany, 741-747.

Dijkstra, E. W. 1959. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik* 1, 269-271.

Edmonds, J., and R. M. Karp. 1972. Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. *J. ACM* 19, 248-264.

Freuder, E. C. 1978. Synthesizing Constraint Expressions. *Comm. ACM* 21, 958-966.

Freuder, E. C. 1982. A Sufficient Condition for Backtrack-Free Search. *J. ACM* 29, 24-32.

Johnson, D. B. 1973. A Note on Dijkstra's Shortest Path Algorithm. *J. ACM* 20, 385-388.

Koubarakis, M., J. Mylopoulos, M. Stanley, and A. Borgida. 1989. Telos: Features and Formalization. Forthcoming Technical Report, Computer Systems Research Institute, University of Toronto, Feb.

Ladkin, P. B. 1988. Satisfying First-Order Constraints About Time Intervals. *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI),* Saint Paul, Minn., 512-517.

Ladkin, P. B. 1989. Unpublished manuscript. Kestrel Institute, Palo Alto, Calif.

Mackworth, A. K. 1977. Consistency in Networks of Relations. *Artificial Intelligence* 8, 99-118.

Montanari, U. 1974. Networks of Constraints: Fundamental Properties and Applications to Picture Processing. *Inform. Sci. 7,* 95-132.

Song, F., and R. Cohen. 1988. The Interpretation of Temporal Relations in Narrative. *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI),* Saint Paul, Minn., 745-750.

Tsang, E. P. K. 1987. The Consistent Labeling Problem in Temporal Reasoning. *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI),* Seattle, Wash., 251-255.

Valdes-Perez, R. E. 1987. The Satisfiability of Temporal Constraint Networks. *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI),* Seattle, Wash., 256-260.

van Beek, P. 1989. Approximation Algorithms for Temporal Reasoning. Department of Computer Science Technical Report CS-89-12, University of Waterloo.

Vilain, M., and H. Kautz. 1986. Constraint Propagation Algorithms for Temporal Reasoning. *Proceedings of the Fifth National Conference on Artificial Intelligence (AAAI),* Philadelphia, Pa., 377-382.