

Temporal Preference Optimization as Weighted Constraint Satisfaction

Michael D. Moffitt and Martha E. Pollack

Department of Electrical Engineering and Computer Science

University of Michigan

Ann Arbor, MI 48109, USA

{mmoffitt, pollackm}@eecs.umich.edu

Abstract

We present a new efficient algorithm for obtaining utilitarian optimal solutions to Disjunctive Temporal Problems with Preferences (DTPPs). The previous state-of-the-art system achieves temporal preference optimization using a SAT formulation, with its creators attributing its performance to advances in SAT solving techniques. We depart from the SAT encoding and instead introduce the Valued DTP (VDTP). In contrast to the traditional semiring-based formalism that annotates legal tuples of a constraint with preferences, our framework instead assigns elementary costs to the constraints themselves. After proving that the VDTP can express the same set of utilitarian optimal solutions as the DTPP with piecewise-constant preference functions, we develop a method for achieving *weighted constraint satisfaction* within a meta-CSP search space that has traditionally been used to solve DTPs without preferences. This allows us to directly incorporate several powerful techniques developed in previous decision-based DTP literature. Finally, we present empirical results demonstrating that an implementation of our approach consistently outperforms the SAT-based solver by orders of magnitude.

Introduction

Several recent studies have addressed the topic of *preferential optimization* in constraint-based temporal reasoning (Khatib *et al.* 2001). In this line of research, traditional temporal constraints (Dechter, Meiri, & Pearl 1991) are augmented with local preference functions that express how well a particular assignment satisfies the corresponding constraint. For instance, these functions might convey that a certain activity should be as long as possible, or that it is desirable for a pair of activities to be scheduled very close to one another. Early versions of this research focused on the problem of maximizing the minimum such preference value (Khatib *et al.* 2003; Peintner & Pollack 2004), although later developments have begun to address the more challenging problem of utilitarian optimization (Kumar 2004; Morris *et al.* 2004; Peintner & Pollack 2005), where the sum of the individual preference values is maximized.

Recently, Sheini *et al.* (2005) introduced a highly efficient approach to finding utilitarian optimal solutions to Disjunc-

tive Temporal Problems with Preferences (DTPPs), a powerful representation that subsumes many other common temporal formalisms. In their work, the logical structure of the DTPP is decomposed into a Mixed Logical Linear Program, where the Boolean literals are linked to simple temporal constraints. The resulting problem is solved by a system named ARIO, in which a top-level SAT solver invokes a special-purpose temporal constraint engine. Experimental results showed that ARIO is several orders of magnitude faster than alternative methods for solving DTPPs (Peintner 2005), and the creators of ARIO credit advances in SAT techniques as the principal reason for this success.

In this paper, we present a new efficient algorithm for obtaining utilitarian optimal solutions to Disjunctive Temporal Problems with Preferences. To facilitate our approach, we introduce the Valued DTP (VDTP). In contrast to the traditional semiring-based formalism that annotates legal tuples of a constraint with preferences, our framework instead assigns elementary costs to the constraints themselves. While this reformulation provides no increase in expressive power, it simplifies some of the computational difficulties related to temporal optimization, since search strategies for disjunctive temporal reasoning typically view constraints as meta-level variables. After proving that the VDTP can express the same set of utilitarian optimal solutions as the DTPP with piecewise-constant preference functions, we develop a method for achieving *weighted constraint satisfaction* within a meta-CSP search space that has traditionally been used to solve DTPs without preferences. This allows us to directly incorporate techniques developed in previous decision-based DTP literature in order to make preferential optimization particularly efficient. Finally, we present empirical results demonstrating that an implementation of our approach consistently outperforms the SAT-based solver by orders of magnitude.

Background

Disjunctive Temporal Problems A *Disjunctive Temporal Problem* (DTP) (Stergiou & Koubarakis 1998) is a constraint satisfaction problem defined by a pair $\langle X, C \rangle$, where each element $X_i \in X$ designates a time point, and each element $C_i \in C$ is a constraint of the form: $c_{i1} \vee c_{i2} \vee \dots \vee c_{in_i}$ where in turn, each c_{ij} is of the form: $a_{ij} \leq x_{ij} - y_{ij} \leq b_{ij}$ with $x_{ij}, y_{ij} \in X$ and $a_{ij}, b_{ij} \in \mathfrak{R}$ (we will refer to the

interval $[a_{ij}, b_{ij}]$ as the *feasible region* for c_{ij}). DTPs are thus a generalization of Simple Temporal Problems (STPs), in which each constraint is limited to a single disjunct.

There are generally two ways of defining a solution to a DTP. The first of these is as an object-level assignment of a numeric value to each of the time points in X , such that all the constraints in C are satisfied. A second type of solution is a *meta-CSP* assignment. Here, instead of directly considering assignments to the time points in X , a meta-variable C_i is created for each constraint in the DTP. The domain $D(C_i)$ is simply the set $\{c_{i1}, c_{i2}, \dots, c_{in_i}\}$, representing the various disjuncts one can choose to satisfy that disjunctive constraint. A complete assignment in the meta-CSP thus involves a selection of a single disjunct for each constraint, commonly referred to as a *component STP*.

Within the meta-CSP formulation, the constraints are implicitly defined by the underlying semantics of the disjuncts: the values (disjuncts) assigned to each meta-variable must be mutually consistent. The consistency of a set \mathcal{S} of such inequalities can be determined by first constructing its *distance graph*, a graph that includes a node for each time point and an arc with weight b from y to x whenever $x - y \leq b$ is in \mathcal{S} . Then \mathcal{S} is consistent if and only if its distance graph contains no negative cycles, which can be determined in polynomial time by computing its all-pairs shortest path (APSP) matrix and checking the entries along the main diagonal (Dechter, Meiri, & Pearl 1991).

Disjunctive Temporal Problems with Preferences To extend a DTP to a DTP with Preferences (DTPP) (Peintner & Pollack 2004), each disjunct $c_{ij} : a_{ij} \leq x_{ij} - y_{ij} \leq b_{ij}$ is augmented with a preference function $\langle f_{ij} : t \in [a_{ij}, b_{ij}] \rightarrow \{0, \mathbb{R}^+\} \rangle$ that maps every allowable temporal difference to a *preference value* expressing its relative utility (Khatib *et al.* 2001).¹ DTPPs subsume STPPs (Simple Temporal Problems with Preferences) in the same way that DTPs subsume STPs. Given a solution S to the DTPP D , the preference value of a disjunctive constraint C_i in C is defined to be the maximum value achieved by any of its disjuncts:

$$val_D(S, C_i) = \max_{c_{ij} \in D(C_i)} f_{ij}(x_{ij} - y_{ij})$$

With the addition of preferences, we are no longer concerned with simply finding a feasible solution; we also want a solution of high quality. This requires us to specify an *objective function* with respect to each of the individual preference functions. In this paper, we will consider the useful *utilitarian* objective, where the global value of a solution S is equal to the sum of the preference values of the individual constraints (Morris *et al.* 2004):

$$val_D(S) = \sum_i val_D(S, C_i)$$

Solving DTPPs Two approaches have been developed in previous literature for performing utilitarian optimization of a DTPP. Both can handle problems containing complex

preference functions, requiring only that they be piecewise-constant in shape.²

The first is based on a SAT reformulation of a DTPP (Sheini *et al.* 2005). It involves the creation of a Mixed Logical Linear Programming (MLLP) problem composed of two types of constraints: logical constraints over Boolean variables, and Unit-Two-Variable-Per-Inequality (UTVPI) integer constraints of the form $ax - by \leq d$, where $a, b \in \{-1, 0, 1\}$. The disjuncts in the DTPP are converted to a set of UTVPI constraints, a Boolean *indicator variable* is created for each constraint, and a SAT problem is constructed in which these indicator variables are used to represent the logical structure of the DTPP. The reformulated problem is then solved by system named ARIO, which is composed of a tightly integrated UTVPI engine and SAT solver. Since this approach can handle only the decision variant of the DTPP, optimization is achieved by repeatedly calling the combined constraint engine on a sequence of satisfaction problems with increasingly higher objectives until no feasible solution can be found. Efficient SAT-solving techniques (Moskewicz *et al.* 2001) make the approach taken by ARIO particularly attractive.

The second approach, named GAPD (*Greedy Anytime Partition algorithm for DTPPs*) was designed exclusively for the purpose of solving DTPPs (Peintner 2005). It is based largely on the GAPS algorithm (Peintner & Pollack 2005) for finding utilitarian optimal solutions to STPPs. It begins by first searching for a consistent component STP S to the DTP D induced by fixing all constraints in the DTPP at their bottommost preference level of 0. It then either uses the GAPS algorithm to find an optimal solution to the STPP S' corresponding to S , or computes another solution S'' to the DTP D , and repeats. In this way, the disjunctive search for feasible solutions is almost completely decoupled from the process of optimization. Unfortunately, the memory requirements of GAPS (and GAPD) are exponential in the size of the STPP (and DTPP). Furthermore, they have both been shown to be several orders of magnitude slower than ARIO for finding optimal solutions. However, as the names suggest, these algorithms have desirable anytime properties, and are indeed complete algorithms.

Valued Disjunctive Temporal Problems

The DTP with Preferences has typically been regarded as a type of *semiring* formulation (Bistarelli, Montanari, & Rossi 1997), in which preference values are attributed to the (infinitely many) legal object-level tuples that comprise a constraint. While being expressive, this modeling of preferences presents some computational challenges, since search strategies for disjunctive temporal reasoning operate on the meta-CSP rather than invoking object-level assignments directly. In response, we will introduce a variation of the DTP where the disjunctive constraints themselves are associated with costs, making our representation comparable to early versions of the Valued CSP formalism for finite-domain constraints (Schiex, Fargier, & Verfaillie 1995).

¹Note that zero is the minimum preference value that can be obtained by an assignment that satisfies the constraint.

²DTPPs containing other preference function shapes can be approximated by piecewise-constant functions via discretization.

Definition: A *Valued Disjunctive Temporal Problem* is a tuple $\langle X, C, S, \varphi \rangle$, where X and C are as in a DTP, S is a valuation structure (E, \otimes, \succ) , and φ is a mapping from C to E . \square

In the valuation structure, E is a set whose elements are called *valuations*, and are totally ordered by \succ . The symbol \otimes denotes a commutative, associative closed binary operation on E , and expresses how to aggregate the individual valuations. The function φ simply maps each constraint C_i to an element of E , indicating how important it is for the constraint to be satisfied.

Rather than explore the entire class of problems that the Valued DTP can represent, we will focus our attention on a special *weighted* case, where $E = \mathbb{R}^+ \cup \{\infty\}$ and $\otimes = +$ (i.e., arithmetic sum), using the usual ordering $<$. In other words, each constraint C_i is associated with a positive numeric weight (which we subsequently refer to as w_i instead of $\varphi(C_i)$ to improve readability), and the objective will be to find an assignment S that imposes the minimal cost, where the *cost* is defined to be the weighted sum of violated constraints in the VDTP D :

$$\text{cost}_D(S) = \sum_i \{w_i | \text{violates}(S, C_i)\}$$

As an example, we present the following (very small) instance of our weighted VDTP:

$$\begin{array}{l} C_1: \{c_{11}: 1 \leq x - y \leq 2\} \\ C_2: \{c_{21}: 3 \leq x - y \leq 4\} \vee \{c_{22}: 5 \leq x - z \leq 6\} \\ C_3: \{c_{31}: 1 \leq y - z \leq 2\} \\ C_4: \{c_{41}: 0 \leq x - z \leq 7\} \end{array} \quad \left| \begin{array}{l} w_1 = 1 \\ w_2 = 2 \\ w_3 = 4 \\ w_4 = \infty \end{array} \right.$$

Clearly there is no assignment that will satisfy all the constraints of this problem, since c_{21} conflicts with c_{11} , and c_{22} conflicts with the constraint induced by the composition of c_{11} and c_{31} . In addition, C_4 is a hard constraint having infinite weight, and therefore must be satisfied in any solution.

Once again, we can consider object-level and meta-level solutions to our VDTP. For instance, the object-level assignment $(x, y, z) \leftarrow (6, 3, 1)$ violates only constraint C_1 . This has a cost of 1, and since we know that no solution exists with a cost of 0, it is an optimal solution. Importantly, when we move to the meta-CSP, a solution is no longer necessarily a total assignment; instead a (meta)-variable may be left unassigned, signifying that none of the disjuncts associated with it should be enforced.

The Relationship Between DTPPs and VDTPs

The previous example illustrates an important distinction between DTPPs and VDTPs. Each constraint in a DTPP plays a dual role, serving both as a hard constraint that requires assignments to be within the feasible region, and as a soft constraint expressing the preference values that the assignments within that region will receive. In contrast, each VDTP constraint is either strictly hard (i.e., having an infinite weight, and thus requiring satisfaction) or strictly soft (i.e., having finite weight, and permitting violation).

Nonetheless, we can show that for purposes of utilitarian optimality, any DTPP D' has an equivalent VDTP D in the following sense: (i) an assignment S is a solution to D' iff

it is a solution to D , and (ii) solution S_1 is at least as preferred as S_2 in D' ($\text{val}_{D'}(S_1) \geq \text{val}_{D'}(S_2)$), iff S_1 is also at least as preferred as S_2 in D ($\text{cost}_D(S_1) \leq \text{cost}_D(S_2)$). In fact, we prove something stronger: we show that in addition, given any VDTP D , there is a DTPP D' that is equivalent in exactly the same sense, and hence the VDTP and DTPP formalisms have equivalent expressive power. This relationship requires only that the preference functions be piecewise-constant, an assumption commonly made in prior DTPP research.

Converting a VDTP into a DTPP

We begin by showing how to convert a VDTP into an equivalent DTPP. Let D be a VDTP with constraints $\{C_1, \dots, C_n\}$ where $C_i = c_{i1} \vee \dots \vee c_{in_i}$ and $c_{ij} = a_{ij} \leq x_{ij} - y_{ij} \leq b_{ij}$ and w_i is the weight of C_i . Create the *derived DTPP* D' by constructing a *derived constraint* C'_i from each constraint C_i in D :

- If C_i is hard (i.e., $w_i = \infty$), let $C'_i = c'_{i1} \vee \dots \vee c'_{in_i}$ where $c'_{ij} = a_{ij} \leq x_{ij} - y_{ij} \leq b_{ij}$ with preference function $\langle f_{ij} : t \in [a_{ij}, b_{ij}] \rightarrow 0 \rangle$. Here, C'_i is a *hard-derived constraint*.
- Otherwise, if C_i is soft (i.e., $w_i \neq \infty$), let $C'_i = c'_{i1} \vee \dots \vee c'_{in_i}$ where $c'_{ij} = -\infty \leq x_{ij} - y_{ij} \leq \infty$ with preference function $\langle f_{ij} : t \in [a_{ij}, b_{ij}] \rightarrow w_i, t \notin [a_{ij}, b_{ij}] \rightarrow 0 \rangle$. Here, C'_i is a *soft-derived constraint*.

In what follows, we distinguish the *constraint component* (i.e., the disjunctive constraints themselves) from the weights or the preference functions. It will also be useful to refer to the *worth* of a VDTP solution S , which is the weighted sum of soft constraints that are satisfied by S :

$$\text{worth}_D(S) = \sum_i \{w_i | w_i \neq \infty \wedge \text{satisfies}(S, C_i)\}$$

Note that the worth of a solution is the inverse of its cost, hence we wish to maximize this metric. We now show that every VDTP has an equivalent DTPP.

Theorem 1. VDTP D and its derived DTPP D' are equivalent.

Lemma 1. An assignment S is a solution to D if and only if it is a solution to D' .

Let $S : X \rightarrow \mathbb{R}$ be an object-level assignment. First, assume S is a solution to the VDTP D . Then, for any arbitrary hard constraint C_i in D , S satisfies C_i . Since the constraint component of each hard-derived constraint C'_i in D' is identical to that of the hard constraint in D from which it was derived, S satisfies all hard-derived constraints in D' ; it also satisfies all soft-derived constraints, since each of those contain only disjuncts with infinite feasible regions, and are thus satisfied by any assignment. So S is a solution to D' . Similarly, assume that S is a solution to the DTPP D' . Since it satisfies all hard-derived constraints C'_i in D' , it must satisfy every hard constraint in D , because again, the corresponding constraints have identical constraint components. Since S need not satisfy the soft constraints in D , it is thus also a solution to D . \square

Lemma 2. If S_1 and S_2 are solutions to D (and D'), then $worth_D(S_1) \geq worth_D(S_2)$ iff $val_{D'}(S_1) \geq val_{D'}(S_2)$.

The worth of the VDTP D is as defined above. For a given solution S , the value of the DTPP D' is:

$$\begin{aligned} val_{D'}(S) &= \sum_i val_{D'}(S, C'_i) \\ &= 0 + \sum_i \{val_{D'}(S, C'_i) | soft_derived(C'_i)\} \\ &= \sum_i \{w_i | w_i \neq \infty \wedge satisfies(S, C_i)\} \end{aligned}$$

The second equality follows from the fact that the preference function for all hard-derived constraints is a constant zero. The third is because, by construction, for every soft-derived constraint C'_i , $val_{D'}(S, C'_i) = 0$ if the constraint from which it was derived (C_i) is not satisfied, and $val_{D'}(S, C'_i) = w_i$ if it is satisfied. Consequently, $worth_D(S) = val_{D'}(S)$, and so the lemma follows trivially. \square

Theorem 1 follows directly from Lemmas 1 and 2.

Converting a DTPP into a VDTP

The conversion of a DTPP to a VDTP is slightly more complicated, and relies on the notion of *preference projections* (Peintner & Pollack 2005). An STPP preference projection ‘‘slices’’ an STPP constraint into a set of intervals that produce a preference value greater than or equal to some specified level l . A DTPP preference projection generalizes this notion to all intervals (disjuncts) in a DTPP constraint.

Definition (STPP Preference Projection). Given an STPP constraint $C_{ij} = \langle a_{ij} \leq x_{ij} - y_{ij} \leq b_{ij}, f_{ij} \rangle$, the **preference projection at level l** for C_{ij} is $\mathcal{P}_{ij}[l] = \{c_1, c_2, \dots, c_n\}$, where $c_k = \langle a_k \leq x_{ij} - y_{ij} \leq b_k \rangle$, $b_k < a_{k+1}$ for $1 \leq k < n$ and $\bigcup_{k=1}^n [a_k, b_k] = \{t | f_{ij}(t) \geq l\}$.

Definition (DTPP Preference Projection). Given a DTPP constraint $C_i = c_{i1} \vee c_{i2} \vee \dots \vee c_{in}$, the **preference projection at level l** for C_i is $\mathcal{P}_i[l] = \bigcup_{j=1}^n \mathcal{P}_{ij}[l]$.

In converting a DTPP into an equivalent VDTP, the basic idea will be to create multiple VDTP constraints for each individual DTPP constraint: one for each distinct preference level. Weights will be assigned in such a way that satisfying all projected constraints through level k will result in a *total* weight of k . The procedure is as follows:

Let D' be a DTPP with constraints $\{C'_1, \dots, C'_n\}$. Then create the *derived* VDTP D as follows. For each constraint C'_i in D' :

- Create a hard constraint $C_{\langle i,0 \rangle}$ in D , where $C_{\langle i,0 \rangle} = \bigvee \mathcal{P}_i[0]$ and $w_{i,0} = \infty$. Set l to zero.
- Find the smallest $l' > l$ such that $\mathcal{P}_i[l'] \neq \mathcal{P}_i[l]$.
- Create a soft constraint $C_{\langle i,l' \rangle}$ in D , where $C_{\langle i,l' \rangle} = \bigvee \mathcal{P}_i[l']$ and $w_{i,l'} = (l' - l)$. Set l to l' .
- Iterate until an l' is reached such that $\mathcal{P}_i[l'] = \emptyset$.

Theorem 2. DTPP D' and its derived VDTP D are equivalent.

Lemma 3. An assignment S is a solution to D' if and only if it is a solution to D .

Suppose S is a solution to D' . By definition, it must necessarily satisfy every C'_i in D' . As in Lemma 1, we note that every hard constraint $C_{\langle i,0 \rangle}$ in D has a constraint component identical to that of the C'_i from which it was derived;

thus, these must be satisfied as well, as must D as a whole (since the soft constraints may be violated by any solution). Now assume that S is a solution to D . It then necessarily satisfies every hard constraint $C_{\langle i,0 \rangle}$ (and observe that these are the only hard constraints). It must then also satisfy every C'_i in D' since the constraint components are again identical. Hence, S is a solution to D' . \square

Lemma 4. If S_1 and S_2 are solutions to D' (and D), then $worth_D(S_1) \geq worth_D(S_2)$ iff $val_{D'}(S_1) \geq val_{D'}(S_2)$.

For an arbitrary constraint C'_i in D' , let $val_{D'}(S, C'_i) = k_i$. Then by construction, S will satisfy any constraint $C_{\langle i,l \rangle}$ in D for which $l \leq k_i$. Furthermore, S will not satisfy any constraint $C_{\langle i,m \rangle}$ in D for which $m > k_i$. Also, by construction we have:

$$\sum_l \{w_{i,l} | l \neq 0 \wedge l \leq k_i\} = k_i$$

Since this holds for all constraints C'_i in D' , we again get $worth_D(S) = val_{D'}(S)$, and the lemma follows immediately.³ \square

Theorem 2 follows directly from Lemmas 3 and 4.

Theorem 3. For the objective of utilitarian optimality, DTPPs and VDTPs are equivalent in expressive power.

This follows directly from Theorems 1 and 2. \square

The fact that a utilitarian aggregation of preference values can be obtained by a Valued DTP should come as little surprise, since this objective function is commonly found in Valued CSP literature (Freuder & Wallace 1992; Schiex, Fargier, & Verfaillie 1995). In addition, similar proofs of equivalence have been drawn between the Valued CSP and Semiring CSP for finite-domains (Bistarelli *et al.* 1999). However, much of the prior work on preferences in TCSPs has had no need to move to the valued representation, in part because these studies have exposed tractable cases of the semiring that require either weaker objective functions (e.g., Weakest Link Optimality (Khatib *et al.* 2003)) or different shapes of preferences (e.g., convex piecewise-linear functions (Morris *et al.* 2004)). In contrast, our focus is on a broad range of intractable instances where this alternative modeling of preferences can be more easily exploited.

Solving Valued DTPs

We have just shown that for utilitarian optimization, any DTPP with piecewise-constant preference functions can be translated into an equivalent VDTP. Using this reformulation, we will describe a branch-and-bound algorithm for finding utilitarian optimal solutions to VDTPs; pseudocode is given in Figure 1. The input variable A is the current set of assignments to meta-variables, and is initially \emptyset ; variable U is the set of unassigned meta-variables (initially the entire set C); $cost$ is the total weighted sum of violated constraints (initially zero); and $upperbound$ is the stored cost of the best solution found so far (initially set to ∞). Note that unlike

³One interesting consequence of this construction is that several disjunctive constraints may be generated from a single non-disjunctive STPP constraint, if its preference function is not semi-convex.

```

Solve-VDTP( $A, U, cost, upperbound$ )


---


If ( $cost \geq upperbound$ ) return
If ( $U = \emptyset$ )
     $best\text{-}solution\text{-}so\text{-}far \leftarrow A$ 
     $upperbound \leftarrow cost$ 
    return
EndIf
 $C_i \leftarrow select\text{-}variable(U), U' \leftarrow U - \{C_i\}$ 
For each disjunct  $c_{ij}$  of  $D(C_i)$ 
     $A' \leftarrow A \cup \{C_i \leftarrow c_{ij}\}$ 
    If (consistent( $A'$ ))
        Solve-VDTP( $A', U', cost, upperbound$ )
    EndIf
EndFor
 $A' \leftarrow A \cup \{C_i \leftarrow \epsilon\}$ 
Solve-VDTP( $A', U', cost + w_i, upperbound$ )

```

Figure 1: A branch-and-bound algorithm for solving VDTPs

ARIO, we require no MLLP module or SAT-solving component, and unlike GAPD, our memory requirements are polynomial in the size of the problem.

This algorithm, which takes an approach similar to that in (Moffitt & Pollack 2005), resembles the meta-CSP backtracking search commonly used for solving traditional DTPs with two notable differences. First, backtracking occurs only when the combined weight of the violated constraints ($cost$) equals or exceeds that of the current best solution ($upperbound$); in a standard DTP solver, backtracking would occur whenever $cost$ became nonzero (i.e., when any constraint had been violated). Second, in addition to the values in the original domains of the meta-variables, there is the possibility of an empty assignment (ϵ) that serves to explicitly violate a constraint, and so the branching factor increases by exactly one. This latter modification, in combination with the meta-CSP search space employed in temporal reasoning, sets our algorithm apart from previous applications of weighted constraint satisfaction to classical CSPs.

The key advantages to our meta-CSP reformulation of DTPP optimization are that (1) it performs optimization without the need to solve a sequence of satisfaction problems, and (2) by encoding the relaxation (or violation) of a constraint as an additional empty disjunct, it allows the direct incorporation of several powerful techniques previously developed in decision-based DTP literature (Stergiou & Koubarakis 1998; Armando, Castellini, & Giunchiglia 1999; Oddi & Cesta 2000; Tsamardinos & Pollack 2003), including incremental forward checking, semantic branching, removal of subsumed variables, conflict-directed backjumping, no-good recording, and several heuristics. Previous CSP-based algorithms applied these only to the small portion of the search space corresponding to the DTPP’s lowest preference level, a possible explanation of their poor performance compared to the SAT-based solver.

In addition to the branch-and-bound algorithm, an alternative *iterative weakening* approach to optimization (Provost 1993) is also available, which begins with infeasibly high objective values and works downwards until consistency is achieved; see (Moffitt & Pollack 2005) for details.

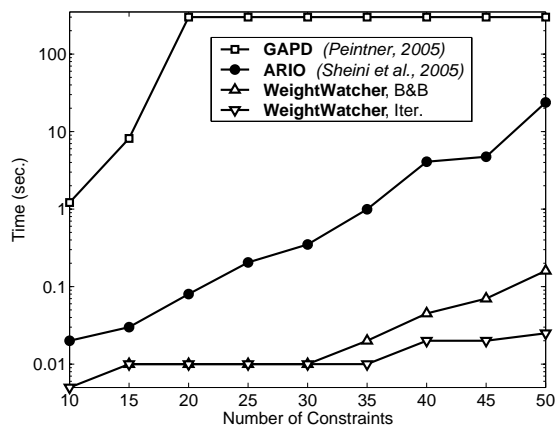


Figure 2: Median running times for GAPD, ARIO, and WEIGHTWATCHER for DTPPs of varying sizes

Experimental Results

In this section, we describe the results of a set of experiments that were performed to compare the weighted constraint satisfaction approach of our solver (which we name WEIGHTWATCHER) against the two previous systems for solving DTPPs: ARIO⁴ (Sheini *et al.* 2005) and the GAPD solver (Peintner 2005). All tests were performed on a 2.26GHz Pentium 4 processor with 1 GB of RAM.

Unfortunately, there remains an absence of real-world benchmarks in temporal preference literature with which to provide an empirical comparison of solvers. Consequently, we must employ the same problem generator used in prior DTPP studies, which takes as parameters $\langle E, C, D_-, D_+, L, R_-, R_+ \rangle$. The DTPP is constructed by generating a set of E events $\{x_1, x_2, \dots, x_E\}$ and a set of C constraints, where each constraint C_i consists of exactly 2 disjuncts. Each disjunct c_{ij} is assigned a pair of events x_{ij} and y_{ij} , and the lower and upper bounds a_{ij} and b_{ij} on the feasible difference between those events are selected from the interval $[D_-, D_+]$. To define the preference function for each temporal difference, the values a_{ij} and b_{ij} serve as the endpoints for preference level 0. To construct preference level l , a reduction factor is chosen from the interval $[R_-, R_+] \subset [0, 1]$ with uniform probability, and is applied to the interval at preference level $l-1$; the resulting (smaller) interval is placed randomly between its endpoints. This process is repeated until preference level L is reached or an interval having zero length is created.

We ran three sets of experiments, in which we varied the problem size, number of preference levels, and constraint density. For all experiments, we generate 30 trials for each setting of parameters, and report the median running time

⁴At the time of this writing, ARIO is the only participant of the recent SMT (satisfiability modulo theories) competition (Barrett, de Moura, & Stump 2005) whose difference-logic engine has been augmented with a reasoning module (in ARIO’s case, an MLLP solver) that allows it to handle the DTPP objective function and solve optimization (as opposed to satisfaction) problems.

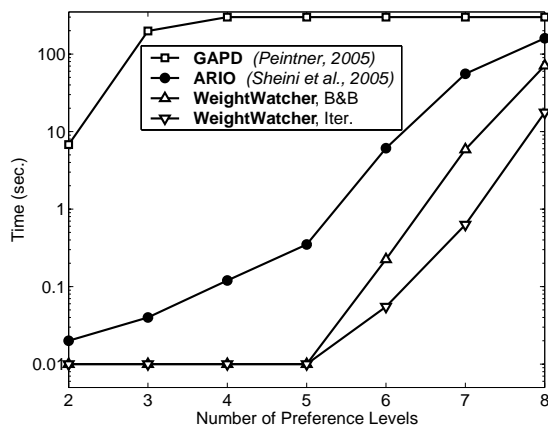


Figure 3: Median running times for GAPD, ARIO, and WEIGHTWATCHER for DTPPs with varying numbers of preference levels

for each solver over the 30 trials. A timeout of 300 seconds is enforced on all problems. The time required to convert the DTPP into the VDTP is included in these runtimes, but is extremely negligible.

Varying Problem Size In our first experiment, we explore the abilities of WEIGHTWATCHER, ARIO, and GAPD to scale with the size of the problem. Our ability to perform well on this set of tests is especially important, since unlike other problem parameters (such as the number of preference levels), the size of the problem is often difficult for a knowledge engineer to control directly. We hold fixed the following parameters: $\{D_- = -50, D_+ = 100, L = 5, R_- = 0.5, R_+ = 0.9\}$. These settings are identical to those used in (Sheini et al. 2005). We vary the number of constraints C from 10 to 50, and set the number of events E to $\frac{4}{3}C$ to maintain a constant constraint density.

Figure 2 displays the results of this experiment. The number of constraints in the problem is shown on the x -axis, and on the y -axis is the median running time (note the logarithmic scale). GAPD quickly reaches the timeout limit of 300 seconds once the number of constraints equals $C = 25$. The median runtime of ARIO consistently remains far below the cutoff threshold, and reaches 23.8 seconds when $C = 50$. Recall that the presence of an efficient SAT solver has been labeled as the key ingredient in achieving this substantial improvement. Yet, the branch-and-bound and iterative versions of WEIGHTWATCHER surpass both GAPD and ARIO on all problem sizes, without the aid of SAT techniques. For $C = 50$ (the largest set of problems), the median runtime of the iterative version is 0.025 seconds, three orders of magnitude faster than ARIO.

Varying the Number of Preference Levels In our second experiment, we examine the effect of the number of preference levels on the performance of these solvers. We hold fixed the parameters $\{E = 24, C = 30, D_- = -50, D_+ = 100, R_- = 0.5, R_+ = 0.9\}$, and vary the number of preference levels L between 2 and 8.

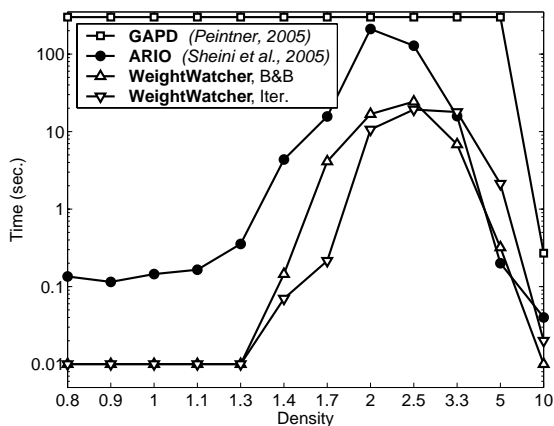


Figure 4: Median running times for GAPD, ARIO, and WEIGHTWATCHER for DTPPs of varying constraint density

Figure 3 provides the results of this experiment. Once again, GAPD tends to be much slower than ARIO, and ARIO is in turn considerably slower than either incarnation of WEIGHTWATCHER. A difference of between one and two orders of magnitude is observed between ARIO and our iterative solver for cases where the number of preference levels is larger than 3.

Varying Constraint Density In our third and final experiment, we explore the abilities of these solvers to scale with constraint density, which is the ratio of constraints to events (C/E). We hold fixed the parameters $\{C = 30, D_- = -50, D_+ = 100, L = 5, R_- = 0.5, R_+ = 0.9\}$, and vary the number of events E between 3 and 36.

In Figure 4 we present the results. For problems having constraint densities less than or equal to 2.5, both WEIGHTWATCHER solvers typically achieve a performance improvement between one and two orders of magnitude over ARIO; for densities larger than this, the difference in speed becomes less evident. It is also near this point that the branch-and-bound version of WEIGHTWATCHER begins to overtake the iterative version. This is likely because the costs of the optimal solutions for these extremely constrained problems are quite large, and thus several iterations are required before a solution is discovered.

In summary, WEIGHTWATCHER consistently outperforms the previous DTPP solvers, including the SAT-based ARIO system, on several dimensions. This suggests that, despite recent successes of SAT in solving decision variants of temporal reasoning (Armando et al. 2004; Nieuwenhuis & Oliveras 2005), there remains considerable room for improvement in the application of these techniques to optimization. Of course, by no means does this imply that the SAT approach is inherently flawed; however, our results clearly demonstrate that well-established CSP-based methods should not be ignored or discounted when constructing algorithms for optimization of temporal preferences.

Conclusion and Future Work

In this paper, we have presented a new efficient algorithm for obtaining utilitarian optimal solutions to Disjunctive Temporal Problems with Preferences (DTPPs). To facilitate our approach, we have introduced the Valued DTP, and have shown that it can express the same set of utilitarian optimal solutions as the DTPP with piecewise-constant preference functions. Using this relationship, we have developed a method for achieving *weighted constraint satisfaction* within a meta-CSP search space that has traditionally been used to solve DTPs without preferences, allowing us to leverage several techniques employed in previous decision-based DTP literature. Experimental results demonstrate that our solver consistently outperforms the state-of-the-art SAT-based approach by orders of magnitude.

This line of research opens the door to several promising avenues of continued progress. For instance, while our focus has been on generating utilitarian optimal solutions, we believe our algorithm can be generalized to handle other objective functions as well. Furthermore, one may wish to combine this approach with our recent addition of conditional bounds and finite-domain constraints to DTPs (Moffitt, Peintner, & Pollack 2005), allowing the encoding of non-temporal constraints that the SAT-based formulation can express quite easily.

Acknowledgements

The authors thank the anonymous reviewers for their many helpful comments. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010 and the Air Force Office of Scientific Research under Contract No. FA9550-04-1-0043. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, the Department of Interior-National Business Center, or the United States Air Force.

References

- Armando, A.; Castellini, C.; Giunchiglia, E.; and Maratea, M. 2004. A SAT-based decision procedure for the boolean combination of difference constraints. In *Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing (SAT-2004)*, 16–29.
- Armando, A.; Castellini, C.; and Giunchiglia, E. 1999. SAT-based procedures for temporal reasoning. In *Proceedings of the 5th European Conference on Planning (ECP-1999)*, 97–108.
- Barrett, C.; de Moura, L.; and Stump, A. 2005. SMT-COMP: Satisfiability Modulo Theories Competition. In *Proceedings of the 17th International Conference on Computer Aided Verification (CAV-2005)*, 20–23.
- Bistarelli, S.; Montanari, U.; Rossi, F.; Schiex, T.; Verfaillie, G.; and Fargier, H. 1999. Semiring-based CSPs and valued CSPs: Frameworks, properties, and comparison. *Constraints* 4(3):199–240.
- Bistarelli, S.; Montanari, U.; and Rossi, F. 1997. Semiring-based constraint satisfaction and optimization. *Journal of the ACM* 44(2):201–236.
- Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artificial Intelligence* 49(1-3):61–95.
- Freuder, E. C., and Wallace, R. J. 1992. Partial constraint satisfaction. *Artificial Intelligence* 58(1-3):21–70.
- Khatib, L.; Morris, P.; Morris, R.; and Rossi, F. 2001. Temporal constraint reasoning with preferences. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-2001)*, 322–327.
- Khatib, L.; Morris, P.; Morris, R.; and Venable, K. B. 2003. Tractable Pareto optimal optimization of temporal preferences. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003)*, 1289–1294.
- Kumar, T. K. S. 2004. A polynomial-time algorithm for simple temporal problems with piecewise constant domain preference functions. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-2004)*, 67–72.
- Moffitt, M. D., and Pollack, M. E. 2005. Partial constraint satisfaction of disjunctive temporal problems. In *Proceedings of the 18th International Florida Artificial Intelligence Research Society Conference (FLAIRS-2005)*, 715–720.
- Moffitt, M. D.; Peintner, B.; and Pollack, M. E. 2005. Augmenting disjunctive temporal problems with finite-domain constraints. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI-2005)*, 1187–1192.
- Morris, P.; Morris, R.; Khatib, L.; Ramakrishnan, S.; and Bachmann, A. 2004. Strategies for global optimization of temporal preferences. In *Proceedings of the 10th International Conference on Principles and Practice of Constraint Programming (CP-2004)*, 408–422.
- Moskewicz, M. W.; Madigan, C. F.; Zhao, Y.; Zhang, L.; and Malik, S. 2001. Chaff: Engineering an Efficient SAT Solver. In *Proceedings of the 38th Design Automation Conference (DAC'01)*, 530–535.
- Nieuwenhuis, R., and Oliveras, A. 2005. DPLL(T) with exhaustive theory propagation and its application to difference logic. In *Proceedings of the 17th International Conference on Computer Aided Verification (CAV-2005)*, 321–334.
- Oddi, A., and Cesta, A. 2000. Incremental forward checking for the disjunctive temporal problem. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-2000)*, 108–112.
- Peintner, B., and Pollack, M. E. 2004. Low-cost addition of preferences to DTPs and TCSPs. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-2004)*, 723–728.
- Peintner, B., and Pollack, M. E. 2005. Anytime, complete algorithm for finding utilitarian optimal solutions to STPPs. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI-2005)*, 443–448.
- Peintner, B. M. 2005. *Algorithms For Constraint-Based Temporal Reasoning With Preferences*. Ph.D. Dissertation, University of Michigan.
- Provost, F. J. 1993. Iterative weakening: Optimal and near-optimal policies for the selection of search bias. In *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI-1993)*, 749–755.
- Schiex, T.; Fargier, H.; and Verfaillie, G. 1995. Valued constraint satisfaction problems: Hard and easy problems. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-1995)*, 631–639.
- Sheini, H. M.; Peintner, B.; Sakallah, K. A.; and Pollack, M. E. 2005. On solving soft temporal constraints using SAT techniques. In *Proceedings of the 11th International Conference on Principles and Practice of Constraint Programming (CP-2005)*, 607–621.
- Stergiou, K., and Koubarakis, M. 1998. Backtracking algorithms for disjunctions of temporal constraints. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, 248–253.
- Tsamardinos, I., and Pollack, M. E. 2003. Efficient solution techniques for disjunctive temporal reasoning problems. *Artificial Intelligence* 151(1-2):43–90.