A Polynomial-time Algorithm for Simple Temporal Problems with Piecewise Constant Domain Preference Functions

T. K. Satish Kumar
Knowledge Systems Laboratory
Stanford University
tksk@ksl.stanford.edu

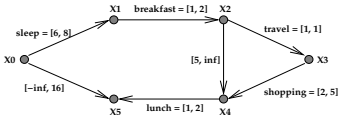

Figure 1: Shows an example of the kind of reasoning possible in STPs. The figure is about an agent who should find a schedule for her day's events that satisfies various constraints (indicated by temporal bounds on the edges). Assuming that X_0 corresponds to 12:00 a.m., the agent is allowed to sleep between 6 and 8 hours, have breakfast between 1 and 2 hours, travel to the market for an hour, shop there between 2 and 5 hours, and have lunch between 1 and 2 hours. Further, she should finish having lunch before 4:00 p.m. and should wait for at least 5 hours between having breakfast and having lunch.

Abstract

In this paper, we provide a polynomial-time algorithm for solving an important class of metric temporal problems that involve simple temporal constraints between various events (variables) and piecewise constant preference functions over variable domains. We are given a graph $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$ where $\mathcal{X} = \{X_0, X_1 \ldots X_n\}$ is a set of events (X_0 is the "beginning of the world" node and is set to 0 by convention) and $e = \langle X_i, X_j \rangle \in \mathcal{E}$, annotated with the bounds $[LB(e), UB(e)]$, is a simple temporal constraint between X_i and X_j indicating that X_j must be scheduled between $LB(e)$ and $UB(e)$ seconds after X_i is scheduled ($LB(e) \leq UB(e)$). A family of stepwise constant preference functions $\mathcal{F} = \{f_{X_i}(t) : \mathcal{R} \rightarrow \mathcal{R}\}$ specifies the preference attached with scheduling X_i at time t. The goal is to find a schedule for all the events such that all the temporal constraints are satisfied and the sum of the preferences is maximized. Our polynomial-time algorithm for solving such problems (which we refer to as extended simple temporal problems (ESTPs)) has important consequences in dealing with limited forms of disjunctions and preferences in metric temporal reasoning that would otherwise require an exponential search space.

Introduction

Expressive and efficient temporal reasoning is central to many areas of Artificial Intelligence (AI). Several tasks in planning and scheduling for example, involve reasoning about temporal constraints between actions and propositions in partial plans (see (Nguyen and Kambhampati 2001) and (Smith *et al.* 2000)). These tasks may include threat resolution between actions in partial order planning, analyzing resource consumption envelopes to guide the search for a good plan (see (Kumar 2003)) etc. Among the important formalisms used for reasoning with metric time are simple temporal problems (STPs) and disjunctive temporal problems (DTPs) (see (Stergiou and Koubarakis 1998) and (Oddi and Cesta 2000)). Unlike DTPs, STPs can be solved in polynomial time, but are not as expressive as DTPs.

An STP is characterized by a graph $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$ where $\mathcal{X} = \{X_0, X_1 \ldots X_n\}$ is a set of events (X_0 is the "beginning of the world" node and is set to 0 by convention) and $e = \langle X_i, X_j \rangle \in \mathcal{E}$, annotated with the bounds $[LB(e), UB(e)]$, is a simple temporal constraint between X_i and X_j indicating that X_j must be scheduled between $LB(e)$ and $UB(e)$ seconds after X_i is scheduled ($LB(e) \leq UB(e)$). Figure 1 shows an example of an STP which (like all other instances of the class) can be solved in polynomial time using shortest paths (see (Dechter *et al.* 1991)).

DTPs are significantly more expressive than STPs and allow for disjunctive constraints. The general form of a DTP is as follows. We are given a set of events $\mathcal{X} = \{X_0, X_1 \ldots X_n\}$ (X_0 is the "beginning of the world" node and is set to 0 by convention) and a set of constraints $\mathcal{C}$. A constraint $c \in \mathcal{C}$ is a disjunction of simple temporal constraints of the form $s_1 \vee s_2 \ldots s_k$. Here, s_i ($1 \leq i \leq k$) is a simple temporal constraint of the form $l \leq X_b - X_a \leq u$ for $0 \leq a, b \leq n$. Figure 2 shows an example of a DTP which expresses disjunctive constraints.

Although DTPs are expressive enough to capture many tasks in planning and scheduling (like threat resolution and plan merging), they require an exponential search space. The principal approach taken to solve DTPs has been to convert the original problem to one of selecting one disjunct from each constraint and then checking that the set of selected disjuncts forms a consistent STP. Checking the

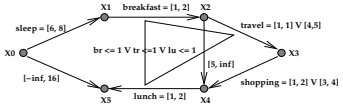

Figure 2: Shows an example of the kind of reasoning possible in DTPs. The scenario is very similar to the that in Figure 1 except that it includes disjunctive constraints like having a travel time between 4 and 5 hours (by walk) or exactly for one hour (by bus), shopping between 1 and 2 hours or 3 and 4 hours, and restricting time spent on at least one of breakfast, lunch or travel time to within an hour.

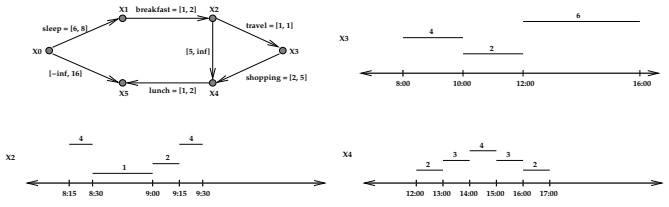

Figure 3: Shows an example of the kind of reasoning allowed by ESTPs. The situation is very similar to that in Figure 1, except that there are preferences attached with various events. If the agent reaches the bus stop between 8:15 a.m. and 8:30 a.m. or between 9:15 a.m. and 9:30 a.m., the buses are readily available and the preference for these intervals is high. Reaching the bus station between 8:30 a.m. and 9:00 a.m. means that she has to wait for at least 15 minutes and therefore has a low preference. Reaching between 9:00 a.m. and 9:15 a.m. means a lower waiting time, and has a slightly higher preference. Similarly if the agent starts shopping early in the morning, the preference is high, it is low during the peak period between 10:00 a.m. and 12:00 a.m. and high again during the non-peak period 12:00 p.m. to 4:00 p.m. Again, for health reasons, starting to have lunch between 2:00 p.m. and 3:00 p.m. is the optimal for the agent and the later the worse beyond 3:00 p.m., and the earlier the worse before 2:00 p.m.

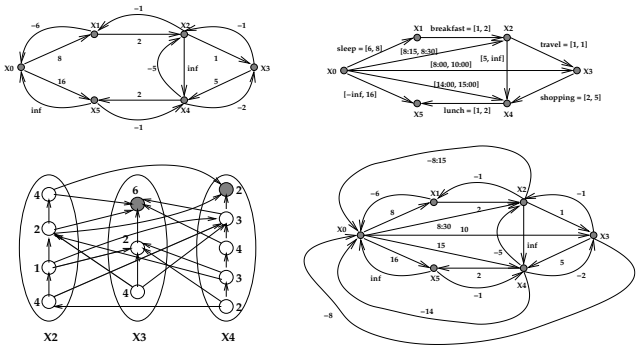

Figure 4: Illustrates how an ESTP (as shown in Figure 3) can be converted to the computation of the largest weighted anti-chain in a POSET. The figure shows the distance graph associated with the temporal constraints to the top-left, and the POSET associated with the size-2 conflicts between various intervals to the bottom-left. The weights on the nodes of the POSET correspond to the preference attached with the respective intervals, the size-2 conflicts are represented by arrows and the size-1 conflicts are represented by dark circles. The top-right figure shows the STP resulting out of replacing the domain preferences with the simple temporal constraints corresponding to the largest weighted anti-chain, and the bottom-right figure shows the corresponding distance graph which is assured of not containing any negative cycle.

consistency of and finding a solution to an STP can be performed in polynomial time using shortest path algorithms (see (Dechter *et al.* 1991)). The computational complexity of solving a DTP comes from the fact that there are an exponentially large number of disjunct combinations possible. The disjunct selection problem can also be cast as a constraint satisfaction problem (CSP) (see (Stergiou and Koubarakis 1998) and (Oddi and Cesta 2000)) or a satisfiability problem (SAT) (see (Armando *et al.* 2000)) and solved using standard search techniques applicable for them. Epilitis is a systems that efficiently solves DTPs using CSP search techniques like conflict-directed backjumping and nogood recording (see (Tsamardinos and Pollack 2003)).

In this paper, we describe a middle ground between STPs and DTPs (which we refer to as extended STPs (ESTPs)) that can deal with limited forms of disjunctions and preferences, and can also be solved in polynomial time. The expressive power of ESTPs along with their tractability makes them a suitable model for many real-life applications. An ESTP is characterized by a graph $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$ where $\mathcal{X} = \{X_0, X_1 \ldots X_n\}$ is a set of events (X_0 is the "beginning of the world" node and is set to 0 by convention) and $e = \langle X_i, X_j \rangle \in \mathcal{E}$, annotated with the bounds $[LB(e), UB(e)]$, is a simple temporal constraint between X_i and X_j indicating that X_j must be scheduled between $LB(e)$ and $UB(e)$ seconds after X_i is scheduled ($LB(e) \leq UB(e)$). A family of stepwise constant preference functions $\mathcal{F} = \{f_{X_i}(t) : \mathcal{R} \rightarrow \mathcal{R}\}$ specifies the preference attached with scheduling X_i at time t. The goal is to find a schedule for all the events such that all the temporal constraints are satisfied and the sum of the preferences is maximized. Figure 3 shows an example of an ESTP.

We present a polynomial-time algorithm for solving ESTPs by reducing a given ESTP to the problem of computing the largest weighted anti-chain in a POSET (partially ordered set) which in turn can be solved using *maxflow* techniques. Our algorithm has important consequences in the context of metric temporal reasoning and dealing with limited forms of disjunctions and preferences that would otherwise require an exponential search space. We note again that the disjunctions and preferences are associated only with scheduling individual events at time t, and not with the relative time difference between two events. The latter problem has been shown to be tractable when one wants to maximize the minimum preference value, and all the preference functions are known to be concave (see (Khatib *et al.* 2001)).[1] In general, the fact that the disjunctions and/or preference functions allow only one argument does not allow encoding of a general DTP as an ESTP.

Solving ESTPs: Reduction to Largest Weighted Anti-Chain Computation

We will present a polynomial-time algorithm for solving ESTPs by reducing a given ESTP to the problem of computing the largest weighted anti-chain in a POSET. We will use the following notation.

Notation. Let the landmarks in the domain of X_i, for which the piecewise constant function $f_{X_i}(t)$ is defined, be $d_{i,1}, d_{i,2} \ldots d_{i,T(i)}$. Let $I_{i,1}, I_{i,2} \ldots I_{i,T(i)+1}$ be intervals defined as follows: $I_{i,1} = (-\infty, d_{i,1}]$, $I_{i,k} = (d_{i,k-1}, d_{i,k}]$ for $2 \leq k \leq T(i)$ and $I_{i,T(i)+1} = (d_{i,T(i)}, +\infty]$. Let $L((a,b]) = a$ and $R((a,b]) = b$. Let the preference associated with scheduling variable X_i at any time $t \in I_{i,k}$ be denoted by $f_{X_i}(I_{i,k})$.

Solving Zero-One ESTPs

We will first deal with the case where $f_{X_i}(I_{i,k})$ is restricted to be either 0 or 1 (referred to as zero-one ESTPs). Figure 5 shows the procedure for solving zero-one ESTPs. We note that disjunctions limited only to the domains of variables can be encoded using zero-one ESTPs, and can therefore be solved in polynomial time.

Lemma 1: A consistent schedule exists for $X_0, X_1 \ldots X_n$ (disregarding the preference functions) in $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$ if and only if the distance graph $D(\mathcal{G})$ does not contain any negative cycles (see Figure 5).

Proof: see (Dechter *et al.* 1991).

[1] Optimizing the sum of preference values (with the preference functions still remaining concave) is an open problem.

ALGORITHM: SOLVE-ZERO-ONE-ESTP
INPUT: An instance $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$ of the ESTP with $f_{X_i}(I_{i,k}) = 0$ or 1.
OUTPUT: A flexible consistent schedule s with only simple temporal constraints that solves $\mathcal{G}$.
(1) Construct the distance graph $D(\mathcal{G})$ on $X_0, X_1 \ldots X_n$ with every edge $e = \langle X_i, X_j \rangle$ in $\mathcal{E}$ compiled to two edges: $\langle X_i, X_j \rangle$ annotated with $UB(e)$, and $\langle X_j, X_i \rangle$ annotated with $-LB(e)$.
(2) Let $dist(X_a, X_b)$ denote the shortest distance from X_a to X_b in $D(\mathcal{G})$.
(3) For every pair of intervals I_{i,k_1} and I_{j,k_2}:
(a) Construct a (directed) size-2 conflict from I_{i,k_1} to I_{j,k_2} (denoted $I_{i,k_1} \rightarrow I_{j,k_2}$) if and only if $R(I_{i,k_1}) + dist(X_i, X_j) - L(I_{j,k_2}) < 0$.
(4) Construct a node-weighted directed graph $P(\mathcal{G})$ as follows:
(a) The nodes of $P(\mathcal{G})$ correspond to the intervals.
(b) Remove all nodes that correspond to size-1 conflicts i.e. remove $I_{i,k}$ from $P(\mathcal{G})$ if $dist(X_i, X_0) + R(I_{i,k}) < 0$ or $dist(X_0, X_i) - L(I_{i,k}) < 0$.
(c) The weight on a node corresponding to interval $I_{i,k}$ is equal to $f_{X_i}(I_{i,k})$.
(d) A directed edge $\langle I_{i,k_1}, I_{j,k_2} \rangle$ in $P(\mathcal{G})$ encodes a size-2 conflict $I_{i,k_1} \rightarrow I_{j,k_2}$.
(5) Compute $Q = \{I_{q_1}, I_{q_2} \ldots I_{q_k}\}$ as the largest weighted anti-chain in $P(\mathcal{G})$.
(6) RETURN:
(a) 'inconsistency' if $|Q| < n$.
(b) $s = D(\mathcal{G}) \cup \{\langle X_0, X_i \rangle$ annotated with $R(I_{i,k_1})$ and $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k_1}) | I_{i,k_1} \in Q\}$.
END ALGORITHM

Figure 5: A polynomial-time algorithm for solving zero-one ESTPs.

Lemma 2: Ensuring that variable X_i takes a value in the interval $I_{i,k}$ requires the addition of the edges $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k})$ and $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k})$ to the distance graph without creating a negative cycle.
Proof: If we have to ensure that the variable X_i is in the interval $I_{i,k}$—i.e. between $L(I_{i,k})$ and $R(I_{i,k})$, we have to make sure that $X_i - X_0 \leq R(I_{i,k})$ and $X_i - X_0 \geq L(I_{i,k})$. Retaining the semantics of the distance graph (where the constraint $X_j - X_i \leq w$ is specified by the edge $\langle X_i, X_j \rangle$ annotated with w), this corresponds to the addition of the edges $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k})$ and $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k})$ to the distance graph without creating an inconsistency (which by the previous Lemma is characterized by the presence of a negative cycle).
Definition 1: We say that an interval $I_{i,k}$ is active if we ensure that the variable X_i takes a value in the interval $I_{i,k}$ by successfully adding the edges $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k})$ and $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k})$ to the distance graph.
Lemma 3: At most one interval can be activated for every variable.
Proof: Consider any variable X_i and two of its intervals I_{i,k_1} and I_{i,k_2}, assuming without loss of generality that $R(I_{i,k_1}) \leq L(I_{i,k_2})$. Activating both these intervals would require the addition of the following edges to the distance graph: $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k_1})$, $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k_1})$, $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k_2})$ and $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k_2})$. This creates the negative cycle $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k_1})$ and $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k_2})$, hence establishing the truth of the Lemma.
Lemma 4: An ESTP $\mathcal{G}$ is consistent if and only if exactly one interval for all variables can be made active simultaneously.
Proof: Suppose exactly one interval for each variable can be made active simultaneously. This means that there is no negative cycle in the resulting distance graph and by Lemma 1, a consistent schedule for $\mathcal{G}$ can be found. Conversely, let $s = \{X_1 = x_1, X_2 = x_2 \ldots X_n = x_n\}$ be a consistent schedule for $\mathcal{G}$. By definition, this means that the distance graph resulting out of adding the edges $\langle X_0, X_i \rangle$ annotated with x_i and $\langle X_i, X_0 \rangle$ annotated with $-x_i$ (for all $1 \leq i \leq n$) does not contain any negative cycles. Let I_{i,k_i} be the interval containing x_i for variable X_i. Clearly, $L(I_{i,k_i}) \leq x_i$ and $x_i \leq R(I_{i,k_i})$. This means that the addition of the edges $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k_i})$ and $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k_i})$ (for all $1 \leq i \leq n$) instead of x_i and $-x_i$ respectively also does not contain any negative cost cycles (since the weights of edges are only increased). The truth of the Lemma then follows from the fact that the addition of these edges corresponds to the activation of these intervals (Lemma 2) and that at most one interval can be activated for any variable (Lemma 3).
Definition 2: A conflict is a set of intervals all of which cannot be made simultaneously active. A minimal conflict is a conflict no proper subset of which is also a conflict.
Lemma 5: A set of intervals can be made simultaneously active if and only if there is no subset of them that constitutes a minimal conflict.
Proof: By the definition of a conflict, a set of intervals can be made simultaneously active if and only if there is no subset of them that constitutes a conflict. Further, the truth of the Lemma follows from the fact that there exists a subset of actions that constitutes a conflict if and only if there exists a subset of intervals that constitutes a minimal conflict.
Lemma 6: The size of a minimal conflict is ≤ 2.
Proof: Suppose we try to activate the set of intervals $I_{i_1,k_1}, I_{i_2,k_2} \ldots I_{i_h,k_h}$. This would involve the addition of the following edges to the distance graph without creating a negative cycle: $\langle X_0, X_{i_p} \rangle$ annotated with $R(I_{i_p,k_p})$ and $\langle X_{i_p}, X_0 \rangle$ annotated with $-L(I_{i_p,k_p})$ for all $1 \leq p \leq h$. We refer to these edges as "special" edges. If a negative cycle is created in our attempt to activate these intervals, it must involve one of the "special" edges. Further, since all "special" edges have X_0 as an end point, the negative cycle must contain X_0. This also means that the cycle can involve at most 2 "special" edges. Since these "special" edges correspond to the activation of intervals, the size of a minimal conflict is ≤ 2.
Lemma 7: An interval $I_{i,k}$ constitutes a size-1 conflict if $-L(I_{i,k}) + dist(X_0, X_i) < 0$ or $R(I_{i,k}) + dist(X_i, X_0) < 0$.
Proof: Continuing the arguments of the previous Lemma, a size-1 conflict contains exactly one "special" edge. In the case that this edge is "incoming" to X_0 (say $\langle X_i, X_0 \rangle$ annotated with $-L(I_{i,k})$ for the interval $I_{i,k}$), the negative cycle must be $-L(I_{i,k}) + dist(X_0, X_i)$. In the case that

this edge is "outgoing" from X_0 (say $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k})$ for the interval $I_{i,k}$), the negative cycle must be $R(I_{i,k}) + dist(X_i, X_0)$.

Lemma 8: The two intervals I_{i,k_1} and I_{j,k_2} constitute a directed size-2 conflict (denoted $I_{i,k_1} \rightarrow I_{j,k_2}$) when $R(I_{i,k_1}) + dist(X_i, X_j) - L(I_{j,k_2}) < 0$.

Proof: Continuing the arguments in the proof of the previous two Lemmas, a size-2 conflict involves exactly one incoming "special" edge (say $\langle X_j, X_0 \rangle$ annotated with $-L(I_{j,k_2})$ and exactly one outgoing "special" edge (say $\langle X_0, X_i \rangle$ annotated with $R(I_{i,k_1})$. The weight of the negative cycle created is then $R(I_{i,k_1}) + dist(X_i, X_j) - L(I_{j,k_2})$ as required.

Lemma 9: The size-2 conflicts are transitive.

Proof: Suppose we have the two directed conflicts $I_{i,k_1} \rightarrow I_{j,k_2}$ and $I_{j,k_2} \rightarrow I_{l,k_3}$. We have to show that $I_{i,k_1} \rightarrow I_{l,k_3}$ is also true. From the previous Lemma we know that $R(I_{i,k_1}) + dist(X_i, X_j) - L(I_{j,k_2}) < 0$ and $R(I_{j,k_2}) + dist(X_j, X_l) - L(I_{l,k_3}) < 0$. Adding the two inequalities and noting that $dist(X_i, X_l) \leq dist(X_i, X_j) + dist(X_j, X_l)$, we have that $R(I_{i,k_1}) + dist(X_i, X_l) - L(I_{j,k_2}) + R(I_{j,k_2}) - L(I_{l,k_3}) < 0$. Further, since $R(I_{j,k_2}) \geq L(I_{j,k_2})$, we have that $R(I_{i,k_1}) + dist(X_i, X_l) - L(I_{l,k_3}) < 0$. This establishes that $I_{i,k_1} \rightarrow I_{l,k_3}$ as required.

Lemma 10: The size-2 conflicts are acyclic.

Proof: Suppose there was a cycle in the directed size-2 conflicts. By the previous Lemma, this would mean that there is some interval $I_{i,k}$ which conflicts with itself—i.e. $I_{i,k} \rightarrow I_{i,k}$. This would then mean that $R(I_{i,k}) + dist(X_i, X_i) - L(I_{i,k}) < 0$. This is clearly false since $dist(X_i, X_i) = 0$ and $R(I_{i_k}) \geq L(I_{i_k})$. By contradiction, therefore, the truth of the Lemma is established.

Lemma 11: The relation $\succeq$ defined as follows forms a POSET—$I_{i,k_1} \succeq I_{j,k_2}$ if and only if $I_{i,k_1} = I_{j,k_2}$ or there is a size-2 conflict $I_{i,k_1} \rightarrow I_{j,k_2}$.

Proof: We have to show that the relation $\succeq$ is *reflexive*, *anti-symmetric* and *transitive*. By definition, $\succeq$ is reflexive since $I_{i,k_1} \succeq I_{i,k_1}$. The transitive and anti-symmetric properties follow from the previous two Lemmas.

Lemma 12: An ESTP $\mathcal{G}$ is consistent if and only if the size of the largest anti-chain in $P(\mathcal{G})$ is $= n$ (see Figure 5).

Proof: From the previous Lemmas, we know that the only minimal conflicts are of size 1 or size 2. $P(\mathcal{G})$ incorporates the deletion of all the size-1 conflicts and any anti-chain in it incorporates the deletion of all size-2 conflicts. The largest anti-chain targets the maximum number of variables that can be assigned a consistent value together, and since exactly one interval must be active for each variable (Lemma 4), the size of the largest anti-chain must be $= n$ for the ESTP to be consistent.

Solving General ESTPs

In this subsection, we will deal with the most general version of ESTPs where $f_{X_i}(I_{i,k})$ is allowed to be an arbitrary positive real number.

Lemma 13: A general ESTP $\mathcal{G}$ is consistent if and only if the largest weighted anti-chain in $P(\mathcal{G})$ (see Figure 5) using the modified weights $f'_{X_i}(I_{i,k}) = f_{X_i}(I_{i,k}) + M$ is $\geq Mn$.

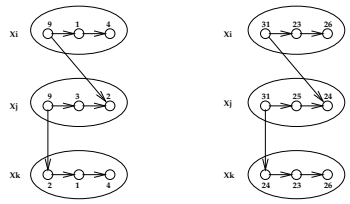

Figure 6: Illustrates the working of Lemmas 13 and 14. The POSET on the left is such that the largest weighted anti-chain in it (9 for X_i and 9 for X_j) is not feasible—i.e. it does not choose exactly one node (interval) for each of the variables X_i, X_j and X_k. The POSET on the right is obtained by adding a factor of 22 to the weights of all the nodes. The largest weighted anti-chain in this POSET (31 for X_i, 25 for X_j, and 26 for X_k) indeed chooses exactly one element from each variable's domain and upon subtracting back the factor 22 from each element in this anti-chain, we get the largest weighted feasible anti-chain for the original POSET (9 for X_i, 3 for X_j, and 4 for X_k). The factor 22 is obtained by summing up the maximum weights in each variable's domain.

Here, $M = \sum_{i=1}^{n} \max\{f_{X_i}(I_{i,k}) : 1 \leq k \leq T(i)+1\}$.

Proof: It suffices to show that any anti-chain that chooses an interval for each variable (feasible) has a greater weight than any anti-chain that does not (infeasible). By the previous Lemma, we know that at most one interval for every variable can be in any anti-chain. This means that the largest value of any infeasible anti-chain is $M(n-1) + \sum_{i=1}^{n} \max\{f_{X_i}(I_{i,k}) : 1 \leq k \leq T(i)+1\}$ and the smallest value for any feasible anti-chain is Mn. Setting $M = \sum_{i=1}^{n} \max\{f_{X_i}(I_{i,k}) : 1 \leq k \leq T(i)+1\}$ therefore ensures the truth of the Lemma.

Lemma 14: The largest weighted anti-chain with the modified weights $f'_{X_i}(I_{i,k}) = f_{X_i}(I_{i,k}) + M$ is the required feasible anti-chain when it is $\geq Mn$.

Proof: Continuing the arguments in the proof of the previous Lemma, the largest anti-chain is guaranteed to pick exactly one interval for each variable with the modified weights. Since any other feasible anti-chain also chooses exactly one interval for each variable, the number of nodes in any feasible anti-chain is equal to n. Further, since the weights of all intervals are increased by the same additive factor, the largest feasible anti-chain using the weights $f'_{X_i}(I_{i,k})$ is also the largest feasible anti-chain using the weights $f_{X_i}(I_{i,k})$, hence establishing the truth of the Lemma.

Figure 6 shows an example of a possible POSET arising in the context of solving an ESTP (unrelated to the example in Figure 3 and Figure 4) and the weight conversions that must be used to find the largest weighted feasible anti-chain in it. In the example in Figure 4, the largest weighted feasible anti-chain is constituted by the "bottom-most" nodes (intervals) for the variables X_2 and X_3, and the "middle" node (interval) for the variable X_4. This means that the agent should replace the domain preference functions with the set of simple temporal constraints of having to reach the bus stop between 8:15 a.m. and 8:30 a.m., start shopping between 8:00 a.m. and 10:00 a.m. and having lunch between 2:00 p.m. and 3:00 p.m. With such a replacement, the resulting STP can be solved in polynomial time using shortest paths (see (Dechter *et al.* 1991)).

Largest Weighted Anti-Chain Computation

Figure 7 presents the algorithm for computing the largest weighted anti-chain in a POSET using *maxflow* techniques.

ALGORITHM: ANTI-CHAIN-WITH-WTS
INPUT: A POSET $P = \langle Y_1, Y_2 \ldots Y_N \rangle$ under the relation $\succeq$ and with weights on nodes respectively $\langle w_1, w_2 \ldots w_N \rangle$.
OUTPUT: The set of nodes Q constituting the largest weighted anti-chain in P.
(1) Construct a bi-partite graph $B = \langle U, V, E \rangle$ where $U = \{Y_1, Y_2 \ldots Y_N\}$, $V = \{Y'_1, Y'_2 \ldots Y'_N\}$ and (Y_i, Y'_j) is an undirected edge in E iff $Y_j \succeq Y_i$ in P and $Y_j \neq Y_i$.
(2) Construct a directed graph D from B as follows:
(a) Add two special nodes S and T.
(b) Add directed edges $\langle S, Y_i \rangle$ with capacity w_i.
(c) Add directed edges $\langle Y'_j, T \rangle$ with capacity w_j.
(d) Impose a direction on all edges (Y_i, Y'_j) in B to get $\langle Y_i, Y'_j \rangle$ and let them be of infinite capacity.
(3) Compute an integral maximum flow F from S to T in D. Let the residual graph be R_F.
(4) Compute $C = \{\langle S, u \rangle | u$ is unreachable from S in $R_F\} \cup \{\langle v, T \rangle | v$ is reachable from S in $R_F\}$.
(5) Compute $V = \{u | \langle S, u \rangle \in C\} \cup \{v | \langle v, T \rangle \in C\}$.
(6) Compute $S = \{Y_i | Y_i \in V \vee Y'_i \in V\}$.
(7) Compute $Q = \{Y_1, Y_2 \ldots Y_N\} \backslash S$.
(8) RETURN Q.
END ALGORITHM

Figure 7: Illustrates the computation of the largest weighted anti-chain in a POSET using *maxflow* techniques.

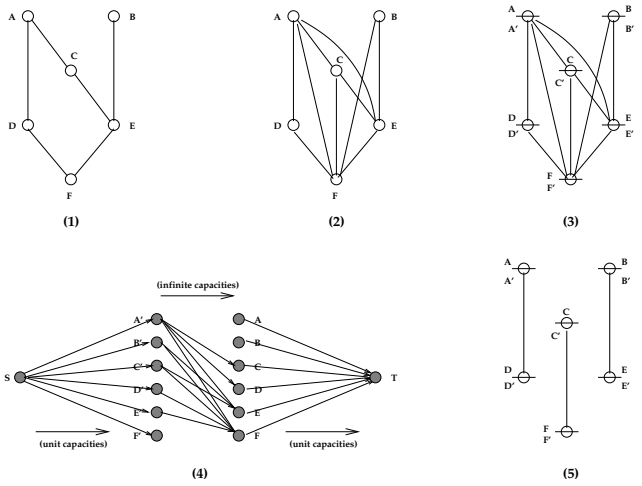

Figure 8: Illustrates the working of the algorithm in Figure 7. Given a POSET under the relation $\succeq$ as in (1), the idea is to first draw all the implicit edges (transitive relationships) to get (2), and split each node into two halves to get (3). Every edge now connects the upper half of some node with the lower half of another and is therefore bi-partite. A *maxflow* (between two auxiliary variables S and T) is staged as in (4) to compute a maximum matching on this bi-partite graph. After the maximum matching is computed, the two halves of every node are merged back to get a chain decomposition of the POSET as in (5). Further, the minimum chain decomposition is related to the maximum anti-chain as in Lemmas 19 to 25.

To keep the proof of its correctness simple, we reiterate a series of Lemmas (see (Cormen *et al.* 1990)) that first establish its correctness for unit weights (imagine setting all $w_i = 1$ in Figure 7). We then prove a single concluding Lemma that generalizes the proof for arbitrary positive weights. We make use of the standard result that when edges have integral capacities in an instance of the *maxflow* problem, a maximum flow with integral amount of flow on all edges can be efficiently computed (hence justifying step 3 in Figure 7) (Cormen *et al.* 1990).

Definition 3: A matching M in a graph G is a set of edges that do not share a common end-point. The size of a matching (denoted $|M|$) is the number of edges in it, and a maximum matching (denoted M^*) is a matching with maximum size. The vertex cover V for a graph G is defined as a set of nodes in G such that all the edges are covered—i.e. for every edge, at least one end point is in V. The size of a vertex cover (denoted $|V|$) is the number of nodes in it and a minimum vertex cover (denoted V^*) is a vertex cover with minimum size.

Lemma 15: If M^* is the maximum matching in B, then $|M^*| = F$.

Proof: For an integral flow, there cannot exist two edges of the form $\langle Y_i, Y'_{j_1} \rangle$ and $\langle Y_i, Y'_{j_2} \rangle$ both with non-zero flows. This is because the edge $\langle S, Y_i \rangle$ has unit capacity and the flow has to be conserved at Y_i. Similarly, there cannot exist two edges of the form $\langle Y_{i_1}, Y'_j \rangle$ and $\langle Y_{i_2}, Y'_j \rangle$ both with non-zero flows (because $\langle Y'_j, T \rangle$ is of unit capacity). Therefore, an integral flow in D defines a matching in B of the same size and hence a maximum flow F in D defines a maximum matching M^* in B of the same size, making $|M^*| = F$.

Lemma 16: For any graph G, if M^* is the maximum matching in G and V^* is the minimum vertex cover in G, then $|V^*| \geq |M^*|$.

Proof: For any edge in M^*, at least one of its end points must be in V^*. Also, since no two edges in M^* share a common end point, they cannot be covered by the same element in V^*. This means that $|V^*| \geq |M^*|$.

Lemma 17: V constructed in step 5 of Figure 7 is a vertex cover for B and $|V| = F$.

Proof: From the construction of V, u (belonging to the set $\{Y_1, Y_2 \ldots Y_N\}$) is in V if and only if $\langle S, u \rangle$ is in C and v (belonging to the set $\{Y'_1, Y'_2 \ldots Y'_N\}$) is in V if and only if $\langle v, T \rangle$ is in C. This means that $|V| = |C|$. Since C is formed out of all edges that have one end reachable from S and the other unreachable in R_F, it forms a minimum cut between S and T in D. From the *maxflow-mincut* theorem, $|C| = F$, and hence $|V| = F$ as required.

Lemma 18: For the bi-partite graph B, if V^* is the minimum vertex cover, then $|V| = |V^*|$ (where V is the vertex cover constructed for B in step 5 of Figure 7).

Proof: From the above Lemmas, we have that $|V| = F$, $|V^*| \geq |M^*|$ and $|M^*| = F$. This implies $|V| = |V^*|$.

Definition 4: A chain c in a POSET P is a set of nodes $Y_{i_1}, Y_{i_2} \ldots Y_{i_k}$ such that there exists a total order among these nodes under the relation $\succeq$. Y_{i_2} is said to cover Y_{i_1} in c (denoted $\langle Y_{i_2}, Y_{i_1} \rangle$) when $Y_{i_2} \succeq Y_{i_1}$, $Y_{i_2} \neq Y_{i_1}$ and for any distinct third element Y_{i_j} in c, $Y_{i_j} \succeq Y_{i_1} \Rightarrow Y_{i_j} \succeq Y_{i_2}$. A chain-decomposition ρ of P is a set of disjoint chains such that all nodes in P are included in exactly one chain. The size of a chain-decomposition ρ (denoted $|\rho|$) is the number of chains constituting it. The minimum chain-decomposition ρ^* of P is a chain-decomposition of P with minimum size.

Lemma 19: If M is a matching in B, merging Y_i and Y'_i in M produces a chain-decomposition ρ for P.

Proof: It suffices to prove that merging Y_i and Y'_i in M does not produce cover relationships that share a common end point. Suppose such relationships were pro-

duced. They must be of the form $(\langle Y_i, Y_j\rangle, \langle Y_k, Y_j\rangle)$ or $(\langle Y_j, Y_i\rangle, \langle Y_j, Y_k\rangle)$. In the first case, this would mean the existence of the edges $\langle Y_i, Y'_j\rangle$ and $\langle Y_k, Y'_j\rangle$ in M, contradicting the fact that no two edges in M share a common point. Similarly, the second case also leads to a contradiction, hence establishing the truth of the Lemma.

Lemma 20: If ρ is a chain-decomposition in P corresponding to a matching M in B, then $|\rho| + |M| = N$.

Proof: Let the chains in ρ be $c_1, c_2 \ldots c_{|\rho|}$. The number of edges (cover relationships) in any chain c_i is one less than the number of nodes in it. Let M_{c_i} denote the edges in M that are present in c_i. We then have $|M_{c_i}| = |c_i| - 1$. Summing over all chains c_i in ρ, we get $|M| = N - |\rho|$ i.e. $|M| + |\rho| = N$.

Lemma 21: If Q^* is the largest anti-chain, $|\rho^*| \geq |Q^*|$.

Proof: By definition, no two nodes in Q^* are comparable and hence must be in different chains of ρ^*. Hence by the pigeonhole principle, we have $|\rho^*| \geq |Q^*|$.

Lemma 22: If the minimum vertex cover for B is V^*, then $S = \{Y_i | Y_i \in V^* \vee Y'_i \in V^*\}$ is a vertex cover for P with $|S| \leq F$.

Proof: For every Y_i in S, at least one of Y_i or Y'_i must be present in V^*. Therefore, $|S| \leq |V^*| = F$. Also, S forms a vertex cover for P, because if there existed an edge $\langle u, v\rangle$ in P not covered by S, then there must exist some $Y_j = u$ and $Y'_k = v$ such that $\langle Y_j, Y'_k\rangle$ is in B and is not covered by V^*. This contradicts that V^* is a vertex cover for B and hence S must be a vertex cover for P.

Lemma 23: $Q = P \backslash S$ is an anti-chain and $|Q| \geq N - F$.

Proof: Q is an anti-chain because if there existed two nodes u and v in Q that were comparable, then the edge $\langle u, v\rangle$ must have been uncovered by S contradicting that S is a vertex cover for P. Also, since $|Q| + |S| = N$ and $|S| \leq F$, $|Q| \geq N - F$.

Lemma 24: Q (calculated in step 7 of Figure 7) is the required largest weighted anti-chain.

Proof: Suppose Q^* was the optimal. By Lemma 23 we have that $|Q^*| \geq |Q| \geq N - F$. By Lemma 21 we also have that $|\rho^*| \geq |Q^*| \geq |Q| \geq N - F$. Again by Lemma 20 we know that $|\rho^*| \leq N - |M^*|$. Since $|M^*| = F$, we get $|\rho^*| = |Q^*| = |Q| = N - F$, hence making Q optimal as required.

Lemma 25: The algorithm presented in Figure 7 works for arbitrary positive weights $w_i > 0$.

Proof: From the foregoing Lemmas, we know that the algorithm works for unit weights—i.e. $w_i = 1$. Now suppose that the weights were positive integers (still not the general case). Conceptually, a new POSET can be constructed where node Y_i with weight w_i is replicated w_i times—each of unit weight and incomparable to each other. If $Y_i \succ Y_j$ (short for saying $Y_i \succeq Y_j$ and $Y_i \neq Y_j$), then all w_i copies of Y_i are made to have a $\succ$ relation to all w_j copies of Y_j. The staged *maxflow* in D will have all w_i copies of Y_i (denoted $y_{i_1}, y_{i_2} \ldots y_{i_{w_i}}$) behaving identically. Also since all edges of the form $\langle y_{i_j}, y'_{k_l}\rangle$ have infinite capacity, we can replace the group of edges $\langle S, y_{i_1}\rangle, \langle S, y_{i_2}\rangle \ldots \langle S, y_{i_{w_i}}\rangle$ (each of unit capacity) with a single edge $\langle S, Y_i\rangle$ of capacity w_i. Similarly we can replace the group of edges $\langle y'_{i_1}, T\rangle, \langle y'_{i_2}, T\rangle \ldots \langle y'_{i_{w_i}}, T\rangle$ (each of unit capacity) with a single edge $\langle Y'_i, T\rangle$ of capacity w_i. All intermediate edges are of infinite capacity and are defined (as previously) using the $\succ$ relation. Now consider the most general case where w_i is positive but need not be an integer. In such a case, the idea is to conceptually scale all the weights by a uniform factor L, such that all of them become integers. The more the precision of the numbers, the larger we can choose L—but since this is only conceptual, L does not have a concrete role in the algorithm. The algorithm can then find the largest anti-chain using the scaled weights and since scaling the weights uniformly in a POSET does not affect the largest anti-chain, the same can be used after scaling down the weights by L. Computationally however, the idea of scaling is not reflected anywhere except in the fact that the weights w_i can be used as they are to define capacities on the edges in D.

Conclusions

We described a class of metric temporal problems (which we referred to as extended STPs (ESTPs)) that formed a middle ground between STPs and DTPs. We showed that ESTPs could be solved in polynomial time and were expressive enough to deal with limited forms of disjunctions and preferences that would otherwise require an exponential search space. Our polynomial-time algorithm for solving ESTPs was based on the idea of reducing a given ESTP to the problem of computing the largest weighted anti-chain in a POSET which in turn could be solved using *maxflow* techniques. The expressive power of ESTPs along with their tractability makes them a suitable model for many real-life applications that involve metric temporal reasoning.

References

Armando A., Castellini C. and Giunchiglia E. SAT-based Procedures for Temporal Reasoning. *ECP 2000.*

Cormen T. H., Leiserson C. E. and Rivest R. Introduction to Algorithms. *Cambridge, MA, 1990.*

Dechter R., Meiri I. and Pearl J. Temporal Constraint Networks. *Artificial Intelligence Journal 49. 1991.*

Khatib L., Morris P. H., Morris R. A. and Rossi F. Temporal Constraint Reasoning With Preferences. *IJCAI 2001.*

Kumar T. K. S. Incremental Computation of Resource-Envelopes in Producer-Consumer Models. *CP 2003.*

Nguyen X. and Kambhampati S. Reviving Partial Order Planning. *IJCAI-2001.*

Oddi A. and Cesta A. Incremental Forward Checking for the Disjunctive Temporal Problem. *ECAI 2000.*

Smith D., Frank J. and Jonsson A. Bridging the Gap Between Planning and Scheduling. *Knowledge Engineering Review 15:1, 2000.*

Stergiou K. and Koubarakis M. Backtracking Algorithms for Disjunctions of Temporal Constraints. *In the 15th National Conference on Artificial Intelligence. 1998.*

Tsamardinos I. and Pollack M. E. Efficient Solution Techniques for Disjunctive Temporal Reasoning Problems. *Artificial Intelligence, 2003.*