Translation-based Approaches for Solving Disjunctive Temporal Problems with Preferences

Enrico Giunchiglia[1], Marco Maratea[1], and Luca Pulina[2]

[1] DIBRIS - University of Genova, Viale F. Causa 15, Genova, Italy.
{enrico.giunchiglia,marco.maratea}@unige.it
[2] POLCOMING - University of Sassari, Viale Mancini 5, 07100 Sassari, Italy.
lpulina@uniss.it

Abstract. Disjunctive Temporal Problems (DTPs) with Preferences (DTPPs) extend DTPs with piece-wise constant preference functions associated to each constraint of the form $l \leq x - y \leq u$, where x, y are (real or integer) variables, and l, u are numeric constants. The goal is to find an assignment to the variables of the problem that maximizes the sum of the preference values of satisfied DTP constraints, where such values are obtained by aggregating the preference functions of the satisfied constraints in it under a "max" semantic. The state-of-the-art approach in the field, implemented in the native DTPP solver MAXILITIS, extends the approach of the native DTP solver EPILITIS.
In this paper we present alternative approaches that translate DTPPs to Maximum Satisfiability of a set of Boolean combination of constraints of the form $l \bowtie x - y \bowtie u$, $\bowtie \in \{<, \leq\}$, that extend previous work dealing with constant preference functions only. We prove correctness and completeness of the approaches. Results obtained with the Satisfiability Modulo Theories (SMT) solvers YICES and MATHSAT on randomly generated DTPPs and DTPPs built from real-world benchmarks, show that one of our translation is competitive to, and can be faster than, MAXILITIS.[3]

1 Introduction

Temporal constraint networks [2] provide a convenient formal framework for representing and processing temporal knowledge. Over the years, a number of extensions to the framework have been presented. Disjunctive Temporal Problems (DTPs) with Preferences (DTPPs) is one of such extensions. DTPPs extend DTPs, i.e. conjunctions of disjunctions of constraints of the form $l \leq x - y \leq u$, where x, y are (real or integer) variables, and l, u are numeric constants, with piece-wise constant preference functions associated to each constraint. The goal is to find an assignment to the variables of the problem that maximizes the sum of the preference values of satisfied disjunctions of constraints (called DTP constraints), where such values are obtained by aggregating the preference functions of the satisfied constraints in it. The DTPP has been employed in a number

[3] This is a an extended and revised version of [1].

of real-world applications, including scheduling meeting requests and the problem of building automatic assistants (see, e.g. [3–5]). We consider an utilitarian aggregation of such DTP constraints values, and a "max" semantic for aggregating preference values within DTP constraints: given a (candidate) solution of a DTPP, the preference value of each DTP constraint is defined to be the maximum value achieved by any of its satisfied disjuncts (see, e.g. [6]). The current state-of-the-art approach that considers such aggregation methods is implemented in the native DTPP solver MAXILITIS, and is based on an extension of the DTP solver EPILITIS [7] to deal with piece-wise constant preference functions. Various other approaches have been designed in the literature to deal with DTPPs [6, 8–10], possibly relying on alternative preference aggregation methods (see, e.g. [11, 12]).

In this paper we present alternative approaches that translate DTPPs to Maximum Satisfiability of a set of Boolean combination of constraints of the form $l \bowtie x - y \bowtie u$, where $\bowtie \in \{<, \leq\}$. In case of unsatisfiable DTPs, our approaches provide a "Max-SAT optimal" solution (defined precisely later), rather than just reporting the problem to be unsatisfiable as done by MAXILITIS. The first approach relies on a very natural modeling of the problem where, for each soft DTP constraint, the generated constraints are mutually exclusive, and each is weighted by a preference value: the set is constructed in order to maximize the degree of satisfaction of the DTP constraint. The second solution we propose is, instead, obtained by extending previous work that dealt with constant preference functions only [13], and reduces each soft DTP constraint to a set of disjunction of constraints, and a non-trivial interplay among their preference values to maximize, as before, the preference value of the DTP constraint. Then, we prove that these translations are correct. In order to test the effectiveness of our proposals, we have randomly generated DTPPs, following the method originally developed by Peintner and Pollack [11], and then employed in all other papers on DTPPs. Moreover, we have also generated non-random benchmarks starting from Job Shop Scheduling problems already employed in [13]. In our framework, each translated problem is represented as a Satisfiability Modulo Theory (SMT) formula in the QF_RDL or QF_IDL logics (depending on the domain of interpretation of variables) plus optimization, and the YICES and MATHSAT SMT solvers, that are able to deal with these logics and optimization issues, are employed. An experimental analysis conducted on a wide set of benchmarks, using also the same benchmarks setting already employed in past papers, shows that our second proposal is competitive to, and can be faster than, MAXILITIS. Moreover, the experiments further show that YICES performs better then MATHSAT on these benchmarks.

To sum up, the main contributions of this paper are the following:

- We design new approaches for solving DTPPs that employ translations to Maximum Satisfiability of a set of Boolean combination of constraints of the form $l \bowtie x - y \bowtie u$, $\bowtie \in \{<, \leq\}$.

- We prove that the translations are correct.
- We implement the translations, by expressing the resulting formulas in the SMT logics QF_RDL or QF_IDL plus optimization.
- We run a wide experimental analysis that shows that one of our encoding, employing state-of-the-art SMT solvers, can be faster than MAXILITIS.

The rest of the paper is structured as follows. Section 2 introduces preliminaries about DTPs, DTPPs and Maximum Satisfiability. Then, in Section 3 we present our translations from DTPPs to Maximum Satisfiability of Boolean combination of constraints. In Section 4 we prove correctness and completeness of the approaches, while the experimental analysis is reported in Section 5. The paper ends by providing a discussion about related work in Section 6 and some conclusions in Section 7.

2 Formal Background

Problems involving disjunction of temporal constraints have been introduced by Stergious and Koubarakis [14], as an extension of the Simple Temporal Problem (STP) [2], which consists of finite conjunction of constraints. The problem was referred for the first time as Disjunctive Temporal Problem (DTP) by Armando et. al [15], and is presented in the first subsection. The remaining subsections introduce DTPPs and Maximum Satisfiability of DTPs.

2.1 DTP

Let $\mathcal{V}$ be a set of *variables*. A *constraint* is an expression of the form $l \leq x - y \leq u$, where $x, y \in \mathcal{V}$, and l, u are numeric constants. A *DTP constraint* is a finite disjunction of constraints. A *DTP formula*, or simply *formula*, is a finite conjunction of DTP constraints. A DTP formula (resp. DTP constraint) can be equivalently seen as a conjunctively (resp. disjunctively) intended set of DTP constraints (resp. constraints).

The semantics of DTP formulas is defined as follows. Let the *domain of interpretation* $\mathcal{D}$ be either the set of the real numbers $\mathcal{R}$ or the set of integers $\mathcal{Z}$. An *assignment* σ is a total function mapping variables to $\mathcal{D}$; $\sigma \models \phi$, i.e. σ satisfies *a formula* ϕ, is defined as follows

- $\sigma \models l \leq x - y \leq u$ if and only if $l \leq \sigma(x) - \sigma(y) \leq u$;
- $\sigma \models \neg\phi$ if and only if it is not the case that $\sigma \models \phi$;
- $\sigma \models (\wedge_{i=1}^{n} \phi_i)$ if and only if for each $i \in [1, n]$, $\sigma \models \phi_i$ $(n \geq 0)$; and
- $\sigma \models (\vee_{i=1}^{n} \phi_i)$ if and only if for some $i \in [1, n]$, $\sigma \models \phi_i$ $(n \geq 0)$.

If $\sigma \models \phi$ then σ is also called a *model*, or a *satisfying assignment* of ϕ. We also say that a formula ϕ is *satisfiable* if and only if there exists a model for ϕ. The DTP is the problem of deciding whether a formula ϕ is satisfiable or not in the given domain of interpretation $\mathcal{D}$.

Example 1. The following formula, where $\mathcal{D}$ is $\mathcal{Z}$

$$(5 \leq x - y \leq 7 \vee -30 \leq z - x \leq -20) \wedge (5 \leq z - y \leq 10)$$

is satisfiable, and a model σ for it assigns, e.g. $x = 8$, $y = 2$ and $z = 10$.

Notice that the satisfiability of a formula may depend on $\mathcal{D}$, e.g. the formula

$$x - y > 0 \wedge x - y < 1$$

is satisfiable if $\mathcal{D}$ is $\mathcal{R}$ but unsatisfiable if $\mathcal{D}$ is $\mathcal{Z}$. However, the problems of checking satisfiability in $\mathcal{Z}$ and in $\mathcal{R}$ are closely related and will be treated uniformly in the following.

2.2 DTPP

We now define our problem of interest, the DTPP. We first extend the concept of DTP constraint, considering that a DTP constraint can be either *hard*, i.e. its satisfaction is mandatory, or *soft*, i.e. its satisfaction is not necessary but preferred, and in case of satisfaction it contributes to the generation of high quality solutions according to the aggregation methods employed and defined later.

A DTPP is defined as a pair $\langle \phi, w_{pc} \rangle$, where

- $\phi := \langle \phi_h, \phi_s \rangle$ is a DTP formula partitioned into a set of hard DTP constraints (denoted ϕ_h) and a set of soft DTP constraints (denoted ϕ_s), and
- w_{pc} is a function that maps the constraints appearing in soft DTP constraints in ϕ_s to piece-wise constant preference functions.

As for the semantics, we start by defining how weights, corresponding to values in the preference functions, are aggregated within a soft DTP constraint d to define the weight of d. In our work, we consider a prominent semantic for this purpose: the *max* semantic. Consider a constraint $dc := l \leq x - y \leq u$, its preference function $w_{pc}(dc)$ is a piece-wise constant function that can be specified by

- partitioning the interval [l,u] into a finite set of convex interval $I_1, \ldots, I_n \subseteq [l, u]$ $(n \geq 1)$, called *preference intervals* of dc, and
- associating a positive integer (the preference value) to each interval I_i, $1 \leq i \leq n$.

The *max* semantic $[9, 6]$ defines the weight $w_c(d)$ of a satisfied soft DTP constraint d as the maximum among the possible preference values of satisfied constraints in d, i.e. given an assignment σ'

$$w_c(d) = max\{w_{pc}(\sigma'(x) - \sigma'(y)) : dc \in d, \sigma' \models dc\}.$$

The semantics of the whole DTPP is based on an utilitarian method for aggregating soft DTP constraints weights. More formally, given a function w_c that maps each soft DTP constraint in ϕ_s to a positive integer number, the goal is to find an assignment σ' for ϕ that

- satisfies ϕ_h, and
- maximizes the sum of weights associated to the satisfied soft DTP constraints in ϕ_s, i.e. maximizes the following linear objective function

$$f = \sum_{d \in \phi_s, \sigma' \models d} w_c(d). \tag{1}$$

Example 2. Consider a simple formula consisting of one soft DTP constraint $d := dc_1 \vee dc_2$, where $dc_1 : 1 \leq x - y \leq 10$ and $dc_2 : 5 \leq z - q \leq 15$. The piece-wise constant preference function associated to dc_1 is

$$w_{pc}(dc_1) = \begin{cases} 1 & 1 \leq x - y \leq 3 \\ 2 & 3 < x - y \leq 7 \\ 1 & 7 < x - y \leq 10 \end{cases} \tag{2}$$

while, regarding dc_2, its preference function is

$$w_{pc}(dc_2) = \begin{cases} 2 & 5 \leq z - q \leq 8 \\ 4 & 8 < z - q \leq 10 \\ 2 & 10 < z - q \leq 15 \end{cases} \tag{3}$$

Of course both difference constraints can be satisfied at the highest preference value, e.g. consider a model σ' that assigns $x = 30$, $y = 25$, $z = 10$ and $q = 1$, the optimal value $w_c(d)$ for the satisfaction of the only soft DTP constraint d in the formula is 4.

2.3 Max-DTP

The idea of our paper is to translate DTPPs to Maximum Satisfiability of formulas composed by hard and soft DTP constraints. The translation requires the extension of the syntax and semantics of DTP formulas in order to allow for arbitrary Boolean combination of constraints allowing also for strict inequalities.

An *Arbitrary* DTP constraint, denoted DTP^A, is a Boolean combination of constraints of the form $l \bowtie x - y \bowtie u$, $\bowtie \in \{<, \leq\}$, and a DTP^A formula $\phi' = \langle \phi_h, \phi'_s \rangle$ consists of two sets ϕ_h and ϕ'_s of hard and arbitrary soft DTP^A constraints, respectively.

The Partial Weighted Maximum Satisfiability problem of a DTP^A formula is formally defined as a pair $\langle \phi', w_c \rangle$. In this case, the goal is to find a satisfying assignment σ' to the variables in ϕ' that

- satisfies ϕ_h, and
- maximizes the sum of the weights associated to satisfied soft DTP^A constraints in ϕ'_s, i.e. maximizes a linear objective function with the form (1).

In the following, for simplicity we will use Max-DTP to refer to the Partial Weighted Maximum Satisfiability problem of mixed (hard) DTP and (soft) DTP^A constraints as defined above.

Example 3. The following formula ϕ, where $\mathcal{D}$ is $\mathcal{Z}$

$$d_1 : (x - y \leq 7 \vee z - x \leq -20) \wedge$$
$$d_2 : x - y \geq 10 \wedge$$
$$d_3 : z - x \geq 0$$

is not satisfiable if each constraint is hard.

If the DTP constraints are, instead, soft with $w_c(d_1)$=3, $w_c(d_2)$=1 and $w_c(d_3)$=1, σ of Example 1 is an optimal solution for ϕ as well as, e.g. σ' that assigns $x = 30$, $y = 2$ and $z = 10$, given that for both assignments the corresponding value of f is 4.

3 Translating DTPPs to Max-DTPs

As we said before, our main idea is to reduce the problem of solving DTPPs to solving Max-DTPs, for which we can rely on efficient solvers, e.g. SMT solvers. Hard DTP constraints remain unchanged in our translation, while soft DTP constraints need special treatment. In the following, given an interval $I = [l, u]$, we write $x - y \in I$ as a shorthand for the constraint $l \leq x - y \leq u$ (and similarly for the related open, left-open and right-open intervals) with preference function w_{pc}, and preference intervals $I_1^{dc}, \ldots, I_n^{dc}$ $(n \geq 1)$.

Given a soft DTP constraint d, we partition each constraint dc of the form $l \leq x - y \leq u$ in d into a set of maximal sub-intervals having the same preference value. More formally, let l_{dc} be the number of different preferences values $v_1 \ldots v_{l_{dc}}$ appearing in the preference function of dc, we partition dc into l_{dc} sets defined through the following function

$$f(dc, v) = \{x - y \in I_i^{dc} : w_{pc}(I_i^{dc}) = v, 1 \leq i \leq n\}.$$

If there is only one preference interval, i.e. the preference function is a constant v', only one pair $f(dc, v')$ is defined consisting of the interval $[l, u]$, i.e. it represents the constraint $l \leq x - y \leq u$, and its preference value is v'.

We now need to "aggregate" the preference values corresponding to different levels of the piece-wise constant functions in the various constraints in order to implement our translation. The idea is to "merge" sets $f(dc, v)$ in the same soft DTP constraint; intuitively, this means that, if the candidate solution satisfies at least one of the constraints in $f(dc, v)$, then a possible preference value for d is v.

More formally, consider a soft DTP constraint

$$d := dc_1 \vee \ldots \vee dc_k. \tag{4}$$

Let $v_1 \ldots v_{l_d}$ be the different preference values appearing in the preference functions of d (of course, $l_d \geq l_{dc_i}, 1 \leq i \leq k$). Then, d and its preference functions are represented by l_d sets defined by the following function

$$g(d, v_j) = \bigcup_{dc \in d} f(dc, v_j) \tag{5}$$

$1 \leq j \leq l_d$.

In the remaining part of the section we present, in separate subsections, the two encodings that we considered. The first corresponds to a very natural modeling of the problem, while the second extends previous work that dealt with constant preference functions only [13].

For simplicity, in the following if we write $g(d, v_i)$ and $g(d, v_j)$ with $i < j$, we assume $v_i < v_j$.

3.1 The first translation

The first solution we considered for our encoding is to express a soft DTP constraint d using soft DTP^A constraints that force the highest preference value associated to satisfied constraints in d to be assigned as weight for d.

A soft DTP constraint d and its preference value are expressed by a set of l_d soft DTP^A constraints: for each $z = 1 \ldots l_d$

$$c_z(d) : \wedge_{i=1}^{z-1} \neg(\vee_{p \in g(d, v_i)}\ p) \wedge (\vee_{p \in g(d, v_z)}\ p) \tag{6}$$

where p is an interval, and

$$w_c(c_z(d)) = v_z \tag{7}$$

is the weight associated to $c_z(d)$.

The set of constraints is mutually exclusive: considering an assignment, at most one of the constraints in the set is satisfied, and the relative weight is assigned to d. If a constraint in (6) is satisfied, this is the constraint leading to the maximum value (according to the candidate solution considered), whose weight is defined in (7).

Example 4. Consider the soft DTP constraint of Example 2. $w_{pc}(dc_1)$ is represented with

$$f(dc_1, 1) = \{1 \leq x - y \leq 3, 7 < x - y \leq 10\}$$

$$f(dc_1, 2) = \{3 < x - y \leq 7\}.$$

Regarding dc_2, its preference function is represented with

$$f(dc_2, 2) = \{5 \leq z - q \leq 8, 10 < z - q \leq 15\}$$

$$f(dc_2, 4) = \{8 < z - q \leq 10\}.$$

We now "merge" the sets (on the three existing levels), whose result is

$$g(d,1) = \{1 \leq x - y \leq 3, 7 < x - y \leq 10\}$$

$$g(d,2) = \{3 < x - y \leq 7, 5 \leq z - q \leq 8, 10 < z - q \leq 15\}$$

$$g(d,4) = \{8 < z - q \leq 10\}.$$

Following (6), the reduction is

$$c_1(d) : 8 < z - q \leq 10$$

$$c_2(d) : \neg c_1 \wedge ((3 < x - y \leq 7) \vee (5 \leq z - q \leq 8) \vee (10 < z - q \leq 15))$$

$$c_3(d) : \neg c_1 \wedge \neg c_2 \wedge (1 \leq x - y \leq 3 \vee 7 < x - y \leq 10)$$

with

$$w_c(c_1(d)) = 4,$$

$$w_c(c_2(d)) = 2,$$

$$w_c(c_3(d)) = 1.$$

3.2 The second translation

A second translation transforms each soft DTP constraint d to a set of l_d soft DTP^A constraints as follows: for each $z = 1 \dots l_d$

$$c'_z(d) : \vee_{i=z-1}^{z} \vee_{p \in g(d,v_i)} p \tag{8}$$

(we assume that $g(d, v_0)$ is empty). The problem is now to define what are the weights associated to each newly defined soft DTP^A constraint, in order to reflect the semantic of our problem. In the previous translation (4), the DTP^A constraints were mutually exclusive; now, instead, the dependencies between them influence constraints weights adaptation and definition. Our solution starts from the following fact: if the constraint c'_{l_d} is satisfied, it is safe to consider that it contributes for at least the minimum preference value v_{l_d}, i.e. the one associated to the set $g(d, v_{l_d})$, from which c'_{l_d} is constructed. Satisfying the constraint c'_{l_d-1} further contributes for $v_{l_d-1} - v_{l_d}$, and given that a constraint c'_z implies all constraints $c'_{z'}$, $z' > z$, these two soft DTP^A constraints together contribute for v_{l_d-1}. This method is recursively applied up to the set of constraints in $g(d, v_1)$, i.e. c'_1, whose preference value is $v_1 - v_2$ and, given that c'_1 implies all other introduced soft DTP^A constraints, satisfying c'_1 correctly corresponds to assign a weight v_1 to d.

More formally, for each $z = 1 \dots l_d$

$$w_c(c'_z(d)) = \begin{cases} v_{l_d} & z = l_d \\ v_z - v_{z+1} & 1 \leq z < l_d \end{cases} \tag{9}$$

and, given an assignment σ, $w_c(d) = \sum_{z \in \{1,\dots,l_d\}, \sigma' \models c'_z} v_z$.

Example 5. Concerning the second translation, the soft DTP^A constraints that express the constraint d with the preference functions in Example 4 are

$$c'_1(d) : 8 < z - q \leq 10$$

$$c'_2(d) : c'_1 \vee (3 < x - y \leq 7 \vee 5 \leq z - q \leq 8 \vee 10 < z - q \leq 15)$$

$$c'_3(d) : c'_2 \vee (1 \leq x - y \leq 3 \vee 7 < x - y \leq 10)$$

where $w_c(c'_1(d)) = 2$, $w_c(c'_2(d)) = 1$ and $w_c(c'_3(d)) = 1$.

Let us now define the whole translation: given a DTPP $\langle \phi, w_{pc} \rangle$, with $\phi := \langle \phi_h, \phi_s \rangle$, and let $\text{REDUCT}(d, w_{pc})$ being the translation of a single soft DTP constraint d presented in (6) (called REDUCT_1 in the following), with weights definition in (7), or (8) (called REDUCT_2 in the following), with weights definition in (9), the resulting Max-DTP formula has

- ϕ_h as hard formula,
- $\bigcup_{d \in \phi_s} \text{REDUCT}(d, w_{pc})$ as soft DTP^A formula, and
- w_c defined as in (7) or (9).

Such translation works correctly if, considering a formula ϕ, no repeated DTP^A constraints appear in the translated formula ϕ'. If this happens, intuitively, we want each single occurrence in ϕ' to count "separately", given that they take into account different contributions from different soft DTP constraints in ϕ. A solution is to consider a *single occurrence* of the resulting soft DTP^A constraint in ϕ' whose weight is the sum of the weights of the various occurrences.

4 Correctness and completeness of the reductions

This section deals with correctness and completeness of the introduced reductions, i.e. the original DTPP formula $\langle \langle \phi_h, \phi_s \rangle, w_{pc} \rangle$, and the reduced DTP^A formula have the same solution space of "optimal" assignments.

We first show that the underlying DTPs $\phi := \phi_h \cup \phi_s$ and $\phi' := \phi_h \cup \bigcup_{d \in \phi_s} reduct(d, w_{pc})$ have the same satisfying assignments[4], i.e. that this holds for ϕ_s and $\bigcup_{d \in \phi_s} reduct(d, w_{pc})$, given that ϕ_h remains unchanged during both reductions.

We assume that no repeated soft DTP^A constraints are in the reduced formula: with this hypothesis, it is enough to prove that the property holds for a single soft DTP constraint d.

At first we deal with the reduction in Section 3.1.

Proposition 1. *Given an assignment σ, σ satisfies d iff σ satisfies* $\text{REDUCT}_1(d, w_{pc})$.

[4] Note that in the case of the second reduction this corresponds to a model, while for the first reduction, where the constraints are mutually exclusive, this is according to the semantic of a Max-SAT solution.

Proof. To prove the thesis, we need to show that an assignment σ that satisfies d also satisfies $\textsc{reduct}_1(d, w_{pc})$, an vice-versa.

(left-to-right) If σ satisfies d, this means that at least a constraint $dc \in d$ is satisfied. Consider now the dc which is satisfied at the highest preference value by σ. We know by construction that dc can occur in more DTP^A constraints of $\textsc{reduct}_1(d, w_{pc})$, in this case divided in preference intervals. We are guaranteed that at least one of its preference intervals satisfies a DTP^A constraint in $\textsc{reduct}_1(d, w_{pc})$: in fact, if it satisfies $c_1(d)$, then the thesis holds, otherwise this means that a preference interval at lower preference value is satisfied, and we know that it satisfies the respective DTP^A constraint $c_i(d)$.

(right-to-left) If σ satisfies $\textsc{reduct}_1(d, w_{pc})$, this means that exactly one DTP^A constraint in $\textsc{reduct}_1(d, w_{pc})$ is satisfied. Such DTP^A constraint is satisfied because of a preference interval of a constraint dc in d, and thus σ satisfies also d.

We now state that, given a satisfying assignment of the underlying DTPs of the two formulas, the two optimal solutions have the same values.

Proposition 2. *Given a satisfying assignment σ of ϕ and ϕ', for each $d \in \phi_s$*

$$w_c(d) = \sum_{\sigma\ satisfies\ c_i(d), c_i(d) \in reduct_1(d, w_{pc}), 1 \leq i \leq z} w_c(c_i(d))$$

Proof. From Proposition 1 we also know that if $c_i(d)$ is satisfied, by construction no DTP^A constraint $c_j(d)$ in $\textsc{reduct}_1(d, w_{pc})$ with both $j < i$ or $j > i$ is satisfied, thus the thesis follows immediately.

Thus, one DTP^A constraint in $\textsc{reduct}_1(d, w_{pc})$ is satisfied, and corresponds to the DTP^A constraint having the maximum possible preference value, which corresponds to the semantic of our problem.

Now, we deal with the reduction in Section 3.2.

Proposition 3. *Given an assignment σ, σ satisfies d iff σ satisfies* $\textsc{reduct}_2(d, w_{pc})$.

Proof. To prove the thesis, we need to show that an assignment σ that satisfies d also satisfies $\textsc{reduct}_2(d, w_{pc})$, an vice-versa.

(left-to-right) If σ satisfies d, this means that at least one constraint $dc \in d$ is satisfied. From (8), we know by construction that dc will occur, possibly divided into its preference intervals, in $\textsc{reduct}_2(d, w_{pc})$; at least one of its preference interval is satisfied, thus also $\textsc{reduct}_2(d, w_{pc})$ is satisfied by σ.

(right-to-left) If σ satisfies $\textsc{reduct}_2(d, w_{pc})$, this means that at least one DTP^A constraint in (8) is satisfied, thus at least one preference interval in it is satisfied. Take this preference interval, by construction we know that it is (part of) a constraint occurring in d, thus also d is satisfied by σ.

Proposition 4. *Given a satisfying assignment σ of ϕ and ϕ', for each $d \in \phi_s$*

$$w_c(d) = \sum_{\sigma \ satisfies \ c_i(d), c_i(d) \in reduct_2(d, w_{pc}), 1 \leq i \leq z} w_c(c_i(d))$$

This proposition follows from the construction of the encoding, and from (9).

5 Experimental Analysis

In this section we present benchmarks and solvers involved in our analysis, as well as the results of our experiments.

5.1 Benchmarks

Randomly generated benchmarks. These benchmarks aim at comparing the considered solvers on two dimensions, namely (i) the size of the benchmarks, and (ii) the number of preference levels in the piece-wise constant preference function, all used in past papers on DTPPs – see, e.g. [6].

In order to generate the benchmarks, the main parameters considered are:

1. the number k of disjuncts per DTP constraint;
2. the number n of arithmetic variables;
3. the number m of DTP constraints;
4. the number l of levels in the preference functions.

Furthermore, we also investigated the performance of the solvers considering two different settings related to the preference values of the preference functions. The preference functions considered are semi-convex piece-wise constant: starting from the lower and upper bounds of the constraints, intervals corresponding to higher preference levels are randomly put within the interval of the immediate lower level, with a reduction factor, up to an highest level. For details see, e.g. [6].

In particular, we consider

– a setting where, given the i-th level l, $w(l) = i$, i.e. the setting used by Moffitt for evaluating Maxilitis [6] ("Model A" in the following);
– a setting where $w(l)$ is randomly generated in the range [1, 100], still ensuring to maintain the same shape for preference functions ("Model B" in the following).

Finally,

– the domain of interpretation for all benchmarks is $\mathcal{Z}$, given that Maxilitis can not deal with real numbers, and

– all generated DTP constraints are soft, i.e. experiments are focused on this challenge setting.

For each tuple of values of the parameters, 25 instances have been generated.

Concerning the first dimension, we randomly generated benchmarks by varying the total amount of DTP constraints, with the following parameters: k={2, 3}, $m \in \{10, \ldots, 80\}$, n=0.8 × m, l=5, lower and upper bounds of each constraint taken in $[-50, 100]$.[5]

Regarding the second dimension, we randomly generated benchmarks by varying the number of levels l in the preference functions in the interval $[2, \ldots, 8]$. The remaining parameters has been set as follows. In the case of $k = 2$, n and m have been set to 24 and 30, respectively. In the case of $k = 3$, n is equal to 32, while $m = 40$. Finally, lower and upper bounds of each constraint is taken again in $[-50, 100]$.

Real-world benchmarks. For these kind of benchmarks we analyze the Job Shop Scheduling problems already employed for DTPPs in [13]. The benchmarks evaluated in [13] are composed of 10 groups, each made of 4 problems, whose preference functions are constant.

We have considered one problem for the smallest group, having $n = 5$, $m = 10$, and $k \in \{1, 2\}$ as parameters, and generated instances having piece-wise constant preference functions with Model A and B. We varied the number l of levels in the interval $[2, \ldots, 5]$, and generated 10 instances for each setting, for a total of 80 instances. Problems in the other groups have very challenge parameters such as k up to 4 and m up to 500, and result in very big and hard DTPPs.

5.2 Solvers evaluated

We have implemented our translations and expressed the resulting formulas as SMT formulas with optimization. We called our system DTPP2MaxSMT, and it can be coupled, in principle, with any MaxSMT solver as back-engine. In particular, we evaluated its performance involving two state-of-the-art MaxSMT solvers, namely MathSAT (ver. 5.2.11) [16, 17] and Yices (ver. 1.0.38) [18, 19]. In the following, we will refer to the systems DTPP2MaxSMT+MathSAT and DTPP2MaxSMT+Yices with dtppMathSAT and dtppYices, respectively.

In the experimental analysis, we compare the systems mentioned above with the solver Maxilitis, an implementation by Moffitt of the approach presented in [6]. To the best of our knowledge, Maxilitis is the state-of-the-art system for solving DTPPs, and it subsumes other previous systems, such as ario [8] and gapd [20]. Maxilitis works as follows: a DTPP is represented as a constraints system named Valued DTP (VDTP), that can express the same solution space of the DTPP. The VDTP is then solved by generalizing meta-CSP approach employed by Epilitis for DTP solving. Known optimization techniques

[5] These benchmarks have been generated using the program provided by Michael D. Moffitt, author of Maxilitis.

in DTP solving, i.e. removal of subsumed variables and semantic branching, are also lifted to VDTP in order to reduce the explored search space. We included in our analysis two variants of MAXILITIS (as provided by its author), namely MAXILITIS-IW and MAXILITIS-BB. MAXILITIS-IW (IW standing for Iterative Weakening) searches for solutions with a progressively increasing number of violated constraints; MAXILITIS-BB uses, instead, branch-and-bound for reaching the optimal solution.

The executable of our solver, together with the benchmarks analyzed, can be found at

`http://www.star.dist.unige.it/~marco/DTPPYices/.`

5.3 Experimental results

The experiments described in this subsection ran on PCs equipped with a processor Intel Core i5 at 3.20 GHz, with 4 GB of RAM, and running GNU Linux Ubuntu 12.04. The timeout for each instance has been set to 300s. All instances have been evaluated considering integer-valued variables.[6]

Randomly generated benchmarks. We first preliminary tested the two translations on the smallest benchmarks generated with Model A. For instance, considering the performance of DTPPYICES (that will prove to be our best option) on benchmarks generated on dimension (i), we notice that, by employing the first translation, it was able to solve only the ones with $m = 10$. In particular, DTPPYICES solves all 25 instances generated with Model A in (cumulative time of) 141.92 seconds, while it solves only 14 out of 25 instances generated with Model B in 1749.28 seconds. At the same time, DTPPYICES on the second translation solves all 50 instances in 0.1 seconds and 0.26 seconds, respectively.

Thus, the first translation looks impractical from a performance point of view. For this reason, in the following we report only the results related to the second translation.

Considering the second translation, the results obtained in the experiments for $k = 2$ are shown in Figure 1, which is organized as follows. Concerning the top-most plots, in the x-axis we show the total amount of DTP constraints, while in the plots in the bottom, the total amount of levels of the piece-wise constant preference function is reported. In the y-axis (in log scale), it is shown the related median CPU time (in seconds). MAXILITIS-BB's performance is depicted by a dotted line with blue triangles, MAXILITIS-IW's by using a dashed line with orange upside down triangles, while DTPPMATHSAT's performance is depicted by a dashed green line with boxes; finally, DTPPYICES performance is denoted

[6] We have tested our solvers on the biggest formulas we could solve but employing real-valued variables, and results are very similar to those when variables are integers.

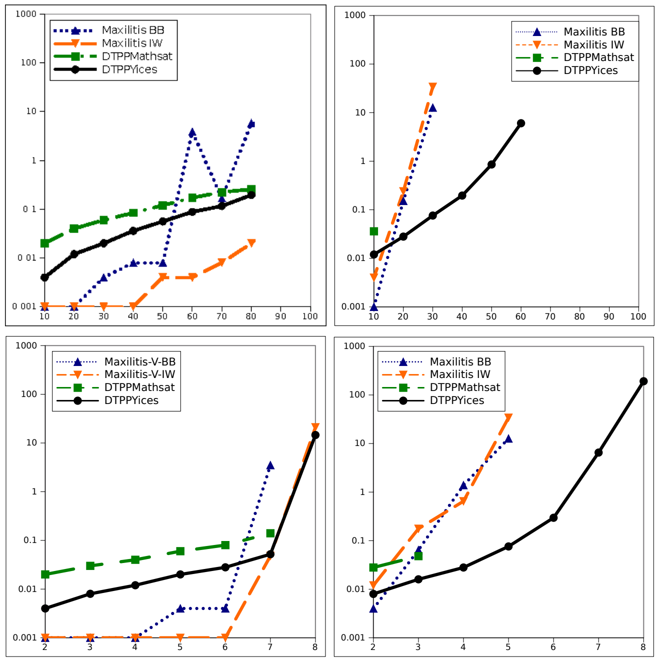

Fig. 1. Results of the evaluated solvers on random DTPPs with $k = 2$ considering the size of the benchmarks (top-most plots) and number of preference levels (bottom). Left-most plots are related to Model A, while right-most plots depict the results related to Model B.

by a solid line with black circles. Plots in the left-most column are related to Model A, while plots in the right-most column are related to Model B.

Looking at Figure 1, and considering the top-left plot, we can see that MAXILITIS-IW is the solver with the best performance, and it is one order of magnitude of CPU time faster than DTPPYICES and DTPPMATHSAT. Considering the same analysis in the case of Model B, we can see (top-right plot) that the picture changes in a noticeable way. Benchmarks are harder than previously: MAXILITIS-BB and MAXILITIS-IW are not able to efficiently cope with benchmarks with $m > 30$, while DTPPMATHSAT stops at $m = 10$. In this case, DTPPYICES is the best solver, and we report that it is able to deal with benchmarks up to $m = 60$.

Looking at the bottom-left plot of Figure 1, we can see that MAXILITIS-IW is the best solver up to $l = 7$, while for $l = 8$, we report that DTPPYICES is slightly faster. Also in this case MAXILITIS-BB does not efficiently deal with the most difficult benchmarks in the suite. Looking now at the plot in the bottom-right, we can see that the performance of both versions of MAXILITIS are very similar, while DTPPYICES is the fastest solver: the median CPU time of both

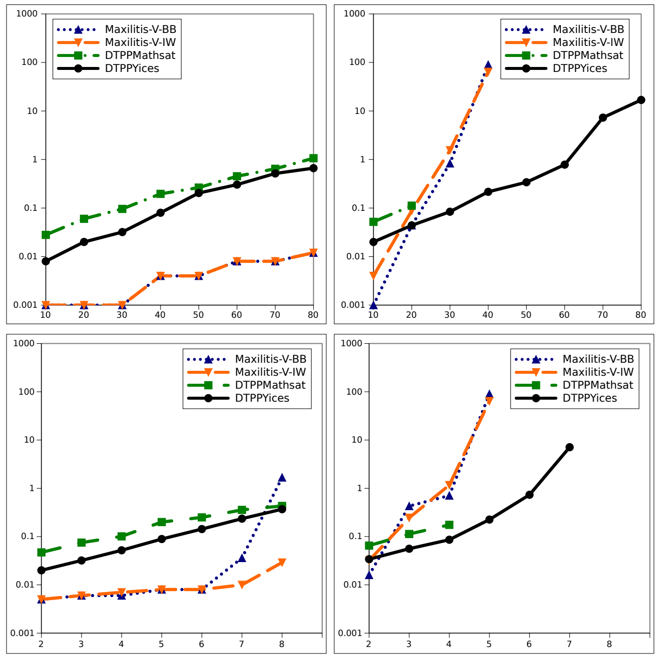

Fig. 2. Results of the evaluated solvers on random DTPPs with $k = 3$ considering the size of the benchmarks (top-most plots) and number of preference levels (bottom). As in Figure 1, left-most plots are related to Model A, while right-most plots depict the results related to Model B.

Maxilitis-BB and Maxilitis-IW runs in timeout for $l > 5$, while dtppYices solves the majority of the benchmarks within the time limit for all levels.

Detailed results related to the plots in Figure 1 are reported in the Appendix, cf. Tables 3 and 4.

Considering the results related to $k = 3$, looking at Figure 2 (which has the same organization of Figure 1), top-left plot, we report for both versions of Maxilitis the very same performance, and they are one order of magnitude faster than both dtppMathSAT and dtppYices. Concerning DTPPs generated with Model B, by looking at the top-right plot of Figure 2, we report that the best solver turned out to be dtppYices, while both versions of Maxilitis stop at $m = 40$, while dtppMathSAT stops at $m = 20$.

Finally, concerning the analysis on preference levels, we can report a picture similar to the one related to $k = 2$. Both versions of Maxilitis outperform both dtppMathSAT and dtppYices on benchmarks generated with Model A (with the exception of Maxilitis-BB for $l = 8$), while dtppYices is by far the best solver on benchmarks generated with Model B (with the exception of the smallest instances having $l = 2$).

Detailed results related to the plots in Figure 2 are reported in the Appendix, cf. Tables 5 and 6.

Real-world benchmarks. Table 1 reports the results of the Job Shop Scheduling problems enhanced with preference functions generated with Model A and B. The table is structured as follow. The first column gives information about the benchmark, where jobshop_lN means the selected problem whose preference functions have N levels. The second column is the number of instances generated, while the third and fourth columns report the results for DTPPMATHSAT and DTPPYICES, respectively. The last two columns are then divided into two sub-columns reporting results for the two generation models, each sub-column being further divided into number of instances solved and cumulative time for solved instances, respectively. MAXILITIS is not included in this analysis given it returns some incorrect answers.

Results confirm the trends observed so far: DTPPYICES is much faster than DTPPMATHSAT, solving all 80 instances; formulas generated with Model B are much more difficult for MATHSAT, while all instances are relatively easy for YICES; and performance decreases while the number of level increases.

Benchmark	**N**	DTPPMATHSAT				DTPPYICES			
		Mod. A		Mod. B		Mod. A		Mod. B	
		#	Time	#	Time	#	Time	#	Time
jobshop_l2	10	8	0.18	3	0.53	10	0.01	10	0.01
jobshop_l3	10	10	3.26	–	–	10	0.04	10	0.05
jobshop_l4	10	6	43.68	–	–	10	0.14	10	0.12
jobshop_l5	10	–	–	–	–	10	0.38	10	0.25

Table 1. Results of DTPPYICES and DTPPMATHSAT on Job Shop Scheduling problems.

5.4 Discussion

In this subsection we give insights in order to more deeply understand the results we have shown in the previous subsection. To this aim, we often employ number of conflicts encoutered during search as a CPU-time independent measure for measuring the workload of a solver.[7] The concept of "conflict" is central in every part of the search, e.g. backtracking, learning, and decision, for a CDCL solver. Our analysis follows a number of direction, devoting one paragraph to each.

First vs. second translation. We focused on the setting where the two translations solve the same set of istances, i.e. the 25 instances built under dimention (i) with $m = 10$, Model A. In the following table we report, for both translations, the 5 numbers (minimum, first quartile, median, third quartile, maximum) of the number of conflicts (DTPP)YICES encoutered during search.

[7] We consider number of conflicts YICES outputs by running it in verbose mode; MATHSAT does not look to output such number.

Translation	Min	Q1	Median	Q3	Max
First	96619	113809	175417	184762	241149
Second	0	3	6	14	58

Table 2. Five-number summary for the number of conflicts of DTPPYICES on formulas arising from the two translations.

As it is clear from Table 2, with the first translation DTPPYICES encounters a number of conflicts bigger by orders of magnitude in comparison to those encountered with the second translation.

A property of the second translation that can contribute to these results is the following: by construction, as soon as a DTP^A constraint gets satisfied, say, c'_z, all DTP^A constraints $c'_{z'}$ coming from the translation of the same soft DTP constraint, and such that $z' > z$ wrt (8), are satisfied as well.

Results on different random models. Model B brings in general to harder formulas for all solvers, given that the weights are not uniform. As far as the specific performance of solvers is concerned, on random benchmarks MAXILITIS in general performs better than our solution on Model A, while this is the opposite on Model B. Unfortunately, we could not rely on CPU time independent measures for MAXILITIS given that it does not output measures of this kind (other than the cost). A possible explanation for this behavior is that Maxilitis is likely to be optimized on formulas generated with Model A, given that this is the only type of formulas analyzed in its paper [6].

Instead, in order to corroborate the results on Model A vs. Model B in our setting, we did a similar analysis to the paragraph above, by comparing the distributions of number of conflicts, employing the same setting. Results are now showed using a boxplot depicted in Figure 3. In this case we can note that the number of conflicts on benchmarks generated with Model B is much higher than the same number of those generated with Model A.

YICES *vs.* MATHSAT*.* We saw that YICES consistently outperforms MATHSAT on our formulas. Our results are consistent with the state of the art of the competition, e.g. Yices won the SMT Competition from 2014 to 2017 on the logics QF_RDL and QF_IDL, which are the basis for our formulas (see the webpage of the (last) SMT Competition at `http://smtcomp.sourceforge.net`).

6 Related Work

We have seen in the introduction that DTPPs have been used in applications. We briefly describe here some of these applications. In [5], a preference model designed to capture user scheduling preferences for over-constrained meeting requests between multiple people has been presented. Solving is done by a constrained scheduling problem with preferences, which is modeled as a DTPP. In [4], instead, DTPP is used in the context of one component of an automatic

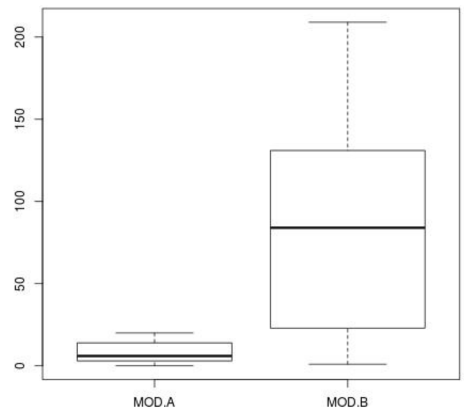

Fig. 3. Distributions of the number of conflicts of DTPPYICES on benchmarks generated with different models.

personal assistant, the Personalized Time Manager (PTIME), i.e. the PTIME constraint reasoner, to reason on temporal constraints and preferences that may arise in this context. The usage of Artificial Intelligent techniques, including DTPs, in the context of intelligent technology for assisting elders with cognitive decline with an Autominder's Plan Manager is overviewed by Pollack [3], where it is described the further advatanges that DTPPs bring. This is connected to the previous mentioned application, but with a specific target on elders.

MAXILITIS [6, 9], WEIGHTWATCHER [10] and ARIO [8] implement different approaches for solving DTPPs as defined in [11]. MAXILITIS is a direct extension of the DTP solver EPILITIS [7], while WEIGHTWATCHER uses a similar (as mentioned in, e.g. [10]) approach based on Weighted Constraints Satisfaction problems, but is less efficient. ARIO, instead, relies on an approach based on Mixed Logical Linear Programming (MLLP) problems. In our analysis we have used MAXILITIS because previous results, e.g. in [6], clearly indicate its superior performance.

About the comparison to MAXILITIS, our solution is easy, yet efficient, and has a number of advantages. On the modeling side, it allows to consider (with no modifications) both integer and real variables, while MAXILITIS can deal with integer variables only. Then, in case of unsatisfiable DTPs, our approaches provide a Max-SAT optimal solution, rather than just reporting the problem to be unsatisfiable as done by MAXILITIS. Moreover, our implementation provides an unique framework for solving DTPPs, while the techniques proposed by Moffitt [6] are implemented in two separate versions of MAXILITIS. Finally, our solution is modular, i.e. it is easy to rely on different back-end solvers (or, on a new version of Yices or MathSAT), thus taking advantages on new algorithms and tools for solving our formulas of interest.

7 Conclusions

In this paper we have introduced translation-based approaches for solving DTPPs, that reduce these problems to Maximum Satisfiability of DTPs as defined in the paper. An experimental analysis performed with SMT solvers on randomly generated and real-world DTPPs shows that our approach, in particular when Yices is employed as SMT solver, is competitive to, and sometimes faster than, the specific implementations of the Maxilitis solver. A possible direction for future research is to consider different aggregation functions.

Acknowledgment. The authors would like to thank the reviewers for useful comments and criticisms. They would like to thank also Michael D. Moffitt for providing his solvers and the program for generating random benchmarks, and Bruno Dutertre for his support about Yices.

References

1. Bourguet, J.R., Maratea, M., Pulina, L.: A reduction-based approach for solving disjunctive temporal problems with preferences. In Baldoni, M., Baroglio, C., Boella, G., Micalizio, R., eds.: Proc. of the 13th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2013): Advances in Artificial Intelligence. Volume 8249 of Lecture Notes in Computer Science., Springer (2013) 445–456
2. Dechter, R., Meiri, I., Pearl, J.: Temporal constraint networks. Artificial Intelligence **49**(1-3) (1991) 61–95
3. Pollack, M.E.: Intelligent technology for an aging population: The use of AI to assist elders with cognitive impairment. AI Magazine **26**(2) (2005) 9–24
4. Berry, P.M., Gervasio, M.T., Uribe, T.E., Pollack, M.E., Moffitt, M.D.: A personalized time management assistant: Research directions. In: Persistent Assistants: Living and Working with AI, Papers from the 2005 AAAI Spring Symposium, Technical Report SS-05-05, Stanford, California, USA, AAAI (2005) 1–6
5. Berry, P.M., Gervasio, M.T., Peintner, B., Yorke-Smith, N.: A preference model for over-constrained meeting requests. In: Proc. of the AAAI 2007 Workshop on Preference Handling for Artificial Intelligence. (2007) 7–14
6. Moffitt, M.D.: On the modelling and optimization of preferences in constraint-based temporal reasoning. Artificial Intelligence **175**(7-8) (2011) 1390–1409
7. Tsamardinos, I., Pollack, M.: Efficient solution techniques for disjunctive temporal reasoning problems. Artificial Intelligence **151** (2003) 43–89
8. Sheini, H.M., Peintner, B., Sakallah, K.A., Pollack, M.E.: On solving soft temporal constraints using SAT techniques. In van Beek, P., ed.: Proc. of the 11th International Conference on Principles and Practice of Constraint Programming (CP 2005). Volume 3709 of Lecture Notes in Computer Science., Springer (2005) 607–621

9. Moffitt, M.D., Pollack, M.E.: Partial constraint satisfaction of disjunctive temporal problems. In Russell, I., Markov, Z., eds.: Proc. of the 18th International Conference of the Florida Artificial Intelligence Research Society (FLAIRS 2005), AAAI Press (2005) 715–720
10. Moffitt, M.D., Pollack, M.E.: Temporal preference optimization as weighted constraint satisfaction. In: Proc. of the 21st National Conference on Artificial Intelligence (AAAI 2006), AAAI Press (2006)
11. Peintner, B., Pollack, M.E.: Low-cost addition of preferences to DTPs and TCSPs. In McGuinness, D.L., Ferguson, G., eds.: Proc. of the 19th National Conference on Artificial Intelligence (AAAI 2004), AAAI Press / The MIT Press (2004) 723–728
12. Peintner, B., Moffitt, M.D., Pollack, M.E.: Solving over-constrained disjunctive temporal problems with preferences. In Biundo, S., Myers, K.L., Rajan, K., eds.: Proc. of the 15th International Conference on Automated Planning and Scheduling (ICAPS 2005), AAAI (2005) 202–211
13. Maratea, M., Pulina, L.: Solving disjunctive temporal problems with preferences using maximum satisfiability. AI Commuications **25**(2) (2012) 137–156
14. Stergiou, K., Koubarakis, M.: Backtracking algorithms for disjunctions of temporal constraints. In Shrobe, H.E., Mitchell, T.M., Smith, R.G., eds.: Proc. of the 15th National Conference on Artificial Intelligence (AAAI 1998), AAAI Press / The MIT Press (1998) 248–253
15. Armando, A., Castellini, C., Giunchiglia, E.: SAT-based procedures for temporal reasoning. In Biundo, S., Fox, M., eds.: Proc. of the 5th European Conference on Planning (ICAPS 1999). Volume 1809 of Lecture Notes in Computer Science., Springer (1999) 97–108
16. Cimatti, A., Griggio, A., Schaafsma, B.J., Sebastiani, R.: The MathSAT5 SMT solver. In Piterman, N., Smolka, S.A., eds.: Proc. of the 19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2013). Volume 7795 of Lecture Notes in Computer Science., Springer (2013) 93–107
17. Sebastiani, R., Trentin, P.: OptiMathSAT: A tool for optimization modulo theories. In Kroening, D., Pasareanu, C.S., eds.: Proc. of the 27th International Conference of Computer Aided Verification (CAV 2015). Volume 9206 of Lecture Notes in Computer Science., Springer (2015) 447–454
18. Dutertre, B., Moura, L.D.: A fast linear-arithmetic solver for DPLL (T). In Ball, T., Jones, R.B., eds.: Proc. of the 18th International Conference on Computer Aided Verification (CAV 2006). Volume 4144 of Lecture Notes in Computer Science., Springer (2006) 81–94
19. Dutertre, B., De Moura, L.: The Yices SMT solver. Tool paper at `http://yices.csl.sri.com/tool-paper.pdf` **2**(2) (2006)
20. Peintner, B.M.: Algorithms for constraint-based temporal reasoning with preferences. University of Michigan (2005)

Appendix

n	m	dtppMathSAT				dtppYices			
		Mod. A		Mod. B		Mod. A		Mod. B	
		#	Time	#	Time	#	Time	#	Time
8	10	25	0.55	18	2.68	25	0.10	25	0.26
16	20	25	1.04	9	0.75	25	0.31	**25**	**0.80**
24	30	25	1.58	4	22.63	25	0.50	**25**	**4.45**
32	40	25	2.92	2	64.61	25	0.78	**25**	**23.44**
40	50	25	3.05	1	0.15	25	1.48	**24**	**245.32**
48	60	25	4.90	–	–	**25**	**2.41**	**21**	**403.39**
56	70	25	6.28	–	–	25	3.58	**12**	**1355.01**
64	80	25	8.50	–	–	**25**	**4.75**	**6**	**414.62**
n	**m**	Maxilitis-BB				Maxilitis-IW			
		Mod. A		Mod. B		Mod. A		Mod. B	
		#	Time	#	Time	#	Time	#	Time
8	10	**25**	**0.00**	**25**	**0.04**	**25**	**0.00**	25	0.31
16	20	25	0.17	25	16.28	**25**	**0.01**	25	60.84
24	30	25	5.61	18	588.56	**25**	**0.02**	21	913.16
32	40	24	67.60	3	80.83	**25**	**0.06**	9	905.42
40	50	22	27.53	1	105.45	**25**	**0.29**	3	327.23
48	60	17	247.24	–	–	25	4.24	–	–
56	70	21	57.79	–	–	**25**	**0.71**	–	–
64	80	16	151.82	–	–	25	7.13	–	–

Table 3. Performance of the selected solvers on random DTPPs with $k = 2$ with different sizes. The first columns ("n") reports the total amount of variables for each pool of DTPPs, while the second one ("m") reports the total amount of constraints. It is followed by two groups of columns, and the label is the solver name. Each group is composed of four columns, reporting the total amount of instances solved within the time limit ("#") and the total CPU time in seconds ("Time") spent, in the case of model A and B (groups "Mod. A" and "Mod. B", respectively). In case a solver does not solve any instance, "–" is reported. Finally, best performance are denoted in bold.

l	DTPPMATHSAT				DTPPYICES			
	Mod. A		Mod. B		Mod. A		Mod. B	
	#	Time	#	Time	#	Time	#	Time
2	25	0.54	23	0.64	25	0.10	**25**	**0.20**
3	25	0.81	18	1.01	25	0.20	**25**	**0.38**
4	25	1.14	12	211.14	25	0.32	**25**	**0.87**
5	25	1.58	4	22.61	25	0.50	**25**	**4.44**
6	25	28.54	–	–	**25**	**1.82**	**25**	**17.28**
7	22	238.73	–	–	**23**	**56.05**	**21**	**528.05**
8	10	56.82	–	–	15	228.16	**13**	**487.39**
l	MAXILITIS-BB				MAXILITIS-IW			
	Mod. A		Mod. B		Mod. A		Mod. B	
	#	Time	#	Time	#	Time	#	Time
2	**25**	**0.01**	25	1.68	**25**	**0.01**	25	2.84
3	**25**	**0.01**	25	7.05	**25**	**0.01**	25	47.26
4	**25**	**0.01**	21	395.00	**25**	**0.01**	25	203.52
5	25	5.62	18	589.33	**25**	**0.04**	21	914.45
6	24	32.66	10	673.87	25	4.80	10	608.04
7	21	230.50	2	68.01	23	129.72	2	59.57
8	12	434.55	2	216.89	**17**	**598.42**	2	303.08

Table 4. Performance of the selected solvers on random DTPPs having $k = 2$, with different levels. In column "l" we report the total amount of levels, while the rest of the table is organized similarly to Table 3.

n	m	dtppMathSAT				dtppYices			
		Mod. A		Mod. B		Mod. A		Mod. B	
		#	Time	#	Time	#	Time	#	Time
8	10	25	0.73	23	77.80	25	0.20	25	0.52
16	20	25	1.51	19	158.56	25	0.47	**25**	**1.22**
24	30	25	2.60	9	28.58	25	0.96	**25**	**2.42**
32	40	25	4.85	6	173.39	25	1.95	**25**	**10.21**
40	50	25	9.36	1	189.91	25	6.02	**25**	**54.78**
48	60	24	13.43	2	0.84	25	9.19	**25**	**70.18**
56	70	25	30.75	–	–	25	11.25	**24**	**299.66**
64	80	25	40.41	–	–	25	20.92	**18**	**844.75**
n	**m**	Maxilitis-BB				Maxilitis-IW			
		Mod. A		Mod. B		Mod. A		Mod. B	
		#	Time	#	Time	#	Time	#	Time
8	10	**25**	**0.01**	**25**	**0.04**	**25**	**0.01**	25	0.09
16	20	**25**	**0.01**	25	2.48	**25**	**0.01**	25	5.53
24	30	**25**	**0.02**	25	231.18	25	0.04	25	209.74
32	40	25	0.12	17	838.47	**25**	**0.11**	18	974.13
40	50	25	0.15	6	671.02	**25**	**0.13**	9	703.24
48	60	25	0.28	2	144.48	**25**	**0.15**	3	380.24
56	70	25	0.64	–	–	**25**	**0.24**	1	22.22
64	80	25	0.33	–	–	**25**	**0.30**	–	–

Table 5. Performance of the selected solvers on random DTPPs with $k = 3$ with different sizes. The table is organized as Table 3.

l	dtppMathSAT				dtppYices			
	Mod. A		Mod. B		Mod. A		Mod. B	
	#	Time	#	Time	#	Time	#	Time
2	25	1.18	23	1.54	25	0.52	**25**	**0.83**
3	25	1.91	21	38.65	25	0.87	**25**	**1.40**
4	25	2.74	20	4.35	25	1.32	**25**	**2.21**
5	25	4.97	6	160.81	25	2.18	**25**	**10.51**
6	25	7.19	1	0.30	25	4.06	**25**	**185.18**
7	25	13.85	–	–	25	8.32	**22**	**391.73**
8	25	15.19	–	–	25	11.23	**7**	**352.83**

l	Maxilitis-BB				Maxilitis-IW			
	Mod. A		Mod. B		Mod. A		Mod. B	
	#	Time	#	Time	#	Time	#	Time
2	**25**	**0.13**	25	2.52	**25**	**0.13**	25	8.39
3	**25**	**0.15**	24	14.92	**25**	**0.15**	25	80.92
4	**25**	**0.16**	25	197.68	25	0.17	25	180.88
5	25	0.20	17	838.47	**25**	**0.19**	18	974.05
6	25	0.76	8	1071.03	**25**	**0.23**	10	1064.63
7	25	180.89	1	163.84	**25**	**0.37**	–	–
8	20	943.89	–	–	**25**	**1.17**	–	–

Table 6. Performance of the selected solvers on random DTPPs with $k = 3$ with different levels. The table is organized as Table 4.