Tableau-based decision procedures for the logics of subinterval structures over dense orderings

Davide Bresolin[1], Valentin Goranko[2], Angelo Montanari[3], and Pietro Sala[3]

[1] Department of Computer Science, University of Verona, Verona, Italy bresolin@sci.univr.it

[2] School of Mathematics, University of the Witwatersrand, Johannesburg, South Africa goranko@maths.wits.ac.za

[3] Department of Mathematics and Computer Science, University of Udine, Udine, Italy {angelo.montanari|pietro.sala}@dimi.uniud.it

Abstract. In this paper, we develop tableau-based decision procedures for the logics of subinterval structures over dense linear orderings. In particular, we consider the two difficult cases: the relation of strict subintervals (with both endpoints strictly inside the current interval) and the relation of proper subintervals (that can share one endpoint with the current interval). For each of these logics, we establish a small pseudo-model property and construct a sound, complete, and terminating tableau that searches systematically for existence of such a pseudo-model satisfying the input formulas. Both constructions are non-trivial, but the latter is substantially more complicated because of the presence of beginning and ending subintervals which require special treatment. We prove PSPACE completeness for both procedures and implement them in the generic tableau-based theorem prover Lotrec.

1 Introduction

Interval-based temporal logics provide a natural framework for temporal representation and reasoning. Unfortunately, in the interval temporal logic setting undecidability is the rule (see, for instance, [16,18]). The quest for decidable fragments and systems of temporal logics with interval-based semantics is one of the main research problems in the area of interval logics. Several decidability results have been established previously by reduction to point-based logics, either by selection of an appropriate subset of temporal operators or by restriction of the semantics, e.g., imposing locality, homogeneity, or other principles that essentially reduce it to point-based semantics [1,2,3,12,13,14,17,19]. Only recently some decidability results for genuinely interval-based logics have been established [4,5,6,7,8,9,10].

This paper deals with interval logics of subinterval structures over dense linear orderings. There are three natural definitions of the subinterval relation: reflexive $\sqsubseteq$ (the current interval is a subinterval of itself), proper $\sqsubset$ (subintervals can share one endpoint with the current interval), and strict $\sqsubset\!\!\cdot$ (both endpoints of the subintervals are strictly inside the current interval). The logic $\mathrm{D}_{\sqsubseteq}$ of reflexive subintervals has been studied first by van Benthem in [22], where it is proved that this logic, if interpreted over dense linear orderings, is equivalent to the standard modal logic S4. The connections between the logic of strict subintervals $\mathrm{D}_{\sqsubset\!\!\cdot}$ and the logic of Minkowski space-time have been explored by Shapirovsky and Shehtman in [21]. The authors proved that the following axiomatic system is sound and complete for $\mathrm{D}_{\sqsubset\!\!\cdot}$ over the class of dense orderings:

- the K axiom;
- transitivity: $[D]p \to [D][D]p$,
- seriality: $\langle D\rangle \top$,
- 2-density: $\langle D\rangle p_1 \wedge \langle D\rangle p_2 \to \langle D\rangle(\langle D\rangle p_1 \wedge \langle D\rangle p_2)$.

By means of a suitable filtration technique, they also proved $\mathrm{D}_{\sqsubset\!\!\cdot}$ decidability and PSPACE completeness [20,21].

In this paper, we develop a sound, complete, and terminating tableau system for $\mathrm{D}_{\sqsubset\!\!\cdot}$. In order to prove soundness and completeness, we introduce a kind of *finite* pseudo-models for $\mathrm{D}_{\sqsubset\!\!\cdot}$, called $\mathrm{D}_{\sqsubset\!\!\cdot}$-structures, and we show that every formula satisfiable in $\mathrm{D}_{\sqsubset\!\!\cdot}$ is satisfiable in such pseudo-models with a bound on their dimension which depends on the size of the formula to be checked for satisfiability, thereby proving small-model property and decidability in PSPACE of $\mathrm{D}_{\sqsubset\!\!\cdot}$ (the result established earlier by Shapirovsky and Shehtman by means of filtration). Inter alia, we show that $\mathrm{D}_{\sqsubset\!\!\cdot}$ is also the logic of subinterval structures over the interval $[0,1]$ of the rational line.

Then, we extend our results to the case of the interval logic $\mathrm{D}_{\sqsubset}$ interpreted in dense interval structures with *proper* (irreflexive) subinterval relation. $\mathrm{D}_{\sqsubset}$ differs substantially from $\mathrm{D}_{\sqsubset\!\!\cdot}$ and is much more difficult to analyze. The presence of the special families of beginning subintervals and ending subintervals of a given interval in a structure with proper subinterval relation leads to considerable complications in the constructions of both pseudo-models and tableaux with respect to the case of $\mathrm{D}_{\sqsubset\!\!\cdot}$. For instance, the formula $(\langle D\rangle p \wedge \langle D\rangle q) \to \langle D\rangle(\langle D\rangle p \wedge \langle D\rangle q)$ is valid in $\mathrm{D}_{\sqsubset\!\!\cdot}$, but not in $\mathrm{D}_{\sqsubset}$ (in $\mathrm{D}_{\sqsubset}$, p and q could be satisfied in respectively beginning and ending subintervals only). Furthermore, the formula

$$\langle D\rangle(p \wedge [D]q) \wedge \langle D\rangle(p \wedge [D]\neg q) \wedge [D]\neg(\langle D\rangle(p \wedge [D]q) \wedge \langle D\rangle(p \wedge [D]\neg q))$$

can only be satisfied in a $\mathrm{D}_{\sqsubset}$-structure, as it forces p to be true at some beginning and at some ending subintervals, a requirement which cannot be imposed in $\mathrm{D}_{\sqsubset\!\!\cdot}$. Note, however, that $\mathrm{D}_{\sqsubset}$ can refer to beginning or ending subintervals, but *it cannot differentiate between them.* This is a subtle but crucial detail: as shown by Lodaya [18], the interval logic BE with modalities respectively for beginning and ending subintervals is *undecidable over the class of dense orderings.*

The paper is structured as follows. In Section 2 we provide some basic definitions, including syntax and semantics of the considered logics. In Section 3 we introduce pseudo-models for $\mathrm{D}_{\sqsubset\!\!\cdot}$ and we use them to prove its decidability and PSPACE completeness. Moreover, we develop an optimal tableau-based decision procedure for $\mathrm{D}_{\sqsubset\!\!\!\cdot}$ formulas, which, from a practical point of view, turned out to be much more efficient than a brute enumeration method directly based on the small-model property. In Section 4 we deal with the more difficult case of $\mathrm{D}_{\sqsubset}$ by providing a non-trivial generalization of the notion of pseudo-model that allows us to prove its decidability and PSPACE completeness. Moreover, we provide an optimal tableau method for it. In Section 5 we describe an implementation of the two tableau methods in Lotrec. Conclusions provide an assessment of the work and outline future research directions.

2 Preliminaries

Let $\mathbb{D} = \langle D, < \rangle$ be a dense linear order. An *interval* over $\mathbb{D}$ is an ordered pair $[b, e]$, where $b < e$. We denote the set of all intervals over $\mathbb{D}$ by $\mathbb{I}(\mathbb{D})$. We consider three *subinterval relations*: the *reflexive subinterval relation* (denoted by $\sqsubseteq$), defined by $[d_k, d_l] \sqsubseteq [d_i, d_j]$ iff $d_i \leq d_k$ and $d_l \leq d_j$, the *proper (or irreflexive) subinterval relation* (denoted by $\sqsubset$), defined by $[d_k, d_l] \sqsubset [d_i, d_j]$ iff $[d_k, d_l] \sqsubseteq [d_i, d_j]$ and $[d_k, d_l] \neq [d_i, d_j]$, and the *strict subinterval relation* (denoted by $\sqsubset\!\!\!\cdot$), defined by $[d_k, d_l] \sqsubset\!\!\!\cdot [d_i, d_j]$ iff $d_i < d_k$ and $d_l < d_j$.

The three modal logics $\mathrm{D}_{\sqsubseteq}$, $\mathrm{D}_{\sqsubset}$, and $\mathrm{D}_{\sqsubset\!\!\!\cdot}$ share the same language, consisting of a set $\mathcal{AP}$ of propositional letters, the propositional connectives $\neg$ and $\vee$, and the modal operator $\langle D \rangle$. The other propositional connectives, as well as the logical constants $\top$ (*true*) and $\bot$ (*false*) and the dual modal operator $[D]$, are defined as usual. Formulae are defined by the following grammar:

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

The semantics of $\mathrm{D}_{\sqsubset\!\!\!\cdot}$, $\mathrm{D}_{\sqsubset}$, and $\mathrm{D}_{\sqsubseteq}$ only differ in the interpretation of the $\langle D \rangle$ operator. For the sake of brevity, we use $\sim \in \{\sqsubset\!\!\!\cdot, \sqsubset, \sqsubseteq\}$ as a shorthand for either one of the three subinterval relations. The semantics of a subinterval logic $\mathrm{D}_{\sim}$ is based on *interval models* $\mathbf{M} = \langle \mathbb{I}(\mathbb{D}), \sim, \mathcal{V} \rangle$. The *valuation function* $\mathcal{V} : \mathcal{AP} \mapsto 2^{\mathbb{I}(\mathbb{D})}$ assigns to every propositional variable p the set of intervals $\mathcal{V}(p)$ over which p holds. The satisfiability relation $\Vdash$ is recursively defined as follows:

- for every propositional variable $p \in \mathcal{AP}$, $\mathbf{M}, [d_i, d_j] \Vdash p$ iff $[d_i, d_j] \in \mathcal{V}(p)$;
- $\mathbf{M}, [d_i, d_j] \Vdash \neg\psi$ iff $\mathbf{M}, [d_i, d_j] \nVdash \psi$;
- $\mathbf{M}, [d_i, d_j] \Vdash \psi_1 \vee \psi_2$ iff $\mathbf{M}, [d_i, d_j] \Vdash \psi_1$ or $\mathbf{M}, [d_i, d_j] \Vdash \psi_2$;
- $\mathbf{M}, [d_i, d_j] \Vdash \langle D \rangle \psi$ iff $\exists [d_k, d_l] \in \mathbb{I}(D)$ such that $[d_k, d_l] \sim [d_i, d_j]$ and $\mathbf{M}, [d_k, d_l] \Vdash \psi$.

A $\mathrm{D}_{\sim}$-formula is $\mathrm{D}_{\sim}$-*satisfiable* if it is true in some interval in some interval model and it is $\mathrm{D}_{\sim}$-*valid* if it is true in every interval in every interval model.

As already pointed out in the introduction, the logic $\mathrm{D}_{\sqsubseteq}$ turns out to be equivalent to the standard modal logic S4. Hence, hereafter we will concentrate our attention on the two logics $\mathrm{D}_{\sqsubset\!\!\!\cdot}$ and $\mathrm{D}_{\sqsubset}$.

We introduce now some basic definitions and notation that will be extensively used in the following. Given a $\mathrm{D}_{\sim}$-formula φ, we define the *closure of* φ (denoted by $\mathrm{CL}(\varphi)$) as the set of all subformulas of φ and their negations. The notion of φ-atom can then be defined as follows.

Definition 1. *Given a* $\mathrm{D}_{\sim}$*-formula* φ*, a* φ*-atom* A *is a subset of* $\mathrm{CL}(\varphi)$ *such that (i) for every* $\psi \in \mathrm{CL}(\varphi)$*,* $\psi \in A$ *if and only if* $\neg\psi \notin A$ *and (ii) for every* $\psi_1 \vee \psi_2 \in \mathrm{CL}(\varphi)$*,* $\psi_1 \vee \psi_2 \in A$ *if and only if* $\psi_1 \in A$ *or* $\psi_2 \in A$*.*

We denote the set of all φ-atoms as $\mathcal{A}_\varphi$. Atoms are connected by the following binary relation D_φ.

Definition 2. *Let* D_φ *be a binary relation over* $\mathcal{A}_\varphi$ *such that, for every pair of atoms* $A, A' \in \mathcal{A}_\varphi$*,* $A \; D_\varphi \; A'$ *holds if and only if for every formula* $[D]\psi \in A$ *both* $\psi \in A'$ *and* $[D]\psi \in A'$*.*

Let A be a φ-atom, we denote the set $\{\langle D\rangle\psi \in \mathrm{CL}(\varphi) : \langle D\rangle\psi \in A\}$ of temporal requests of A by $\mathrm{REQ}(A)$. We have that if $\langle D\rangle\psi \notin \mathrm{REQ}(A)$, then $[D]\neg\psi \in A$ (by definition of φ-atom) and thus $\mathrm{REQ}(A)$ identifies all temporal formulas in A. We denote by REQ_φ the set of all $\langle D\rangle$-formulae in $\mathrm{CL}(\varphi)$.

Both logics $\mathrm{D}_{\sqsubset}$ and $\mathrm{D}_{\dot\sqsubset}$ will be interpreted over special classes of directed graphs. A *directed graph* is a pair $\mathbb{G} = \langle V, E\rangle$, where V is a set of *vertices* and $E \subseteq V \times V$ is a set of *edges*. We say that a vertex $v \in V$ is *reflexive* if the edge (v, v) belongs to E, otherwise we say that v is an *irreflexive vertex*. Given two vertices $v, v' \in V$, we say that v' is a *successor* of v if the edge (v, v') belongs to the graph and we say that v' is a *descendant* of v if v' is reachable from v via a path of successors.

The size of a graph $\mathbb{G} = \langle V, E\rangle$ depends on two parameters: (i) the *breadth* of the graph, which is the maximum number of outgoing edges of a vertex, and (ii) the *depth* of the graph, which is the maximum length of a repetition-free path of vertices. Both parameters will come into play in the decidability proofs for both $\mathrm{D}_{\dot\sqsubset}$ and $\mathrm{D}_{\sqsubset}$: we will show that a formula is satisfiable if and only if it is satisfiable in a graph of bounded breadth and depth.

3 The logic $\mathrm{D}_{\dot\sqsubset}$ of strict subinterval structures

3.1 Structures for $\mathrm{D}_{\dot\sqsubset}$ formulas

To devise a decision procedure for $\mathrm{D}_{\dot\sqsubset}$, we interpret it over a special class of graphs, that we call $\mathrm{D}_{\dot\sqsubset}$-graphs. We will prove that a $\mathrm{D}_{\dot\sqsubset}$ formula is satisfiable in a dense interval structure if and only if it is "satisfiable" in a $\mathrm{D}_{\dot\sqsubset}$-graph. As matter of fact, it will turn out that this is equivalent to satisfiability in an interval structure over the interval $[0, 1]$ of the rational line.

Definition 3. *A finite directed graph* $\mathbb{G} = \langle V, E\rangle$ *is a* $\mathrm{D}_{\dot\sqsubset}$-graph *if (and only if) the following conditions hold:*

1. *there exists an irreflexive vertex* $v_0 \in V$*, called the* root *of* $\mathbb{G}$*, such that any other vertex* $v \in V$ *is reachable from it;*

2. *every irreflexive vertex $v \in V$ has a unique successor v_D, which is reflexive;*
3. *every successor of a reflexive vertex v, different from v, is irreflexive.*

An example of $\mathrm{D}_{\sqsubset}$-graph is depicted in Figure 1. $\mathrm{D}_{\sqsubset}$-graphs are finite by definition, but they may include loops involving irreflexive vertices.

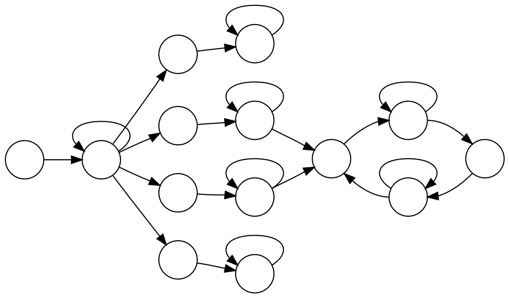


Fig. 1. An example of $\mathrm{D}_{\sqsubset}$-graph.

A $\mathrm{D}_{\sqsubset}$-structure is a $\mathrm{D}_{\sqsubset}$-graph paired with a labeling function that associates an $\mathcal{A}_\varphi$ atom with every vertex in the graph. It is formally defined as follows.

Definition 4. *A* $\mathrm{D}_{\sqsubset}$-structure *is a pair* $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$*, where* $\langle V, E\rangle$ *is a* $\mathrm{D}_{\sqsubset}$*-graph and* $\mathcal{L} : V \rightarrow \mathcal{A}_\varphi$ *is a* labeling function *that assigns to every vertex* $v \in V$ *an atom* $\mathcal{L}(v)$ *such that, for every edge* $(v, v') \in E$*,* $\mathcal{L}(v)\ D_\varphi\ \mathcal{L}(v')$*. Let* v_0 *be the root of* $\langle V, E\rangle$*. If* $\varphi \in \mathcal{L}(v_0)$*, we say that* $\mathbf{S}$ *is a* $\mathrm{D}_{\sqsubset}$-structure for φ.

$\mathrm{D}_{\sqsubset}$-structures can be viewed as tentative 'pseudo-models' for $\mathrm{D}_{\sqsubset}$. Formulas devoid of temporal operators are satisfied by definition of φ-atom; moreover, $[D]$ formulas are satisfied by definition of D_φ. To guarantee the satisfiability of $\langle D\rangle$ formulas, we introduce the notion of *fulfilling* $\mathrm{D}_{\sqsubset}$*-structures.*

Definition 5. *A* $\mathrm{D}_{\sqsubset}$*-structure* $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ *is* fulfilling *if and only if for every vertex* $v \in V$ *and every formula* $\langle D\rangle\psi \in \mathcal{L}(v)$*, there exists a descendant* v' *of* v *such that* $\psi \in \mathcal{L}(v')$*.*

A fulfilling $\mathrm{D}_{\sqsubset}$-structure for the formula $\varphi = \langle D\rangle p \wedge \langle D\rangle\neg p \wedge \langle D\rangle q$ is shown in Figure 2.

Theorem 1. *Let* φ *be a* $\mathrm{D}_{\sqsubset}$ *formula which is satisfied by a strict interval model. Then, there exists a fulfilling* $\mathrm{D}_{\sqsubset}$*-structure* $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ *such that* $\varphi \in \mathcal{L}(v_0)$*, where* v_0 *is the root of* $\langle V, E\rangle$*.*

Proof. Let $\mathbf{M} = \langle \mathbb{I}(\mathbb{D}), \sqsubset, \mathcal{V}\rangle$ be a strict interval model and let $[b_0, e_0] \in \mathbb{I}(\mathbb{D})$ be an interval such that $\mathbf{M}, [b_0, e_0] \Vdash \varphi$. We recursively build a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ for φ as follows.

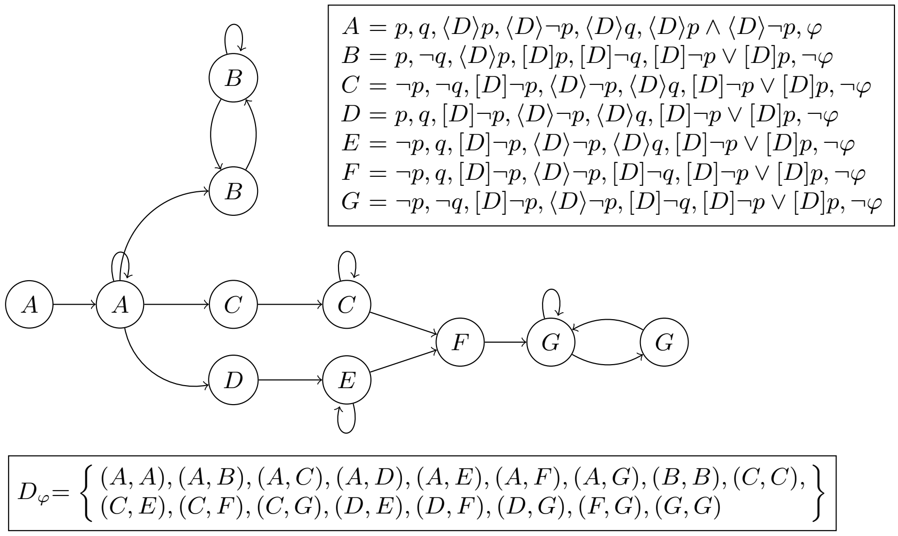


Fig. 2. A fulfilling $\mathrm{D}_{\sqsubset}$-structure for the formula $\varphi = \langle D\rangle p \wedge \langle D\rangle \neg p \wedge \langle D\rangle q$.

We start with the one-node graph $\langle \{v_0\}, \emptyset \rangle$ and the labeling function $\mathcal{L}$ such that $\mathcal{L}(v_0) = \{\psi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_0, e_0] \Vdash \psi\}$.

Next, for every formula $\langle D\rangle\psi \in \mathcal{L}(v_0)$, we pick up an interval $[b_\psi, e_\psi]$ such that $[b_\psi, e_\psi] \sqsubset [b_0, e_0]$ and $\mathbf{M}, [b_\psi, e_\psi] \Vdash \psi$. Since $\mathbb{D}$ is a dense ordering and $\mathrm{CL}(\varphi)$ is a finite set of formulas, there exist two intervals $[b_1, e_1]$ and $[b_2, e_2]$ such that:

– $[b_2, e_2] \sqsubset [b_1, e_1] \sqsubset [b_0, e_0]$;
– for every interval $[b_\psi, e_\psi]$, $[b_\psi, e_\psi] \sqsubset [b_2, e_2]$;
– $[b_1, e_1]$ and $[b_2, e_2]$ satisfy the same formulas of $\mathrm{CL}(\varphi)$.

Since $\mathbf{M}$ is a model and $[b_\psi, e_\psi] \sqsubset [b_2, e_2]$ for every interval $[b_\psi, e_\psi]$, $[b_2, e_2]$ satisfies $\langle D\rangle\psi$ for every $\langle D\rangle\psi \in \mathcal{L}(v_0)$. Moreover, since $[b_1, e_1]$ and $[b_2, e_2]$ satisfy the same formulas of $\mathrm{CL}(\varphi)$ and $[b_2, e_2] \sqsubset [b_1, e_1]$, for every $[D]\psi \in \mathrm{CL}(\varphi)$, if $[b_2, e_2]$ satisfies $[D]\psi$, then it satisfies ψ as well.

Accordingly, we add a new (reflexive) vertex v_D and the edges (v_0, v_D) and (v_D, v_D) to the graph and we label v_D by $\mathcal{L}(v_D) = \{\xi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_2, e_2] \Vdash \xi\}$. Furthermore, for every interval $[b_\psi, e_\psi]$, we add a new (irreflexive) vertex v_ψ, together with the edge (v_D, v_ψ), and we label it by $\mathcal{L}(v_\psi) = \{\xi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_\psi, e_\psi] \Vdash \xi\}$. Then, to obtain a $\mathrm{D}_{\sqsubset}$-structure for φ, we recursively apply the above construction to the vertices $v_{\psi_1}, \ldots, v_{\psi_k}$.

To keep the construction finite, whenever the above procedure requests us to introduce a successor v'' of a reflexive (resp., irreflexive) node $v \in V$, but there exists an irreflexive (resp., reflexive) node $v' \in V$ such that $\mathcal{L}(v') = \mathcal{L}(v'')$, we replace the addition of the node v'' with the addition of an edge from v to v'. Since the set of atoms is finite, this guarantees the termination of the construction process. □

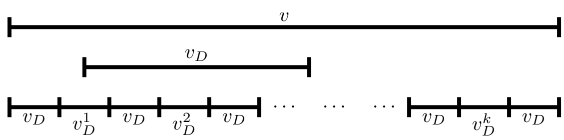
Let $\mathbf{S}$ be a fulfilling $\mathrm{D}_{\sqsubset}$-structure for a formula φ. We will prove that φ is satisfiable in a strict interval structure. Moreover, we will show that such a structure can be constructed on the interval $[0, 1]$ of the rational line. To begin with, we define a function $f_{\mathbf{S}}$ connecting intervals in $\mathbb{I}([0, 1])$ to vertices in $\mathbf{S}$. Such a function will allow us to define a model for φ.

Definition 6. *Let $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ be a $\mathrm{D}_{\sqsubset}$-structure. The function $f_{\mathbf{S}} : \mathbb{I}([0, 1]) \mapsto V$ is recursively defined as follows:*

- $f_{\mathbf{S}}([0, 1]) = v_0$;
- *let $[b, e]$ be an interval such that $f_{\mathbf{S}}([b, e]) = v$ and $f_{\mathbf{S}}$ has not been yet defined over any of its subinterval. We distinguish two cases. If v is irreflexive, let v_D be its unique reflexive successor. If v is irreflexive, let $v_D = v$. Two alternatives must be taken into consideration.*
  1. v_D has no successors different from itself. *In such a case, we put $f_{\mathbf{S}}([b', e']) = v_D$ for every proper subinterval $[b', e']$ of $[b, e]$.*
  2. v_D has at least one successor different from itself. *Let $v_D^1, \dots, v_D^k$ be the successors of v_D different from v_D. We consider the intervals defined by the points $b, b + p, b + 2p \dots, b + 2kp, b + (2k + 1)p = e$, with $p = \frac{e-b}{2k+1}$. The function $f_{\mathbf{S}}$ over such intervals is defined as follows:*
     - *for every $i = 1, \dots, k$, we put $f_{\mathbf{S}}([b + (2i - 1)p, b + 2ip]) = v_D^i$.*
     - *for every $i = 0, \dots, k$, we put $f_{\mathbf{S}}([b + 2ip, b + (2i + 1)p]) = v_D$.*

     We complete the construction by putting $f_{\mathbf{S}}([b', e']) = v_D$ for every subinterval $[b', e']$ of $[b, e]$ which is not a subinterval of any of the intervals $[b + ip, b + (i + 1)p]$.

The structure induced by Definition 6 is depicted below.



It is easy to show that $f_{\mathbf{S}}$ satisfies the following properties (proof omitted).

Lemma 1.
1. *For every pair of intervals $[b, e], [b', e'] \in \mathbb{I}([0, 1])$ such that $[b', e'] \sqsubset [b, e]$, $f_{\mathbf{S}}([b', e'])$ is reachable from $f_{\mathbf{S}}([b, e])$.*
2. *For every interval $[b, e] \in \mathbb{I}([0, 1])$, if $f_{\mathbf{S}}([b, e]) = v$ and v' is reachable from v, then there exists $[b', e'] \sqsubset [b, e]$ such that $f_{\mathbf{S}}([b', e']) = v'$.*

Given a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S}$ for φ, let $\mathbf{M}_{\mathbf{S}}$ be the triplet $\langle\mathbb{I}([0, 1]), \sqsubset, \mathcal{V}\rangle$, where $\mathcal{V}(p) = \{[b, e] : p \in \mathcal{L}(f_{\mathbf{S}}([b, e]))\}$ for every $p \in \mathcal{AP}$. It turns out that $\mathbf{M}_{\mathbf{S}}$ is a model for φ.

Theorem 2. *Let $\mathbf{S}$ be a fulfilling $\mathrm{D}_{\sqsubset}$-structure for φ. Then $\mathbf{M}_{\mathbf{S}}, [0, 1] \Vdash \varphi$.*

Proof. We prove that for every interval $[b, e] \in \mathbb{I}([0,1])$ and every formula $\psi \in \mathrm{CL}(\varphi)$, $\mathbf{M_S}, [b, e] \Vdash \psi$ if and only if $\psi \in \mathcal{L}(f_\mathbf{S}([b, e])$. The proof is by induction on the structure of the formula.

- the case of propositional letters as well as those of Boolean connectives are straightforward and thus omitted;
- let $\psi = \langle D\rangle\xi$, and suppose that $\psi \in \mathcal{L}(f_\mathbf{S}([b, e])$. Since $\mathbf{S}$ is fulfilling, there exists a vertex v', which is reachable from $f_\mathbf{S}([b, e])$, such that $\xi \in \mathcal{L}(v')$. By Lemma 1, there exists $[b', e'] \sqsubset [b, e]$ such that $f_\mathbf{S}([b', e']) = v'$. By inductive hypothesis, $\mathbf{M_S}, [b', e'] \Vdash \xi$ and thus $\mathbf{M_S}, [b, e] \Vdash \langle D\rangle\xi$.
  To prove the opposite implication, suppose by reductio ad absurdum that $\mathbf{M_S}, [b, e] \Vdash \langle D\rangle\xi$ but $\langle D\rangle\xi \notin \mathcal{L}(f_\mathbf{S}([b, e]))$. By definition of φ-atom, this implies that $[D]\neg\xi \in \mathcal{L}(f_\mathbf{S}([b, e]))$. Thus, by Lemma 1, we have that, for every $[b', e'] \sqsubset [b, e]$, $\neg\xi \in \mathcal{L}(f_\mathbf{S}([b', e'])$. By inductive hypothesis, this implies that $\mathbf{M_S}, [b', e'] \Vdash \neg\xi$ for every $[b', e'] \sqsubset [b, e]$, which is a contradiction.

Let v_0 be the root of $\mathbf{S}$. Since $\varphi \in \mathcal{L}(v_0)$ and $f_\mathbf{S}([0, 1]) = v_0$, it immediately follows that $\mathbf{M_S}, [0, 1] \Vdash \varphi$. $\square$

In Figure 3, we show how to turn the fulfilling $\mathrm{D}_\sqsubset$-structure for the formula $\varphi = \langle D\rangle p \wedge \langle D\rangle\neg p \wedge \langle D\rangle q$ depicted in Figure 2 into a model for φ.

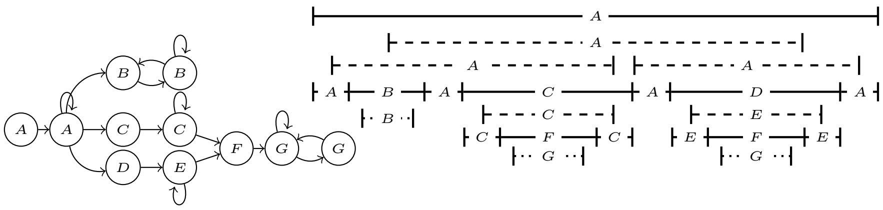


Fig. 3. Turning a fulfilling $\mathrm{D}_\sqsubset$-structure into a model.

3.2 A small-model theorem for $\mathrm{D}_\sqsubset$-structures

In this section we will prove a small-model theorem for $\mathrm{D}_\sqsubset$, that is, we will show that a $\mathrm{D}_\sqsubset$ formula is satisfiable if and only if there exists a fulfilling $\mathrm{D}_\sqsubset$-structure of *bounded size*. In particular, we will demonstrate that the breadth and the depth of such a fulfilling $\mathrm{D}_\sqsubset$-structure are linear in the size of the formula.

Theorem 3. *For every satisfiable* $\mathrm{D}_\sqsubset$ *formula* φ*, there exists a fulfilling* $\mathrm{D}_\sqsubset$*-structure whose breadth and depth are bounded by* $2 \cdot |\varphi|$.

Proof. Let $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ be a fulfilling $\mathrm{D}_\sqsubset$-structure for φ. The following algorithm builds a fulfilling $\mathrm{D}_\sqsubset$-structure $\mathbf{S}' = \langle\langle V', E'\rangle, \mathcal{L}'\rangle$ for φ with the requested property.

Initialization. Initialize $\mathbf{S}'$ as the one-vertex $D_{\sqsubset}$-structure $\langle\langle\{v_0\},\emptyset\rangle,\mathcal{L}'\rangle$, where v_0 is the root of $\mathbf{S}$ and $\mathcal{L}'(v_0)=\mathcal{L}(v_0)$. Call the procedure *Expansion* on v_0.

Expansion(v). If v is irreflexive, execute *Step 1*; otherwise, execute *Step 2*.

Step 1. Let v' be unique reflexive successor of v in $\mathbf{S}$. Add v' to V' and $(v,v'),(v',v')$ to E'; moreover, put $\mathcal{L}'(v')=\mathcal{L}(v')$. Call the procedure *Expansion* on v'.

Step 2. Let $\mathrm{REQ}(\mathcal{L}'(v))=\{\langle D\rangle\psi_1,\ldots,\ \langle D\rangle\psi_k\}$. Since $\mathbf{S}$ is fulfilling, for every formula $\langle D\rangle\psi_i\in\mathrm{REQ}(\mathcal{L}'(v))$, there exists a descendant v_i of v in $\mathbf{S}$ such that $\psi_i\in\mathcal{L}(v_i)$. For $i=1\ldots,k$, add v_i to V' and (v,v_i) to E'; moreover, put $\mathcal{L}'(v_i)=\mathcal{L}(v_i)$. Next, for every v_i such that $\mathrm{REQ}(\mathcal{L}'(v_i))=\mathrm{REQ}(\mathcal{L}'(v))$, add an edge (v_i,v) to E'. For the remaining vertices v_i, it holds that $\mathrm{REQ}(\mathcal{L}'(v_i))\subset\mathrm{REQ}(\mathcal{L}'(v))$, because every $[D]$ formula in $\mathcal{L}'(v)$ also belongs to $\mathcal{L}'(v_i)$ and there exists at least one ψ_j such that $\langle D\rangle\psi_j\in\mathrm{REQ}(\mathcal{L}'(v))$ and $[D]\neg\psi_j\in\mathrm{REQ}(\mathcal{L}'(v_i))$. For every vertex v_i in this latter set, call the procedure *Expansion* on v_i.

It is easy to show that the algorithm terminates and that it produces a fulfilling $D_{\sqsubset}$-structure $\mathbf{S}'$ for φ. To prove that both the breadth and the depth of $\mathbf{S}'$ are less than or equal to $2\cdot|\varphi|$, it suffices to observe that:

- every irreflexive vertex has exactly one outgoing edge;
- the number of outgoing edges of reflexive vertices is bounded by the number of $\langle D\rangle$ formulas in $\mathrm{CL}(\varphi)$, not exceeding the size of the formula;
- in *Step 2*, the procedure *Expansion* is called only on those vertices v_i such that $\mathrm{REQ}(\mathcal{L}'(v_i))\subset\mathrm{REQ}(\mathcal{L}'(v))$. It follows that at every step the number of $\langle D\rangle$ formulas strictly decreases. As a consequence, we have that every path in $\mathbf{S}'$ devoid of repetitions includes at most $|\varphi|$ different irreflexive vertices. Since in every repetition-free path reflexive and irreflexive vertices alternate, the depth of the $D_{\sqsubset}$-structure is bounded by $2\cdot|\varphi|$. □

Given a formula φ, let n be the number of $\langle D\rangle$ formulas in $\mathrm{CL}(\varphi)$. From Theorem 3 it follows that there exists a fulfilling $D_{\sqsubset}$-structure for φ whose repetition-free vertex paths $v_0,...,v_k$, starting from the root, have length at most $2n$.

A PSPACE non-deterministic algorithm for satisfiability checking of $D_{\sqsubset}$ formulas can be defined as follows.

procedure $D_{\sqsubset}$-sat(φ)
  A = a non-deterministically generated atom containing φ;
  $D_{\sqsubset}$-step(A);

procedure $D_{\sqsubset}$-step(A)
  A' = a non-deterministically generated reflexive
  atom such that $A\ D_\varphi\ A'$ and $\mathrm{REQ}(A')=\mathrm{REQ}(A)$;
  $\forall\langle D\rangle\psi\in\mathrm{REQ}(A')$ **do**
    A'' = a non-deterministically generated atom
    such that $A'\ D_\varphi\ A''$ and $\psi\in A''$;

if REQ(A'') $\neq$ REQ(A') **then** $D_{\sqsubset}$-step(A");

The above procedure fails whenever an atom with the requested properties cannot be generated. It is easy to show that it has a successful (terminating) computation if and only if there exists a fulfilling $D_{\sqsubset}$-structure for φ. The procedure does not exceed polynomial space because the number of nested calls of the procedure $D_{\sqsubset}$-step is bounded by $\mathcal{O}(n)$ (the maximum length of a repetition-free path) and every call needs $\mathcal{O}(n)$ memory space for local operations.

3.3 A tableau method for $D_{\sqsubset}$

A tableau for a $D_{\sqsubset}$ formula φ is a finite, tree-like graph, in which every node is a subset of CL(φ). Nodes are grouped into *macronodes*, that is, finite subtrees of the tableau, dealt with by the expansion rules. Branching inside a macronode corresponds to disjunctions. Macronodes and edges connecting them represent vertices and edges in the $D_{\sqsubset}$-structure for φ. We distinguish two types of rules: *local rules*, that generate new nodes belonging to the same macronode, and *global rules*, that generate new nodes belonging to new macronodes.

Given two nodes n and n' such that n' is a descendant of n, we say that n' is a *local descendant* of n (or, equivalently, that n is a *local ancestor* of n') if n and n' belong to the same macronode and that n' is a *global descendant* of n (n is a *global ancestor* of n') if n and n' belong to different macronodes.

Local Rules.

(NOT) $\dfrac{\neg\neg\psi, F}{\psi, F}$ (OR) $\dfrac{\psi_1 \vee \psi_2, F}{\psi_1, F \mid \psi_2, F}$ (AND) $\dfrac{\neg(\psi_1 \vee \psi_2), F}{\neg\psi_1, \neg\psi_2, F}$

(REFL) $\dfrac{[D]\psi, F}{\psi, [D]\psi, F}$ where ψ does not occur in any local ancestor of the node

Global Rules.

(2-DENS) $\dfrac{[D]\psi_1, \ldots, [D]\psi_m, \langle D\rangle\varphi_1, \ldots, \langle D\rangle\varphi_n, F}{\psi_1, \ldots, \psi_m, [D]\psi_1, \ldots, [D]\psi_m, \langle D\rangle\varphi_1, \ldots, \langle D\rangle\varphi_n}$
where $m, n \geq 0$ and F contains no temporal formulas;

(STEP) $\dfrac{[D]\psi_1, \ldots, [D]\psi_m, \langle D\rangle\varphi_1, \ldots, \langle D\rangle\varphi_n, F}{G_1 \mid \ldots \mid G_n}$
where $m \geq 0$, $n > 0$, F contains no temporal formulas and, for every $i = 1, \ldots, n$, $G_i = \{\varphi_i, \psi_1, \ldots, \psi_m, [D]\psi_1, \ldots, [D]\psi_m\}$

Reflexive and irreflexive macronodes. Like the vertices of a $D_{\sqsubset}$-graph, macronodes can be either reflexive or irreflexive. A macronode is *irreflexive* if (i) the macronode contains the initial node of the tableau or (ii) the macronode has been created by an application of the STEP rule. In the other cases, viz., when the macronode has been created by an application of the 2-DENS rule, the macronode is *reflexive*. A node of the tableau is reflexive (resp., irreflexive) if it belongs to a reflexive (resp., irreflexive) macronode.

Expansion strategy. Given a formula φ, the tableau for φ is obtained from the one-node initial tableau $\langle\{\{\varphi\}\}, \emptyset\rangle$ by recursively applying the following expansion strategy, until it cannot be applied anymore:

1. do not apply the expansion rules to nodes of the tableau containing both ψ and $\neg\psi$, for some $\psi \in \mathrm{CL}(\varphi)$;
2. apply the NOT, OR, and AND rules to both reflexive and irreflexive nodes;
3. apply the 2-DENS rule to irreflexive nodes;
4. apply the REFL and STEP rules to reflexive nodes;

To keep the construction finite, when the application of the 2-DENS (resp., STEP) rule to a node n would generate a new reflexive (resp., irreflexive) node such that there exists another reflexive (resp., irreflexive) node n' in the tableau with the same set of formulas, add an edge from n to n' instead of generating such a new node.

Open and closed tableau. A node n in a tableau $\mathcal{T}$ is *closed* if one of the following conditions holds:

1. there exists a formula $\psi \in \mathrm{CL}(\varphi)$ such that $\psi, \neg\psi \in n$;
2. in the tableau construction, the NOT, OR, AND, 2-DENS, or REFL rules have been applied to n and *all* the immediate successors of n are closed;
3. in the tableau construction, the STEP rule has been applied to n and *at least one* of the immediate successors of n is closed.

A tableau $\mathcal{T}$ is *closed* if its initial node is closed; otherwise, it is *open*.

Example of application. The construction of the tableau is exemplified in Figure 4, where a closed tableau for the unsatisfiable formula $\langle D\rangle\langle D\rangle p \wedge \langle D\rangle q \wedge [D]([D]\neg p \vee [D]\neg q)$ is reported. For the sake of readability, macronodes are represented by boxes and formulas to which local rules are applied are underlined.

3.4 Termination, complexity, soundness, and completeness

The following theorem ensures the termination of the proposed tableau method:

Theorem 4. *Given a* $\mathrm{D}_{\sqsubset}$ *formula* φ*, every tableau* $\mathcal{T}$ *for* φ *is finite.*

Theorem 4 easily follows from an analysis of the construction rules:

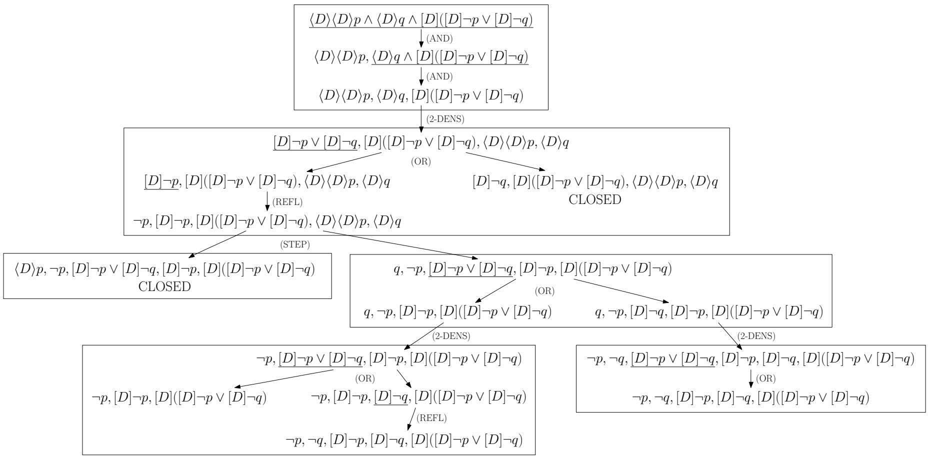


Fig. 4. The tableau for $\langle D\rangle\langle D\rangle p \wedge \langle D\rangle q \wedge [D]([D]\neg p \vee [D]\neg q)$.

- the local rules can be applied only finitely many times to the nodes of a macronode (the application of the NOT, AND, and OR rules to a node generates one or two nodes where one of the formulas in the original node has been replaced by one or two formulas of strictly lower size; moreover, for any branch in the subtree associated with a macronode and any $[D]$ formula in $\mathrm{CL}(\varphi)$, the REFL rule can be applied at most one time);
- the 2-DENS (resp., STEP) rule generates a new node only when the set of temporal formulas of such a node is different from that of the other reflexive (resp. irreflexive) nodes in the tableau, and thus it can be applied only finitely many times.

As for the complexity, we have shown that a formula φ is satisfiable if and only if there exists a $\mathrm{D}_{\sqsubseteq}$-structure for it whose breadth and depth are linear in $|\varphi|$. The same property holds for the tableau method as formally stated by the following theorem.

Theorem 5. *Let $\mathcal{T}$ be a tableau for a $\mathrm{D}_{\sqsubseteq}$ formula φ. Then, the depth and the breadth of $\mathcal{T}$ are linear in $|\varphi|$.*

It is easy to prove that such a bound holds for any tableau $\mathcal{T}$ for φ. Let n be the number of $\langle D\rangle$ formulas in $\mathrm{CL}(\varphi)$. The number of outgoing edges of a node is bounded by n. Moreover, as in the case of $\mathrm{D}_{\sqsubseteq}$-structures, every repetition-free path of macronodes starting from the root is of length at most $2 \cdot n$. Hence, both the breadth and the depth of $\mathcal{T}$ are linear in $|\varphi|$.

It immediately follows that any tableau $\mathcal{T}$ for φ can be non-deterministically generated and explored by using a polynomial amount of space. Thus, we obtain the following corollary.

Corollary 1. *The proposed tableau method for $\mathrm{D}_{\sqsubseteq}$ has a PSPACE complexity.*

We conclude the section by proving soundness and completeness of the proposed tableau-based decision procedure.

Theorem 6. *(*SOUNDNESS*) Let φ be a $\mathrm{D}_{\sqsubset}$ formula and $\mathcal{T}$ be a tableau for it. If $\mathcal{T}$ is open, then φ is satisfiable.*

Proof. Let $\mathcal{T}$ be an open tableau for φ. We build a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ for φ in an incremental way, starting from the root n_0 of $\mathcal{T}$. Since $\mathcal{T}$ is open, then n_0 is not closed. We generate a one-node $\mathrm{D}_{\sqsubset}$-graph $\langle\{v_0\}, \emptyset\rangle$ and we put the formulas that belong to n_0 in $\mathcal{L}(v_0)$. Now, let n be a non-closed node in $\mathcal{T}$ and let v be the corresponding vertex in the $\mathrm{D}_{\sqsubset}$-graph. We distinguish four possible cases, depending on the expansion rule that has been applied to n in the tableau construction.

- *One among the NOT, AND, and REFL rules has been applied to n.* Since n is not closed, its unique successor n' is not closed either. We add the formulas contained in n' to $\mathcal{L}(v)$ and then we proceed by taking into consideration the node n' and the corresponding vertex v (notice that different nodes can be associated with the same vertex of the $\mathrm{D}_{\sqsubset}$-structure).
- *The OR rule has been applied to n.* Since n is not closed, it has at least one successor n' that is not closed either. We add the formulas contained in n' to $\mathcal{L}(v)$ and then we proceed by taking into consideration the node n' and the corresponding vertex v.
- *The 2-DENS rule has been applied to n.* Since n is not closed, its unique successor n' is not closed either. We must distinguish two cases: either n' has been already taken into account during the construction of the $\mathrm{D}_{\sqsubset}$-structure or not. In the former case, we simply add an edge (v, v') to E, where v' is the vertex corresponding to n' in the $\mathrm{D}_{\sqsubset}$-graph. In the latter case, we add a new reflexive vertex v' to V, we add the edges (v, v') and (v', v') to E, and we put the formulas belonging to n' in $\mathcal{L}(v')$. Then, we proceed by taking into consideration the node n' and the corresponding vertex v'.
- *The STEP rule has been applied to n.* Since n is not closed, all its successors $n_1, \ldots, n_k$ are not closed either. For every n_i, we must distinguish two cases: either n' has been already taken into account during the construction of the $\mathrm{D}_{\sqsubset}$-structure or not. In the former case, there exists a vertex v_i that corresponds to n_i in the $\mathrm{D}_{\sqsubset}$-graph and we simply add an edge (v, v_i) to E. In the latter case, we add an irreflexive vertex v_i to V, we add the edge (v, v_i) to E, and we put the formulas belonging to n_i in $\mathcal{L}(v_i)$. Then, we proceed by taking into consideration the node n_i and the corresponding vertex v_i.

Since any tableau for φ is finite, such a construction process terminates. However, the resulting pair $\langle\langle V, E\rangle, \mathcal{L}\rangle$ is not necessarily a $\mathrm{D}_{\sqsubset}$-structure: while $\langle V, E\rangle$ respects the definition of $\mathrm{D}_{\sqsubset}$-graph, the function $\mathcal{L}$ is not necessarily a labeling function. Since in the tableau construction we add new formulas only when necessary, there may exist a vertex $v \in V$ and a formula $\psi \in \mathrm{CL}(\varphi)$ such that neither ψ nor $\neg\psi$ belongs to $\mathcal{L}(v)$. Let $v \in V$ and $\psi \in \mathrm{CL}(\varphi)$ such that neither ψ nor $\neg\psi$ belongs to $\mathcal{L}(v)$. We can complete the definition of $\mathcal{L}$ as follows (by induction on the complexity of ψ):

– if $\psi = p$, with $p \in \mathcal{AP}$, we put $\neg p \in \mathcal{L}(v)$;
– If $\psi = \neg\xi$, we put $\psi \in \mathcal{L}(v)$ if and only if $\xi \notin \mathcal{L}(v)$;
– If $\psi = \psi_1 \vee \psi_2$, we put $\psi_1 \vee \psi_2 \in \mathcal{L}(v)$ if and only if $\psi_1 \in \mathcal{L}(v)$ or $\psi_2 \in \mathcal{L}(v)$;
– If $\psi = \langle D\rangle\xi$, we put $\psi \in \mathcal{L}(v)$ if and only if there exists a descendant v' of v such that $\xi \in \mathcal{L}(v')$.

The resulting $\mathrm{D}_{\sqsubset}$-structure $\langle\langle V, E\rangle, \mathcal{L}\rangle$ is a fulfilling $\mathrm{D}_{\sqsubset}$-structure for φ. From Theorem 2, it follows that φ is satisfiable. □

Theorem 7. (COMPLETENESS) *Let φ be a $\mathrm{D}_{\sqsubset}$ formula and $\mathcal{T}$ be a tableau for it. If $\mathcal{T}$ is closed, then φ is unsatisfiable.*

Proof. We prove a stronger claim. Given a node n in a tableau, we say that (the set of formulas belonging to) n is *consistent* if there exists a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ such that one of the following conditions holds:

– n belongs to an irreflexive macronode and there exists an irreflexive vertex $v \in V$ such that $\mathcal{L}(v)$ contains all formulas in n;
– n belongs to a reflexive macronode and there exists a reflexive vertex $v \in V$ such that $\mathcal{L}(v)$ contains all formulas in n.

We prove that for any node n in a tableau $\mathcal{T}$, if n is closed, then n is inconsistent.

If there exists a formula $\psi \in \mathrm{CL}(\varphi)$ such that n contains both ψ and $\neg\psi$, then n is obviously inconsistent. In the other cases, we proceed by induction, from the leaves to the root, on the expansion rule that has been applied to the node n in the construction of the tableau. Since any tableau is finite, we eventually reach the initial node of $\mathcal{T}$, thus concluding that φ is an inconsistent formula.

– *The NOT rule has been applied to n.* Then n is of the form $\neg\neg\psi, F$ and it has a unique successor $n' = \psi, F$. If n' is closed then, by inductive hypothesis, ψ, F is an inconsistent set. Hence, $\neg\neg\psi, F$ is inconsistent.
– *The OR rule has been applied to n.* Then n is of the form $\psi_1 \vee \psi_2, F$ and it has two immediate successors $n_1 = \psi_1, F$ and $n_2 = \psi_2, F$. If both n_1 and n_2 are closed then, by inductive hypothesis, both ψ_1, F and ψ_2, F are inconsistent sets. Hence, $\psi_1 \vee \psi_2, F$ is inconsistent.
– *The AND rule has been applied to n.* Then n is of the form $\neg(\psi_1 \vee \psi_2), F$ and it has a unique successor $n' = \neg\psi_1, \neg\psi_2, F$. If n' is closed then, by inductive hypothesis, $\neg\psi_1, \neg\psi_2, F$ is an inconsistent set. Hence, $\neg\psi_1 \wedge \neg\psi_2, F$ is inconsistent.
– *The REFL rule has been applied to n.* In this case, n belongs to a reflexive macronode and it is of the form $[D]\psi, F$. Suppose, for contradiction, that n is closed, but consistent. This implies that there exists a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ and a reflexive vertex $v \in V$ such that $n \subseteq \mathcal{L}(v)$. Since v is reflexive, we have that $(v, v) \in E$ and thus $\mathcal{L}(v)\ D_{\varphi}\ \mathcal{L}(v)$. Hence, we have that $\psi \in \mathcal{L}(v)$ and thus the set of formulas $\psi, [D]\psi, F$ is consistent. By inductive hypothesis, this implies that the successor n' of n is not closed, which contradicts the hypothesis that n is closed.

– *The 2-DENS rule has been applied to n.* In this case, n belongs to an irreflexive macronode and it is of the form $[D]\psi_1, \ldots, [D]\psi_m, \langle D\rangle\varphi_1, \ldots, \langle D\rangle\varphi_h, F$. Suppose, for contradiction, that n is closed, but consistent. Hence, there exists a fulfilling $\mathrm{D}_{\sqsubseteq}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ and an irreflexive vertex $v \in V$ such that $n \subseteq \mathcal{L}(v)$. By the definition of $\mathrm{D}_{\sqsubseteq}$-structure, v has a reflexive successor v' such that $\mathcal{L}(v)\ D_\varphi\ \mathcal{L}(v')$. Hence, v' is such that $\{\psi_1, \ldots, \psi_m, [D]\psi_1, \ldots, [D]\psi_m, \langle D\rangle\varphi_1, \ldots, \langle D\rangle\varphi_h\} \subseteq \mathcal{L}(v')$. This proves that the successor n' of n in the tableau is consistent. By inductive hypothesis, n' is not closed, which contradicts the hypothesis that n is closed.
– *The STEP rule has been applied to n.* In this case, n belongs to a reflexive macronode and it is of the form $[D]\psi_1, \ldots, [D]\psi_m, \langle D\rangle\varphi_1, \ldots, \langle D\rangle\varphi_h, F$. Suppose, for contradiction, that n is closed, but consistent. Hence, there exists a fulfilling $\mathrm{D}_{\sqsubseteq}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}\rangle$ and a reflexive vertex $v \in V$ such that $n \subseteq \mathcal{L}(v)$. Since $\mathbf{S}$ is fulfilling, for every formula $\langle D\rangle\varphi_i$ there exists a descendant v_i of v such that $\varphi_i \in \mathcal{L}(v_i)$. This implies that, for every $i = 1, \ldots, h$, the set of formulas $G_i = \{\varphi_i, \psi_1, \ldots, \psi_m, [D]\psi_1, \ldots, [D]\psi_m\}$ is consistent. By inductive hypothesis, this implies that every immediate successor of n is not closed, which contradicts the hypothesis that n is closed.

□

4 The logic $\mathrm{D}_{\sqsubset}$ of proper subinterval structures

4.1 Structures for $\mathrm{D}_{\sqsubset}$ formulas

In this section we extend the notion of $\mathrm{D}_{\sqsubseteq}$-graph and $\mathrm{D}_{\sqsubseteq}$-structures given in Section 3.1 to the case of the logic $\mathrm{D}_{\sqsubset}$.

The presence of the special families of beginning subintervals and ending subintervals complicates the construction of pseudo-models for $\mathrm{D}_{\sqsubset}$. Given an interval $[b, e]$, a *beginning subinterval* of $[b, e]$ is an interval $[b, e']$, with $e' < e$, an *ending subinterval* of $[b, e]$ is an interval $[b', e]$, with $b < b'$, and an *internal subinterval* of $[b, e]$ is an interval $[b', e']$, with $b < b'$ and $e' < e$. To represent infinite chains of beginning (resp., ending) subintervals of a given interval, we need to introduce the notion of *cluster* of reflexive nodes. Given a graph $\mathbb{G} = \langle V, E\rangle$, we define a *cluster* as a maximal strongly connected subgraph $\mathcal{C}$ which includes reflexive vertices only. By abuse of notation, we say that a *cluster $\mathcal{C}$ is a successor of a vertex v* if v does not belong to $\mathcal{C}$ and there exists a successor v' of v in $\mathcal{C}$. Conversely, a *vertex v is a successor of $\mathcal{C}$* if v does not belong to $\mathcal{C}$ and there exists a predecessor v' of v in $\mathcal{C}$. The definition of $\mathrm{D}_{\sqsubseteq}$-graph can be extended to $\mathrm{D}_{\sqsubset}$ as follows.

Definition 7. *A finite directed graph $\mathbb{G} = \langle V, E\rangle$ is a* $\mathrm{D}_{\sqsubset}$-graph *if:*
1. *there exists an irreflexive vertex $v_0 \in V$, called the* root *of $\mathbb{G}$, such that any other vertex $v \in V$ is reachable from it;*
2. *every irreflexive vertex $v \in V$ has exactly two clusters as successors: a* beginning successor cluster *$\mathcal{C}_b$ and an* ending successor cluster *$\mathcal{C}_e$;*

3. $\mathcal{C}_b$ *and* $\mathcal{C}_e$ *have a unique common successor* v_c*, which is a reflexive vertex;*
4. *every successor of* v_c*, different from* v_c *itself, is irreflexive;*
5. *there exists at most one edge exiting the clusters* $\mathcal{C}_b$ *and* $\mathcal{C}_e$ *and reaching an irreflexive node;*
6. *apart from the edge leading to* v_c*, there are no edges exiting from* $\mathcal{C}_b$ *(resp.* $\mathcal{C}_e$*) that reach a reflexive vertex.*

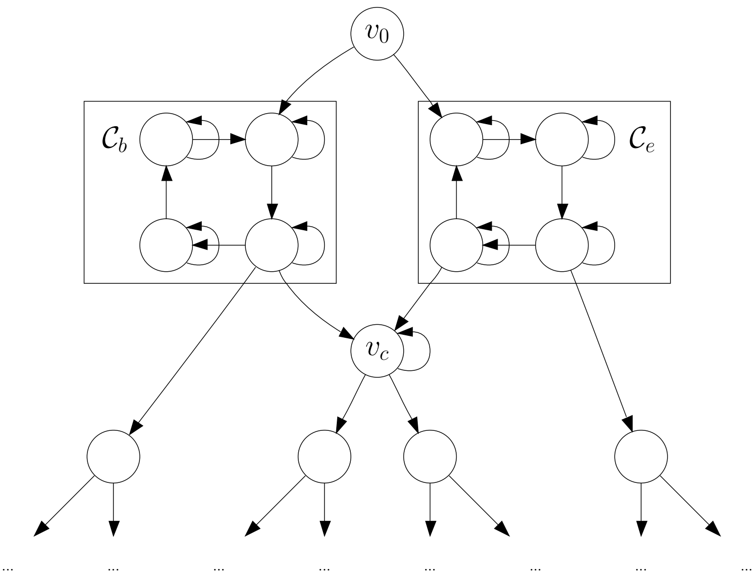


Fig. 5. An example of $\mathrm{D}_{\sqsubset}$-graph.

Figure 5 depicts a portion of a $\mathrm{D}_{\sqsubset}$-graph. The root v_0 has two successor clusters $\mathcal{C}_b$ and $\mathcal{C}_e$ of four vertices each. Both $\mathcal{C}_b$ and $\mathcal{C}_e$ have exactly one irreflexive successor. Their common reflexive successor v_c has two irreflexive successors.

Let φ be a $\mathrm{D}_{\sqsubset}$ formula. $\mathrm{D}_{\sqsubset}$-structures are defined by pairing a $\mathrm{D}_{\sqsubset}$-graph with a labeling function that associates an $\mathcal{A}_\varphi$ atom with each vertex of the graph.

Definition 8. *A* $\mathrm{D}_{\sqsubset}$-structure *is a quadruple* $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$*, where:*

1. $\langle V, E\rangle$ *is a* $\mathrm{D}_{\sqsubset}$*-graph;*
2. $\mathcal{L} : V \rightarrow \mathcal{A}_\varphi$ *is a* labeling function *that assigns to every vertex* $v \in V$ *an atom* $\mathcal{L}(v)$ *such that for every edge* $(v, v') \in E$*,* $\mathcal{L}(v)\ D_\varphi\ \mathcal{L}(v')$*;*
3. $\mathcal{B} : V \rightarrow 2^{\mathrm{REQ}_\varphi}$ *and* $\mathcal{E} : V \rightarrow 2^{\mathrm{REQ}_\varphi}$ *are mappings that assign to every vertex the sets of its* beginning *and* ending requests*, respectively;*
4. *for every irreflexive vertex* $v \in V$*, with successor clusters* $\mathcal{C}_b$ *and* $\mathcal{C}_e$*, we have that:*

- *v_c, the common reflexive successor of $\mathcal{C}_b$ and $\mathcal{C}_e$, is such that* $\mathcal{E}(v_c) = \mathcal{B}(v_c) = \emptyset$ *and* $\mathrm{REQ}(\mathcal{L}(v_c)) = \mathrm{REQ}(\mathcal{L}(v)) - (\mathcal{B}(v) \cup \mathcal{E}(v))$*,*
- *every reflexive vertex $v' \in \mathcal{C}_b$ is such that $\mathcal{B}(v') = \mathcal{B}(v)$, $\mathcal{E}(v') = \emptyset$, and* $\mathrm{REQ}(\mathcal{L}(v')) = \mathrm{REQ}(\mathcal{L}(v_c)) \cup \mathcal{B}(v)$*,*
- *the unique irreflexive successor v'' of $\mathcal{C}_b$ (if any) is such that $\mathcal{B}(v) \cap \mathcal{L}(v'') \subseteq \mathcal{B}(v'')$ (requests which have been classified as initial in a given vertex cannot be reclassified in its descendants),*
- *every reflexive vertex $v' \in \mathcal{C}_e$ is such that $\mathcal{E}(v') = \mathcal{E}(v)$, $\mathcal{B}(v') = \emptyset$, and* $\mathrm{REQ}(\mathcal{L}(v')) = \mathrm{REQ}(\mathcal{L}(v_c)) \cup \mathcal{E}(v)$*,*
- *the unique irreflexive successor v'' of $\mathcal{C}_e$ (if any) is such that $\mathcal{E}(v) \cap \mathcal{L}(v'') \subseteq \mathcal{E}(v'')$ (requests which have been classified as ending in a given vertex cannot be reclassified in its descendants).*

Let v_0 be the root of $\langle V, E\rangle$. If $\varphi \in \mathcal{L}(v_0)$, we say that $\mathbf{S}$ is a $\mathrm{D}_{\sqsubset}$-structure for φ.

Beginning and ending requests associated with a vertex v can be viewed as requests that must be satisfied over respectively beginning and ending subintervals of any interval corresponding to v (possibly over both of them), but not over its internal subintervals.

Every $\mathrm{D}_{\sqsubset}$-structure can be regarded as a Kripke model for $\mathrm{D}_{\sqsubset}$, where the valuation is determined by the labeling. As in the case of $\mathrm{D}_{\sqsubseteq}$-structures, we restrict our attention to fulfilling structures.

Definition 9. *A $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ is* fulfilling *if for every $v \in V$ and every $\langle D\rangle\psi \in \mathcal{L}(v)$, there exists $v' \in V$ such that v' is a descendant of v and $\psi \in \mathcal{L}(v')$.*

Theorem 8. *Let φ be a $\mathrm{D}_{\sqsubset}$ formula which is satisfied in a proper interval model. Then, there exists a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ for φ.*

Proof. Let $\mathbf{M} = \langle \mathbb{I}(\mathbb{D}), \sqsubset, \mathcal{V}\rangle$ be a proper interval model and let $[b_0, e_0] \in \mathbb{I}(\mathbb{D})$ be an interval such that $\mathbf{M}, [b_0, e_0] \Vdash \varphi$. We recursively build a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ for φ as follows.

We start with the one-node graph $\langle \{v_0\}, \emptyset\rangle$ and a labeling function $\mathcal{L}$ such that $\mathcal{L}(v_0) = \{\psi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_0, e_0] \Vdash \psi\}$. Then, we partition the set $\mathrm{REQ}(\mathcal{L}(v_0))$ into the following three sets of formulas:

Beginning requests: B_{v_0} contains all $\langle D\rangle\xi \in \mathrm{REQ}(\mathcal{L}(v_0))$ such that ξ is satisfied over beginning subintervals of $[b_0, e_0]$, but not over internal subintervals of $[b_0, e_0]$;

Ending requests: E_{v_0} contains all $\langle D\rangle\xi \in \mathrm{REQ}(\mathcal{L}(v_0))$ such that ξ is satisfied over ending subintervals of $[b_0, e_0]$, but not over internal subintervals of $[b_0, e_0]$;

Internal requests: $I_{v_0} = (\mathrm{REQ}(\mathcal{L}(v_0)) \setminus B_{v_0}) \setminus E_{v_0}$, that is, the set of all $\langle D\rangle\xi \in \mathrm{REQ}(\mathcal{L}(v_0))$ such that ξ is satisfied over internal subintervals of $[b_0, e_0]$.

We put $\mathcal{B}(v_0) = B_{v_0}$ and $\mathcal{E}(v_0) = E_{v_0}$. Then, for every formula $\langle D\rangle\psi \in \mathcal{L}(v_0)$, we choose an interval $[b_\psi, e_\psi]$, with $[b_\psi, e_\psi] \sqsubset [b_0, e_0]$, such that $\mathbf{M}, [b_\psi, e_\psi] \Vdash \psi$. If

$\langle D\rangle\psi \in I_{v_0}$, then $b_0 < b_\psi < e_\psi < e_0$, else if $\langle D\rangle\psi \in B_{v_0}$, then $b_0 = b_\psi < e_\psi < e_0$, else ($\langle D\rangle\psi \in E_{v_0}$) $b_0 < b_\psi < e_\psi = e_0$.

Since $\mathbb{D}$ is a dense ordering and $\mathrm{CL}(\varphi)$ is a finite set of formulas, there exist two beginning intervals $[b_0, e_1]$ and $[b_0, e_2]$ such that:

- for every interval $[b_\psi, e_\psi]$, with $\langle D\rangle\psi \in B_{v_0} \cup I_{v_0}$, $[b_\psi, e_\psi] \sqsubset [b_0, e_2] \sqsubset [b_0, e_1]$;
- $[b_0, e_1]$ and $[b_0, e_2]$ satisfy the same formulas of $\mathrm{CL}(\varphi)$.

We start the construction of the beginning successor cluster $\mathcal{C}_b$ of v_0 by adding a new vertex v_b and a pair of edges (v_0, v_b) and (v_b, v_b), and by putting $\mathcal{L}(v_b) = \{\xi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_0, e_1] \Vdash \xi\}$, $\mathcal{B}(v_b) = B_{v_0}$ and $\mathcal{E}(v_b) = \emptyset$. Next, for every $\langle D\rangle\psi \in \mathcal{B}(v_b)$ we establish e whether or not we must add a vertex v_ψ in $\mathcal{C}_b$ as follows. Let $[b_0, e_\psi]$ be a beginning subinterval such that $\mathbf{M}, [b_0, e_\psi] \Vdash \psi$. We add a reflexive vertex v_ψ to $\mathcal{C}_b$ if $[b_0, e_\psi]$ satisfies the same temporal formulas $[b_0, e_1]$ satisfies. Moreover, we put $\mathcal{L}(v_\psi) = \{\xi \in \mathrm{CL}(\varphi) : \mathbf{M},\ [b_0, e_\psi] \Vdash \xi\}$, $\mathcal{B}(v_\psi) = \mathcal{B}(v_b)$, and $\mathcal{E}(v_\psi) = \emptyset$. Let $\{v_1, \ldots, v_k\}$ be the resulting set of vertices added to $\mathcal{C}_b$. For $i = 1, \ldots, k-1$, we add an edge (v_i, v_{i+1}) to E; furthermore, we add the edges (v_b, v_1) and (v_k, v_b) to E. If for all formulas $\langle D\rangle\psi \in \mathcal{B}(v_b)$ there exists a corresponding vertex v_ψ in $\mathcal{C}_b$, we are done. Otherwise, let Γ_B be the set of the remaining formulas $\langle D\rangle\psi \in \mathcal{B}(v_b)$ and let $[b_0, e_B^{max}]$ be a beginning subinterval such that, for every formula $\langle D\rangle\psi \in \Gamma_B$, we have that $\mathbf{M}, [b_0, e_B^{max}] \Vdash \psi$ or $\mathbf{M}, [b_0, e_B^{max}] \Vdash \langle D\rangle\psi$. We add a new irreflexive vertex v_b^{max} and an edge connecting an arbitrary vertex in $\mathcal{C}_b$ to it, say (v_b, v_b^{max}), and we define its labeling as $\mathcal{L}(v_b^{max}) = \{\xi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_0, e_B^{max}] \Vdash \xi\}$.

The ending successor cluster $\mathcal{C}_e$ of v_0 is built in the very same way.

To complete the first phase of the construction, we must introduce the common reflexive successor v_c of $\mathcal{C}_b$ and $\mathcal{C}_e$. Since $\mathbb{D}$ is a dense ordering and $\mathrm{CL}(\varphi)$ is a finite set of formulas, there exist two intervals $[b_3, e_3]$ and $[b_4, e_4]$ such that:

- for every interval $[b_\psi, e_\psi]$, with $\langle D\rangle\psi \in I_{v_0}$, $[b_\psi, e_\psi] \sqsubset [b_4, e_4] \sqsubset [b_3, e_3]$;
- $[b_3, e_3]$ and $[b_4, e_4]$ satisfy the same formulas of $\mathrm{CL}(\varphi)$.

We add a new vertex v_c, together with the edges (v_b, v_c), (v_e, v_c), and (v_c, v_c), and we put $\mathcal{L}(v_c) = \{\xi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_3, e_3] \Vdash \xi\}$, $\mathcal{B}(v_c) = \mathcal{E}(v_c) = \emptyset$.

For every formula $\langle D\rangle\psi \in I_{v_0}$, we add a new vertex v_ψ and an edge (v_c, v_ψ), and we define its labeling as $\mathcal{L}(v_\psi) = \{\xi \in \mathrm{CL}(\varphi) : \mathbf{M}, [b_\psi, e_\psi] \Vdash \xi\}$.

Then, we recursively apply the above procedure to the irreflexive vertices we have introduced. To keep the construction finite, whenever there exists an irreflexive vertex $v' \in V$ such that $\mathcal{L}(v_\psi) = \mathcal{L}(v')$ for some v_ψ, we simply add an edge to v' instead of creating a new vertex v_ψ and an edge entering it. Since the set of atoms is finite, the construction is guaranteed to terminate. $\square$

Let $\mathbf{S}$ be a fulfilling $\mathrm{D}_\sqsubset$-structure for a formula φ. To build a model for φ, we consider the interval $[0, 1]$ of the rational line and define a function $f_\mathbf{S}$ mapping intervals in $\mathbb{I}([0, 1])$ to vertices in $\mathbf{S}$.

Definition 10. *Let* $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ *be a* $\mathrm{D}_{\sqsubset}$*-structure. The function* $f_{\mathbf{S}} : \mathbb{I}([0,1]) \mapsto V$ *is defined recursively as follows. First,* $f_{\mathbf{S}}([0,1]) = v_0$*. Now, let* $[b, e]$ *be an interval such that* $f_{\mathbf{S}}([b, e]) = v$ *and* $f_{\mathbf{S}}$ *has not been yet defined over any of its subinterval. We distinguish two cases.*

Case 1: *v is an irreflexive vertex. Let $\mathcal{C}_b$ and $\mathcal{C}_e$ be the reflexive successor beginning and ending clusters of v, respectively, and v_c be their common reflexive successor. Let v_b^{max} be the irreflexive successor of $\mathcal{C}_b$ (if any), v_e^{max} be the irreflexive successor of $\mathcal{C}_e$ (if any), and $v_1, \ldots, v_k$ be the k irreflexive successors of v_c (if any). Let $p = \frac{e-b}{2k+3}$. The function $f_{\mathbf{S}}$ is defined as follows (the definition is very close to that for $\mathrm{D}_{\sqsubseteq}$-structures given in Section 3):*

1. *we put $f_{\mathbf{S}}([b, b+p]) = v_b^{max}$ and $f_{\mathbf{S}}([e-p, e]) = v_e^{max}$;*
2. *for every $i = 1, \ldots, k$, we put $f_{\mathbf{S}}([b+2ip, b+(2i+1)p]) = v_i$;*
3. *for every $i = 1, \ldots, k+1$, we put $f_{\mathbf{S}}([b+(2i-1)p, b+2ip]) = v_c$;*
4. *for every* strict *subinterval $[b', e']$ of $[b, e]$ which is not a subinterval of any of the intervals $[b+ip, b+(i+1)p]$, we put $f_{\mathbf{S}}([b', e']) = v_c$.*

To complete the construction, we need to define $f_{\mathbf{S}}$ over the beginning subintervals $[b, e']$ such that $b + p < e' < e$ and the ending subintervals $[b', e]$ such that $b < b' < e - p$. We map such beginning (resp., ending) subintervals to vertices in $\mathcal{C}_b$ (resp., $\mathcal{C}_e$) in such a way that for any beginning subinterval $[b, e']$ (resp., ending subinterval $[b', e]$) and any $v_b \in \mathcal{C}_b$ (resp., $v_e \in \mathcal{C}_e$), there exists a beginning subinterval $[b, e'']$, with $[b, b+p] \sqsubset [b, e''] \sqsubset [b, e']$ (resp., ending subinterval $[b'', e]$, with $[e-p, e] \sqsubset [b'', e] \sqsubset [b', e]$) such that $f_{\mathbf{S}}([b, e'']) = v_b$ (resp., $f_{\mathbf{S}}([b'', e]) = v_e$)[1].

Case 2: *v is a reflexive vertex. The case in which v belongs to $\mathcal{C}_b$ or $\mathcal{C}_e$ has been already dealt with. Thus, we only need to consider the case of vertices v_c with irreflexive successors only (apart from themselves). We distinguish two cases:*

1. v_c has no successors apart from itself. *In such a case, we put $f_{\mathbf{S}}([b', e']) = v_c$ for every subinterval $[b', e']$ of $[b, e]$.*
2. v_c has at least one successor different from itself. *Let $v_c^1, \ldots, v_c^k$ be the k successors of v_c different from v_c. We consider the intervals defined by the points $b, b+p, b+2p \ldots, b+2kp, b+(2k+1)p = e$, with $p = \frac{e-b}{2k+1}$. The function $f_{\mathbf{S}}$ over such intervals is defined as follows:*
   - *for every $i = 1, \ldots, k$, we put $f_{\mathbf{S}}([b+(2i-1)p, b+2ip]) = v_c^i$.*
   - *for every $i = 0, \ldots, k$, we put $f_{\mathbf{S}}([b+2ip, b+(2i+1)p]) = v_c$.*

   We complete the construction by putting $f_{\mathbf{S}}([b', e']) = v_c$ for every subinterval $[b', e']$ of $[b, e]$ which is not a subinterval of any of the intervals $[b+ip, b+(i+1)p]$.

As in the case of $\mathrm{D}_{\sqsubseteq}$-structures, the function $f_{\mathbf{S}}$ satisfies some basic properties.

Lemma 2.

[1] Notice that the density of the rational interval $[0, 1]$ plays here an essential role.

1. *For every interval $[b,e] \in \mathbb{I}([0,1])$, if $f_{\mathbf{S}}([b,e]) = v$ and v' is reachable from v, then there exists an interval $[b',e']$ such that $f_{\mathbf{S}}([b',e']) = v'$ and $[b',e'] \sqsubset [b,e]$.*
2. *For every pair of intervals $[b,e]$ and $[b',e']$ in $\mathbb{I}([0,1])$ such that $[b',e'] \sqsubset [b,e]$, we have that for every formula $[D]\psi \in \mathcal{L}(f_{\mathbf{S}}([b,e]))$, both ψ and $[D]\psi$ belong to $\mathcal{L}(f_{\mathbf{S}}([b',e']))$.*

Proof. Condition 1 can be easily proved by observing that it trivially holds for all successors of v by definition of $f_{\mathbf{S}}$ and then extending the result to every descendant v' of v by induction on the length of the shortest path from v to v'.

As for condition 2, let $[b,e]$ and $[b',e']$ be two intervals in $\mathbb{I}([0,1])$ such that $[b',e'] \sqsubset [b,e]$, $v = f_{\mathbf{S}}([b,e])$, and $v' = f_{\mathbf{S}}([b',e'])$. If v' is a descendant of v in the $\mathrm{D}_{\sqsubset}$-graph, then condition 2 holds by definition of D_{φ}. When we apply the construction step defined by Case 1, Point 4, of Definition 10, it may happen that $[b',e'] \sqsubset [b,e]$ but v' is not reachable from v in the $\mathrm{D}_{\sqsubset}$-graph. In such a case, both $[b,e]$ and $[b',e']$ are internal subintervals, and thus, by definition of the labeling functions $\mathcal{B}$ and $\mathcal{E}$, Condition 2 is satisfied. □

Theorem 9. *Given any fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S}$ for φ, there exists an interval model $\mathbf{M}_{\mathbf{S}} = \langle \mathbb{I}([0,1]), \sqsubset, \mathcal{V} \rangle$ over the rational interval $[0,1]$ such that $\mathbf{M}_{\mathbf{S}}, [0,1] \Vdash \varphi$.*

Proof. For every $p \in \mathcal{AP}$, let $\mathcal{V}(p) = \{[b,e] : p \in \mathcal{L}(f_{\mathbf{S}}([b,e]))\}$. We can prove by induction on the structure of formulas $\psi \in \mathrm{CL}(\varphi)$ that for every interval $[b,e] \in \mathbb{I}([0,1])$:

$$\mathbf{M}_{\mathbf{S}}, [b,e] \models \psi \text{ iff } \psi \in \mathcal{L}(f_{\mathbf{S}}([b,e])).$$

The atomic case immediately follows from definition of $\mathcal{V}$. The Boolean cases follow from the definition of atom. Finally, the case of temporal formulas follows from Lemma 2. This allows us to conclude that $\mathbf{M}_{\mathbf{S}}, [0,1] \models \varphi$. □

4.2 A small-model theorem for $\mathrm{D}_{\sqsubset}$-structures

Given a fulfilling $\mathrm{D}_{\sqsubset}$-structure, we can remove from it those vertices which are not necessary to fulfill any $\langle D \rangle$ formula to obtain a smaller $\mathrm{D}_{\sqsubset}$-structure of bounded size, as proved by the following theorem.

Theorem 10. *For every satisfiable $\mathrm{D}_{\sqsubset}$ formula φ, there exists a fulfilling $\mathrm{D}_{\sqsubset}$-structure with breadth and depth bounded by $2 \cdot |\varphi|$.*

Proof. Consider a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S}$. The size of the structure can be safely reduced as follows:

– we remove from every cluster $\mathcal{C}$ all vertices that either do not fulfill any $\langle D \rangle$ formula or fulfill only formulas that are fulfilled by some descendant of it. Let $\mathcal{C}$ be the resulting cluster. We select a minimal subset $\mathcal{C}' \subseteq \mathcal{C}$ that fulfills all formulas that are fulfilled only inside $\mathcal{C}$ and we replace $\mathcal{C}$ with $\mathcal{C}'$ (if $\mathcal{C}'$ is empty, we replace $\mathcal{C}$ with one of its vertices);

– for every common reflexive successor v_c of a pair of clusters, we select a minimal subset of its irreflexive successors whose vertices satisfy all $\langle D \rangle$ formulas in v_c.

The execution of the first removal process produces a $\mathrm{D}_{\sqsubset}$-structure where the size of every cluster is at most $|\varphi|$ and every vertex in a cluster of size at least 2 fulfills some ψ formulas which are not fulfilled elsewhere, while the execution of the second removal process produces a $\mathrm{D}_{\sqsubset}$-structure where every vertex has at most $|\varphi|$ immediate successors.

Since whenever we exit from a cluster or we move from a reflexive node to an irreflexive one the number of requests strictly decreases, we can conclude that the length of every loop-free path is at most $2 \cdot |\varphi|$. ☐

As a direct consequence of Theorem 10, we have that a fulfilling $\mathrm{D}_{\sqsubset}$-structure for a formula φ (if any) can be generated and explored by a non-deterministic procedure that uses only a polynomial amount of space. This gives the following complexity bound to the decision problem for $\mathrm{D}_{\sqsubset}$.

Theorem 11. *The decision problem for* $\mathrm{D}_{\sqsubset}$ *is in PSPACE.*

Proof. A PSPACE non-deterministic algorithm to check the satisfiability of a $\mathrm{D}_{\sqsubset}$ formula φ can be obtained as follows. We non-deterministically generate an atom containing φ, we associate it with the root v of a $\mathrm{D}_{\sqsubset}$-structure, and we guess the subsets $\mathcal{B}(v)$ and $\mathcal{E}(v)$ of $\mathrm{REQ}(\mathcal{L}(v))$. Then, we apply the following recursive procedure on v:

– If the vertex v is irreflexive, we generate three new vertices v_b, v_c and v_e. Then, we non-deterministically generate three labelings for them such that (if such labelings do no exist, the procedure fails):
  (i) v_b, v_c, and v_e are successors of v, that is, $\mathcal{L}(v) D_\varphi \mathcal{L}(v_x)$ for every $x \in \{b, c, e\}$;
  (ii) v_b, v_c, and v_e are reflexive vertices, that is, $\mathcal{L}(v_x) D_\varphi \mathcal{L}(v_x)$ for every $x \in \{b, c, e\}$;
  (iii) v_c is a successor of both v_b and v_e, that is, $\mathcal{L}(v_b)\ D_\varphi\ \mathcal{L}(v_c)$ and $\mathcal{L}(v_e)\ D_\varphi\ \mathcal{L}(v_c)$;
  (iv) $\mathcal{B}(v) \subseteq \mathcal{L}(v_b)$, $\mathcal{E}(v) \subseteq \mathcal{L}(v_e)$, and $\{[D]\neg\psi \mid \langle D \rangle \psi \in \mathcal{B}(v) \cup \mathcal{E}(v)\} \subseteq \mathcal{L}(v_c)$.

  The vertex v_b is the first vertex of the beginning cluster $\mathcal{C}_b$ associated with v, the vertex v_e is the first vertex of the ending cluster $\mathcal{C}_e$, and the vertex v_c is the common reflexive successor of $\mathcal{C}_b$ and $\mathcal{C}_e$. We recursively call the procedure separately on v_b, v_c, and v_e. If one of these three calls returns $fail$, the procedure ends with failure; otherwise, it returns *success*.
– If the vertex v is the common reflexive successor of a pair of clusters, we generate an irreflexive successor v_ψ of it for every $\langle D \rangle \psi \in \mathrm{REQ}(\mathcal{L}(v))$ and we guess a labeling $\mathcal{L}(v_\psi)$ such $\psi \in \mathcal{L}(v_\psi)$ and $\mathcal{L}(v) D_\varphi \mathcal{L}(v_\psi)$ (if no such a labeling exists, the procedure fails). If $\mathrm{REQ}(\mathcal{L}(v_\psi)) \neq \mathcal{L}(v)$, we recursively call the procedure on v_ψ; otherwise, we add an edge from v_ψ to v in the

$\mathrm{D}_{\sqsubset}$-structure. If one of these calls fails, then we return *fail*; otherwise, we return *success*.

– If the node v belongs to a beginning cluster $\mathcal{C}_B$ (the ending case is symmetric), we guess a formula $\langle D\rangle\psi \in \mathcal{B}(v)$ such that $\langle D\rangle\psi \in \mathcal{L}(v)$ and no vertex in $\mathcal{C}_B$ satisfies ψ (if any). If such a formula does not exist, the procedure ends with *success*; otherwise, the procedure non-deterministically chooses to satisfy $\langle D\rangle\psi$ either in $\mathcal{C}_B$ or outside it. In the former case, it adds a new reflexive node v' to $\mathcal{C}_B$ and it guesses a labeling $\mathcal{L}(v')$ such that $\psi \in \mathcal{L}(v')$, $\mathcal{L}(v')\ D_\varphi\ \mathcal{L}(v')$, $\mathcal{L}(v)\ D_\varphi\ \mathcal{L}(v')$, and $\mathrm{REQ}(\mathcal{L}(v')) = \mathrm{REQ}(\mathcal{L}(v))$ (if no such a labeling exists, the procedure fails). In the latter case, it adds an irreflexive successor v' to $\mathcal{C}_B$ and it guesses a labeling $\mathcal{L}(v')$ such that $\psi \in \mathcal{L}(v')$, $\mathcal{L}(v)\ D_\varphi\ \mathcal{L}(v')$ and $\mathrm{REQ}(\mathcal{L}(v')) \subset \mathrm{REQ}(\mathcal{L}(v))$ (if no such a labeling exists, the procedure fails). Finally, we recursively call the procedure on v'.

As for the computational complexity, every call to the procedure needs to store only the path from the root to the current vertex, and it generates at most $|\varphi|$ distinct recursive calls. Moreover, every recursive call either satisfies a $\langle D\rangle\psi$ formula (and thus it strictly decreases the number of remaining $\langle D\rangle$ formulas) or it generates a beginning, an ending, or a 'middle' reflexive vertex. Since every step at which a new beginning, ending, or middle reflexive vertex is generated is followed by a step at which a vertex that satisfies a $\langle D\rangle$ formula is generated, the maximum number of nested calls to the procedure is bounded by $2 \cdot |\varphi|$. This allows us to conclude that the procedure is of PSPACE complexity. □

The very same reduction that has been used to prove $\mathrm{D}_{\dot\sqsubset}$ PSPACE hardness can be applied to $\mathrm{D}_{\sqsubset}$, thus proving the PSPACE completeness of the satisfiability problem for $\mathrm{D}_{\sqsubset}$.

4.3 The tableau method for $\mathrm{D}_{\sqsubset}$

In this section we present a tableau system for $\mathrm{D}_{\sqsubset}$. From the model-theoretic results in the previous section, we have that a $\mathrm{D}_{\sqsubset}$ formula φ is satisfiable if and only if there exists a fulfilling $\mathrm{D}_{\sqsubset}$-structure for it. The tableau method attempts systematically to build such a structure if there is any, returning "satisfiable" if it succeeds and "unsatisfiable" otherwise.

The nodes of the tableau are sets of locally consistent formulas (i.e., parts of atoms). At the root of the tableau, we place a set containing only the formula φ the satisfiability of which is being tested. We then proceed recursively to expand the tableau, following the expansion rules described below. Every disjunctive branch of the tableau describes an attempt to construct a fulfilling $\mathrm{D}_{\sqsubset}$-structure for the atom at the root. Going down the branch roughly corresponds to going deeper into subintervals of the interval corresponding to the root. The applicability of an expansion rule at a given node depends on the formulas in the node and on the part of $\mathrm{D}_{\sqsubset}$-structure we are building. The expansion of the tableau proceeds as follows.

1. We start with the *current vertex* (at the beginning, the root) v_0 of the $D_{\sqsubset}$-structure that is being constructed and we apply the usual Boolean rules to decompose Boolean operators.
2. Then, we impose a suitable marking on $\langle D \rangle$ formulas to partition them into four sets: the set of formulas that are satisfied only on beginning subintervals, that of formulas that are satisfied only on ending subintervals, that of formulas that are satisfied both on beginning and ending subintervals, and that of formulas that are satisfied on internal subintervals.
3. The third phase of the procedure is the construction of the first vertex v_b of the beginning successor cluster $\mathcal{C}_b$, the first vertex v_e of the ending successor cluster $\mathcal{C}_e$, and their common successor v_c.
4. Next, we proceed in parallel with the construction of the clusters $\mathcal{C}_b$ and $\mathcal{C}_e$ by guessing the $\langle D \rangle$ formulas from the set $\mathrm{REQ}(\mathcal{L}(v_0))$ that should be satisfied inside each of them.
5. Then, we build the irreflexive successor v_b^{max} of $\mathcal{C}_b$, the irreflexive successor v_e^{max} of $\mathcal{C}_e$, and the irreflexive successors of v_c, if needed, and proceed recursively with their expansion from Step 1 above.

During the expansion of the tableau, we restrict our search to models with the property stated in Theorem 10. In particular, during the construction of a cluster we explicitly satisfy only those $\langle D \rangle$ formulas that should be satisfied inside the cluster and can never be satisfied outside it. In this way we have the following advantages:

i) we consider a $\langle D \rangle$ formula only once on a given branch of the tableau.

ii) when we exit a cluster, we can add the negation of every $\langle D \rangle$ formula that has been explicitly satisfied inside that cluster, thus reducing the search space of the successive expansion steps.

The rules of the tableau. Before describing the tableau rules in details, we need to introduce some preliminary notation. A formula of the form $\langle D \rangle \psi \in CL(\varphi)$ can be possibly marked as follows:

$$\langle D \rangle^M \psi, \langle D \rangle^B \psi, \langle D \rangle^{BC} \psi, \langle D \rangle^{BNC} \psi, \langle D \rangle^E \psi, \langle D \rangle^{EC} \psi, \langle D \rangle^{ENC} \psi, \langle D \rangle^{BE} \psi.$$

This notation has the following intuitive meaning. The markings $\langle D \rangle^M \psi$, $\langle D \rangle^B \psi$, $\langle D \rangle^E \psi$, and $\langle D \rangle^{BE}$ appear when we try to construct an irreflexive interval node and we guess that the formula $\langle D \rangle \psi$ should be satisfied over an internal (middle) subinterval, only over a beginning subinterval, only over an ending subinterval, or both over a beginning and over an ending (but not over middle) subinterval of the current one. The markings $\langle D \rangle^{BC} \psi$ or $\langle D \rangle^{BNC} \psi$ (resp. $\langle D \rangle^{EC} \psi$,$\langle D \rangle^{ENC} \psi$) substitute a previously marked $\langle D \rangle^B \psi$ (resp. $\langle D \rangle^E \psi$) formula when we try to construct a beginning cluster and we guess that the formula ψ should be satisfied in the current cluster ($\langle D \rangle^{BC} \psi$ marking) or not ($\langle D \rangle^{BNC} \psi$ marking). The marking is only used for bookkeeping purposes, to facilitate the correct choice of the rules to be applied. It does not affect the existence of a contradiction; we say that a *node is closed* iff once we remove the marking from every formula in it, it then contains both ψ and $\neg\psi$ for some $\psi \in CL(\varphi)$.

Given a set Φ of possibly marked formulas, the set $TF(\Phi)$ (the *temporal fragment of* Φ) is the set of all the formulas in Φ of the types $\langle D\rangle\psi$ and $[D]\psi$ (ignoring the markings). Given a set of formulas Γ, we use $(D)\Gamma$, where $(D) \in \{[D], \langle D\rangle, \langle D\rangle^M, \langle D\rangle^B, \langle D\rangle^{BC}, \langle D\rangle^{BNC}, \langle D\rangle^E, \langle D\rangle^{EC}, \langle D\rangle^{ENC}, \langle D\rangle^{BE}\}$, as a shorthand for $\{(D)\psi \mid \psi \in \Gamma\}$. Likewise, $\neg\Gamma$ stands for $\{\neg\psi \mid \psi \in \Gamma\}$ and $\Gamma \vee (D)\Gamma$ for $\{\psi \vee (D)\psi \mid \psi \in \Gamma\}$.

We now describe the rules used to expand the tableau nodes. In order to help the reader in understanding them, they are introduced and briefly explained in the order they appear in the procedure. We start with an initial tableau consisting of only one node containing the formula φ that we want to check for satisfiability. We apply the following **Boolean Rules** to $\{\varphi\}$ and to the newly generated nodes until these rules are no longer applicable:

$$\frac{\Phi, \neg\neg\psi}{\Phi, \psi} \qquad \frac{\Phi, \psi_1 \vee \psi_2}{\Phi, \psi_1 \mid \Phi, \psi_1} \qquad \frac{\Phi, \neg(\psi_1 \vee \psi_2)}{\Phi, \neg\psi_1, \neg\psi_2}$$

Next, we focus on a node to which the Boolean Rules are no more applicable. At this stage the node contains only atomic formulas and a subset of the temporal fragment of an atom (there may exist a formula $\langle D\rangle\psi \in \mathrm{REQ}(\varphi)$ for which neither $\langle D\rangle\psi$ nor $[D]\neg\psi$ belongs to the current node). In order to obtain a complete temporal fragment, we apply the following **Completion Rule** to the current node and to all newly generated nodes:

$$\frac{\Phi}{\Phi, \langle D\rangle\psi \mid \Phi, [D]\neg\psi} \quad \text{where } \langle D\rangle\psi \in CL(\varphi),\ \langle D\rangle\psi \notin \Phi, \text{ and } [D]\neg\psi \notin \Phi.$$

Given a node with a complete temporal fragment, we have to classify every formula of the form $\langle D\rangle\psi$ belonging to it as a *beginning, middle, ending*, or *both beginning and ending* one. This is done by the following **Marking Rule**:

$$\frac{\Phi, \langle D\rangle\psi}{\Phi, \langle D\rangle^B\psi \mid \Phi, \langle D\rangle^M\psi \mid \Phi, \langle D\rangle^E\psi \mid \Phi, \langle D\rangle^{BE}\psi} \quad \begin{array}{c}\text{where neither } \langle D\rangle^B\psi \text{ nor } \langle D\rangle^E\psi\\ \text{belongs to an ancestor}\\ \text{of the current node.}\end{array}$$

The conditions for the application of this rule will be explained later.

Given an irreflexive node with a complete temporal fragment, whose $\langle D\rangle$ formulas have been classified and marked, we generate its two reflexive successors, together with their common reflexive successor. This operation is performed by applying once the following **Reflexive Step Rule**:

$$\frac{\Phi, \langle D\rangle^B\Gamma, \langle D\rangle^M\mathrm{M}, \langle D\rangle^{BE}\Theta, \langle D\rangle^E\Lambda, [D]\Delta}{\begin{array}{c|c|c} \langle D\rangle^B\Gamma, \langle D\rangle^B\Theta, \langle D\rangle^M\mathrm{M}, & \langle D\rangle^M\mathrm{M}, & \langle D\rangle^E\Lambda, \langle D\rangle^E\Theta, \langle D\rangle^M\mathrm{M}, \\ [D]\neg\Lambda, [D]\Delta, \neg\Lambda, \Delta & [D]\neg\Gamma, [D]\neg\Theta, [D]\neg\Lambda, & [D]\neg\Gamma, [D]\Delta, \neg\Gamma, \Delta \\ & [D]\Delta, \neg\Gamma, \neg\Theta, \neg\Lambda, \Delta & \end{array}}$$

This rule splits the requests over three nodes accordingly to their classification. If a request cannot appear in a node, it introduces the corresponding negation. The generated nodes have a complete temporal fragment and are reflexive since all box arguments belong to them.

We deal with the middle node in a way similar to the case of $\mathrm{D}_{\sqsubset}$-structures (see Section 3). First, we apply the Boolean Rules until they are no longer applicable. Then, we apply the following **Middle Step Rule**:

$$\frac{\Phi, \langle D\rangle^M \mu_1, ..., \langle D\rangle^M \mu_h, [D]\Gamma}{\mu_1, \Gamma, [D]\Gamma \,|\, ... \,|\, \mu_h, \Gamma, [D]\Gamma}$$

For every request in the current node, this rule creates an irreflexive successor of it. Then, we re-apply the expansion procedure from the beginning for every newly generated node.

The expansion of a beginning node takes place as follows. As usual, we first apply the Boolean Rules to it, and to the newly generated nodes, until they are applicable. Then, for any $\langle D\rangle^B\psi$ formula in the current node, we distinguish two cases: $\langle D\rangle^B\psi$ can be fulfilled in the cluster or it can be fulfilled in one of its descendants. They are dealt with the following **Build Beginning Cluster Rule**:

$$\frac{\Phi, \langle D\rangle^B\psi, \langle D\rangle^B\Gamma_B, \langle D\rangle^{BC}\Gamma_{BC}, \langle D\rangle^{BNC}\Gamma_{BNC}, \langle D\rangle^M \mathrm{M}, [D]\Delta}{\begin{array}{c|c} \psi, \langle D\rangle^B\Gamma_B, \langle D\rangle^{BC}(\Gamma_{BC}\cup\{\psi\}), & \Phi, \langle D\rangle^B\Gamma_B, \langle D\rangle^{BC}\Gamma_{BC}, \\ \langle D\rangle^{BNC}\Gamma_{BNC}, \langle D\rangle^M\mathrm{M}, [D]\Delta, \Delta & \langle D\rangle^{BNC}(\Gamma_{BNC}\cup\{\psi\}), \langle D\rangle^M\mathrm{M}, [D]\Delta \end{array}}$$

The former case is handled by the first branch, which marks the request as $\langle D\rangle^{BC}\psi$ (in order to avoid loops) and satisfies ψ in a new cluster node with the same temporal fragment as the current one. The latter case is handled by the second branch that simply reclassifies the request as $\langle D\rangle^{BNC}\psi$ without moving to another cluster node. Such a procedure is iterated until every $\langle D\rangle^B\psi$ is remarked as $\langle D\rangle^{BC}\psi$ or $\langle D\rangle^{BNC}\psi$.

The case of ending nodes is dealt with in a very similar way by means of the following **Build Ending Cluster Rule**:

$$\frac{\Phi, \langle D\rangle^E\psi, \langle D\rangle^E\Gamma_E, \langle D\rangle^{EC}\Gamma_{EC}, \langle D\rangle^{ENC}\Gamma_{ENC}, \langle D\rangle^M \mathrm{M}, [D]\Delta}{\begin{array}{c|c} \psi, \langle D\rangle^E\Gamma_E, \langle D\rangle^{EC}(\Gamma_{EC}\cup\{\psi\}), & \Phi, \langle D\rangle^E\Gamma_E, \langle D\rangle^{EC}\Gamma_{EC}, \\ \langle D\rangle^{ENC}\Gamma_{ENC}, \langle D\rangle^M\mathrm{M}, [D]\Delta, \Delta & \langle D\rangle^{ENC}(\Gamma_{ENC}\cup\{\psi\}), \langle D\rangle^M\mathrm{M}, [D]\Delta \end{array}}$$

Once we reach a cluster node such that no Boolean rules are applicable and every $\langle D\rangle^B\psi$ request has been reclassified as $\langle D\rangle^{BC}\psi$ or $\langle D\rangle^{BNC}\psi$, we proceed as follows. If the node does not include any $\langle D\rangle^{BNC}\psi$ request, we are done (all requests have been satisfied in the cluster). Otherwise (there exists at least one marked formula of the form $\langle D\rangle^{BNC}\psi$), we generate an irreflexive successor of the cluster that, for every formula $\langle D\rangle^{BNC}\psi$, satisfies either ψ or $\langle D\rangle^B\psi$. This last case is handled by the formulas $\Gamma_{BNC} \vee \langle D\rangle^B\Gamma_{BNC}$ introduced by the following **Exit Beginning Cluster Rule**:

$$\frac{\Phi, \langle D\rangle^{BC}\Gamma_{BC}, \langle D\rangle^{BNC}\Gamma_{BNC}, \langle D\rangle^{M}\mathrm{M}, [D]\Delta}{\Gamma_{BNC} \vee \langle D\rangle^{B}\Gamma_{BNC}, [D]\neg\Gamma_{BC}, [D]\Delta, \Delta} \text{ where } \Gamma_{BNC} \neq \emptyset.$$

The case of the ending cluster is dealt with in a very similar way by means of the following **Exit Ending Cluster Rule**:

$$\frac{\Phi, \langle D\rangle^{EC}\Gamma_{EC}, \langle D\rangle^{ENC}\Gamma_{ENC}, \langle D\rangle^{M}\mathrm{M}, [D]\Delta}{\Gamma_{ENC} \vee \langle D\rangle^{E}\Gamma_{ENC}, [D]\neg\Gamma_{EC}, [D]\Delta, \Delta} \text{ where } \Gamma_{ENC} \neq \emptyset.$$

Then, we apply again all steps from the beginning, with only a little difference in the application of the Marking Rule. The Completion Rule may produce some requests $\langle D\rangle\psi$ devoid of any markings. For all these requests, we must check whether they have been marked as $\langle D\rangle^{B}\psi$ or $\langle D\rangle^{E}\psi$ in an ancestor of the current node and, if this is the case, we must guarantee the downward propagation of their markings. To this end, before applying the Marking Rule, we apply the following **Persistent Beginning** and **Persistent Ending Rules**:

$$\frac{\Phi, \langle D\rangle\psi}{\Phi, \langle D\rangle^{B}\psi} \qquad \frac{\Phi, \langle D\rangle\psi}{\Phi, \langle D\rangle^{E}\psi}$$

whenever $\langle D\rangle^{B}\psi$ (resp., $\langle D\rangle^{E}\psi$) belongs to an ancestor of the current node.

Building the tableaux. As in the case of $\mathrm{D}_{\sqsubseteq}$ logic, a tableau for a $\mathrm{D}_{\sqsubset}$ formula φ is a finite graph $\mathcal{T} = \langle V, E\rangle$, whose vertices are subsets of $\mathrm{CL}(\varphi)$ and whose edges are generated by the application of expansion rules. The construction of the tableau starts with the *initial tableau*, which is the single node graph $\langle\{\{\varphi\}\}, \emptyset\rangle$. To describe such a construction process, we take advantage of macronodes, which can be viewed as the counterpart of vertices of $\mathrm{D}_{\sqsubset}$-structures.

Given a set $V' \subseteq V$, let $E(V')$ be the restriction of E to vertices in V. Moreover, let the Reflexive Step, Middle Step, Build Beginning/Ending Cluster and Exit Beginning/Ending Cluster rules be called **Step Rules**. Macronodes are defined as follows.

Definition 11. *Let $\langle V, E\rangle$ be a tableau for a $\mathrm{D}_{\sqsubset}$ formula φ. A* macronode *is a set $V' \subseteq V$ such that:*

- *$\langle V', E(V')\rangle$ is a tree;*
- *the root of $\langle V', E(V')\rangle$ is either the initial node of the tableau or a node generated by an application of a Step Rule;*
- *every edge in $E(V')$ is generated by the application of an expansion rule which is not a Step Rule;*
- *the only expansion rule that can be applied to the leaves of $\langle V', E(V')\rangle$ is a Step Rule.*

A macronode m is reflexive *if its root is generated by the application of the Reflexive Step Rule or of the Build Beginning/Ending Cluster Rules; otherwise, it is* irreflexive.

We say that a rule is applicable to a node n if it generates at least one successor node. The construction of a tableau for a $D_{\sqsubset}$ formula φ starts with the initial tableau $\langle\{\{\varphi\}\},\emptyset\rangle$ and proceeds by applying the following *expansion strategy* to the leaves of the current tableau, until it cannot be applied anymore.

Apply the first rule in the list whose condition is satisfied:

1. a Boolean Rule is applicable;
2. the Completion Rule is applicable;
3. the node belongs to an irreflexive macronode and the Persistent Beginning Rule is applicable;
4. the node belongs to an irreflexive macronode and the Persistent Ending Rule is applicable;
5. the node belongs to an irreflexive macronode and the Marking Rule is applicable;
6. the node belongs to an irreflexive macronode and the Reflexive Step Rule is applicable;
7. the node belongs to a reflexive macronode with only M markings and the Middle Step Rule is applicable;
8. the node belongs to a reflexive macronode with B markings or E markings and the Build Beginning/Ending Cluster Rules are applicable;
9. the node belongs to a reflexive macronode with B markings or E markings and the Exit Beginning/Ending Cluster Rules are applicable.

Termination is ensured by the following *looping conditions*:

– if an application of the Reflexive Rule generates a node which is the root of an existing reflexive macronode, then add an edge from the current node to this node instead of creating the new one.
– if the Middle Step Rule is applied to a node n and one of the successor nodes it generates, say n', is such that $TF(n') = TF(n)$, then add the edge (n', n) to the tableau. Do not apply any expansion rule to n'.

We say that a node n in a tableau is *closed* if one of the following conditions holds:

– there exists ψ such that both ψ and $\neg\psi$ belong to n;
– a Middle Step Rule or a Reflexive Step Rule have been applied to n and *at least one* of its successors is closed;
– a rule different from the Middle Step Rule and the Reflexive Step Rule has been applied to n and *all* its successors are closed;
– n is a descendant of a node n' to which an Exit Beginning/Ending Cluster Rule has been applied and $TF(n') = TF(n)$.

A node in a tableau is *open* if it is not closed. A tableau is *open* if and only if its root is open. We will prove that a formula is satisfiable if and only if there exists an open tableau for it.

As for computational complexity, it is not difficult to show that the proof of Theorem 10 can be adapted to the proposed tableau method. The only difference is that at any step of the tableau construction we either expand a node or mark one of its formulas. As a consequence, any node of a $D_{\sqsubset}$-structure corresponds to a path of at most $|\varphi|$ nodes in the tableau. Hence, the depth of the tableau is bounded by $2 \cdot |\varphi|^2$. Since the breadth of the tableau is $2 \cdot |\varphi|$, we can conclude that the proposed tableau-based decision procedure is in $PSPACE$ (and thus optimal).

Theorem 12. *(Complexity) The proposed tableau procedure is in* $PSPACE$.

Example of application. Here we give an example of the above-described expansion strategy at work. Consider the formula $\varphi = \langle D\rangle p \wedge \langle D\rangle q \wedge [D]\neg(\langle D\rangle p \wedge \langle D\rangle q)$, which states that the given interval has a subinterval where p holds and a subinterval where q holds, but no subintervals covering both of them. It is easy to see that in any model for this formula p and q respectively hold in a beginning and an ending subinterval only, or vice versa. Part of the tableau for φ is depicted in Figure 6. Due to space limitations, we restrict our attention to the non-closed region of the tableau and we skip the details about the application of Boolean Rules. We start with the root A, whose temporal fragment is complete, and we apply the Marking Rule. For the sake of conciseness, we only consider a correct marking, which inserts $\langle D\rangle^B p$ and $\langle D\rangle^E q$ in B. Once all $\langle D\rangle$ formulas have been marked, we apply the Reflexive Step Rule, that generates the three successors of B. The first successor is node C that contains the request $\langle D\rangle^B p$ and the negation of the request $\langle D\rangle^E q$, namely, $[D]\neg q$. The second one is node E that contains the request $\langle D\rangle^E q$ and the negation of the request $\langle D\rangle^B p$, namely, $[D]\neg p$. The third one is node D that contains the negation of the two requests (such a node represents the middle reflexive vertex of the corresponding $D_{\sqsubset}$-structure). Node D contains no $\langle D\rangle$ formulas and thus it cannot be expanded anymore. Since it does not include any contradiction, we declare it open. Consider now node C. According to the expansion strategy, we apply the Build Beginning Cluster Rule to $\langle D\rangle^B p$ in node C, that generates nodes F and G. Node F includes p and, accordingly, replaces $\langle D\rangle^B p$ with $\langle D\rangle^{BC} p$. It does not contain $\langle D\rangle^{BNC}$ formulas and no expansion rules are applicable to it. Since it does not include any contradiction, we declare it open. The same argument can be applied to nodes E and H. This allows us to conclude that the tableau is open (and thus φ is satisfiable).

To better explain the proposed tableau method, we include in Figure 6 additional nodes which are not strictly necessary to conclude that the tableau is open. This is the case with node G that replaces $\langle D\rangle^B p$ with $\langle D\rangle^{BNC} p$, thus postponing the satisfaction of p. According to the expansion strategy, we apply the Exit Beginning Cluster Rule to G, that generates the irreflexive node L. Such a node contains the formula $\langle D\rangle^B p \vee p$, stating that p is satisfied either in L or in some descendant of it. The application of the Or Rule to $\langle D\rangle^B p \vee p$ generates nodes M and N. Node M includes again the formula $\langle D\rangle^B p$ and, since $TF(M) = TF(G)$, we declare it closed. As for node N, that satisfies p, we apply

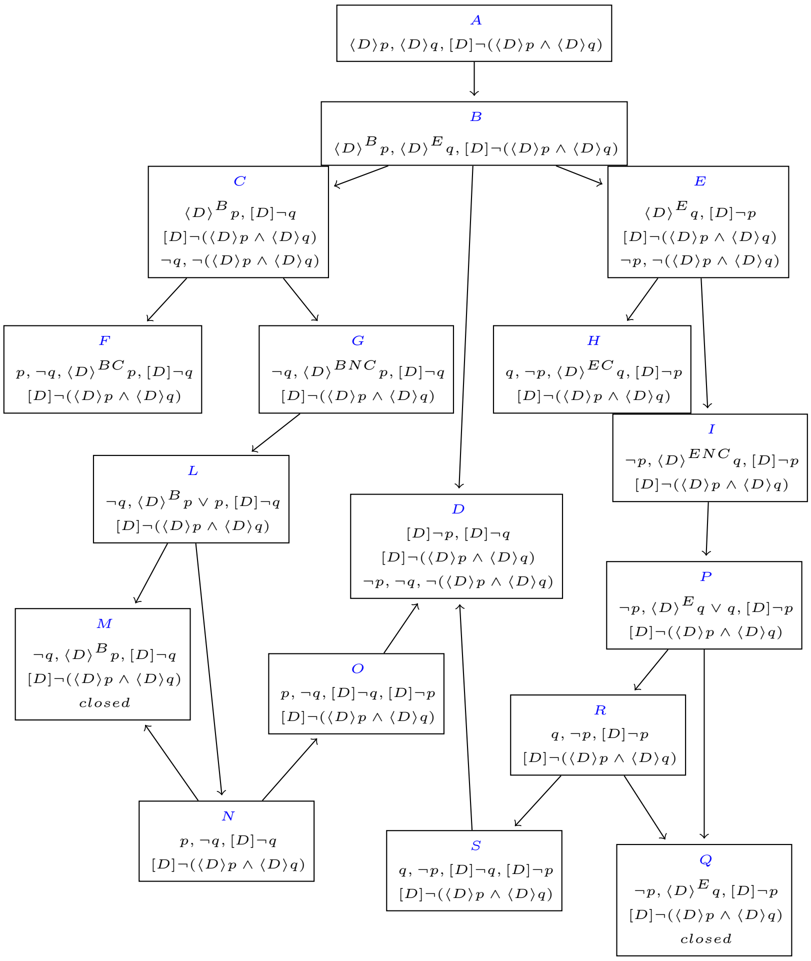


Fig. 6. (Part of) the tableau for $\varphi = \langle D\rangle p \wedge \langle D\rangle q \wedge [D]\neg(\langle D\rangle p \wedge \langle D\rangle q)$.

the Completion Rule (neither $\langle D\rangle p$ nor $[D]\neg p$ belongs to N), that generates its two successors. The first successor turns out to be identical to M and thus we add an edge from N to M instead of adding a new node; the second successor is node O, with $TF(O) \subset TF(G)$. Then, we apply Reflexive Step Rule to node O. Since it does not contain any $\langle D\rangle$ formula, its three reflexive successors coincides with node D. Hence, we add an edge from O to D and we stop the expansion of (this part of) the tableau.

4.4 Soundness and completeness

We conclude the section by proving soundness and completeness of the tableau method.

Theorem 13. *(*SOUNDNESS*) Let φ be a $\mathrm{D}_{\sqsubset}$ formula and $\mathcal{T}$ be a tableau for it. If $\mathcal{T}$ is open, then φ is satisfiable.*

Proof. We build a fulfilling $\mathrm{D}_{\sqsubset}$-structure $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ for φ step by step, starting from the root of $\mathcal{T}$ and proceeding according to the expansion rules that have been applied in the construction of the tableau.

Let n_0 be the root of $\mathcal{T}$. We generate the one-node $\mathrm{D}_{\sqsubset}$-graph $\langle\{v_0\}, \emptyset\rangle$ and we put formulas belonging to n_0 in $\mathcal{L}(v_0)$. Now, let n be an open node in $\mathcal{T}$ and let v be the corresponding vertex in the $\mathrm{D}_{\sqsubset}$-graph. The way in which we develop the $\mathrm{D}_{\sqsubset}$-structure depends on the expansion rule that has been applied to n during the construction of the tableau.

- *A Boolean Rule has been applied.* Then, at least one successor n' of n is open. We add formulas belonging to n' to $\mathcal{L}(v)$ and we proceed by taking into consideration the tableau node n' and the vertex v.
- *The Completion Rule has been applied.* Then, at least one successor n' of n is open. As in the previous case, we add formulas belonging to n' to $\mathcal{L}(v)$ and we proceed by taking into consideration the tableau node n' and the vertex v.
- *The Marking/Persistent Beginning/Persistent Ending Rule has been applied.* Let $\langle D\rangle\psi$ be the formula to which the rule has been applied and let n' be one of the open successors of n. Four cases may arise, depending on which marking has been applied to the considered formula in n':
  - if $\langle D\rangle^B\psi \in n'$, then we put $\langle D\rangle\psi \in \mathcal{B}(v)$;
  - if $\langle D\rangle^E\psi \in n'$, then we put $\langle D\rangle\psi \in \mathcal{E}(v)$;
  - if $\langle D\rangle^{BE}\psi \in n'$, then we add $\langle D\rangle\psi$ to both $\mathcal{B}(v)$ and $\mathcal{E}(v)$;
  - if $\langle D\rangle^M\psi \in n'$, then the marking does not influence the construction of the $\mathrm{D}_{\sqsubset}$-structure.

  In all cases, we proceed recursively by taking into consideration the tableau node n' and the current vertex v.
- *The Reflexive Step Rule has been applied.* Since $\mathcal{T}$ is open, all successors of n are open either. Let n_b, n_c, and n_e be the first, second, and third successor of n, respectively. We add three reflexive vertices v_b, v_c, and v_e to V and the edges (v, v_b), (v, v_e), (v_b, v_c), (v_e, v_c), (v_b, v_b), (v_c, v_c), and (v_e, v_e) to E. The labeling of v_b, v_c, and v_e is defined as follows: $\mathcal{L}(v_b) = n_b$, $\mathcal{L}(v_c) = n_c$, and $\mathcal{L}(v_e) = n_e$. We recursively apply the construction by taking into consideration the node n_b with the corresponding vertex v_b, the node n_c with the corresponding vertex v_c, and the node n_e with the corresponding vertex v_e.
- *The Middle Step Rule has been applied.* Since n is open, all its successors $n_1, ..., n_h$ are open either. We add h new vertices $v_1, ..., v_h$ to V and the edges $(v, v_1), ..., (v, v_h)$ to E, and we define their labeling in such a way that for $i = 1, \ldots, h$, $\mathcal{L}(v_i) = n_i$. We recursively apply the construction to every node n_i paired with the corresponding vertex v_i.

– *The Build Beginning/Ending Cluster Rule has been applied.* Suppose that the rule has been applied to a formula $\langle D\rangle^B\psi \in n$ (the case of $\langle D\rangle^E\psi$ is analogous) and let n' be an open successor of n. Two cases may arise:
  1. $\langle D\rangle^{BC}\psi \in n'$ ($\langle D\rangle\psi$ has been satisfied in the cluster). We introduce a new node v' in the cluster of v by adding the edges $(v, v'), (v', v')$, and (v', v) to E. The labeling $\mathcal{L}(v')$ of v' consists of the set of formulas belonging to n'. We proceed by taking into consideration the node n' and the corresponding vertex v'.
  2. $\langle D\rangle^{BNC}\psi \in n'$ (satisfaction of $\langle D\rangle\psi$ has been postponed). We do not add any vertex to the $D_\sqsubset$-structure, but simply proceed by taking into consideration the node n' and the current vertex v.
– *The Exit Beginning/Ending Cluster Rule has been applied.* Since $\mathcal{T}$ is open, the unique successor n' of n is open and it is the root of an irreflexive macronode. We add a new irreflexive vertex v' to V and an edge (v, v') to E. Moreover, we set the labeling of v' as the set of formulas belonging to n'. Then, we proceed by taking into consideration the node n' with the corresponding vertex v'.

To keep the construction finite, whenever the procedure reaches a tableau node n' that has been already taken into consideration, instead of adding a new vertex to the $D_\sqsubset$-structure, it simply adds an edge from the current vertex v to the vertex v' corresponding to n'.

Since any tableau for φ is finite, such a construction is terminating. However, the resulting structure $\langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ is not necessarily a $D_\sqsubset$-structure: there may exist a vertex $v \in V$ and a non-temporal formula $\psi \in \mathrm{CL}(\varphi)$ such that neither ψ nor $\neg\psi$ belongs to $\mathcal{L}(v)$. To overcome this problem, we can consistently extend the labeling $\mathcal{L}(v)$ as follows:

– if $\psi = p$, with $p \in \mathcal{AP}$, we put $\neg p \in \mathcal{L}(v)$;
– If $\psi = \neg\xi$, we put $\psi \in \mathcal{L}(v)$ if and only if $\xi \notin \mathcal{L}(v)$;
– If $\psi = \psi_1 \vee \psi_2$, we put $\psi_1 \vee \psi_2 \in \mathcal{L}(v)$ if and only if $\psi_1 \in \mathcal{L}(v)$ or $\psi_2 \in \mathcal{L}(v)$.

The resulting $D_\sqsubset$-structure $\langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ is a fulfilling $D_\sqsubset$-structure for φ and thus φ is satisfiable. ☐

Theorem 14. (COMPLETENESS) *Let φ be a $D_\sqsubset$ formula. If φ is satisfiable, then there exists an open tableau for it.*

Proof. Let $\mathbf{S} = \langle\langle V, E\rangle, \mathcal{L}, \mathcal{B}, \mathcal{E}\rangle$ be a fulfilling $D_\sqsubset$-structure that satisfies φ. We take advantage of such a structure to show that there exists an open tableau $\mathcal{T}$ for φ. In particular, we will define a correspondence between (some) nodes in $\mathcal{T}$ and vertices in $\mathbf{S}$ that satisfies the following constraints:

(1) if n is associated with an irreflexive vertex v, then n belongs to an irreflexive macronode;
(2) if n is associated with a reflexive vertex v, then n belongs to a reflexive macronode;
(3) ff n is associated with a vertex v, then, for every formula $\psi \in n$, $\psi \in \mathcal{L}(v)$.

Let n_0 be the root of the tableau. We associate it with the root v_0 of $\mathbf{S}$. Since n_0 belongs to an irreflexive macronode, v_0 is an irreflexive vertex, and $\varphi \in \mathcal{L}(v_0)$, all constraints are satisfied.

Let n be the current node of the tableau, v be the vertex of $\mathbf{S}$ associated with it, and, by inductive hypothesis, n and v satisfy the constraints. We proceed by taking into consideration the rule that, according to the expansion strategy, is applicable to node n.

- *One of the Boolean Rules is applicable.* We consider the application of the OR Rule to a formula of the form $\psi_1 \vee \psi_2$ (the other cases are simpler and thus omitted). Since $\psi_1 \vee \psi_2 \in n$, by Constraint (3), $\psi_1 \vee \psi_2 \in \mathcal{L}(v)$ and thus $\psi_1 \in \mathcal{L}(v)$ or $\psi_2 \in \mathcal{L}(v)$. If $\psi_1 \in \mathcal{L}(v)$, then we associate the successor n_1 of n, that contains ψ_1, with v; otherwise, we associate the successor n_2 of n, that contains ψ_2, with v. In either cases, all constraints are satisfied.
- *The Completion Rule is applicable.* Let us consider the application of the Completion Rule to the formula $\langle D\rangle\psi$. Since $\mathcal{L}(v)$ is an atom, either $\langle D\rangle\psi \in \mathcal{L}(v)$ or $[D]\neg\psi \in \mathcal{L}(v)$. In the former case, we associate the successor n_1 of n, that contains $\langle D\rangle\psi$, with v; in the latter case, we associate the successor n_2 of n, containing $[D]\neg\psi$, with v. In either cases, all constraints are satisfied.
- *The Marking Rule is applicable.* Let us consider the application of the Marking Rule to the formula $\langle D\rangle\psi$. According to the expansion strategy, n belongs to an irreflexive macronode and thus, by inductive hypothesis, v is an irreflexive vertex. Let $\mathcal{C}_b$ be the beginning successor cluster of v, $\mathcal{C}_e$ the ending successor cluster of v, and v_c their common reflexive successor (see Definition 7). Four cases may arise:
  1. $\langle D\rangle\psi$ appears in $\mathcal{C}_b$, but not in $\mathcal{C}_e$ and v_c. In this case, we associate the successor n' of n, which includes $\langle D\rangle^B\psi$, with v.
  2. $\langle D\rangle\psi$ appears in $\mathcal{C}_e$, but not in $\mathcal{C}_b$ and v_c. In this case, we associate the successor n' of n, which includes $\langle D\rangle^E\psi$, with v.
  3. $\langle D\rangle\psi$ appears in $\mathcal{C}_b$ and $\mathcal{C}_e$, but not in v_c. In this case, we associate the successor n' of n, which includes $\langle D\rangle^{BE}\psi$, with v.
  4. $\langle D\rangle\psi$ appears in $\mathcal{C}_b$, $\mathcal{C}_e$, and v_c. In this case, we associate the successor n' of n, which includes $\langle D\rangle^M\psi$, with v.
- *The Persistent Beginning/Ending Rule is applicable.* We associate the unique successor n' of n with v.
- *The Reflexive Step Rule is applicable.* According to the expansion strategy, n belongs to an irreflexive macronode and thus, by inductive hypothesis, v is an irreflexive vertex. Let v_b be a node in the beginning successor cluster of v, v_e a node in the ending successor cluster of v, and v_c the common reflexive successor of the two clusters. According to the expansion strategy, when such a rule turns out to be applicable, all $\langle D\rangle$ formulas have already been marked in accordance with $\mathbf{S}$. Let $n = \{\Phi, \langle D\rangle^B\Gamma, \langle D\rangle^M\mathrm{M}, \langle D\rangle^{BE}\Theta, \langle D\rangle^E\Lambda, [D]\Delta\}$, where Φ only contains atomic formulas. We have that $\{\langle D\rangle\Gamma, \langle D\rangle\Theta, \langle D\rangle\mathrm{M}, [D]\neg\Lambda, [D]\Delta, \neg\Lambda,\ \Delta\} \subseteq \mathcal{L}(v_b)$, that $\{\langle D\rangle\Lambda, \langle D\rangle\Theta, \langle D\rangle\mathrm{M}, [D]\neg\Gamma, [D]\Delta, \neg\Gamma, \Delta\} \subseteq \mathcal{L}(v_e)$, and that $\{\langle D\rangle\mathrm{M}, [D]\neg\Gamma, [D]\neg\Theta[D]\neg\Lambda, [D]\Delta,\ \neg\Gamma,\ \neg\Theta,\ \neg\Lambda,\ \Delta\} \subseteq \mathcal{L}(v_c)$. We associate the first successor of n with v_b, the second one with v_e, and the third one with v_c.

- *The Middle Step Rule is applicable.* According to the expansion strategy, n belongs to a macronode whose root is the middle node generated by an application of the Reflexive Step Rule and thus, by inductive hypothesis, n is associated with a middle reflexive vertex v_c. Since $\mathbf{S}$ is fulfilling, for every formula $\langle D\rangle\psi \in n$ there exists a successor v_ψ of v_c such that $\psi \in \mathcal{L}(v_\psi)$ and for every $[D]\theta \in n$, $\theta, [D]\theta \in \mathcal{L}(v_\psi)$. For all $\langle D\rangle\psi \in n$, we associated the successor n_ψ of n with v_ψ.
- *The Build Beginning Cluster Rule is applicable.* Given the expansion strategy, by inductive hypothesis we have that n is associated with a node v that belongs to a beginning cluster $\mathcal{C}$. Let us consider the application of the rule to the formula $\langle D\rangle^B\psi$. Two cases may arise: either $\mathbf{S}$ fulfills $\langle D\rangle\psi$ outside $\mathcal{C}$ or not. In the former case, we associate the successor n' of n, that contains $\langle D\rangle^{BNC}\psi$, with v; in the latter case, there exists a node $v' \in \mathcal{C}$ such that $\psi \in \mathcal{L}(v')$ and we associate the successor n' of n, that contains both ψ and $\langle D\rangle^{BC}\psi$, with v'.
- *The Build Ending Cluster Rule is applicable.* This case is analogous to the previous one and thus omitted.
- *The Exit Beginning Cluster Rule is applicable.* Given the expansion strategy, by inductive hypothesis we have that n is associated with a node v that belongs to a beginning cluster $\mathcal{C}$. Let v' be the unique irreflexive successor of $\mathcal{C}$. We have that, for every formula $\langle D\rangle^{BNC}\psi \in n$, $\psi \in \mathcal{L}(v')$ or $\langle D\rangle\psi \in \mathcal{L}(v')$. The labeling of the unique successor node n' of n is thus consistent with v' and we can associate n' with v'.
- *The Exit Ending Cluster Rule is applicable.* This case is analogous to the previous one and thus omitted.

At the end of the above construction, we have obtained (a portion of) a tableau for φ. Since all its nodes are open, we can conclude that there exists an open tableau for φ. $\square$

5 Implementation

In this section we briefly describe an implementation of the proposed tableau methods in Lotrec, a generic theorem prover for modal and description logics [11,15]. We start with a short account of the main features of Lotrec. Then, we will point out the distinctive features of the proposed implementations of the tableau methods for $\mathrm{D}_{\sqsubseteq}$ and $\mathrm{D}_{\sqsubset}$, with a special attention to that for $\mathrm{D}_{\sqsubset}$. Finally, we will illustrate the behavior of the system on a concrete example.

5.1 A short overview of Lotrec

Lotrec is a generic prover that can be used for most modal logics studied in the literature. It 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.

A Lotrec program is divided into three parts. The first part defines the syntax of the considered logic. The second part defines a set of expansion rules to apply to the nodes of the graph during the computation. Rules have the following form:

rule *ruleName*
    if $condition_1$
        $\vdots$
    if $condition_n$
    do $action_1$
        $\vdots$
    do $action_m$
end

The *conditional part* of a rule, which consists of all lines starting with **if**, defines the conditions that must be satisfied in order to apply the rule to a certain node of the graph. When a rule is applied to a node, the current labeled graph is modified in accordance with the *action part* of the rule, which consists of the lines starting with **do**. As an example, consider the following two rules that encode the **Exit Beginning/Ending Cluster Rule** of the tableau for $D_{\sqsubset}$:

rule clusterStep

    if isMarked $node_0$ CLUSTER
    if hasElement $node_0$ ($\langle D \rangle$ (variable A))
    if isMarkedExpression $node_0$ ($\langle D \rangle$ (variable A)) DELAY
    do createOneSuccessor $node_1$ (R)
    do add $node_1$ (variable A) $\vee$ ($\langle D \rangle$ (variable A))

end

rule clusterDelay

    if isMarked $node_0$ CLUSTER
    if isLinked $node_0$ $node_1$ (R)
    if hasElement $node_0$ ($\langle D \rangle$ (variable A))
    if isMarkedExpression $node_0$ ($\langle D \rangle$ (variable A)) DELAY
    do add $node_1$ (variable A) $\vee$ ($\langle D \rangle$ (variable A))

end

The first rule deals with the case when the current node has no successors and it creates a unique irreflexive successor of a cluster (by the createOneSuccessor action). The rule is applicable only to nodes containing at least one $\langle D\rangle$-formula with a DELAY label, that is, a formula which is not satisfied in the current cluster, and it puts the formula $(variable\ A) \vee \langle D\rangle(variable\ A)$ in the labeling of the new node. The second rule completes the labeling of the new successor by adding the formula $(variable\ A) \vee \langle D\rangle(variable\ A)$ to it for every formula $\langle D\rangle(variable\ A)$ with a DELAY label in the current node. In a similar way, all rules of our tableau methods can be encoded in Lotrec.

A Lotrec program ends with a third part devoted to the definition of the expansion strategy to be used to construct the tableau. As an example, the expansion strategy of the tableau for $D_{\sqsubseteq}$ (see Section 3) can be easily encoded as follows:

```
strategy strictStrategy
  repeat
    firstRule
      stopRule
      notRule
      andRule
      orRule
      reflRule
      twoDensRule
      stepRule
      blockingRule
    end
  end
end
```

In such a case, the computation proceeds by repeatedly applying the first expansion rule that is applicable, following the order given in the strategy. The *stopRule* simply checks for contradictions and stop with failure if a node contains both a formula ψ and its negation $\neg\psi$. The *notRule*, *andRule*, *orRule*, *reflRule*, *twoDensRule*, and *stepRule* are Lotrec translations of the (NOT), (AND), (OR), (REFL), (2-DENS), and (STEP) Rules of the tableau method for $D_{\sqsubseteq}$. Finally, *blockingRule* impements the blocking conditions that avoid infinite expansions of the tableau.

5.2 Lotrec implementation of the tableau for $D_{\sqsubset}$

The tableau methods for $D_{\sqsubseteq}$ and $D_{\sqsubset}$ can be implemented in Lotrec by appropriately encoding their expansion rules and their expansion strategies. In the case of $D_{\sqsubset}$, we take advantage of Lotrec rewriting system to remove the *Completion Rule*.

The tableau method for $D_{\sqsubset}$ presented in Section 4 applies expansion rules only to the leaves of the current tableau. Moreover, the labeling of a node is defined when the node is created and it is not changed by the application of successive expansion steps. Because of this, we need the *Completion Rule* to completely expand the temporal fragment of a node to make it possible to immediately detect possible contradictions.

The application of Lotrec rules is not confined to leaves, and it can modify nodes which are ancestors of the current one. This feature allows us to remove the Completion Rule from the Lotrec implementation of the tableau method for $D_{\sqsubset}$ and to substitute it with the following rule that propagates back new $\langle D\rangle$-formulas as soon as they appear in a node.

rule backwardDiamond

> **if** hasElement $node_0$ ($\langle D\rangle$ (variable A))
> **if** isLinked $node_1$ $node_0$ (R)
> **if** hasNotElement $node_1$ ($\langle D\rangle$ (variable A))
> **do** add $node_1$ ($\langle D\rangle$ (variable A))

end

This rule allows us to add a new $\langle D\rangle$-formula to a node when it appears in some of its descendants, instead of guessing its belonging to the labeling of the node when the node is created, thus saving computation time in the average case. When this rule is applied, Lotrec behaves as follows. First, it adds $\langle D\rangle$ (variable A) to the ancestor $node_1$ of $node_0$. If this causes a contradiction, the current branch of the tableau is declared closed. Otherwise, it proceeds with the expansion following the given expansion strategy.

5.3 An application example

In this section we describe the tableau generated by Lotrec for the following unsatisfiable formula:

$$\varphi = \langle D\rangle p \wedge \langle D\rangle q \wedge [D]\neg(\langle D\rangle p \wedge \langle D\rangle q) \wedge [D](\langle D\rangle r \rightarrow [D]\neg p \wedge [D]\neg q) \wedge \langle D\rangle\langle D\rangle r$$

The formula φ imposes that p and q must be satisfied on opposite sides of the given interval, that r must be satisfied over some subinterval $[b_r, e_r]$ of the given one, and that every superinterval of $[b_r, e_r]$ is such that both $[D]\neg p$ and $[D]\neg q$ hold over it. The formula φ can be easily shown to be unsatisfiable. Assume p to be satisfied over some initial subinterval $[b_0, e_p]$, and take an initial subinterval $[b_0, e]$ that covers both $[b_0, e_p]$ and $[b_r, e_r]$. Since $[b_0, e_p] \sqsubset [b_0, e]$, $[b_0, e]$ satisfies both $\langle D\rangle p$ and $\langle D\rangle r$, contradicting φ requests. The case in which p is satisfied over an ending interval leads to an analogous contradiction.

Such an example allows us to display the role of backward propagation of diamond formulas in order to ensure correctness. The tableau for φ generated by Lotrec is depicted in Figure 7. Its construction starts with the root A and proceeds with the creation of its three reflexive successors B, C, and D. Let us assume that the partition rule classifies $\langle D\rangle p$ as a beginning request, $\langle D\rangle q$ as an ending request, and $\langle D\rangle\langle D\rangle r$ as a middle one. At this stage, the formula $\langle D\rangle r$ does not appear explicitly in any node of the tableau. Hence, B (resp., C) can satisfy p (resp., q) in its cluster without contradiction. The request $\langle D\rangle r$ appears for the first time in node E when Lotrec tries to fulfill the $\langle D\rangle\langle D\rangle r$ request of node D.

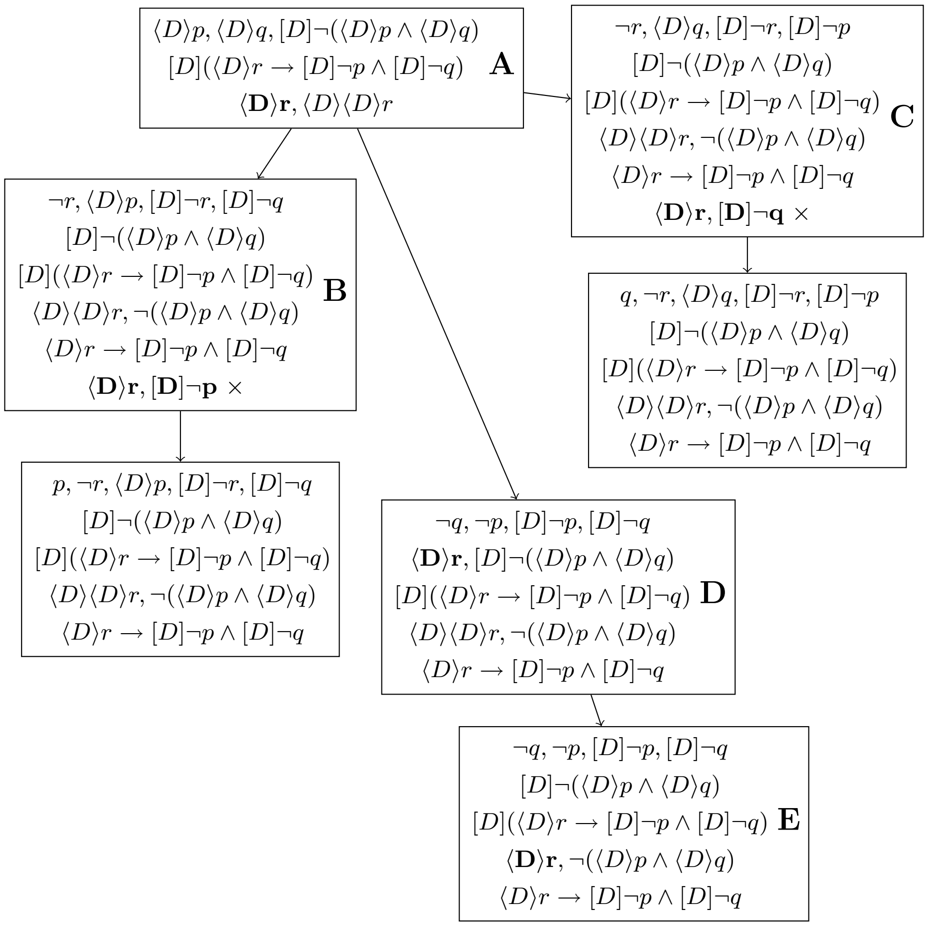


Fig. 7. An example of Lotrec tableau.

At this point, Lotrec propagates $\langle D\rangle r$ backward to D and then to A. When $\langle D\rangle r$ is added to A, it must be classified either as a begin, an end, or a middle request. The only admissible option is to classify it as a middle request (otherwise, we should put $[D]r$ in D and immediately close the tableau). Since $\langle D\rangle r$ is a middle request, it is propagated downward from A to B and C. The presence of both $\langle D\rangle r$ and $\langle D\rangle r \rightarrow ([D]\neg p \wedge [D]\neg q)$ in B forces the addition of $[D]\neg p$ in

B, thus causing a contradiction with $\langle D \rangle p$. An analogous contradiction occurs in C between $[D]\neg q$ and $\langle D \rangle q$. In both cases, at least one successor of the root is declared closed and thus the entire tableau is closed. In Figure 7 formulas in bold are those added to a node as an effect of the backward propagation of $\langle D \rangle r$.

6 Conclusions

In this paper, we investigated the decidability problem for logics of subinterval structures over dense linear orderings. First, we systematically studied the logic $\mathrm{D}_{\sqsubset}$ of the strict subinterval relation. We provided an alternative proof of its decidability by a non-standard small-model theorem and we developed an optimal tableau-based decision procedure for it. Then, we showed how to refine structures and techniques for $\mathrm{D}_{\sqsubset}$ to cope with the logic $\mathrm{D}_{\sqsubset}$ of the proper subinterval relation. We proved that it is decidable as well, and we devise an optimal tableau-based decision procedure for it. Finally, we implemented both tableau methods in Lotrec, a generic theorem prover for modal and description logics.

We restricted our attention to subinterval logics (strict, proper, and reflexive) over dense linear orders excluding point-intervals. However, the proposed decidability results and tableau constructions carry over to subinterval structures with point-intervals with minor modifications. In the two difficult cases (strict and proper subinterval relations), point-intervals are indeed easily definable by means of the formula $\langle D \rangle \bot$.

We are currently investigating the decision problem for the logic of subinterval structures over discrete and arbitrary linear orders that seems to be much more difficult to deal with.

Acknowledgements

We are grateful to the anonymous referees for their useful remarks and suggestions. This work has been funded by the bilateral project “Temporal logics in computer and information sciences”, supported by the Italian Ministero degli Affari Esteri and the National Research Foundation of South Africa, under the Joint Italy/South Africa Science and Technology Agreement. In addition, Davide Bresolin, Angelo Montanari, and Pietro Sala have been supported by the Italian PRIN project on “Constraints and preferences as a unifying formalism for system analysis and solution of real-life problems” and by the European INTAS project on “Algebraic and Deduction Methods in Non Classical Logics and their Applications to Computer Science”.

References

1. A.A. Aaby and K.T. Narayana. Propositional temporal interval logic is pspace complete. In *Proc. of 9th Inter. Conf. Automated Deduction*, volume 193 of *LNCS*, pages 218–237. Springer, 1985.
2. H. Bowman and S. Thompson. A tableau method for Interval Temporal Logic with projection. In *Proc. of TABLEAUX 1998*, volume 1397 of *LNAI*, pages 108–134. Springer, 1998.

3. H. Bowman and S. Thompson. A decision procedure and complete axiomatization of finite interval temporal logic with projection. *Journal of Logic and Computation*, 13(2):195–239, 2003.
4. D. Bresolin, V. Goranko, A. Montanari, and P. Sala. Tableau-based decision procedure for the logic of proper subinterval structures over dense orderings. In C. Areces and S. Demri, editors, *Proc. of the 5th Int. Workshop on Methods for Modalities (M4M)*, pages 335–351, 2007.
5. D. Bresolin, V. Goranko, A. Montanari, and P. Sala. Tableau Systems for Logics of Subinterval Structures over Dense Orderings. In N. Olivetti, editor, *Proc. of TABLEAUX 2007*, volume 4548 of *LNAI*, pages 73–89. Springer, 2007.
6. D. Bresolin, V. Goranko, A. Montanari, and G. Sciavicco. On Decidability and Expressiveness of Propositional Interval Neighborhood Logics. In *Proc. of the International Symposium on Logical Foundations of Computer Science (LFCS)*, volume 4514 of *LNCS*, pages 84–99. Springer, 2007.
7. D. Bresolin and A. Montanari. A tableau-based decision procedure for a branching-time interval temporal logic. In H. Schlingloff, editor, *Proc. of the 4th Int. Workshop on Methods for Modalities (M4M)*, pages 38–53, 2005.
8. D. Bresolin and A. Montanari. A tableau-based decision procedure for Right Propositional Neighborhood Logic. In B. Beckert, editor, *Proc. of TABLEAUX 2005*, volume 3702 of *LNAI*, pages 63–77. Springer, 2005.
9. D. Bresolin, A. Montanari, and P. Sala. An optimal tableau-based decision algorithm for Propositional Neighborhood Logic. In *Proc. of the 24th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 4393 of *LNCS*, pages 549–560. Springer, 2007.
10. D. Bresolin, A. Montanari, and G. Sciavicco. An optimal decision procedure for Right Propositional Neighborhood Logic. *Journal of Automated Reasoning*, 38(1-3):173–199, 2007.
11. M. Castilho, L. Fariñas del Cerro, O. Gasquet, and A. Herzig. Modal tableaux with propagation rules and structural rules. *Fundamenta Informaticae*, 32(3/4), 1997.
12. Z. Chaochen, M.R. Hansen, and P. Sestoft. Decidability and undecidability results for duration calculus. In *Proc. of the 10th Symposium on Theoretical Aspects of Computer Science*, number 665 in LNCS, pages 58–68. Springer, 1993.
13. L.K. Dillon, G. Kutty, L.E. Moser, P. M. Melliar-Smith, and Y.S. Ramakrishna. A real-time interval logic and its decision procedure. In *Proc. of the 13th Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 761 of *LNCS*, pages 173–192. Springer, 1993.
14. L.K. Dillon, P.M. Melliar-Smith, L.E. Moser, G. Kutty, and Y.S. Ramakrishna. Interval logics and their decision procedures. Part I: an interval logic. *Theoretical Computer Science*, 166(1–2):1–47, 1996.
15. L. Fariñas del Cerro, D. Fauthoux, O. Gasquet, A. Herzig, D. Longin, and F. Massacci. Lotrec: the generic tableau prover for modal and description logics. In *International Joint Conference on Automated Reasoning*, LNCS, page 6. Springer, 2001.
16. J. Y. Halpern and Y. Shoham. A propositional modal logic of time intervals. *Journal of the ACM*, 38(4):935–962, October 1991.
17. D. Van Hung and D.P. Guelev. Completeness and decidability of a fragment of duration calculus with iteration. In P.S. Thiagarajan and R. Yap, editors, *Advances in Computing Science*, volume 1742 of *LNCS*, pages 139–150. Springer, 1999.

18. K. Lodaya. Sharpening the undecidability of interval temporal logic. In *Proc. of 6th Asian Computing Science Conference*, number 1961 in LNCS, pages 290–298. Springer, 2000.
19. A. Montanari, G. Sciavicco, and N. Vitacolonna. Decidability of interval temporal logics over split-frames via granularity. In *Proc. the 8th European Conference on Logic in Artificial Intelligence (JELIA)*, number 2424 in LNAI, pages 259–270. Springer, 2002.
20. I. Shapirovsky. On PSPACE-decidability in Transitive Modal Logic. In R. Schmidt, I. Pratt-Hartmann, M. Reynolds, and H. Wansing, editors, *Advances in Modal Logic*, volume 5, pages 269–287. King's College Publications, London, 2005.
21. I. Shapirovsky and V. Shehtman. Chronological future modality in Minkowski spacetime. In P. Balbiani, N. Y. Suzuki, F. Wolter, and M. Zakharyaschev, editors, *Advances in Modal Logic*, volume 4, pages 437–459. King's College Publications, London, 2003.
22. J. van Benthem. *The Logic of Time: A Model-Theoretic Investigation into the Varieties of Temporal Ontology and Temporal Discourse, Second Edition.* Kluver, 1991.