First Steps towards Automated Synthesis of Tableau Systems for Interval Temporal Logics

Dario Della Monica*, Angelo Montanari†, Guido Sciavicco‡ and Dmitri Tishkovsky§
* ICE-TCS, School of Computer Science, Reykjavik University
Reykjavik, Iceland
Email: dariodm@ru.is
† Department of Mathematics and Computer Science, University of Udine
Udine, Italy
Email: angelo.montanari@uniud.it
‡ Department of Information Engineering and Communications, University of Murcia
Murcia, Spain
Email: guido@um.es
§ School of Computer Science, University of Manchester
Manchester, UK
Email: dmitry@cs.man.ac.uk

***Abstract*—Interval temporal logics are difficult to deal with in many respects. In the last years, various meaningful fragments of Halpern and Shoham's modal logic of time intervals have been shown to be decidable with complexities that range from NP-complete to non-primitive recursive. However, even restricting the attention to finite interval structures, the step from model-theoretic decidability results to the actual implementations of tableau-based decision procedures is quite challenging. In this paper, we investigate the possibility of making use of automated tableau generators. More precisely, we exploit the generator METTEL^2 to implement a tableau-based decision procedure for the future fragment of the logic of temporal neighborhood over finite linear orders. We explore and contrast two alternative solutions: a *concrete* tableau system, that operates on a concrete interval structure explicitly built over a finite, linearly-ordered set of points, and an *abstract* one, that operates on an interval frame which is forced to be isomorphic to a concrete interval structure by suitably constraining its accessibility relation.**

***Keywords*–Interval temporal logics; satisfiability; tableaux; automated tableau generation.**

I. INTRODUCTION

In this paper, we make some initial steps towards the automated synthesis of tableau systems for interval temporal logics. It is well-known that turning (optimal) declarative, tableau-based systems for decidable temporal logics into effective decision procedures is far from being trivial. Such a transition turns out to be particularly complex in the case of interval temporal logics. In the last years, it has been experimented for two specific logics, namely, the temporal logic of subintervals D, interpreted over dense linear orders [1], and the future fragment of the logic of temporal neighborhood A, interpreted over finite linear orders [2]. However, in both cases the proposed solution is tailored to the logic under consideration, and thus it lacks generality. In this paper, we explore the possibility of exploiting a general tool for the automated synthesis of tableau systems, namely, the generator METTEL^2, to deal with interval temporal logics. Even though we will apply the proposed solution to the logic A only (this makes it possible to compare the performance of the generated system with that of the procedure given in [2]), there is no any limitation that prevents its application to other interval temporal logics.

Propositional interval temporal logics play a significant role in computer science, as they provide a natural framework for representing and reasoning about temporal properties in a number of application domains [3]. This is the case, for instance, of computational linguistics, where significant interval-based logical formalisms have been developed to represent and reason about tenses and temporal prepositions [4]. As another example, the possibility of encoding and reasoning about various constructs of imperative programming in interval temporal logic has been systematically explored by Moszkowski in [5]. Other meaningful applications of interval temporal logics can be found in knowledge representation, systems for temporal planning and maintenance, qualitative reasoning, theories of action and change, specification and design of hardware components, concurrent real-time processes, event modeling, and temporal databases. Modalities of interval temporal logics correspond to binary relations between time intervals. In particular, Halpern and Shoham's modal logic of time intervals HS [6] features one modality for each Allen's interval relation [7]. In [6], the authors showed that HS is undecidable over all meaningful classes of linear orders. Since then, a lot of work has been devoted to the study of HS fragments, mainly to disclose their computational properties and relative expressiveness. The classification of HS fragments with respect to the status (decidable/undecidable) of their satisfiability problem is now almost completed. In this paper, we focus our attention on the class of finite linear orders, which comes into play in a variety of application domains, e.g., in planning problems. A complete classification of HS fragments over finite linear orders is given in [8]. It shows that there are 62 non-equivalent (with respect to expressiveness) decidable HS fragments, which can be partitioned into four complexity classes, ranging from NP-complete to non-primitive recursive. For each decidable fragment, an optimal, tableau-based decision procedure has been devised. However, since each of such procedures has been given a declarative formulation, no one of them is available as a working system, apart from the tableau-based decision procedure for the fragment A reported in [2]. The only attempt to apply a generic theorem prover to an interval temporal logic can be found in [1], where a tableau-based decision procedure for the fragment D, interpreted over dense linear orders, has been developed in LoTREC [9], [10]. LoTREC is a generic prover for modal and description logics that can be used to prove validity and satisfiability of formulas. Whenever a formula is satisfiable, it returns a model for it; whenever a formula is not valid, it returns a counter-model for it. In LoTREC, a tableau is a special kind of labeled graph that is built, and possibly revised, according to a set of user-specified rules. Every node of the graph is labeled with a set of formulae and can be enriched by auxiliary markings, if needed. Unfortunately, LoTREC, as well as most generic theorem provers, cannot be exploited to deal with other interval temporal logics because (i) they do not support the management of world labels explicitly, and (ii) they support closing conditions based on loop checks, but do not allow explicit checks on the number of worlds generated during the construction of a tentative model. Such limitations are overcome by the current version of METTEL[2] [11], which provides the user with a flexible language for specifying propositional syntaxes and tableau calculi.

In the following, we make use of METTEL[2] to implement a tableau-based decision procedure for A over finite linear orders. We explore and contrast two alternative solutions: a *concrete* tableau system, that operates on a concrete interval structure explicitly built over a finite, linearly-ordered set of points, and an *abstract* one, that operates on an interval frame which is forced to be isomorphic to a concrete interval structure by suitably constraining its accessibility relation (using the specification language provided by METTEL[2]). The main contributions of the paper can be summarized as follows: *(i)* it is the first general attempt of using an automated generator to synthesize a tableau system for an interval temporal logic (D over dense linear orders is a very special case because, due to its properties, it bears strong resemblance to standard modal logic); *(ii)* while METTEL[2] works perfectly on a variety of other logics (see, e.g., [12] and Section III), it required a small, but not trivial, modification to be able to formulate closing conditions for A; *(iii)* the abstract version of the tableau system, based on a suitable representation theorem, renews the interest in the areas of temporal knowledge representation and reasoning, and representation theorems [7], [13], [14].

The paper is structured as follows. In the next section, we present the logic A; in Section III, we give the necessary overview of the system METTEL[2], and in Section IV we present our A-prover. Section V presents an account of the results, and in Section VI we conclude the paper.

II. The Interval Temporal Logic A

Given a linearly ordered set $\mathbb{D}$, a (strict) *interval* $[a, b]$ is a pair $a < b$, where $a, b \in \mathbb{D}$. There are 12 different relations (excluding the identity) between two intervals on a linear order, often referred to as *Allen's relations* [7]: the six relations depicted in Fig. 1, namely $R_A, R_L, R_B, R_E, R_D, R_O$, and the inverse ones, defined in the standard way, that is,

Modality	Relation
$\langle A\rangle$	$[a,b]R_A[c,d] \Leftrightarrow b = c$
$\langle L\rangle$	$[a,b]R_L[c,d] \Leftrightarrow b < c$
$\langle B\rangle$	$[a,b]R_B[c,d] \Leftrightarrow a = c, d < b$
$\langle E\rangle$	$[a,b]R_E[c,d] \Leftrightarrow b = d, a < c$
$\langle D\rangle$	$[a,b]R_D[c,d] \Leftrightarrow a < c, d < b$
$\langle O\rangle$	$[a,b]R_O[c,d] \Leftrightarrow a < c < b < d$

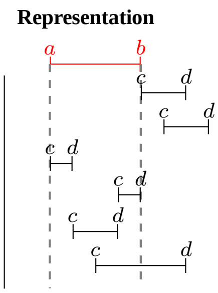

Figure 1. Allen's interval relations and the corresponding HS modalities.

$R_{\overline{X}} = (R_X)^{-1}$, for each $X \in \{A, L, B, E, D, O\}$. Intuitively, an interval structure over a linear order $\mathbb{D}$ consists of the set of all the intervals on $\mathbb{D}$, along with a set of Allen's relations. We treat interval structures as Kripke structures, where the Allen's relations play the role of the accessibility relations, and we associate a modal operator $\langle X\rangle$ with each Allen's relation R_X. Given an operator $\langle X\rangle$ associated to the relation R_X, with $X \in \{A, L, B, E, D, O\}$, its *transpose* is the operator $\langle\overline{X}\rangle$, corresponding to the inverse relation $R_{\overline{X}}$ of R_X.

Syntax and (Concrete) Semantics. Halpern and Shoham's logic HS [6] is a multi-modal logic with formulae built from a finite, non-empty set $\mathcal{AP}$ of atomic propositions, the propositional connectives $\vee$ and $\neg$, and a set of modal operators associated with all Allen's relations. With every subset $\{R_{X_1}, \ldots, R_{X_k}\}$ of these relations, we associate the fragment $\mathsf{X_1X_2\ldots X_k}$ of HS, whose formulae are defined by the grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle X_1\rangle\varphi \mid \ldots \mid \langle X_k\rangle\varphi,$$

where $p \in \mathcal{AP}$. The other propositional connectives and constants (e.g., $\wedge$, $\rightarrow$, and $\top$) can be derived in the standard way, as well as the dual modal operators (e.g., $[A]\varphi \equiv \neg\langle A\rangle\neg\varphi$). In this paper, we will focus on the particular case of the fragment A, so that for all purposes we can assume that formulae are generated by the following restricted grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle A\rangle\varphi.$$

The concrete semantics of HS is given in terms of *concrete interval models*.

Definition 1: Let $\mathbb{D}$ be a linearly ordered set, and $\mathbb{I}(\mathbb{D})$ be the set of all (strict) intervals over $\mathbb{D}$. A *concrete interval structure* is a pair $S = \langle\mathbb{D}, \mathbb{I}(\mathbb{D})\rangle$, and a *concrete interval model* is a pair $M = \langle S, V\rangle$, where S is a concrete interval structure, and V is a *valuation function* $V : \mathcal{AP} \rightarrow 2^{\mathbb{I}(\mathbb{D})}$, which assigns to every atomic proposition $p \in \mathcal{AP}$ the set of intervals $V(p)$ on which p holds.

The *truth* of a formula is evaluated with respect to a concrete interval model M and an interval $[a, b]$ on it, by structural induction on formulae as follows:

- $M, [a, b] \Vdash p$ iff $[a, b] \in V(p)$, for each $p \in \mathcal{AP}$;
- $M, [a, b] \Vdash \neg\psi$ iff it is not the case that $M, [a, b] \Vdash \psi$;
- $M, [a, b] \Vdash \varphi \vee \psi$ iff $M, [a, b] \Vdash \varphi$ or $M, [a, b] \Vdash \psi$;
- $M, [a, b] \Vdash \langle X\rangle\psi$ iff there exists an interval $[c, d]$ such that $[a, b]R_X[c, d]$ and $M, [c, d] \Vdash \psi$, for each modal operator $\langle X\rangle$.

For the purpose of the present paper, we explicitly instantiate below the semantic clause for the modality $\langle A \rangle$:

$$M, [a, b] \Vdash \langle A \rangle \varphi \text{ iff there exists } c > b \text{ s.t. } M, [b, c] \Vdash \varphi.$$

Formulae of HS can be interpreted with respect to several interesting classes of concrete interval models, depending on the particular class of linear orders over which the models are built. In this paper, we focus on the class of (concrete interval models built over) finite linear orders, for which the following small model theorem holds [15].

Theorem 1: Let φ be any A-formula. Then, φ is finitely satisfiable if and only if it is satisfiable on a model whose domain has cardinality strictly less than $2^m \cdot m + 1$, where m is the number of diamonds and boxes in φ.

The above result immediately provides a termination condition that can be used to implement a *fair* procedure that exhaustively searches for a model of size smaller than the bound.

Abstract Semantics. As we have already pointed out, METTEL² is flexible enough to allow us to devise an alternative, *abstract* version of tableau system for A, based on a different, but equivalent, set of semantic conditions. To this end, we first define a suitable class of interval frames for A, called finite abstract interval A-structures, whose distinctive features are expressed by a set of first-order conditions, and then we show that any such frame is isomorphic to a concrete interval structure. It is worth noticing that such an abstract semantics, that takes intervals as first-class citizens, is quite common in the field of interval temporal logics, while modal and point-based temporal logics do not present this duality. In the AI community, the dualism between abstract and concrete interval structures is well-known since the early stages of interval-based temporal reasoning. The variety of binary relations between intervals in linear orders was first studied systematically by Allen et al. [7], [13], [14], who explored their use in systems for time management and planning. Allen's work and its follows up are based on the assumption that time can be represented as a dense line, and that points are excluded from the semantics. As it has been shown in the early work of Allen and Hayes [16] and van Benthem [17], interval temporal reasoning can be formalized as an extension of first-order logic with equality with one or more relations; the resulting formalization will also depend on the choices for certain semantic parameters, specifically, the class of linear orders over which we construct our interval structures. Given the dual nature of time intervals (i.e., they can be abstract first-order individuals with specific characteristics, or they can be defined as ordered pairs over a linear order), one of the problems that have been studied is the so-called *representation theorem*. Consider a class of linear orders: given a specific extension of first-order logic with a set of interval relations (such as, for example, *meets* and *during*), does there exist a set of axioms in this language which constrain (abstract) models in this signature to be isomorphic to concrete ones? In other words, can we produce an isomorphism into concrete models whose domain is the set of intervals over the considered linear order, and whose relations are the concrete interval relations? In the relevant literature, we find a number of representation theorems for languages that include interval relations: van Benthem [17], over rationals and with the interval relations *during* and *before*; Allen and Hayes [16], for the dense unbounded case without point intervals and for the relation *meets*; Ladkin [18], for point-based structures with a quaternary relation that encodes meeting of two intervals; Venema [19], for structures with the relations *starts* and *finishes*; Goranko, Montanari, and Sciavicco [20], that generalize the results for structures with *meets* and *met-by*; and Coetzee [21], for dense structures with *overlaps* and *meets*.

In our specific case, we need a representation theorem to suitably constrain a generic finite Kripke frame $\langle W, R \rangle$. Among all possible choices, we will consider only the relation *meets*, in the line of the original result by Allen and Hayes [16]. Moreover, let $R_L, R_B, \ldots$ denote the first-order relations corresponding to the other Allen's relations (so that $R = R_A$; see Fig. 1). It is worth pointing out that our characterization is fully general with respect to finite abstract vs. concrete interval structures. Nevertheless, not all conditions are needed in the actual implementation, as explained at the end of this section.

Definition 2: Let W be a non-empty set, and $R \subseteq W \times W$ be a binary relation on it. We call the pair $\mathfrak{S} = \langle W, R \rangle$ a *finite abstract interval* A*-structure* if and only if the following conditions are respected:

1) $\forall x \neg(xRx)$ (irreflexivity);
2) $\forall x, y(xRy \wedge yRx \rightarrow x = y)$ (antisymmetry);
3) $\forall x, y(xRy \rightarrow \exists z(\forall t(tRz \leftrightarrow tRx) \wedge \forall t(zRt \leftrightarrow yRt)))$ (composition);
4) $\forall x, y, z, t((xRy \wedge yRt \wedge xRz \wedge zRt) \rightarrow y = z)$ (linearity);
5) $\exists x(\forall y(\neg(yRx)))$ (left-boundedness);
6) $\exists x(\forall y(\neg(xRy)))$ (right-boundedness);
7) $\forall x, y(x = y \vee xR_A y \vee xR_L x \vee \ldots \vee yR_A x \vee yR_L x \vee \ldots)$ (joint exhaustivity).

The aim of our representation theorem is to prove that the above conditions are enough to make sure that every finite abstract interval structure is isomorphic to a finite concrete one, and the other way around. Notice that condition 7 is written in an extended language; in [16] it is proven that every Allen's relation can be expressed in the first-order language by using only $R = R_A$ (originally, the result is stated under the density and unboundedness condition, which are, in fact, not necessary), so that condition 7 can be considered as a shortcut for a longer formula. As we will not implement it in the abstract tableau, this form of condition 7 is actually not a problem. It is convenient to consider, here, concrete interval structures of the type $S = \langle \mathbb{D}, \mathbb{I}(\mathbb{D}), R_A \rangle$, that is, where the relation R_A that corresponds to our modal operator (*meets*) is made explicit. Proving that every such concrete structure respects the above conditions 1-7 is trivial; as for the other direction, it is taken care of in the next theorem, whose proof is omitted for space reasons.

Theorem 2: Every finite abstract interval A-structure is isomorphic to a finite concrete one.

In conclusion, every finite abstract interval structure is a frame over which we can interpret the fragment A. We can easily re-define the notion of model for A as a pair $M = \langle \mathfrak{S}, V \rangle$, where $\mathfrak{S}$ is a finite abstract interval A-structure, and $V : \mathcal{AP} \mapsto 2^{\mathbb{W}}$, so that the modal truth clause can be written as:

$$M, i \Vdash \langle A \rangle \psi \text{ iff there exists } j \in W \text{ s.t. } iRj \text{ and } M, j \Vdash \psi.$$

As a final note, observe that, in fact, one can limit himself to implement only conditions 1-4, paired with a suitable cardinality constraint, that is, with a suitable interval version of the concrete constraint that comes from Theorem 1. The finiteness of $\mathbb{D}$ (conditions 5 and 6) comes as a consequence of such a constraint, and the joint exhaustivity of the Allen's relations (condition 7) is no longer essential: every branch is limited in length by the number of different world *comparable* with the starting one, and no incomparable world is ever created.

III. Automated Synthesis of Tableau Calculi and MetTeL^2

Tableau reasoning methods represent a powerful tool to reason about logical formalisms. They have been extensively used to devise decision procedures for description and modal logics [22], [23], as well as for intuitionistic logics, conditional logics, logics of metric and topology, and hybrid logics. In [24], the authors devise a method for automatically generating tableau calculi from a first-order specification of a formal semantics of a logic. The underlying idea is turning such a specification into a set of inference rules giving rise to a sound, complete, and terminating deduction calculus for the logic, provided that the logic has the finite model property.

The tableau synthesis method introduced in [24] works as follows. The user defines the formal semantics of the given logic in a many-sorted first-order language so that certain well-definedness conditions hold. The semantic specification of the logic is then automatically reduced to Skolemised implicational forms which are further transformed into tableau inference rules. Combined with a set of default closure and equality rules, the generated rules provide a sound and complete calculus for the logic. Under certain conditions the set of rules can be further refined [25]. If the logic has the finite model property, then the generated calculus can be automatically turned into a terminating calculus by adding a suitable blocking mechanism.

The tableau prover generator MetTeL^2 has been implemented to complement the theoretical tableau synthesis framework [11]. MetTeL^2 produces Java code of a tableau prover from specifications of a logical syntax and a tableau calculus for given logic. It is intended to provide an easy-to-use system for non-technical users and allow technical users to extend the implementation of generated provers. MetTeL^2 has been successfully employed to produce tableau provers for modal logics, description logics, epistemic logics, and temporal logics with cardinality constraints. It is worth pointing out that prior implementations of systems for automated synthesis of tableau calculi already existed. We mention, for instance, LoTREC [9], [10] and The Tableau Work Bench (TWB) [26], that are prover engineering platforms most closely related to MetTeL^2. Although MetTeL^2 does not give the user the same possibilities for programming and controlling derivations as these systems, its specification language is more expressive. For example, Skolem terms are allowed both in premises and conclusions of rules. The expressive specification language also allows specifications of syntaxes of arbitrary propositional logics and makes MetTeL^2 able to deal with the interval temporal logic A (which we focus on in this paper) and possibly with most of the other fragments of HS.

IV. Tableau Provers for A

In this section, we describe specifications of two tableau provers which are based on the concrete semantics and the abstract semantics for the fragment A.

The steps for obtaining the specifications are common for both the provers and are as follows. First, we apply the tableau synthesis framework [24] to the semantics of A. We notice that both concrete and abstract semantics for A consist of connective definitions and the background theory. Thus the well-definedness conditions for them in [24] are trivially fulfilled. Therefore, the generated calculi are automatically sound and (constructively) complete for the logic A. Next, we apply the atomic refinement [25] to the rules of the obtained calculi by moving negated atomic formulae in the rule conclusions to its premises while changing their signs. While retaining soundness and (constructive) completeness of the calculi, this reduces branching factor of the rules and makes tableau algorithms based on the calculi more efficient. Finally, we extend the tableau languages with additional constructs which replace the first-order predicates in the original calculi. This further simplifies the calculi, makes them more readable and specifiable in MetTeL^2.

The tableau specifications for the concrete and abstract semantics of A in MetTeL^2 specification language are listed in Fig. 2. The symbol / separates premises of a rule from its conclusions and the symbol || separates branches of the rule. A priority value is assigned to each rule with the keyword $priority$. The less the value the more eagerly the rule is applied during derivation.

The tableau specification for the concrete semantics of A is based on two logical sorts: the sort of points and the sort of logical formulae. Disjunction $p \vee q$ is represented in the specification as p|q, negation $\neg p$ is represented as ~p, and <A> represents the modal operator $\langle A \rangle$. Constructs which are additional to the language of the logic are the ordering predicate $<$ on the sort of points ($a < b$ is represented as {a<b}), the equality predicate ({{a=b}} represents $a = b$), a Skolem function f for generating new terms of the sort of points, and expressions of the form $[a, b] : \varphi$ which are formulae φ of A labeled by intervals $[a, b]$ where a and b are points. The rules on the lines 1–8 of the concrete tableau enforce $<$ to be a strict linear ordering. The rule on the line 10 ensures that all the intervals are not degenerative. The remaining rules are standard for modal-like logics. It is worth noting that the rules on the lines 1–8 and the line 15 are obtained by the atomic refinement from the rules generated by the tableau synthesis framework. For example, the rule [a,b]:~(<A>p) {b < c} / [b,c]:~p is obtained by the refinement from the generated rule [a,b]:~(<A>p) / ~{b < c} || [b,c]:~p. As a consequence of the results in [25], the calculus is sound and (constructively) complete for the standard interval semantics of the fragment A.

The tableau specification for the abstract semantics is also based on two sorts: the sort of intervals and the sort of logical formulae. The additional constructs are two Skolem functions f and g, the equality predicate, and a binary relational symbol R on the sort of intervals. The tableau operates on labeled formulae $@_i\varphi$ (@i p in the specification) where φ is a formula of A and i is an interval. The lines 1–7 of the abstract tableau

```
1  {a < a} / priority 0;
2  {a < b} {b < c} / {a < c} priority 3;
3  {a < b} {c < d} /
4       {{c = a}} || {c < a} || {a < c} {c < b} ||
5       {{c = b}} || {b < c} priority 7;
6  {a < b} {c < d} /
7       {{d = a}} || {d < a} || {a < d} {d < b} ||
8       {{d = b}} || {b < d} priority 7;
9  [a,b]:p [a,b]:~p / priority 0;
10 [a,b]:p / {a < b} priority 1;
11 [a,b]:~(~p) / [a,b]:p priority 1;
12 [a,b]:(p|q) / [a,b]:p || [a,b]:q priority 5;
13 [a,b]:~(p|q) / [a,b]:~p [a,b]:~q priority 3;
14 [a,b]:<A>p / [b,f(b,p)]:p priority 9;
15 [a,b]:~(<A>p) {b < c} / [b,c]:~p priority 4;
```

```
1  R i i / priority 0;
2  R i j R j i / priority 0;
3  R i j R k g(i,j) / R k i priority 4;
4  R i j R k i / R k g(i,j) priority 10;
5  R i j R g(i,j) k / R j k priority 4;
6  R i j R j k / R g(i,j) k priority 10;
7  R i j R j k R i l R l k / {{ j = l }} priority 6;
8  @i p @i ~p / priority 0;
9  @i ~(~p) / @i p priority 1;
10 @i (p|q) / @i p || @i q priority 5;
11 @i ~(p|q) / @i ~p @i ~q priority 3;
12 @i <A>p / R i f(i,p) @f(i,p) p priority 9;
13 @i ~(<A>p) R i j / @j ~p priority 4;
```

Figure 2. Tableau specifications for concrete (left) and abstract (right) semantics.

define the theory of the relation R and correspond to the conditions 1–4 in Definition 2. While the rest of the rules are similar to standard rules for modal-like logics and can be specified in tableau development platforms like LoTREC and TWB, the four rules listed on the lines 3–6 are special. All the four rules uses same Skolem function g, and, moreover, the rules on the lines 3 and 5 have the Skolem function g in their premises. Allowing specifications of tableau rules where Skolem functions occur in the rule premises is a distinctive feature of METTEL^2 prover generator which demonstrate expressiveness of the METTEL^2 specification language. Similarly to the concrete tableau, the rules on the lines 1–7 and the line 13 are obtained by the atomic refinement. Therefore, the calculus is sound and (constructively) complete for the relational semantics of the fragment A.

Termination property of both the provers is achieved by modification of the generated Java code to ignore branches which exceed allowed limit of points or intervals (Theorem 1).

V. Testing and Results

We have tested our implementations against the same benchmark of problems used in [2], although the absolute speed results cannot be immediately compared since the two experiments used a different hardware. These problems are divided into two classes. First, we tested the scalability of the implementation with respect to a set of combinatorial problems of increasing complexity (COMBINATORICS), where the n-th combinatorial problem is defined as the problem of finding a model for a formula that contains n conjuncts, each one of the form $\langle A\rangle p_i$ ($0 \leq i \leq n$), plus $\frac{n(n+1)}{2}$ conjuncts of the form $[A]\neg(p_i \wedge p_j)$ ($i \neq j$). (Notice that there are $n(n+1)$ different conjuncts of the pointed out form. However, a conjunct with indices i, j is equivalent to another one with indices j, i. This is why $\frac{n(n+1)}{2}$ is posed.) Then, we considered the set of 72 purely randomized formulas used in [27] to evaluate an evolutionary algorithm for the same fragment (RANDOMIZED). Table I summarizes the outcome of the experiments. For each class of problems, the corresponding table shows, for each instance n, the time (in milliseconds) necessary to solve the problem taking into account, when appropriate, the specific policy that has been used; in particular, the concrete version has been run under both the 'breadth first' and the 'depth first' (left branch first) policies. A time-out of 1 minute was used to stop instances running for too long.

At first sight, it is clear that the relational (abstract) version of the tableau system is more (time) efficient than the standard (concrete) one; however, the number of instances that generated a memory error indicates that the latter uses less memory, which can be considered an interesting result on its own. All the experiments were executed on Java 1.7.0_25 OpenJDK 64-Bit Server VM under the Java heap size limit of 3Gb on a hardware based on Intel® Core™ i7-880 CPU (3.07GHz, 8Mb), with a total memory of 8Gb (1333MHz), under the 64-bit Fedora Linux 17 operating system.

VI. Conclusions and Future Work

In this paper, we described the outcomes of a first experiment in automated generation of a tableau-based decision procedure for an interval temporal logic, using the automatic prover generator METTEL^2. Thanks to its expressivity and flexibility, we explored and contrasted two alternative implementations: a concrete and an abstract one (at the best of our knowledge, this is the first tableau-based decision procedure for interval temporal logics based on an abstract frame semantics). Although the performance of the developed systems is not particularly exciting, the use of generators like METTEL^2 provides a general and effective way of implementing tableau systems for interval temporal logics. We believe it possible to make the concrete tableau system more efficient, provided that we represent the linear order by a list of points. This would remedy the exponential blow-up of inequality formulae in the tableau derivation, but, unfortunately, lists cannot be represented in the language of METTEL^2 yet. The addition of such a feature to METTEL^2 and practical investigations of its effects are left to future work. As for the abstract tableau system, in principle, it allows us to compare more than one (equivalent) version of the first-order constraints for the same fragment. Last but not least, we are going to validate the proposed approach on other more expressive HS fragments.

Acknowledgements

The authors acknowledge the support from the Spanish fellowship program *'Ramon y Cajal' RYC-2011-07821* (G. Sciavicco), the projects *Processes and Modal Logics* (project nr. 100048021) and *Decidability and Expressiveness for Interval Temporal Logics* (project nr. 130802-051) of the Icelandic Research Fund (D. Della Monica), the Italian GNCS project

Table I. EXPERIMENTAL RESULTS (IN MILLISECONDS; ‘-’: “OUT OF TIME”; ‘M’: “OUT OF MEMORY”; ‘Y’: “SATISFIABLE”; ‘N’: “UNSATISFIABLE”).

COMBINATORICS

	CON		ABS	
n	DF	BF		sat
1	10	10	0	y
2	60	100	0	y
3	270	420	10	y
4	920	1360	30	y
5	2930	4010	70	y

	CON		ABS	
n	DF	BF		sat
6	7890	9850	150	y
7	19420	23670	300	y
8	47220	51220	560	y
9	-	-	1000	y
10	-	-	1790	y

	CON		ABS	
n	DF	BF		sat
11	-	-	3440	y
12	-	-	4660	y
13	-	-	7600	y
14	-	-	11560	y
15	-	-	17170	y

	CON		ABS	
n	DF	BF		sat
16	-	-	25160	y
17	-	-	35610	y
18	-	-	50740	y
19	-	-	-	-
20	-	-	-	-

RANDOMIZED

	CON		ABS	
n	DF	BF		sat
1	-	-	-	-
2	0	0	0	y
3	10	0	0	y
4	0	10	0	y
5	-	-	-	-
6	0	10	0	y
7	-	-	-	-
8	10	10	0	y
9	20	20	10	y
10	10	10	0	y
11	-	-	-	-
12	10	10	0	y
13	10	10	0	y
14	10	10	0	y
15	-	-	-	-
16	10	20	0	y
17	30	50	10	y
18	-	-	-	-

	CON		ABS	
n	DF	BF		sat
19	30	50	0	y
20	-	-	-	-
21	20	50	10	y
22	-	-	-	-
23	-	-	-	-
24	20	20	0	y
25	-	-	-	-
26	-	-	-	-
27	-	-	-	-
28	-	-	-	-
29	-	-	-	-
30	-	-	-	-
31	10	10	10	n
32	-	-	-	-
33	-	-	M	-
34	60	70	10	y
35	-	-	-	-
36	-	-	-	-

	CON		ABS	
n	DF	BF		sat
37	-	-	M	-
38	-	-	M	-
39	-	-	M	-
40	-	-	M	-
41	-	-	-	-
42	-	-	-	-
43	-	-	-	-
44	-	-	-	-
45	-	-	M	-
46	-	-	-	-
47	-	-	-	-
48	-	-	-	-
49	-	-	-	-
50	-	-	M	-
51	-	-	M	-
52	-	-	-	-
53	-	-	M	-
54	-	-	-	-

	CON		ABS	
n	DF	BF		sat
55	-	-	M	-
56	-	-	M	-
57	-	-	M	-
58	-	-	-	-
59	-	-	M	-
60	-	-	M	-
61	-	M	M	-
62	-	-	-	-
63	M	-	-	-
64	-	-	-	-
65	-	-	M	-
66	-	-	-	-
67	M	-	-	-
68	-	-	-	-
69	M	-	-	-
70	-	-	M	-
71	M	M	M	-
72	-	-	-	-

Extended Game Logics (A. Montanari), and the research grant EP/H043748/1 of the UK EPSRC (D. Tishkovsky).

REFERENCES

[1] D. Bresolin, V. Goranko, A. Montanari, and P. Sala, “Tableaux for logics of subinterval structures over dense orderings,” J. of Logic and Computation, vol. 20, no. 1, 2010, pp. 133–166.

[2] D. Bresolin, D. Della Monica, A. Montanari, and G. Sciavicco, “A tableau system for right propositional neighborhood logic over finite linear orders: an implementation,” in Proc. of the 22nd TABLEAUX, ser. LNCS, vol. 8123, 2013, pp. 74–80.

[3] V. Goranko, A. Montanari, and G. Sciavicco, “A road map of interval temporal logics and duration calculi,” J. of Applied Non-Classical Logics, vol. 14, no. 1–2, 2004, pp. 9–54.

[4] I. Pratt-Hartmann, “Temporal prepositions and their logic,” Artificial Intelligence, vol. 166, no. 1-2, 2005, pp. 1–36.

[5] B. Moszkowski, “Reasoning about digital circuits,” Tech. Rep. STAN-CS-83-970, Dept. of Computer Science, Stanford University, Stanford, CA, 1983.

[6] J. Halpern and Y. Shoham, “A propositional modal logic of time intervals,” J. of the ACM, vol. 38, no. 4, 1991, pp. 935–962.

[7] J. Allen, “Maintaining knowledge about temporal intervals,” Communications of the ACM, vol. 26, no. 11, 1983, pp. 832–843.

[8] D. Bresolin, D. Della Monica, A. Montanari, P. Sala, and G. Sciavicco, “Interval temporal logics over finite linear orders: the complete picture,” in Proc. of the 20th ECAI, 2012, pp. 199–204.

[9] F. del Cerro et al., “Lotrec: the generic tableau prover for modal and description logics,” in Proc. of the 1st IJCAR, ser. LNCS, vol. 2083. Springer, 2001, pp. 453–458.

[10] O. Gasquet, A. Herzig, D. Longin, and M. Sahade, “LoTREC: Logical Tableaux Research Engineering Companion,” in Proc. of the 14th TABLEAUX, ser. LNCS, vol. 3702, 2005, pp. 318–322.

[11] D. Tishkovsky, R. A. Schmidt, and M. Khodadadi, “The tableau prover generator METTEL^2,” in Proc. of the 13th JELIA, 2012, pp. 492–495.

[12] M. Khodadadi, R. A. Schmidt, D. Tishkovsky, and M. Zawidzki, “Terminating tableau calculi for modal logic K with global counting operators,” 2012, technical report. Available at http://www.mettel-prover.org/papers/KEn12.pdf.

[13] P. J. Hayes and J. F. Allen, “Short time periods,” in Proc. of the 10th IJCAI, Milano, Italy, 1987, pp. 981–983.

[14] J. F. Allen and G. Ferguson, “Actions and events in interval temporal logic,” J. Log. Comput., vol. 4, no. 5, 1994, pp. 531–579.

[15] D. Bresolin, A. Montanari, and G. Sciavicco, “An optimal decision procedure for Right Propositional Neighborhood Logic,” J. of Automated Reasoning, vol. 38, no. 1-3, 2007, pp. 173–199.

[16] J. F. Allen and P. J. Hayes, “A common-sense theory of time,” in Proc. of the 9th IJCAI, Los Angeles, CA, USA, 1985, pp. 528–531.

[17] J. Benthem, The Logic of Time, 2nd ed. Kluwer Academic Press, 1991.

[18] P. Ladkin, “The logic of time representation,” Ph.D. dissertation, University of California, Berkeley, 1987.

[19] Y. Venema, “A modal logic for chopping intervals,” Journal of Logic and Computation, vol. 1, no. 4, 1991, pp. 453–476.

[20] V. Goranko, A. Montanari, and G. Sciavicco, “Propositional interval neighborhood temporal logics,” J. of Universal Computer Science, vol. 9, no. 9, 2003, pp. 1137–1167.

[21] C. J. Coetzee, “Representation theorems for classes of interval structures,” Master’s thesis, Department of Mathematics, University of Johannesburg, 2009.

[22] F. Baader and U. Sattler, “An overview of tableau algorithms for description logics,” Studia Logica, vol. 69, no. 1, 2001, pp. 5–40.

[23] R. Goré, “Tableau methods for modal and temporal logics,” in Handbook of Tableau Methods. Springer Netherlands, 1999, pp. 297–396.

[24] R. A. Schmidt and D. Tishkovsky, “Automated synthesis of tableau calculi,” Logical Methods in Computer Science, vol. 7, no. 2:6, 2011, pp. 1–32. [Online]. Available: http://arxiv.org/abs/1104.4131

[25] D. Tishkovsky and R. A. Schmidt, “Refinement in the tableau synthesis framework,” CoRR, vol. abs/1305.3131, 2013.

[26] P. Abate and R. Goré, “The Tableau Workbench,” Electronic Notes in Theoretical Computer Science, vol. 231, 2009, pp. 55 – 67.

[27] D. Bresolin, F. Jiménez, G. Sánchez, and G. Sciavicco, “Finite satisfiability of propositional interval logic formulas with multi-objective evolutionary algorithms,” in Proc. of the 12th FOGA, 2013, pp. 25–36.