Artificial Intelligence 175 (2011) 1390–1409

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

On the modelling and optimization of preferences in constraint-based temporal reasoning ☆

Michael D. Moffitt

11400 Burnet Rd., Austin, TX, United States

ARTICLE INFO

Article history:
Received 28 February 2009
Received in revised form 12 August 2010
Accepted 15 September 2010
Available online 8 December 2010

Keywords:
Preferences
Overconstrained problems
Constraint satisfaction
Optimization
Branch and bound
Temporal reasoning

ABSTRACT

In this paper, we consider both the modelling and optimization of preferences in problems of constraint-based temporal reasoning. The Disjunctive Temporal Problems with Preferences (DTPP) – a formulation that combines the rich expressive power of the Disjunctive Temporal Problem with the introduction of metric preference functions – is studied, and transformed into a corresponding constraint system that we name the Valued DTP (VDTP). We show that for a broad family of optimization criteria, the VDTP can express the same solution space as the DTPP, under the assumption of arbitrary piecewise-constant preference functions. We then generalize the powerful search strategies from decision-based DTP literature to accomplish the efficient optimization of temporal preferences. In contrast to the previous state-of-the-art system (which addresses the optimization of temporal preferences using a SAT formulation), we instead employ a meta-CSP search space that has traditionally been used to solve DTPs without preferences. Our approach supports a variety of objective functions (such as utilitarian optimality or maximin optimality) and can accommodate any compliant valuation structure. We also demonstrate that key pruning techniques commonly used for temporal satisfiability (particularly, the removal of subsumed variables and semantic branching) are naturally suited to prevent the exploration of redundant search nodes during optimization that may otherwise be encountered when resolving a typical VDTP derived from a DTPP. Finally, we present empirical results showing that an implementation of our approach consistently outperforms prior algorithms by orders of magnitude.

1. Introduction

The need to accommodate preferences has increasingly become an important problem in many fields related to artificial intelligence. While the topic spans several subjects – including decision theory, planning and scheduling, and machine learning – the area of *constraint satisfaction* [9,29] affords many of the greatest opportunities for both the representation and reasoning of preferences [6,28]. The application of preferences to constraint-based systems presents at least three principal challenges. The first of these challenges is determining a means to achieve preference *elicitation*, either through explicit mechanisms or by indirect inference through a series of observations and interactions with a user [34,35]. Secondly, one must address the adequate *modelling* of local preference values and their global aggregation; the burden of translating the known preference values of the real world to a specific standalone representation is seldom straightforward, and, in some cases, impossible. The third challenge deals with the necessary adaptation of classical search strategies to transform the goal of *satisfaction* into one of *optimization*, requiring the generalization of highly specialized techniques to navigate a richer

☆ This paper includes and extends preliminary work from Moffitt and Pollack (2005, 2006) [18,19].

E-mail address: mdmoffitt@us.ibm.com.

0004-3702/$ – see front matter
doi:10.1016/j.artint.2010.11.016

search space [16]. Whether optimization can be done practically depends largely on the choice of model and the choice of search strategy, resulting in a well-known tradeoff between expressive power and efficient reasoning.

To be sure, existing literature on classical finite-domain CSPs is rich with techniques for encoding and resolving preference criteria. Among the various formalisms proposed, two popular representations are dominant: The Valued CSP [30] and the Semiring CSP [7]. In the Valued CSP, constraints are annotated with scalar valuations that reflect the cost of their violation. In the Semiring CSP, the various relational tuples that may satisfy a constraint are themselves labeled with preferences. Aside from the unique ability of the semiring to encode a partial ordering over the solution space, the two representations are comparable, and can be readily converted to one another [8]. Both approaches unify the expressive power of the classical CSP and the efficient algorithms for their satisfaction under the larger scope of preference optimization.

Closely related to the finite-domain CSP, the temporal CSP is a common representation for many planning and scheduling problems in which the relationships between events (in time) are expressed by constraining their pairwise difference. Preferences have been proposed in this framework as well, with the most common variant augmenting traditional temporal constraints [10] with local preference functions that express how well a particular assignment satisfies the corresponding constraint [13]. 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. While the literature-to-date has introduced representations with generously expressive preference functions, they have often been applied to relatively *inexpressive* underlying constraint systems, limiting the reasoning of complex preferences to only a relatively small class of problems [14,15,20,24]. As preference-based temporal models continue to become more and more commonplace in industrial constraint engines [5,4,25], modern optimization tools must be able to respond to a wider range of problem instances.

The Disjunctive Temporal Problems with Preferences (DTPPs) [23], a powerful representation that subsumes many common temporal formalisms, attempts to provide the best of both worlds: a rich model for complex disjunctive constraints (to handle, for instance, non-overlap conditions that commonly arise in planning and scheduling) in addition to a rich language for expressing preference profiles over their domains. Early work in this problem space focused on maximizing the minimum of such preference values, while later developments have begun to address the more challenging problem of utilitarian optimization [26], where the sum of the individual preference values is maximized. In either case, the problem of optimization is divided from the core satisfiability engine (i.e., of the wealth of techniques used to find *feasible* solutions, relatively few are extended toward finding *good* solutions). This division is due, in part, to a fundamental dichotomy between the preference model and the search space of temporal problems: preference values are attributed to grounded differences between temporal events, whereas the meta-CSP algorithms for temporal reasoning refrain from instantiating these object-level variables. Recent work has shown that SAT formulations can demonstrate orders of magnitude of improvement as compared to these techniques [31], suggesting that advances in SAT technology may be the key to rapid search. Hence, prior art has yet to reveal a unified approach for both the expressive modelling *and* efficient optimization of preferences within a classic CSP framework.

In this work, we consider an alternative to the DTPP – the Valued DTP (or VDTP). We show that for a broad family of optimization criteria, the VDTP can express the same solution space as the DTPP, under the assumption of arbitrary piecewise-constant preference functions. We furthermore argue that the valued constraint representation eliminates the dichotomy between the object-level preference model and the meta-CSP solution space, and thus offers unique advantages when used to guide the search strategies employed in temporal constraint satisfaction algorithms. We then generalize decision-based DTP literature to accomplish the efficient optimization of temporal preferences. Our approach supports a variety of objective functions (such as utilitarian optimality or maximin optimality) and can accommodate any compliant valuation structure. We also demonstrate that key pruning techniques commonly used for temporal satisfiability (particularly, the removal of subsumed variables and semantic branching) are naturally suited to prevent the exploration of redundant search nodes during optimization that may otherwise be encountered when resolving a typical VDTP derived from a DTPP. Finally, we present empirical results showing that an implementation of our approach consistently outperforms prior algorithms by orders of magnitude, including the SAT-based approach.

The remainder of the paper is organized as follows. Section 2 covers background material related to classical CSPs and their corresponding preference models. Sections 3 and 4 cover constraint-based temporal reasoning and extensions to temporal preferences, respectively. In Sections 5 and 6, we present the Valued DTP, and establish its relationship with the DTP with Preferences. In Section 7, we demonstrate how to adopt the preference model of the VDTP to construct a meta-CSP search framework for optimization. In Section 8, we provide an empirical evaluation of approaches. Finally, we conclude in Section 9 with a summary of our approach along with future works.

2. Preference optimization in finite-domain constraint networks

We begin by briefly reviewing the formulation of classical CSPs, followed by a description of two preference models that have been used to augment the original framework.

2.1. Finite-domain constraint networks

A *Constraint Satisfaction Problem* (or *constraint network*) [9] is defined by a triple $\langle X, D, C\rangle$, where $X = \{x_1, \ldots, x_n\}$ is a set of variables, $D = \{D_1, \ldots, D_n\}$ contains a domain $D_i = \{v_1, \ldots, v_k\}$ for each variable that lists the possible values it

may take, and $C = \{C_1, \ldots, C_t\}$ is a set of constraints, where each constraint C_i is a relation R_i defined over a subset of variables $S_i \subseteq X$. A solution to a constraint network is an assignment $\bar{a} = (a_1, \ldots, a_n)$ such that each $a_i \in D_i$, and for each constraint C_i, the projected assignment $\bar{a}[S_i] \in R_i$.

A solution to a CSP is typically found by way of a recursive, depth-first, backtracking search, in which values are chosen for variables one at a time until an inconsistency is detected (i.e., if there are no legal values remaining in the domain of some variable). If a search node cannot be expanded, backtracking occurs and a different value for the most recently instantiated variable is chosen. The search process continues in this way until a solution has been found or the space of assignments has been exhausted. This computationally expensive procedure is improved with the use of *heuristics* and *inference*. A heuristic uses information about the current state in search to determine an effective variable and value ordering. In contrast, inference attempts to rule out possible values for the uninstantiated variables, given only those decisions that have been made in search thus far.

2.2. Semiring CSPs

A constraint satisfaction problem can be associated with a semiring structure that specifies preference values for each tuple of values of the variables' domains. Depending on the choice of aggregation operation, different global objective functions may be constructed based on the local semiring values.

A *semiring* is a tuple $(A, +, \times, 0, 1)$ such that:

- A is a set and $0, 1 \in A$;
- $+$, the additive operation, is a closed (i.e., $a, b \in A$ implies that $a+b \in A$), commutative (i.e., $a+b = b+a$) and associative (i.e., $a + (b + c) = (a + b) + c$) operation such that $a + 0 = 0 = 0 + a$ (i.e., 0 is its unit element);
- $\times$, the multiplicative operation, is a closed and associative operation such that 1 is its unit element and $a \times 0 = 0 = 0 \times a$ (i.e., 0 is its absorbing element);
- $\times$ distributes over $+$ (i.e., $a \times (b + c) = (a \times b) + (a \times c)$).

One of the principal advantages of using a semiring to model preferences in a constraint system is that a large family of optimization criteria can be expressed as instantiations of the same broad problem definition. For instance, a classical CSP (i.e., one where the only notion of preference is the distinction between *feasible* and *infeasible*) is an SCSP with the c-semiring $S_{CSP} = \langle \{0, 1\}, \vee, \wedge, 0, 1 \rangle$. Weighted constraint satisfaction (where the sum of preferences values is maximized) is expressed with the semiring $S_{WCSP} = \langle \Re^{\top}, \max, +, -\infty, 0 \rangle$.

2.3. Valued CSPs

A *valuation structure* is a tuple $\langle E, \circledast, \succ \rangle$ such that:

- E is a set (whose elements are called of valuations) and is totally ordered by $\succ$, with a maximum element $\top$ and a minimum element $\bot$;
- $\circledast$ is a commutative, associative closed binary operation on E that satisfies:
  - *Identity*: $\forall a \in E, a \circledast \bot = a$;
  - *Monotonicity*: $\forall a, b, c \in E, (a \succcurlyeq b) \Rightarrow ((a \circledast c) \succcurlyeq (b \circledast c))$;
  - *Absorbing element*: $\forall a \in E, (a \circledast \top) = \top$.

A *Valued CSP* is defined by a classical CSP $\langle V, D, C \rangle$, a valuation structure $S = (E, \circledast, \succ)$, and an application φ from C to E, where $\varphi(c)$ maps any constraint of a classical CSP to a valuation denoting the impact of its violation.

The *valuation* of an assignment A of the variables $W \subset V$ is defined by:

$$V_P(A) = \circledast_{c \in C, V_c \subset W, violates(A,c)} \left[\varphi(c)\right]$$

The VCSP effectively establishes a total ordering over the set of complete assignments, giving highest preference to assignments that achieve the *minimum* valuation.

2.4. Comparison

The SCSP and the VCSP cast the modelling of preferences in fairly different lights. In one aspect, there is a distinction between whether preference values are attributed to the tuples of a constraint, or instead to the constraint themselves. Beyond that, the two formulations also employ different mathematical mechanisms to achieve the aggregation of local preferences. It has been shown that under the assumption of a total order, the two formulations admit an equivalent solution space [8], allowing one to freely pass between the two without loss of expressive power. However, the methods used to achieve pruning, inference, and bounding are dependent on the representation.

3. Temporal Constraint Satisfaction Problems

In contrast to classical CSPs (whose variables span a finite number of unordered values), temporal CSPs define variables whose domains span the set of reals (or integers), representing time. Hence, they model flavors of scheduling problems in which the relationships between events (in time) are expressed by constraining their pairwise difference.

3.1. Simple Temporal Problems

The *Simple Temporal Problem* (STP) is the most restricted form of quantitative TCSP. It is defined by a pair $\langle X, C\rangle$, where X is a set of time points having continuous domains, and C is a set of constraints of the form:

$$a_{ij} \leqslant X_j - X_i \leqslant b_{ij}$$

This dual-bounded inequality can be converted into a single-bounded form simply by constructing a pair of linear inequalities for each constraint. An STP has an equivalent graph-based network, where each node in the graph G corresponds to a time point in the STP, and each directed edge between nodes i and j corresponds to a linear inequality.

3.2. Binary TCSPs

The *Binary Temporal Constraint Satisfaction Problem* (Binary TCSP) [10] admits a greater expressive power than the STP. It is defined by a pair $\langle X, C\rangle$, where X is a set of time points having continuous domains, and C is a set of disjunctive constraints of the form:

$$a_{ij1} \leqslant X_j - X_i \leqslant b_{ij1} \vee \cdots \vee a_{ijn} \leqslant X_j - X_i \leqslant b_{ijn}$$

As with the STP, this dual-bounded representation can be converted into a single-bounded representation, although the size of the encoding grows exponentially with the number of *disjuncts* in each constraint.

A *solution* to a Binary TCSP may be viewed in one of two ways. The first is as an assignment of a numeric value to each of the time points in X that satisfies all the constraints in C. Another type of solution – called a *meta-CSP* solution – considers the creation of a meta-variable C_i for every constraint C_i, as described formally in the following section.

3.3. Disjunctive Temporal Problems

A *Disjunctive Temporal Problem* (DTP) [32] 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: $c_{i1} \vee c_{i2} \vee \cdots \vee c_{in_i}$ where in turn, each c_{ij} is of the form: $a_{ij} \leqslant x_{ij} - y_{ij} \leqslant b_{ij}$ with $x_{ij}, y_{ij} \in X$ and $a_{ij}, b_{ij} \in \Re$ (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, and Binary Temporal Constraint Satisfaction Problems (Binary TCSPs), where the same pair of time points must participate in every disjunct belonging to a single constraint.

To illustrate the Disjunctive Temporal Problem, consider the following scenario. Meetings A and B are both to last 40 minutes, and cannot overlap. Furthermore, meeting A must begin between 11:00 and 11:30, and meeting B must finish between 11:30 and 12:00. This problem can be cast as a DTP with constraints shown in Fig. 1(a). The time points $[A_S, A_E]$ represent the start and end of meeting A, and likewise, the time points $[B_S, B_E]$ represent the start and end of meeting B. A temporal reference point T_R represents an arbitrary fixed time (such as midnight), and is used to express constraints with respect to wall clock time.

As with the Binary TCSP, 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 $\{c_{i1}, c_{i2}, \ldots, c_{in_i}\}$, 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*. One meta-CSP solution may correspond to many feasible object-level solutions, and vice versa.

Within this 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 $x - y \leqslant b$ 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 [10].

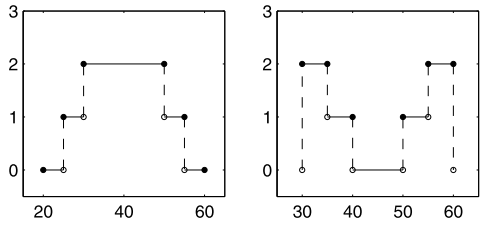
In the meeting example, the assignment:

$$(C_1, C_2, C_3, C_4, C_5) \leftarrow (c_{11}, c_{21}, c_{31}, c_{41}, c_{51})$$

is the only consistent meta-CSP solution. One cannot choose c_{32} as the value for C_3 (requiring meeting A to precede meeting B) since this would force meeting B to finish late even if meeting A begins as early as possible.

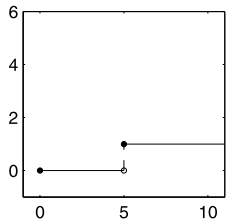
C_1:	$\{c_{11} : 40 \leq A_E - A_S \leq 40\}$
C_2:	$\{c_{21} : 40 \leq B_E - B_S \leq 40\}$
C_3:	$\{c_{31} : 0 \leq A_S - B_E\} \vee \{c_{32} : 0 \leq B_S - A_E\}$
C_4:	$\{c_{41} : 660 \leq A_S - T_R \leq 690\}$
C_5:	$\{c_{51} : 690 \leq B_E - T_R \leq 720\}$

(a)

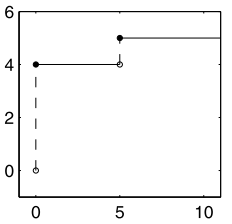


(b) $A'_E - A'_S$

(c) $B'_E - B'_S$



(d) $A'_S - B'_E$



(e) $B'_S - A'_E$

Fig. 1. (a) The constraints in the meeting example. (b)–(e) Preference functions over disjuncts c'_{11}, c'_{21}, c'_{31}, and c'_{32}.

4. Temporal preferences: formalisms and algorithms

The earliest adaptation of preferences in temporal reasoning problems [13] constructed a formalism known as the *Simple Temporal Problem with Preferences*. While this structure overcame the inability of the Simple Temporal Problem to encode optimization variants, it remained fundamentally limited in the same way that the STP could not encompass the broad range of problems allowed by the DTP.

In this section, we focus specifically on preferences in the case of disjunctive constraints. It will be shown in future sections that the ability to reason about disjunctions can be useful even when all constraints are simple in nature.

4.1. Disjunctive Temporal Problems with Preferences

A DTP can be extended to a DTP with Preferences (DTPP) [23] by augmenting each disjunct c_{ij}: $a_{ij} \leqslant x_{ij} - y_{ij} \leqslant b_{ij}$ with a preference function $\langle f_{ij} : t \in [a_{ij}, b_{ij}] \rightarrow \{0, \Re^+\}\rangle$, mapping every feasible temporal difference to a *preference value* expressing its relative utility [13].[1] 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:

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

Example. Consider a variation on the earlier meeting example, in which the duration of meeting A is no longer fixed, and can instead have length between 20 and 60 minutes, with values closer to 40 being more preferable. A preference function reflecting this over the new variables A'_E and A'_S is shown in Fig. 1(b). Fig. 1(c) expresses that meeting B can last between 30 and 60 minutes, with durations closer to either of these extremes being preferred over values in between. Also, imagine that the content of meeting A covers much of the material that meeting B builds upon, and so it is highly desired that meeting A precede meeting B. Figs. 1(d) and 1(e) reflect this strong preference, along with a weaker preference that 5 minutes be allocated between meetings for a short break.[2]

[1] Variations on this model – such as replacing feasibility intervals with universally defined preference functions – have been suggested [17] and can be extended to subsequent formulations without issue.

[2] For the sake of simplicity, we leave C_4 and C_5 as hard constraints with preference functions that map to the constant 2 within their feasible regions.

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 a scheme for aggregating preference values across constraints. One of the earliest objectives to be considered is *maximin* or *Weakest Link Optimality* (WLO) [13], in which the global value of an assignment S is equal to the minimal preference value satisfied by the assignment:

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

WLO compares solutions myopically using only the "weakest link" in each solution; no credit is given for satisfying other constraints at very high levels. Despite this drawback, WLO can be appropriate in some situations [14].

An alternative optimality criterion is the useful *utilitarian* objective, where the global value of a solution S is equal to the sum of the preference values of the all constraints [20]:

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

Utilitarianism corresponds to optimality with the weighted semiring, and is considerably more sensitive than WLO to the levels at which individual constraints are satisfied.

Example. The solution $S : (A'_S, A'_E, B'_S, B'_E) \leftarrow (690, 730, 650, 690)$ is feasible, as it falls within the feasible region of each temporal constraint. For the case of utilitarian optimality, it results in a global preference value of 7:

$$f_1(S) + f_2(S) + \cdots + f_5(S) = 2 + 1 + 0 + 2 + 2 = 7$$

where f_i corresponds to the highest preference value achieved within constraint C_i for a solution S. In contrast, WLO optimality would give S the score of 0, as this is the lowest level at which any constraint is satisfied. We can obtain a better solution S' with the assignment $(A'_S, A'_E, B'_S, B'_E) \leftarrow (660, 685, 690, 720)$, which gives us a utilitarian preference value of 12 and a WLO score of 1:

$$f_1(S') + f_2(S') + \cdots + f_5(S') = 1 + 2 + 5 + 2 + 2 = 12$$

By inspection, we observe that this is a utilitarian optimal solution, meaning that there exists no other object-level assignment that can achieve a higher objective value.

Other forms of preference value aggregation (corresponding to different instantiations of the c-semiring) are possible, though have yet to be considered explicitly in the literature on disjunctive temporal reasoning.

4.2. Solving DTPPs

4.2.1. Weakest link optimality

When introduced, the DTPP was accompanied with an algorithm to compute WLO-optimal solutions with an approach not entirely dissimilar to a traditional DTP search. It began by treating the DTPP as if all preference values were fixed (or *projected*) at their the lowest levels of 0. Using the meta-CSP approach common to many DTP solvers, it would search this space looking for a satisfying solution. Upon the discovery of a consistent solution (which, by construction, would have a WLO score of 0), all constraints in the DTP would be tightened to match the next highest preference level of the DTPP (perhaps level 1). Search would then resume, with each subsequent solution triggering another tightening of the DTP's constraints. This process continues until the highest level has been reached, or search has been exhausted.

The overall approach is described as "low-cost", for the reason that the space explored was guaranteed to be no larger than that of $|A|$ copies of the largest projected DTP, where A is the set of all preference levels.

4.2.2. Utilitarian optimality

Since the development of the WLO algorithm, two approaches have been proposed for performing utilitarian optimization of a DTPP. Both can handle problems containing complex preference functions, requiring only that they be piecewise-constant in shape.[3]

The first is based on a SAT reformulation of a DTPP [31]. 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 $ax - by \leqslant d$, 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 a Satisfiability Modulo Theories (SMT) system named **ARIO**, 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

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

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 [21] 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 [26]. It is based largely on the GAPS algorithm [24] for computing 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 decoupled from the process of optimization. Although both **GAPS** and **GAPD** have been shown to be several orders of magnitude slower than **ARIO** for finding optimal solutions, these algorithms have been shown to exhibit desirable anytime properties.

5. Valued disjunctive temporal problems

The DTP with Preferences has typically been regarded as a type of *semiring* formulation [7], in which preference values are attributed to the (infinitely many) legal object-level tuples that comprise a constraint. While expressive, this model of preferences makes the task of optimization somewhat nebulous. Recall that search strategies for disjunctive temporal reasoning operate on the meta-CSP rather than invoking object-level assignments directly; therefore, the set of object-level solutions to a single meta-CSP solution may vary wildly in quality, and no one preference value can be determined for a component STPP. In response, we 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 [30].

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

For instance, consider the *weighted* case of the VDTP, where $E = \Re^+ \cup \{\infty\}$ and $\circledast = +$ (i.e., arithmetic sum), using the usual ordering $<$. In other words, each constraint C_i is associated with a positive numeric weight $\varphi(C_i)$, and the objective is 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:

$$cost_D(S) = \sum_i \{\varphi(C_i) \mid violates(S, C_i)\}$$

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

C_1: $\{c_{11}\colon 1 \leqslant x - y \leqslant 2\}$	$\varphi(C_1) = 1$
C_2: $\{c_{21}\colon 3 \leqslant x - y \leqslant 4\} \vee \{c_{22}\colon 5 \leqslant x - z \leqslant 6\}$	$\varphi(C_2) = 2$
C_3: $\{c_{31}\colon 1 \leqslant y - z \leqslant 2\}$	$\varphi(C_3) = 4$
C_4: $\{c_{41}\colon 0 \leqslant x - z \leqslant 7\}$	$\varphi(C_4) = \infty$

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.

6. The relationship between DTPPs and VDTPs

The previous example illustrates an important distinction between DTPPs and VDTPs. While constraints in both formalisms serve to bound the pairwise temporal difference between events, the preferences apply to orthogonal elements of the encoding. In the DTPP, a preference value is ascribed to a specific value of an object-level assignment; hence, an assignment to the meta-CSP (i.e., the selection of one disjunct per constraint) is alone insufficient to determine the value of any object-level solution. In contrast, each VDTP constraint has a single valuation that expresses the cost of its violation (with infinite valuation reflecting a truly hard constraint). Thus, a consistent partial assignment to the meta-CSP (in which some meta-level variables are uninstantiated) has known cost.

Despite the difference in preference model, we can show that when assuming a total ordering over solutions, 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' ($S_1 \succ_{D'} S_2$), iff S_1 is also at least as preferred as S_2 in D ($S_1 \succ_D S_2$). In

fact, we show that in addition, given any VDTP D, there is an equivalent DTPP D', and hence the VDTP and DTPP formalisms share the same expressive power. This relationship requires only that the preference functions be *piecewise-constant*, i.e., that each function $f(x)$ may be expressed as

$$\sum_{i=0}^{n} \alpha_i \chi_i(x)$$

where $\chi_i(x)$ returns 1 if x lies within the range of interval i, and 0 otherwise. The piecewise-constant assumption is made in virtually all prior work on DTPPs.

6.1. 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 $\{C_1, \ldots, C_n\}$ where $C_i = c_{i1} \vee \cdots \vee c_{in_i}$ and $c_{ij} = a_{ij} \leqslant x_{ij} - y_{ij} \leqslant b_{ij}$ and valuation structure $S = \langle E, \circledast, \succ \rangle$ with $\varphi(C_i)$ being the valuation of C_i. Create the *derived DTPP* D' with c-semiring $\langle E, +, \circledast, \top, \bot \rangle$ by constructing a *derived constraint* C'_i from each constraint C_i in D, where $C'_i = c'_{i1} \vee \cdots \vee c'_{in_i}$ and $c'_{ij} = -\infty \leqslant x_{ij} - y_{ij} \leqslant \infty$ with preference function $\langle f_{ij} : t \in [a_{ij}, b_{ij}] \to \bot, t \notin [a_{ij}, b_{ij}] \to \varphi(C_i) \rangle$.

In what follows, we distinguish the *constraint component* (i.e., the disjunctive constraints themselves) from the valuations or the preference functions. It will also be useful to refer to the *valuation* of a VDTP solution S, which is the aggregation of valuations of constraints violated by S:

$$V(S) = \circledast_{C_i \in C, violates(S, C_i)} \big[\varphi(C_i)\big]$$

We now show that every VDTP has an equivalent DTPP.

Lemma 6.1. *An assignment S is a solution to D if and only if it is a solution to D'.*

Let $S : X \to \Re$ 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.

Lemma 6.2. *If S_1 and S_2 are solutions to D (and D'), then $S_1 \succ_D S_2$ iff $S_1 \succ_{D'} S_2$.*

As per [8], a total ordering over solutions is preserved when moving from the valuation structure to the c-semiring.

Theorem 6.1. *VDTP D and its derived DTPP D' are equivalent* (*following directly from Lemmas* 6.1 *and* 6.2).

6.2. 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* [24]. 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 $C_{ij} = \langle a_{ij} \leqslant x_{ij} - y_{ij} \leqslant b_{ij}, f_{ij} \rangle$, the **preference projection at level** l for C_{ij} is $\mathcal{P}_{ij}[l] = \{c_1, c_2, \ldots, c_n\}$, where $c_k = \langle a_k \leqslant x_{ij} - y_{ij} \leqslant b_k \rangle$, $b_k < a_{k+1}$ for $1 \leqslant k < n$ and $\bigcup_{k=1}^{n} [a_k, b_k] = \{t | f_{ij}(t) \geqslant l\}$.

Definition *(DTPP preference projection).* Given a DTPP constraint $C_i = c_{i1} \vee c_{i2} \vee \cdots \vee c_{in}$, 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. Valuations will be assigned in such a way that satisfying all projected constraints through level k will result in an *aggregate* value of k. The procedure is as follows:

Let D' be a DTPP with constraints $\{C'_1, \ldots, C'_n\}$. 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 $\varphi(C_{i,0}) = \bot$. Set l to zero.
- Find the smallest $l' > l$ such that $\mathcal{P}_i[l'] \neq \mathcal{P}_i[l]$.

Disjunctive constraints		Valuations (weighted case)	Valuations (maximin case)
$C_{1,0}$:	$20 \leqslant A_E - A_S \leqslant 60$	$\varphi(C_{1,0}) = \infty$	$\varphi(C_{1,0}) = 0$
$C_{1,1}$:	$25 \leqslant A_E - A_S \leqslant 55$	$\varphi(C_{1,1}) = 1$	$\varphi(C_{1,1}) = 1$
$C_{1,2}$:	$30 \leqslant A_E - A_S \leqslant 50$	$\varphi(C_{1,2}) = 1$	$\varphi(C_{1,2}) = 2$
$C_{2,0}$:	$30 \leqslant B_E - B_S \leqslant 60$	$\varphi(C_{2,0}) = \infty$	$\varphi(C_{2,0}) = 0$
$C_{2,1}$:	$30 \leqslant B_E - B_S \leqslant 40 \vee 50 \leqslant B_E - B_S \leqslant 60$	$\varphi(C_{2,1}) = 1$	$\varphi(C_{2,1}) = 1$
$C_{2,2}$:	$30 \leqslant B_E - B_S \leqslant 35 \vee 55 \leqslant B_E - B_S \leqslant 60$	$\varphi(C_{2,2}) = 1$	$\varphi(C_{2,2}) = 2$
$C_{3,0}$:	$0 \leqslant A_S - B_E \vee 0 \leqslant B_S - A_E$	$\varphi(C_{3,0}) = \infty$	$\varphi(C_{3,0}) = 0$
$C_{3,1}$:	$5 \leqslant A_S - B_E \vee 0 \leqslant B_S - A_E$	$\varphi(C_{3,1}) = 1$	$\varphi(C_{3,1}) = 1$
$C_{3,4}$:	$0 \leqslant B_S - A_E$	$\varphi(C_{3,4}) = 3$	$\varphi(C_{3,4}) = 4$
$C_{3,5}$:	$5 \leqslant B_S - A_E$	$\varphi(C_{3,5}) = 1$	$\varphi(C_{3,5}) = 5$
$C_{4,0}$:	$660 \leqslant A_S - T_R \leqslant 690$	$\varphi(C_{4,0}) = \infty$	$\varphi(C_{4,0}) = 0$
$C_{4,2}$:	$660 \leqslant A_S - T_R \leqslant 690$	$\varphi(C_{4,2}) = 2$	$\varphi(C_{4,2}) = 2$
$C_{5,0}$:	$690 \leqslant B_E - T_R \leqslant 720$	$\varphi(C_{5,0}) = \infty$	$\varphi(C_{5,0}) = 0$
$C_{5,2}$:	$690 \leqslant B_E - T_R \leqslant 720$	$\varphi(C_{5,2}) = 2$	$\varphi(C_{5,2}) = 2$

Fig. 2. The weighted VDTP and maximin VDTP corresponding to the meeting example with preferences.

- Create a soft constraint $C_{\langle i,l' \rangle}$ in D, where $C_{\langle i,l' \rangle} = \bigvee \mathcal{P}_i[l']$ such that $\varphi(C_{i,l'}) \circledast \varphi(C_{i,l}) = l'$. Set l to l'.
- Iterate until an l' is reached such that $\mathcal{P}_i[l'] = \varnothing$.

Just as in the conversion of an SCSP to a VCSP [8], the number of constraints typically increases when passing from a DTPP to a VDTP.

Example. As illustration of this procedure, Fig. 2 shows the VDTP corresponding to our DTPP example in Fig. 1. Many of the weighted valuations (shown in the second-to-last column) have unit cost; this is because our preference functions typically change by increments of 1 unit (one exception to this is $C_{3,4}$, which has a valuation of 3; this is due to the strong preference that meeting A precede meeting B). Also, due to the non-convexity of f_2, the preference projections of the simple temporal constraint C_2 have been projected into a pair of disjunctive constraints $C_{2,1}$ and $C_{2,2}$. One can verify that the optimal solution $(A_S, A_E, B_S, B_E) \leftarrow (660, 685, 690, 720)$ identified earlier satisfies all constraints except $C_{1,2}$, and thus has a cost of 1.

Lemma 6.3. *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 6.1, 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'.

Lemma 6.4. *If S_1 and S_2 are solutions to D' (and D), then $S_1 \succ_D S_2$ iff $S_1 \succ_{D'} S_2$.*

Again, a total ordering over solutions is preserved when moving from the c-semiring to the valuation structure, as shown in [8].

Theorem 6.2. *DTPP D' and its derived VDTP D are equivalent* (*following directly from Lemmas* 6.3 *and* 6.4).

Theorem 6.3. *For the assumption of a total order, DTPPs and VDTPs are equivalent in expressive power.*

This follows directly from Theorems 6.1 and 6.2.

For a DTPP with $|C|$ constraints and $|A|$ distinct preference levels (where A is the set of all such levels), the number of constraints in the corresponding VDTP is bounded from above by $|C| \times |A|$. Of course, due to the meta-CSP encoding, these additional constraints in fact become meta-level variables, and thus contribute to additional branches in the corresponding search tree.

The VDTP easily captures the representation of various optimality criteria that arise from different instantiations of the valuation structure. For instance, in the final column of Fig. 2, we provide a different set of valuations – here, each constraint is assigned its original preference level, as opposed to the difference in between it and the next lowest level. If aggregated with the *min* operator, this alternate VDTP corresponds to the case of Weakest Link Optimality.

Observe that in this construction, several (valued) disjunctive constraints may be generated from a single non-disjunctive STPP constraint in the event that its preference function is not semi-convex. A similar (though not equivalent) observation is made by Peintner [26]. Any non-semi-convex STPP constraint can be converted into a DTPP constraint whose disjuncts exclusively contain semi-convex preference functions. The difference in VDTP conversion is the extent to which such disjunctions must be generated. In particular, disjunctive splits need only occur above the level at which non-convexity begins;

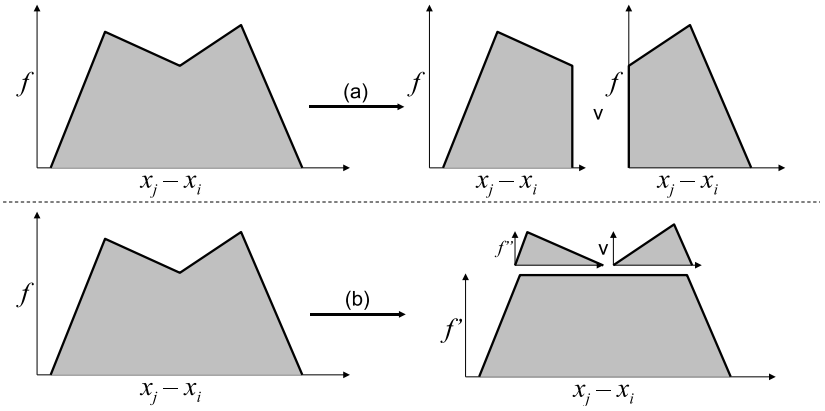


Fig. 3. (a) The semi-convex DTPP conversion suggested by Peintner [26]. (b) The semi-convex VDTP conversion.

Solve-VDTP(A, U, *cost*, *upperbound*)
If ($cost > upperbound$) return
If ($U = \emptyset$)
    best-solution-so-far $\leftarrow A$
    upperbound $\leftarrow$ *cost*
    return
EndIf
$C_i \leftarrow$ *select-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', $cost \circledast \varphi(C_i)$, *upperbound*)

Fig. 4. Solving a VDTP by searching the space of partial component STPs.

any preference projections below this point are maintained as contiguous simple constraints. The distinction is illustrated in Fig. 3. This formulation offers the advantage that the algorithm may freely commit to lower preference levels while deferring the decision of whether to commit to a higher region (and, if so, which of the regions resulting from the disjunction).

7. Solving valued DTPs

We have just shown that any DTPP with piecewise-constant preference functions can be translated into an equivalent VDTP, in which individual constraints are tagged with valuations. To allow the possibility that any one of these constraints may be violated, a partial assignment in the meta-CSP space must be extended to include an *empty* value (which we denote 'ϵ') in the domain of each meta-variable. The ϵ-relaxation, originally proposed to enable nonweighted partial constraint satisfaction [18,19], is required to support the explicit violation of constraints at the meta-level space. A solution that violates a constraint C_i with an assignment of ϵ will incur cost $\varphi(C_i)$.[4]

The augmented partial assignment for the alternative VDTP representation gives rise to a means to search through the space of *partial* component STPs.

7.1. Searching the space of partial component STPs

Pseudocode for the solving the VDTP is given in Fig. 4. 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 aggregate of the valuations of violated constraints (initially zero); and *upperbound* is the stored cost of the best solution found so far. Note that unlike **ARIO**, we require no MLLP module or SAT-solving component, and unlike **GAPD** our memory requirements are polynomial in the size of the problem.

[4] The ϵ-relaxation bears a strong resemblance to the clause selectors used in Max-SAT, as both serve to disable constraints. However, the ϵ relation corresponds to a *variable* in the original search space of the decision problem (i.e., the meta-CSP), whereas the clause selector directly corresponds to a *constraint* in the original search space.

This algorithm resembles the meta-CSP backtracking search commonly used for solving traditional DTPs with two notable differences. First, backtracking occurs only when the combined valuation 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 the empty assignment 'ϵ' that serves to explicitly violate a constraint, increasing the branching factor by exactly one. This latter modification, in combination with the meta-CSP search space employed in temporal reasoning, sets the algorithm apart from previous applications of preference optimization to classical CSPs.

The basic framework of Fig. 4 may be invoked in one of two ways to achieve optimization:

- **branch-and-bound (B&B)**: By initially setting *upperbound* to ∞, a single call to **Solve-VDTP** will cause solutions of progressively higher quality to be found and stored. The costs of solutions found early in search are used to set bounds on the allowed number of violations in the tree.
- **iterative weakening (I.W.)**: An initial setting of *upperbound* to 0 will cause **Solve-VDTP** to conservatively search for solutions with no violations. In the event of failure, a second call to **Solve-VDTP** will search more broadly for solutions with a single violation. The process continues by executing a sequence of independent searches until a solution with the specified cost is found.

In cases where relatively few violations are required, the approach taken by iterative weakening tends to explore small trees with comparatively limited depth.

7.2. Advantages of the meta-CSP

The meta-CSP reformulation of DTPP optimization offers several key advantages as opposed to the approaches taken by **GAPD** and **ARIO**. First and foremost, it embeds classical cost-based pruning (i.e., elimination of search nodes based on the lower-bound of partial solution value) into all nodes of search; in contrast, the decision-based **ARIO** solver is forced to explore a sequence of increasingly harder satisfaction problems, many of which are similar to one another and require redundant search. Second, by encoding the relaxation (or violation) of a constraint as an additional empty disjunct, it allows the direct incorporation of powerful techniques previously developed in decision-based DTP literature [32,1,22,33]. 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. Finally, it generalizes to any VDTP with a compliant valuation structure (i.e., having totally ordered valuations and a commutative, associative closed binary operator), extending to the popular criteria of utilitarian and maximin optimality.

7.3. Prevention of superfluous search

The algorithm described in the previous section has one apparent deficiency: no attempt has been made to ensure that meta-assignments made to constraints projected from a single preference function do not draw from different areas of the preference profile. This nuance is a direct consequence of the meta-CSP, and can potentially lead to superfluous or redundant partial assignments. As an example, consider a single disjunctive DTPP constraint projected at several levels:

$$
\begin{aligned}
&C_1\colon \{c_{\langle 1,1\rangle}\colon a_{1,1} \leqslant x_1 - y_1 \leqslant b_{1,1}\} \vee \{c_{\langle 2,1\rangle}\colon a_{2,1} \leqslant x_2 - y_2 \leqslant b_{2,1}\}, \quad \varphi(C_1) = k_1\\
&C_2\colon \{c_{\langle 1,2\rangle}\colon a_{1,2} \leqslant x_1 - y_1 \leqslant b_{1,2}\} \vee \{c_{\langle 2,2\rangle}\colon a_{2,2} \leqslant x_2 - y_2 \leqslant b_{2,2}\}, \quad \varphi(C_2) = k_2\\
&C_3\colon \{c_{\langle 1,3\rangle}\colon a_{1,3} \leqslant x_1 - y_1 \leqslant b_{1,3}\} \vee \{c_{\langle 2,3\rangle}\colon a_{2,3} \leqslant x_2 - y_2 \leqslant b_{2,3}\}, \quad \varphi(C_3) = k_3\\
&\quad\ldots\\
&C_L\colon \{c_{\langle 1,L\rangle}\colon a_{1,L} \leqslant x_1 - y_1 \leqslant b_{1,L}\} \vee \{c_{\langle 2,L\rangle}\colon a_{2,L} \leqslant x_2 - y_2 \leqslant b_{2,L}\}, \quad \varphi(C_L) = k_L
\end{aligned}
$$

We make no assumptions about the specific temporal intervals on these constraints, nor the specific values of the valuations, with the exception that for all i, j, $a_{i,j} \geqslant a_{i,j+1}$ and $b_{i,j} \leqslant b_{i,j+1}$ (in other words, preference profiles must tighten monotonically at higher levels). Naturally, we would expect that a solution will satisfy either the difference $x_1 - y_1$ or the difference $x_2 - y_2$ up to a particular (not necessarily the highest) preference level. Following this logic, there should be no more than $2 \times L$ possible ways in which to satisfy this set of constraints. The following two assignments reflect this expectation:

$$
\begin{aligned}
&(C_1, C_2, C_3, \ldots, C_{L-1}, C_L) \leftarrow (c_{\langle 1,1\rangle}, c_{\langle 1,2\rangle}, c_{\langle 1,3\rangle}, \ldots, c_{\langle 1,L-1\rangle}, \epsilon)\\
&(C_1, C_2, C_3, \ldots, C_{L-1}, C_L) \leftarrow (c_{\langle 2,1\rangle}, c_{\langle 2,2\rangle}, c_{\langle 2,3\rangle}, \ldots, \epsilon, \epsilon)
\end{aligned}
$$

However, note that in the process of creating these valued constraints, no direct links have been established to tie together disjuncts in the VDTP that were obtained from the same ancestral disjunct in the DTPP. For instance, any of the following assignments are legitimate meta-CSP solutions:

$$
\begin{aligned}
&(C_1, C_2, C_3, \ldots, C_{L-1}, C_L) \leftarrow (c_{\langle 1,1\rangle}, \epsilon, c_{\langle 2,3\rangle}, \ldots, c_{\langle 1,L-1\rangle}, \epsilon)\\
&(C_1, C_2, C_3, \ldots, C_{L-1}, C_L) \leftarrow (c_{\langle 2,1\rangle}, c_{\langle 1,2\rangle}, \epsilon, \ldots, c_{\langle 2,L-1\rangle}, \epsilon)
\end{aligned}
$$

These assignments are peculiar in two aspects: first, disjuncts are drawn from portions of *both* preference profiles, as opposed to only one. Second, we observe that some ϵ-relaxations occur at preference levels below those of constraints that are legitimately instantiated (i.e., a more difficult constraint is satisfied while a strictly less restrictive constraint is violated). Thus, if one were to explore all possible combinations, a grand total of 3^L assignments would be enumerated. In the general case of a DTPP constraint with m disjuncts, a worst case scenario would explore $(m+1)^L$ partial assignments instead of the necessary $m * L$. The search space is further exacerbated by the combinatorial explosion of satisfying *all* constraints in the original problem, making this subtle peculiarity a serious practical concern for achieving optimization. We refer to any such superfluous assignment encountered during search as a *degenerate* solution.

Definition *(degenerate solution).* Given a meta-CSP solution $S = \{C_{\langle i,L\rangle} \rightarrow D(C_{\langle i,L\rangle})\}$ to a VDTP D constructed from a DTPP D' as described in Section 6.2, we say that S is a *degenerate solution* iff there exist two constraints $C_{\langle i,l\rangle}$ and $C_{\langle i,l'\rangle}$ such that $l' > l$, and the disjunct $c_{\langle i,l'\rangle} = S(C_{\langle i,l'\rangle})$ does not subsume $c_{\langle i,l\rangle} = S(C_{\langle i,l\rangle})$ (i.e., the disjuncts are drawn from different regions of the preference profile).[5]

The algorithms presented in [26] and [31] make explicit attempts to prevent the combinatorial explosion that might result from such allowances. In the former case, a tree-based encoding ensures that assignments proceed strictly upwards in the profile, neither skipping levels nor borrowing from adjacent regions. In the latter case, additional boolean clauses are added to prevent the SAT representation from encountering such solutions.

In this section, we argue that these mechanisms are largely unnecessary if traditional DTP pruning strategies are invoked. In particular, we show how the well-established techniques of *removal of subsumed variables* and *semantic branching* prevent unwanted redundancy, regardless of whether meta-level variable heuristics are made to prefer low-level or high-level preference projections. Both of these techniques are unique to the meta-CSP formulation, and have been known for nearly a decade; their efficacy in reducing computational effort in temporal reasoning has been extensively studied in prior work [33], with semantic branching providing the most significant savings. Here, we expose their effect on the structural relationship between constraints that are created in the conversion of a DTP with Preferences to a Valued DTP.

7.3.1. High-level assignments preceding low-level assignments

Consider a search tree in which meta-variables corresponding to high preference levels occur earlier than those corresponding to lower levels. A portion of such a tree is depicted in Fig. 5; to reflect that other constraints may be considered earlier or later in search, incoming and outgoing arrows are placed on nodes at the top and bottom of the figure.

In the absence of traditional meta-CSP pruning techniques, the following possible progression of assignments is possible:

Search node #	Partial solution
1	$C_L \leftarrow c_{\langle 1,L\rangle}$
	...
k	$C_L \leftarrow c_{\langle 1,L\rangle}$ $C_{L-1} \leftarrow c_{\langle 2,L-1\rangle}$
$k+1$	$C_L \leftarrow c_{\langle 1,L\rangle}$ $C_{L-1} \leftarrow c_{\langle 2,L-1\rangle}$ $C_{L-2} \leftarrow c_{\langle 1,L-2\rangle}$
	...
$k+m$	$C_L \leftarrow c_{\langle 1,L\rangle}$ $C_{L-1} \leftarrow c_{\langle 2,L-1\rangle}$ $C_{L-2} \leftarrow c_{\langle 2,L-2\rangle}$
	...
$k+m+n$	$C_L \leftarrow c_{\langle 1,L\rangle}$ $C_{L-1} \leftarrow c_{\langle 2,L-1\rangle}$ $C_{L-2} \leftarrow \epsilon$
	...

Observe that all partial assignments considered between nodes k and $k+m+n$ include one disjunct taken from the preference profile of c_1, and another disjunct from the preference profile of c_2 (where c_1 and c_2 are disjuncts derived from an ancestral DTPP constraint $C = c_1 \vee c_2$).

However, consider the application of *removal of subsumed variables* [22], a pruning technique that refrains from making assignments to any meta-variable if an inequality contained in one of its disjuncts is less constraining than the constraint imposed by the existing temporal network. The invocation of the disjunct $c_{\langle 1,L\rangle}$ (i.e., $a_{1,L} \leqslant x_1 - y_1 \leqslant b_{1,L}$) has the effect of subsuming constraint C_{L-1}, which contains the strictly looser disjunct $c_{\langle 1,L-1\rangle}$ (i.e., $a_{1,L-1} \leqslant x_1 - y_1 \leqslant b_{1,L-1}$). This area of pruned space is labeled **B**. Furthermore, all disjuncts $c_{\langle 1,i\rangle}$ (and thus all meta-variables C_i) such that $i < L$ are subsumed by this assignment to C_L (for instance, the region labeled **A** corresponds to the removal of subsumed variable C_{L-2}).

Similarly, when C_L is assigned the disjunct $c_{\langle 2,L\rangle}$ (i.e., $a_{2,L} \leqslant x_2 - y_2 \leqslant b_{2,L}$), all other constraints C_i such that $i < L$ are again subsumed. In Fig. 5, these pruned regions are labeled **C**, **D**, **E**, and **F**.

After both of these disjuncts have been attempted, C_L is given the empty value ϵ; since such an assignment does not invoke any changes to the temporal network, no meta-variables are subsumed. However, when assignments to C_{L-1} are made, we encounter a familiar pattern: the selection of $c_{\langle 1,L-1\rangle}$ subsumes constraints at all lower levels (pruning the region labeled **G**), as does the selection of $c_{\langle 2,L-1\rangle}$ (pruning the regions labeled **H** and **I**).

[5] Note that $c_{\langle i,l\rangle}$ may, in fact, be the empty assignment ϵ.

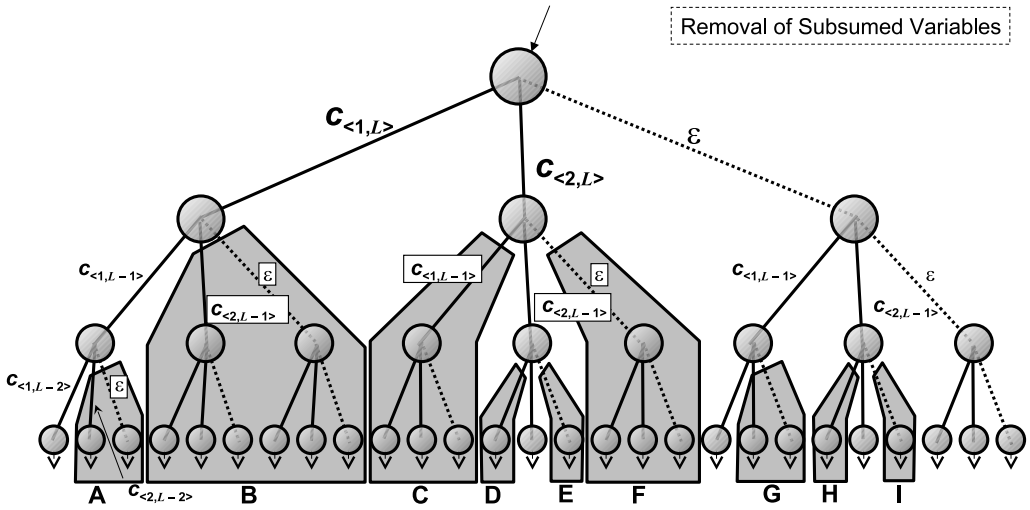


Fig. 5. Removal of subsumed variables prevents the exploration of redundant search nodes when high-level assignments precede low-level assignments.

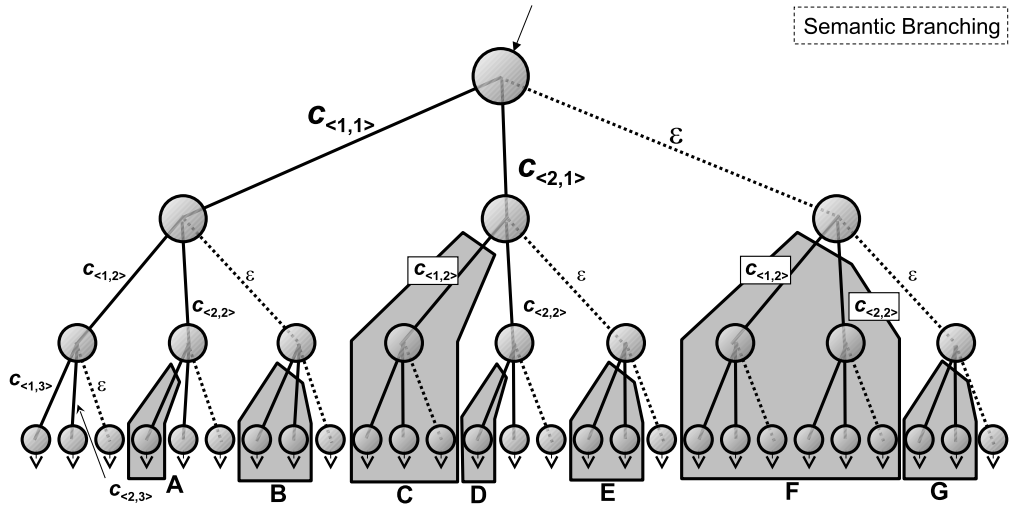


Fig. 6. Semantic branching prevents the exploration of redundant search nodes when low-level assignments precede high-level assignments.

7.3.2. Low-level assignments preceding high-level assignments

Consider a search tree in which meta-variables corresponding to low preference levels occur earlier than those corresponding to high levels. A portion of such a tree is depicted in Fig. 6.

In the absence of traditional meta-CSP pruning techniques, the following possible progression of assignments is possible:

Search node #	Partial solution
1	$C_1 \leftarrow c_{\langle 1,1\rangle}$
2	$C_1 \leftarrow c_{\langle 1,1\rangle}$ $C_2 \leftarrow c_{\langle 1,2\rangle}$
	...
k	$C_1 \leftarrow c_{\langle 1,1\rangle}$ $C_2 \leftarrow c_{\langle 2,2\rangle}$
$k+1$	$C_1 \leftarrow c_{\langle 1,1\rangle}$ $C_2 \leftarrow c_{\langle 2,2\rangle}$ $C_3 \leftarrow c_{\langle 1,3\rangle}$
	...
$k+m$	$C_1 \leftarrow c_{\langle 1,1\rangle}$ $C_2 \leftarrow c_{\langle 2,2\rangle}$ $C_3 \leftarrow c_{\langle 2,3\rangle}$
	...

Once again, we observe that some partial assignments include disjuncts from the preference profiles of both c_1 and c_2. In particular, the assignment at search node $k+1$ includes $C_2 \leftarrow c_{\langle 2,2\rangle}$ and $C_3 \leftarrow c_{\langle 1,3\rangle}$. While the latter of these values subsumes the disjunct $c_{\langle 1,2\rangle}$, its corresponding meta-variable C_2 cannot be subsumed since it is instantiated prior to C_3 (and thus cannot receive an alternate assignment).

Now consider the application of *semantic branching* [1], a pruning technique that enforces the negation of a constraint at a particular level in search if all assignments extending that constraint have been explored exhaustively; the addition of the negated expression has the effect of tightening the temporal network and preventing the expansion of unnecessary subsequent nodes. The exploration of the disjunct $c_{\langle 1,2\rangle}$ (i.e., $a_{1,2} \leqslant x_1 - y_1 \leqslant b_{1,2}$) will be followed by a subtree in which both $c_{\langle 2,2\rangle}$ (i.e., $a_{2,2} \leqslant x_2 - y_2 \leqslant b_{2,2}$) and $\neg c_{\langle 1,2\rangle}$ (i.e., $a_{1,2} > x_1 - y_1 \vee x_1 - y_1 > b_{1,2}$) are enforced. Note that this latter constraint is in direct conflict with $c_{\langle 1,3\rangle}$, a phenomenon that can be understood intuitively: if the satisfaction of a low level of search has been explored (including extensions that do and do not satisfy the next highest level), then there is no benefit in reexamining those assignments that satisfy the next higher level again. This area of pruned space is labeled **A**. In fact, all disjuncts $c_{\langle 1,i\rangle}$ such that $i > 2$ will be ignored at deeper levels of search.

Similarly, when the disjunct $c_{\langle 2,2\rangle}$ has been explored and C_2 is assigned the empty value ϵ, there will be two negations enforced on all nodes below: $\neg c_{\langle 1,2\rangle}$ and $\neg c_{\langle 2,2\rangle}$. Since all remaining unassigned meta-variables belong to higher preference levels, the only viable assignments that do not conflict with these negations are the ϵ-relaxations. In Fig. 6, we see the effect of this pruning in the region labeled **B**.

As failures propagate higher and higher into the tree, the effect of semantic branching becomes more and more significant. This is due to looser values toward the top of search tree providing tighter negations upon retraction, thereby preventing redundant extensions to several subsequent preference levels that would otherwise follow.

7.3.3. Optimality of degenerate assignments

While the superfluous search space created by preference projections creates a combinatorial explosion in theory, we have shown that it is prevented in practice by enabling the well-established pruning techniques of removal of subsumed variables and semantic branching. Even if one chooses not to employ these advanced pruning techniques, we can nevertheless guarantee that the partial component STP corresponding to any degenerate meta-level assignment encompasses a set of object-level solutions to a non-degenerate assignment of equal or greater value. This final step ensures the soundness of the algorithm, e.g., that the final solutions it generates will not have suboptimal global preference values.

Theorem 7.1. *For any degenerate solution S, there exists a non-degenerate solution S' of equal or greater value whose object-level solutions are a superset of those in S.*

Proof. To construct S', consider any pair of assignments $C_{\langle i,l'\rangle} \leftarrow c_{\langle i,l'\rangle}$ and $C_{\langle i,l\rangle} \leftarrow c_{\langle i,l\rangle}$ in S, where $l' > l$. If $child(c_{\langle i,l'\rangle}, l)$ denotes the disjunct in the preference projection at level l that falls directly under $c_{\langle i,l'\rangle}$ in the preference profile of D', we can safely replace the disjunct $c_{\langle i,l\rangle}$ with $child(c_{\langle i,l'\rangle}, l)$, since $child(c_{\langle i,l'\rangle}, l)$ is strictly less constraining than the currently enforced constraint $c_{\langle i,l'\rangle}$.

This process is repeated for all pairs of mismatched disjuncts until there remains no evidence of degeneracy. Since no additional ϵ-relaxations are introduced, the cost of the solution cannot degrade. Furthermore, since each replacement is performed without tightening the temporal network, the set of object-level solutions in the final global assignment S' includes all assignments contained in S. Hence, any solution extracted from the degenerate solution S also belongs to the non-degenerate solution S', and is guaranteed to be optimal provided that S is itself an optimal meta-level assignment. □

8. Experimental results

In this section, we describe the results of a set of experiments that were performed to compare an implementation of valued constraint satisfaction approach (which we name **MAXILITIS-V**) against the two previous systems for solving DTPPs: **ARIO** [31] and the **GAPD** solver [26]. **MAXILITIS-V** incorporates both semantic branching and removal of subsumed variables, though it does not utilize the variants of conflict-directed backjumping that have been used by some prior solvers. We also employ a *minimum remaining values* (or MRV) variable ordering heuristic [12] that dynamically chooses variables to instantiate based on the fewest number of feasible disjunctive clauses remaining in the domain. As in [33], several tie breaking methods are employed, including the number of pairwise conflicts with other disjuncts, as well as the amount of slack on the edges involved in the inequalities. Since the meta-variables of the VDTP correspond directly to constraints, we also easily modify the heuristic to prefer the instantiation of any "hard" constraint that must be satisfied prior to any "soft" constraints having finite valuations (constraints of both types are common in converted DTPP instances). Many other strategies for variable ordering – including variants of those explored in recent studies of finite-domain CSPs [2] – are possible and are worthy of continued research.

Unfortunately, there remains an absence of real-world benchmarks in temporal preference literature with which to provide an empirical comparison of solvers.[6] 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 $\{x_1, x_2, \ldots, x_E\}$ 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

[6] The benchmarks used in the SMT competition [3] contain no optimization component, and are thus incapable of representing the local and global preference objectives in the problems of interest.

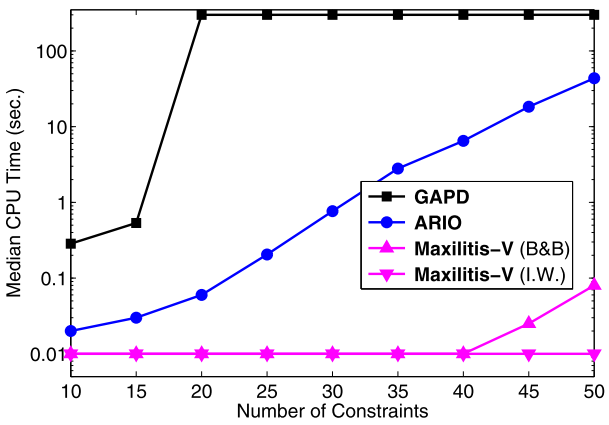


Fig. 7. Median running times for **GAPD**, **ARIO**, and **MAXILITIS-V** for DTPPs of varying sizes.

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 four sets of experiments, in which we evaluated the effect of problem size, the number of preference levels, the constraint density, and their relevance to anytime performance. For all experiments, we generate 50 trials for each setting of parameters, and report the median running time for each solver over the 50 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 negligible.

8.1. Varying problem size

In the first experiment, we explore the abilities of **ARIO**, **GAPD**, and **MAXILITIS-V** to scale with the size of the problem. The 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 previously published work [31]. We vary the number of constraints C from 10 to 50, and set the number of events E to $\frac{4}{5}C$ to maintain a constant constraint density.

Fig. 7 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 43.62 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 weakening versions of **MAXILITIS-V** 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 0.01 seconds, over three orders of magnitude faster than **ARIO**.

8.2. Varying the number of preference levels

In the 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.

Fig. 8 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 **MAXILITIS-V**. For the case where the number of preference levels is seven, their respective median running times are 300, 62.71, and 0.16 seconds (this last being for the iterative version; the branch-and-bound variant requires 10.735 seconds on average). When eight preference levels are allowed, all solvers experience significant difficulty. Overall, a difference of one to two orders of magnitude is observed between **ARIO** and the iterative weakening incarnation of valued constraint satisfaction for cases where the number of preference levels is between four and seven.

8.3. Varying constraint density

In the third 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.

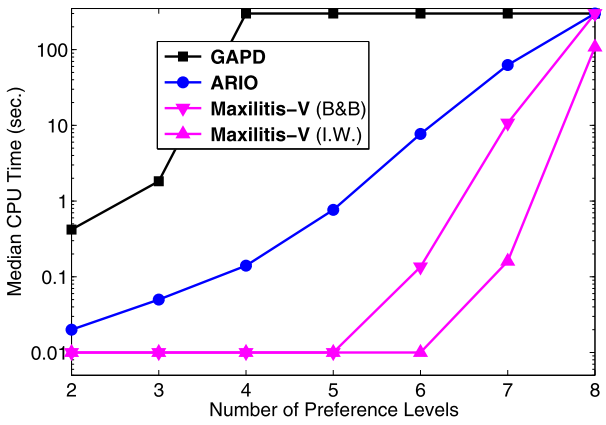


Fig. 8. Median computation time for **GAPD**, **ARIO**, and **MAXILITIS-V** for DTPPs with varying numbers of preference levels.

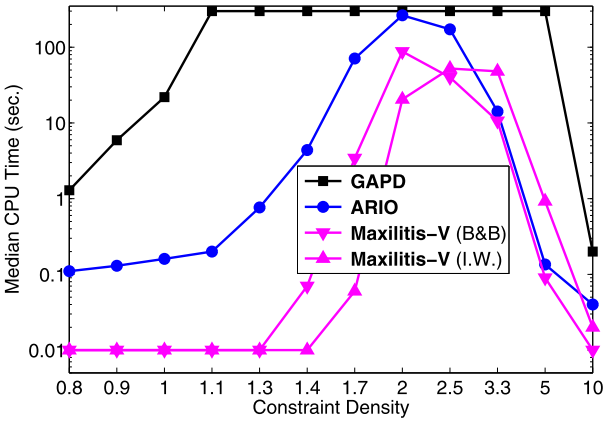


Fig. 9. Median running times for **GAPD**, **ARIO**, and **MAXILITIS-V** for DTPPs of varying constraint density.

In Fig. 9 we present the results. For problems having constraint densities less than or equal to 2.5, both incarnations of **MAXILITIS-V** 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 **MAXILITIS-V** 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.

8.4. Anytime performance

Prior work has underscored the particular importance for a temporal optimization engine to exhibit desirable anytime properties [26]. The reason for this is clear: extremely large, complex problems that contain many preference levels and temporal events are often too difficult to solve to optimality. Furthermore, optimality is often of little use in practice, especially if there is considerable error involved in the preference modelling phase. If computational methods for preferential optimization are incapable of producing high-quality solutions early in search, then there is little guarantee for successful integration of this technology into applications that stress the real-time response of a system.

In order to determine the anytime behavior of **MAXILITIS-V**, we conducted one final set of experiments. Nine different problem sizes were obtained by varying two dimensions of the generator's parameters: the number of preference levels (selected from the range $\{5, 10, 15\}$), and the number of events/constraints (selected from the following settings: 10 events/30 constraints, 20 events/40 constraints, and 40 events/100 constraints). We hold fixed the remaining parameters at $\{D_- = -400, D_+ = 500, R_- = 0.5, R_+ = 0.9\}$, and enforce a timeout limit of 60 seconds. These parameters are all identical to those used in [26] for a similar set of anytime experiments. As in previous tests, 50 instances of each problem size were created, and all graphs reflect an average solution cost over all problem instances as a function of time.

The results for the simplest set of problems (shown in Fig. 10) suggest that when the number of preference levels is small, **ARIO** consistently outperforms **GAPD** in regards to time taken to generate an initial solution as well as the final average solution cost; when the number of levels is increases, **GAPD** becomes more effective in the very early stages of search, but is eventually overtaken by **ARIO**. In contrast, **MAXILITIS-V** outperforms both solvers on either measure and for all

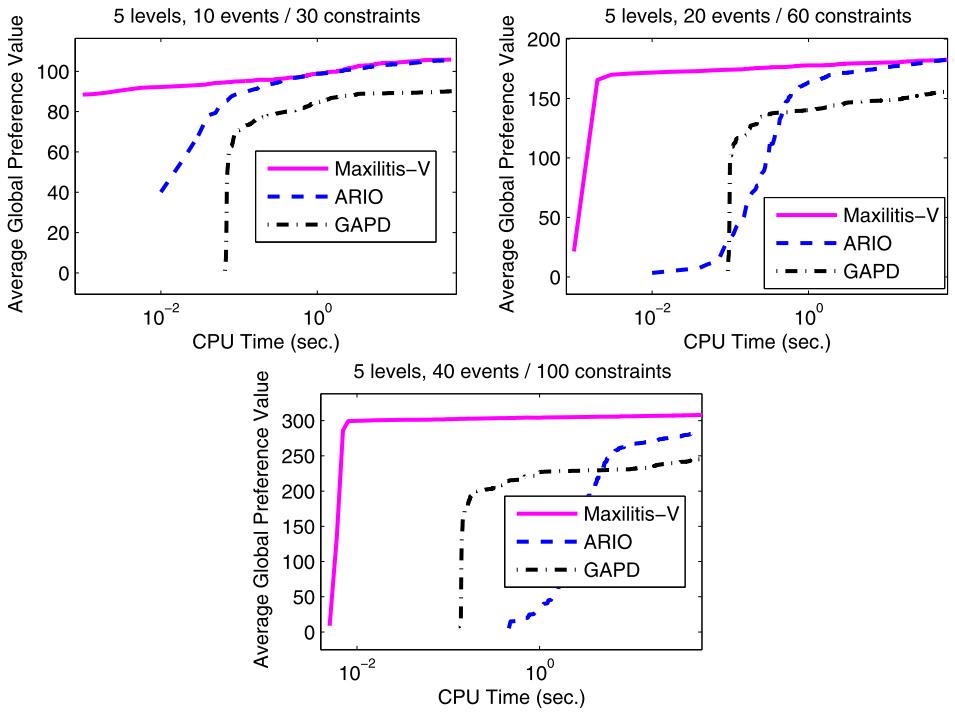


Fig. 10. Anytime curves for **GAPD**, **ARIO**, and **MAXILITIS-V** on DTPPs with a small number of preference levels ($L = 5$).

preference level parameters. Not only does **MAXILITIS-V** find an initial solution almost immediately, but the quality of the solution is often improved to an extraordinary level before either **ARIO** or **GAPD** begin producing output of any kind. This effect is most noticeable on the problems that have the greatest size.

The results for problems with moderate and large numbers of preference levels are shown in Figs. 11 and 12. While the relative performance of **GAPD** continues to improve as problems become larger in size and taller in their preference profiles, we observe that **MAXILITIS-V** remains considerably far ahead of both its competitors.

8.5. Analysis of results

In summary, the valued constraint satisfaction approach – achieved by adapting a state-of-the-art DTP engine to accommodate the ϵ-relaxation and a branch-and-bound pruning strategy – consistently outperforms the previous DTPP solvers, including the SAT-based **ARIO** system and the specialized **GAPD** solver, on several dimensions (including the runtime needed to obtain optimal solutions, and the anytime performance on large problems that are too difficult to solve completely). While we suspect that continued focus on optimization techniques within the SAT and Satisfiability Modulo Theories communities will lead to improved performance in future solvers, these results clearly demonstrate that well-established CSP-based methods remain among the most successful approaches for constraint-based optimization to date.

9. Conclusions

In this paper, we have explored the modelling and optimization of preferences within the context of metric temporal reasoning. We have proposed the Valued DTP, a cousin of the DTP with Preferences that provides an alternative framework for expressing optimization variants of the DTP. We have proven the equivalence of the VDTP and the DTPP when profiles are assumed to be piecewise-constant. We have identified the principal advantages of the Valued DTP representation; namely, that by exploiting the ϵ-relaxation to achieve the explicit violation of constraints (and their associated meta-variables), we permit the straightforward representation of a meta-level assignment as a *partial* component STP. This enables the creation of a powerful algorithm for computing optimal solutions to the Valued DTP, deviating only slightly from the existing meta-CSP framework used to solving traditional temporal constraint problems. We have demonstrated that the pruning techniques commonly used in state-of-the-art engines – namely, the removal of subsumed variables and semantic branching – naturally cope with redundant solutions in the context of optimization, eliminating the need for additional mechanisms that previous algorithms have incorporated into their internal encodings. We have shown empirically that the runtime of **MAXILITIS-V** is considerably faster than other engines; in problems with varying size, constraint density, and numbers of preference levels, the efficiency of **MAXILITIS-V** was consistently competitive. For large problems, several orders of magnitude improvement was

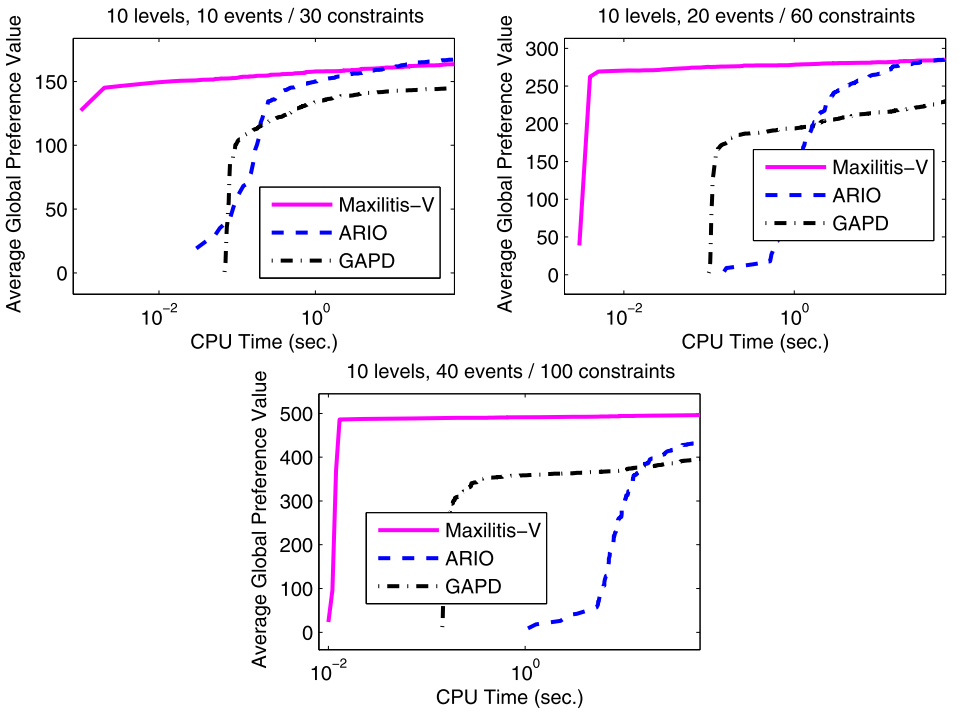


Fig. 11. Anytime curves for **GAPD**, **ARIO**, and **MAXILITIS-V** on DTPPs with a moderate number of preference levels ($L = 10$).

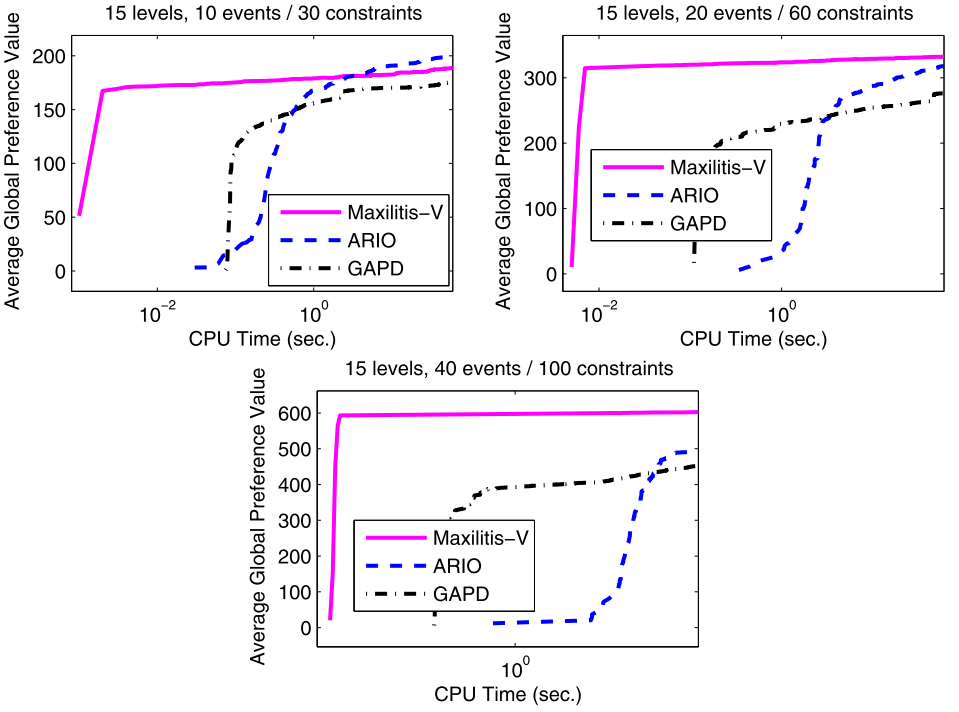


Fig. 12. Anytime curves for **GAPD**, **ARIO**, and **MAXILITIS-V** on DTPPs with a large number of preference levels ($L = 15$).

observed. Furthermore, we have shown empirically that the anytime properties of **MAXILITIS-V** far surpass those of previous solvers (including those designed specifically for the purpose of anytime performance) on a wide variety of benchmarks.

There remains great potential for continued research in the context of preference optimization for constraint-based temporal reasoning. First, the question of which encoding to use (either the VDTP or the DTPP) may depend on the application.

While the approach presented here exploits a one-way conversion, some domains may be more naturally expressed in one or the other. In addition, more complex aggregation models may be required when considering the translation of temporal constraints to the conjunctive normal form of the DTPP (to prevent the "doubling-up" of preference contribution from correlated constraints). Finally, it would be valuable to consider a class of problems where the preference profiles of temporal constraints are known only partially, and optimization must thus move forward with an incomplete model of the valuation structure [11,27].

Acknowledgements

The author is indebted to Prof. Martha E. Pollack for insight and assistance with preliminary versions of this work. The author is also grateful to Dr. Bart Peintner and Dr. Neil Yorke-Smith for several useful discussions, and to the anonymous reviewers for their valuable feedback. Dr. Moffitt is supported by the Josef Raviv Memorial Postdoctoral Fellowship.

References

[1] A. Armando, C. Castellini, E. Giunchiglia, SAT-based procedures for temporal reasoning, in: Proceedings of the 5th European Conference on Planning (ECP 1999), 1999, pp. 97–108.
[2] T. Balafoutis, K. Stergiou, On conflict-driven variable ordering heuristics, in: Proceedings of the ERCIM workshop on Constraint Solving and Constraint Logic Programming (CSCLP-2008), 2008.
[3] C. Barrett, L. de Moura, A. Stump, SMT-COMP: Satisfiability modulo theories competition, in: Proceedings of the 17th International Conference on Computer Aided Verification (CAV 2005), 2005, pp. 20–23.
[4] P. Berry, M. Gervasio, B. Peintner, N. Yorke-Smith, The design of a user-centric scheduling system for multi-faceted real-world problems, in: Proceedings of ICAPS'07 Workshop on Moving Planning and Scheduling Systems Into the Real World, 2007.
[5] P. Berry, M. Gervasio, B. Peintner, N. Yorke-Smith, A preference model for over-constrained meeting requests, in: Proceedings of AAAI 2007 Workshop on Preference Handling for Artificial Intelligence, 2007.
[6] S. Bistarelli, Semirings for Soft Constraint Solving and Programming, Springer, 2004.
[7] S. Bistarelli, U. Montanari, F. Rossi, Semiring-based constraint satisfaction and optimization, Journal of the ACM 44 (2) (1997) 201–236.
[8] S. Bistarelli, U. Montanari, F. Rossi, T. Schiex, G. Verfaillie, H. Fargier, Semiring-based CSPs and valued CSPs: Frameworks, properties, and comparison, Constraints 4 (3) (1999) 199–240.
[9] R. Dechter, Constraint Processing, Morgan Kaufmann, 2003.
[10] R. Dechter, I. Meiri, J. Pearl, Temporal constraint networks, Artificial Intelligence 49 (1–3) (1991) 61–95.
[11] M. Gelain, M.S. Pini, F. Rossi, K.B. Venable, Dealing with incomplete preferences in soft constraint problems, in: Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming (CP 2007), 2007, pp. 286–300.
[12] R.M. Haralick, G.L. Elliott, Increasing tree search efficiency for constraint satisfaction problems, Artificial Intelligence 14 (3) (1980) 263–313.
[13] L. Khatib, P. Morris, R. Morris, F. Rossi, Temporal constraint reasoning with preferences, in: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), 2001, pp. 322–327.
[14] L. Khatib, P. Morris, R. Morris, K.B. Venable, Tractable Pareto optimal optimization of temporal preferences, in: Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003), 2003, pp. 1289–1294.
[15] T.K.S. Kumar, 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), 2004, pp. 67–72.
[16] J. Larrosa, F. Heras, S. de Givry, A logical approach to efficient Max-SAT solving, Artificial Intelligence 172 (2–3) (2008) 204–233.
[17] M.D. Moffitt, B. Peintner, N. Yorke-Smith, Multi-criteria optimization of temporal preferences, in: Proceedings of the 8th International Workshop on Preferences and Soft Constraints, 2006.
[18] M.D. Moffitt, M.E. Pollack, Partial constraint satisfaction of disjunctive temporal problems, in: Proceedings of the 18th International FLAIRS Conference (FLAIRS 2005), 2005, pp. 715–720.
[19] M.D. Moffitt, M.E. Pollack, Temporal preference optimization as weighted constraint satisfaction, in: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006), 2006, pp. 110–116.
[20] P. Morris, R. Morris, L. Khatib, S. Ramakrishnan, A. Bachmann, Strategies for global optimization of temporal preferences, in: Proceedings of the 10th International Conference on Principles and Practices of Constraint Programming (CP 2004), 2004, pp. 408–422.
[21] M.W. Moskewicz, C.F. Madigan, Y. Zhao, L. Zhang, S. Malik, Chaff: Engineering an efficient SAT solver, in: Proceedings of the 38th Design Automation Conference (DAC 2001), 2001, pp. 530–535.
[22] A. Oddi, A. Cesta, Incremental forward checking for the disjunctive temporal problem, in: Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000), 2000, pp. 108–112.
[23] B. Peintner, M.E. Pollack, Low-cost addition of preferences to DTPs and TCSPs, in: Proceedings of the 19th National Conference on Artificial Intelligence (AAAI 2004), 2004, pp. 723–728.
[24] B. Peintner, M.E. Pollack, Anytime, complete algorithm for finding utilitarian optimal solutions to STPPs, in: Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005), 2005, pp. 443–448.
[25] B. Peintner, P. Viappiani, N. Yorke-Smith, Preferences in interactive systems: Technical challenges and case studies, AI Magazine 29 (4) (2008) 13–24.
[26] B.M. Peintner, Algorithms for constraint-based temporal reasoning with preferences, PhD thesis, University of Michigan, 2005.
[27] M.S. Pini, F. Rossi, K.B. Venable, T. Walsh, Incompleteness and incomparability in preference aggregation, in: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), 2007, pp. 1464–1469.
[28] F. Rossi, Preference reasoning, in: Proceedings of the 11th International Conference on Principles and Practice of Constraint Programming (CP 2005), 2005, pp. 9–12.
[29] F. Rossi, P. van Beek, T. Walsh, Handbook of Constraint Programming, Elsevier, 2006.
[30] T. Schiex, H. Fargier, G. Verfaillie, Valued constraint satisfaction problems: Hard and easy problems, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 1995), 1995, pp. 631–639.
[31] H.M. Sheini, B. Peintner, K.A. Sakallah, M.E. Pollack, On solving soft temporal constraints using SAT techniques, in: Proceedings of the 11th International Conference on Principles and Practice of Constraint Programming (CP 2005), 2005, pp. 607–621.
[32] K. Stergiou, M. Koubarakis, Backtracking algorithms for disjunctions of temporal constraints, in: Proceedings of the 15th National Conference on Artificial Intelligence (AAAI 1998), 1998, pp. 248–253.

[33] I. Tsamardinos, M.E. Pollack, Efficient solution techniques for disjunctive temporal reasoning problems, Artificial Intelligence 151 (1–2) (2003) 43–90.
[34] T. Walsh, Uncertainty in preference elicitation and aggregation, in: Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI 2007), 2007, pp. 3–8.
[35] T. Walsh, Complexity of terminating preference elicitation, in: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), 2008, pp. 967–974.