On Disallowing Punctual Intervals in Reflexive Semantics of Halpern-Shoham Logic

Przemysław Andrzej Wałęga

1 University of Oxford, United Kingdom
2 University of Warsaw, Poland

Abstract

Halpern-Shoham logic (HS) is a very expressive and elegant formalism for interval temporal reasoning in which the satisfiability problem is undecidable. One of the methods to obtain HS-fragments of lower computational complexity is to adopt the softened (reflexive) semantics of the accessibility relations. In the paper we consider disallowing punctual intervals in reflexive semantics. We show that in this case we gain additional expressive power, which over discrete orders of time points results in PSpace-hardness of the Horn fragment of HS without diamond modal operators is and in undecidability of the core fragment of HS.

1 Introduction

The logic of Halpern and Shoham (HS in short) is one of the most well-known interval modal logics for temporal knowledge representation and reasoning [9, 8]. Its modal operators correspond to the set of binary relations between intervals known as the Allen’s relations, namely begins (rel_B), during (rel_D), ends (rel_E), overlaps (rel_O), adjacent to (rel_A), later than (rel_L), and their converses: rel_B, rel_D, rel_E, rel_O, rel_A, rel_L [1]. A model of HS may be seen as consisting of two layers. The first layer is a time-line, i.e., a set of time points ordered by an ‘earlier-later’ relation, and the second layer is a set of intervals over this time-line and a set of relations between intervals. In the seminal paper of Halpern and Shoham, the authors assumed that the order of time points is almost arbitrary (only imposed the so-called linear interval property is assumed), an interval is any pair of time points ⟨x, y⟩ such that y is not before x (hence, the punctual intervals starting and ending in the same time point are allowed), and that the relations between intervals correspond to the Allen’s relations [9].

Since the satisfiability problem of HS-formulas is undecidable a number of syntactical and semantical modifications of the logic have been studied [8, 3, 4, 5, 6, 11]. One of the ideas is to weaken semantics of Allen’s relations, by the so-called softening [10]. The obtained relations are known as reflexive semantics, as most of them become reflexive. Other modifications related to this paper consist of imposing additional conditions on the order of time points (e.g., discreteness or density), disallowing punctual intervals, and syntactically restricting the set of well-formed formulas [2, 5].

The most relevant results for this paper are depicted in Table 2, where < denotes irreflexive semantics, ≤ reflexive semantics, Non-S (non-strict semantics) allowing punctual intervals,
S (strict semantics) disallowing punctual intervals, Dis discrete timelines, and Den dense timelines. The satisfiability problem of HS-formulas in the Horn form (this fragment is denoted by HS\textsubscript{horn}) is undecidable under any combination of the above mentioned distinctions, whereas the satisfiability of HS-formulas in the core form (HS\textsubscript{core}) is undecidable under irreflexive semantics and its complexity under reflexive semantics is an open problem. Disallowing diamond modal operators (i.e., allowing only boxes) in the languages of HS\textsubscript{horn} and HS\textsubscript{core} results in HS\textsubscript{2}\textsubscript{horn} and HS\textsubscript{2}\textsubscript{core}, respectively. The satisfiability problem in both of them is tractable under (\textless, Non-S, Den), (\textless, S, Den), (\textless, Non-S, Dis), (\textless, Non-S, Den), and (\textless, S, Den).

Table 1: Computational complexity of HS fragments, where ‘undec’, ‘h’, and ‘co’ stand for undecidable, hard, and complete, respectively. Our results are written in bold.

<table>
<thead>
<tr>
<th></th>
<th>Irreflexive (\textless)</th>
<th>Reflexive (\leq)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Non-Strict (Non-S)</td>
<td>Strict (S)</td>
</tr>
<tr>
<td></td>
<td>Dis</td>
<td>Den</td>
</tr>
<tr>
<td>HS\textsubscript{horn}</td>
<td>undec</td>
<td>undec</td>
</tr>
<tr>
<td>HS\textsubscript{2\textsubscript{horn}}</td>
<td>P-co</td>
<td>P-co</td>
</tr>
<tr>
<td>HS\textsubscript{core}</td>
<td>undec</td>
<td>undec</td>
</tr>
<tr>
<td>HS\textsubscript{2\textsubscript{core}}</td>
<td>PS\textit{PSPACE}\text{-h}</td>
<td>in P</td>
</tr>
</tbody>
</table>

The main results of this paper are that under (\textless, S, Dis) the satisfiability problem in:

- $\text{HS}_{\text{horn}}$ is PSPACE-hard;

- $\text{HS}_{\text{core}}$ is undecidable.

Our results show that the combination of a lack of punctual intervals and discreteness of a time-line gives additional expressive power which was not available in other cases under reflexive semantics. In particular, we will show that under (\textless, S, Dis) the satisfiability problem of $\text{HS}_{\text{horn}}$ formulas reduces to the PSPACE-complete problem of checking whether a Turing machine which uses a polynomial memory on the empty input diverges on the empty input and the satisfiability problem for $\text{HS}_{\text{core}}$ formulas reduces to the undecidable problem of checking whether a given Turing machine diverges on the empty input.

To obtain the new results we show a trick that enables us to mimic under (\textless, S, Dis) proof techniques used in the irreflexive semantics [5]. We observe that under reflexive semantics the adjacent relation between intervals is not irreflexive if punctual intervals are allowed. Indeed, in this case each punctual interval is adjacent to itself. However, if punctual intervals are disallowed, then the adjacent relation becomes irreflexive. This observation allows us to introduce a formula forcing an alternating placement of two propositional variables over a sequence of subsequent unit intervals (i.e., intervals of length 1). Then, we show how to use these alternating propositional variables to pass information from one interval to another, and consequently how to simulate computation of a Turing machine.

The remaining part of the paper is organized as follows. In Section 2 we introduce syntax and semantics of HS and its fragments. Afterwards, we show in Section 3 that under (\textless, S, Dis) $\text{HS}_{\text{horn}}$ is PSPACE-hard and $\text{HS}_{\text{core}}$ is undecidable. We conclude the paper in Section 4.
2 Halpern-Shoham logic

The language of Halpern-Shoham logic consists of a set of propositional variables \( \text{PROP} \), classical propositional connectives \( \neg, \land, \lor, \) and 12 modal operators of the form \( \langle R \rangle \), where \( R \in \{ \mathbb{B}, \mathbb{E}, \mathbb{D}, \mathbb{O}, \mathbb{L} \} \) (in what follows, we denote this set by \( \text{HS}_{\text{rel}} \)). Well-formed \( \text{HS} \)-formulas are defined by the following abstract grammar:

\[
\varphi := p \mid \neg \varphi \mid \varphi \land \varphi \mid \langle R \rangle \varphi,
\]

where \( p \in \text{PROP} \) and \( R \in \text{HS}_{\text{rel}} \). \( \top, \bot, \lor, \land, \to \) are defined as usual, and \( \langle R \rangle \) is a dual modal operator to \( \langle R \rangle \) for any \( R \in \text{HS}_{\text{rel}} \). An \( \text{HS} \)-frame is a tuple \( \mathcal{F} = (\mathbb{D}, I(\mathbb{D}), \{ \text{rel}_R \}_{R \in \text{HS}_{\text{rel}}}) \) such that:

- \( \mathbb{D} = (\mathbb{D}, \leq) \) is an unbounded linear order;
- \( I(\mathbb{D}) \subseteq \mathbb{D} \times \mathbb{D} \) is a set of intervals over \( \mathbb{D} \);
- \( \text{rel}_R \subseteq I(\mathbb{D}) \times I(\mathbb{D}) \) is a binary relation between distinct intervals over \( \mathbb{D} \), for any \( R \in \text{HS}_{\text{rel}} \).

An \( \text{HS} \)-model is a tuple of the form \( \mathcal{M} = (\mathbb{D}, I(\mathbb{D}), \{ \text{rel}_R \}_{R \in \text{HS}_{\text{rel}}}, V) \), where \( (\mathbb{D}, I(\mathbb{D}), \{ \text{rel}_R \}_{R \in \text{HS}_{\text{rel}}}) \) is an \( \text{HS} \)-frame and \( V : \text{PROP} \to \mathcal{P}(I(\mathbb{D})) \). The satisfaction relation for an \( \text{HS} \)-model \( \mathcal{M} = (\mathbb{D}, I(\mathbb{D}), \{ \text{rel}_R \}_{R \in \text{HS}_{\text{rel}}}, V) \) and an interval \( \langle x, y \rangle \in I(\mathbb{D}) \) is defined inductively as follows:

\[
\mathcal{M}, \langle x, y \rangle \models p \quad \text{iff} \quad \langle x, y \rangle \in V(p), \text{ for any } p \in \text{PROP};
\]
\[
\mathcal{M}, \langle x, y \rangle \models \neg \varphi \quad \text{iff} \quad \mathcal{M}, \langle x, y \rangle \nmid \varphi;
\]
\[
\mathcal{M}, \langle x, y \rangle \models \varphi \land \psi \quad \text{iff} \quad \mathcal{M}, \langle x, y \rangle \models \varphi \text{ and } \mathcal{M}, \langle x, y \rangle \models \psi;
\]
\[
\mathcal{M}, \langle x, y \rangle \models \langle R \rangle \varphi \quad \text{iff} \quad \text{there is } \langle x', y' \rangle \text{ such that } \langle x, y \rangle \text{rel}_R \langle x', y' \rangle \text{ and } \mathcal{M}, \langle x', y' \rangle \models \varphi;
\]

for any \( R \in \text{HS}_{\text{rel}} \). A convenient representation of an \( \text{HS} \)-frame is obtained by treating an interval \( \langle x, y \rangle \) as a point in a two-dimensional Cartesian space \( \mathbb{D} \times \mathbb{D} \) such that the abscissa of this point has value \( x \) and its ordinate has value \( y \) [13]. In compass representation non-punctual intervals correspond to points lying in the north-western half-plane of \( \mathbb{D} \times \mathbb{D} \) (the points whose abscissa is strictly smaller than ordinate). Points lying on the diagonal correspond to punctual intervals. Let us fix any interval \( \langle x, y \rangle \). Then, intervals accessible from \( \langle x, y \rangle \) with \( \text{HS} \) modal operators may be determined on the basis of the relative position of the corresponding points in the two-dimensional Cartesian space as presented in Figure 1.

It is easy to see that any \( \text{HS} \)-formula can be transformed into an equisatisfiable formula which is a conjunction of implications (clauses), and vice versa (in the spirit of separation normal form introduced by [7]), i.e., into a formula generated by the following grammar:

\[
\varphi := \lambda \mid \neg \lambda \mid [U](\lambda \land \ldots \land \lambda \rightarrow \lambda \lor \ldots \lor \lambda) \mid \varphi \land \varphi,
\]

where \([U]\) is the universal modality, i.e., \([U] \psi \) is satisfied iff \( \psi \) is satisfied in every \( \langle x, y \rangle \in I(\mathbb{D}) \) whereas \( \lambda \), the so-called positive temporal literal, is a formula defined by the grammar:

\[
\lambda := \top \mid \bot \mid p \mid \langle R \rangle \lambda \mid [R] \lambda,
\]

where \( p \in \text{PROP} \) and \( R \in \text{HS}_{\text{rel}} \).

- \( \text{HS}_{\text{horn}} \) is obtained by restricting (1) to the grammar:

\[
\varphi := \lambda \mid [U](\lambda \land \ldots \land \lambda \rightarrow \lambda) \mid \varphi \land \varphi.
\]
On Disallowing Punctual Intervals in Reflexive Semantics of HS
Przemysław Andrzej Wałęga

Figure 1: Standard (a) and two-dimensional (b) representations of an HS-model, in which \(a, b\) is earlier than \(x, y\), and \(x, c\) is begun by \(x, y\).

- \(\text{HS}_{\text{horn}}^0\) is obtained by additional limitation imposed on \(\text{HS}_{\text{horn}}\), namely the grammar of positive temporal literals (2) is restricted to:
  \[\lambda := \top | \bot | p | [R]\lambda.\]

- \(\text{HS}_{\text{core}}\) is obtained by restricting (1) to the grammar:
  \[\varphi := \lambda | [U](\lambda \rightarrow \lambda) | [U](\lambda \lor \lambda) | [U](\lambda \land \lambda \rightarrow \bot) | \varphi \land \varphi.\]

In what follows we define restrictions imposed on HS semantics. First, the distinction between irreflexive and reflexive semantics is obtained by adopting the definitions of \(\text{rel}_R\) for \(R \in \text{HS}_{\text{rel}}\) as presented in Table 2.

Second, in the non-strict semantics the set \(I(D)\) is defined as:
\[
\{\langle x, y \rangle \mid x, y \in D \text{ and } x \leq y\},
\]
whereas in strict semantics \(I(D)\) is defined as:
\[
\{\langle x, y \rangle \mid x, y \in D \text{ and } x < y\}.
\]
Hence, in non-strict semantics punctual intervals are allowed, whereas in strict semantics they are forbidden.

3 Computational complexity

In what follows we show that under \((\leq, S, \text{Dis})\) the satisfiability problem for \(\text{HS}_{\text{horn}}^0\)-formulas is PSPACE-hard and for \(\text{HS}_{\text{core}}\)-formulas the problem is undecidable. The proofs are based on encodings of a Turing machine computation.
Table 2: Definitions of irreflexive and reflexive semantics of intervals relations.

<table>
<thead>
<tr>
<th>Irreflexive semantics:</th>
<th>Reflexive semantics:</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
<tr>
<td>(</td>
<td>x, y</td>
</tr>
</tbody>
</table>

Let us fix the notation, in which a deterministic Turing machine is \(M = (\Gamma, Q, q_{start}, q_{halt}, \delta)\), where \(\Gamma\) is \(M\)'s alphabet (containing the blank symbol \(\sqcup\) and the start symbol \(\triangleright\)), \(Q\) is the set of \(M\)'s states, \(q_{start} \in Q\) and \(q_{halt} \in Q\) are start and halting states, respectively, and \(\delta : \{Q - q_{halt}\} \times \Gamma \rightarrow Q \times \{L, R\}\) is a transition function which given a state of the machine and a symbol read by its head determines a new state and a symbol to be written in the current position of the head or movement of the head one cell to the left (L) or one cell to the right (R). We will denote \(M\)'s configuration in the \(n\)'th step of computation by an infinite sequence \(C_n(1), C_n(2), C_n(3), \ldots \) where \(C_n(m) \in \Gamma^+\) and \(\Gamma^+ = \Gamma \cup \{\sqcup \times \Gamma\}\) is the content of the \(m\)'th cell in the \(n\)'th step such that \((q, x)\) denotes that the cell contains \(x\), the head of \(M\) is above this cell, and \(M\) is in the state \(q\). Hence, the initial configuration is \((q_{start}, \triangleright), \sqcup, \sqcup, \sqcup, \ldots \). If \(M\) starts with empty input and with head above the first cell, then in the first \(n\) steps the head may visit only the first \(n\) cells. It follows that in the \(n\)'th step of computation all cells to the right of the \(n\)'th cell contain the blank symbol \(\sqcup\).

3.1 Horn fragment without diamonds

It is well known that given a Turing machine \(M\) and a polynomial \(f\) such that the computation of \(M\) with empty input uses at most \(f(|M|)\) amount of memory, where \(|M|\) is the size of \(M\), checking whether \(M\) diverges on empty input is a \(\text{PSPACE}\)-complete problem \([12]\). We will denote this problem by \(\text{PSPACE-BOUND BLANK-NON-HALTING}\) and in what follows we will show that it reduces to the satisfiability problem of \(\text{HS}_{\text{horn}}\) formulas under \((\leq, S, \text{Dis})\).

The main part of the proof consists of showing that \(\text{HS}_{\text{horn}}\) under \((\leq, S, \text{Dis})\) enables us to
force an alternating placement of propositional variables unit\textsubscript{1} and unit\textsubscript{2} in the consecutive unit intervals (i.e., intervals of length 1) as depicted in Figure 2. Then, we will show that such a placement of unit\textsubscript{1} and unit\textsubscript{2} allows us to use a technique known from the literature [5, Theorem 4.1 and Theorem 4.2] to encode PSPACE-BOUND BLANKNON-HALTING.

Figure 2: Placement of propositional variables, which is forced by $\varphi_{\text{unit1,2}}$ satisfied in $\langle u_0, u_1 \rangle$, where $\langle \leftarrow \text{next}_k, q_k \rangle$ for $k \in \{1, 2\}$ denotes a vertical line such that next\textsubscript{k} and q\textsubscript{k} are satisfied in all points belonging to this line.

Fix a polynomial $f$ such that the Turing machine $M$ starting its computation with empty input uses at most $f(|M|) = N$ tape cells. Notice that in step $i$ of $M$’s computation it suffices to consider $C_i(j)$ such that $j < N$ since the rest of the cells in $i$’th step contain the blank symbol. The formula forcing intended placement of unit\textsubscript{1} and unit\textsubscript{2} is defined as follows, where $k \in \{1, 2\}$:

$$
\varphi_{\text{unit1,2}} := \text{unit1} \land \\
\left[ U \right] (\text{unit}_k \rightarrow [E]p_k) \land \left[ U \right] (\text{unit}_k \rightarrow [B]A[q_k]) \land \\
\left[ U \right] (p_k \land q_k \rightarrow \bot) \land \\
\left[ U \right] (\text{unit}_1 \rightarrow [A]\text{next}_1) \land \left[ U \right] ([E]\text{next}_1 \rightarrow \text{unit}_2) \land \\
\left[ U \right] (\text{unit}_2 \rightarrow [A]\text{next}_2) \land \left[ U \right] ([E]\text{next}_2 \rightarrow \text{unit}_1) \land \\
\left[ U \right] (\text{unit}_1 \land \text{unit}_2 \rightarrow \bot) \land \\
[A][B][\text{unit} \land [U] (\text{unit} \land \text{unit}_k \rightarrow \bot)).
$$

(3) \hspace{1cm} (4) \hspace{1cm} (5) \hspace{1cm} (6) \hspace{1cm} (7) \hspace{1cm} (8) \hspace{1cm} (9)

**Lemma 1.** Let $M$ be any HS-model under ($\leq$, $S$, $Dis$) semantics. Assume that $M, \langle u_0, u_1 \rangle \models \varphi_{\text{unit1,2}}$ for some interval $\langle u_0, u_1 \rangle$ and let $u_1 < u_2 < \ldots$ be the infinite sequence of immediate $<$-successors. Then, $u_1$ is the immediate $<$-successor of $u_0$ and for any interval $\langle x, y \rangle$ the following conditions are satisfied:
1) \( \mathcal{M}, (x, y) \models \text{unit}_1 \) iff \( x = u_i \) and \( y = u_{i+1} \) for some even \( i \in \mathbb{N} \);

2) \( \mathcal{M}, (x, y) \models \text{unit}_2 \) iff \( x = u_i \) and \( y = u_{i+1} \) for some odd \( i \in \mathbb{N} \).

Proof sketch. Let \( \varphi_{\text{unit}_{1,2}} \) be satisfied in an interval \( (u_0, u_1) \) and let and \( u_0, u_1, u_2, \ldots \) be the sequence of immediate \(<\)-successors. By (3) the propositional variable \( \text{unit}_1 \) is satisfied in \( (u_0, u_1) \), and by (4) and (5) each interval which satisfies \( \text{unit}_1 \) or \( \text{unit}_2 \) is punctual, so \( (u_0, u_1) \) is punctual. Then, (6) and (7) states that an interval with \( \text{unit}_1 \) is always followed by an interval with \( \text{unit}_2 \), and vice versa. By (8) propositional variables \( \text{unit}_1 \) and \( \text{unit}_2 \) cannot be satisfied in the same interval and by (9) all intervals \( (x, y) \) such that \( x < u_0 \) do not satisfy \( \text{unit}_1 \) nor \( \text{unit}_2 \). It follows that \( \text{unit}_1 \) is satisfied exactly in intervals \( (u_i, u_{i+1}) \) for even \( i \) and \( \text{unit}_2 \) in intervals \( (u_i, u_{i+1}) \) for odd \( i \).

Next, we will slightly modify formulas from the proofs in [5, Theorem 4.1 and Theorem 4.2] to encode computation of a Turing machine \( M \). For \( k \in \{1, 2\} \) define \( \varphi_{\text{horn}M} \) as the conjunction of the following formulas, where \( N = f(|M|) \) :

\[
\text{cell}^{i,(q_{\text{var},D})}_k \bigwedge_{i < N | i \geq 1} \text{cell}^{i+1}_{k} \tag{10}
\]

\[
[U](\text{unit}_k \land \text{cell}^i_{k, x} \rightarrow [A]\text{cell}^i_{k, x, aux}) \tag{11}
\]

\[
[U](|E|\text{cell}^i_{k, x, aux} \rightarrow \text{cell}^i_{k, aux}) \land [U](|E|\text{cell}^i_{k, x, aux} \rightarrow \overline{\text{cell}}^{i}_{k, x}) \tag{12}
\]

\[
[U](\text{cell}^i_{k, x} \rightarrow \text{unit}_k) \tag{13}
\]

\[
\bigwedge_{i < N, x \in \Gamma^+}[U](\text{cell}^{i,(q_{\text{var},x})}_k \rightarrow \bot), \tag{14}
\]

and for any \( (q, x), (q', x') \in \Gamma^+ \) such that \( \delta(q, x) = (q', x') \):

\[
\bigwedge_{i < N}[U](\overline{\text{cell}}^{i,(q,x)}_k \rightarrow \text{cell}^{i,(q',x')}_k) \tag{15}
\]

\[
\bigwedge_{i,j < N | i \neq j}[U](\overline{\text{cell}}^{i,(q,x)}_k \land \text{cell}^{j,(y)}_k \rightarrow \text{cell}^{j,(y)}_k), \tag{16}
\]

and for any \( (q, x) \in \Gamma^+ \), \( q' \in Q \) such that \( \delta(q, x) = (q', R) \):

\[
\bigwedge_{i < N}[U](\overline{\text{cell}}^{i,(q,x)}_k \rightarrow \text{cell}^{i,x}_k) \tag{17}
\]

\[
\bigwedge_{i < N-1}[U](\overline{\text{cell}}^{i,(q,x)}_k \land \overline{\text{cell}}^{i+1,(y)}_k \rightarrow \text{cell}^{i+1,(q',y)}_k) \tag{18}
\]

\[
\bigwedge_{i < N-1, j < N | j \neq i, j+1}[U](\overline{\text{cell}}^{i,(q,x)}_k \land \overline{\text{cell}}^{j,(y)}_k \rightarrow \text{cell}^{j,(y)}_k), \tag{19}
\]
and for any \( (q, x) \in \Gamma^+ \), \( q' \in Q \) such that \( \delta(q, x) = (q', L) \):

\[
\bigwedge_{i < N | i \neq 0} [U]((cell_k^{i,(q,x)} \rightarrow cell_k^{i,x})
\]
(20)

\[
\bigwedge_{i < N | i \neq 0} [U](cell_k^{i,(q,x)} \land cell_k^{i-1,y} \rightarrow cell_k^{i-1,(q',y)})
\]
(21)

\[
\bigwedge_{i < N-1, j < N | i \neq 0 \neq j, i-1} [U](cell_k^{i,(q,x)} \land cell_k^{j,y} \rightarrow cell_k^{j,y}).
\]
(22)

**Lemma 2.** Let \( M \) be a deterministic Turing machine, which uses a polynomial number of tape cells when starting computation with empty input. Then, the following conditions are equivalent:

1. \( \varphi_{unit 1, 2} \land \varphi_{horn M} \) is HS-satisfiable under \((\leq, S, Dis)\);

2. \( M \) diverges with empty input.

**Proof sketch.** Intuitively, our aim is to represent that \( C_n(i) = x \) (in the \( n \)'th step of computation \( i \)'th cell contains \( x \)) with \( cell_k^{i,x} \) being satisfied in \( \langle u_n, u_{n+1} \rangle \), where \( k = 1 \) if \( n \) is even and \( k = 2 \) if \( n \) is odd. Formula (10) encodes the content of \( M \) in the first step of computation, (11)–(13) introduce auxiliary variables which pass an information about the content of a cell in a previous step of computation, and by (14) the halting state is never reached. Then, (15)–(16) encode the transition function of a Turing machine in the case when a new symbol is to be written on the tape, (17)–(19) in the case when the head is to be moved one cell to the right, and (20)–(22) when the head is to be moved one cell to the left. It is straight forward to check that \( \varphi_{horn M} \) is satisfiable if an only if the Turing machine diverges.

Hence, PSpace-Bound BlankNon-Halting reduces polynomially to HS\(^\square\)horn-satisfiability under \((\leq, S, Dis)\), so we obtain the following complexity result.

**Theorem 1.** HS\(^\square\)horn-satisfiability under \((\leq, S, Dis)\) is PSpace-hard.

### 3.2 Core fragment

Next, we will show that HS\(_{core}\)-satisfiability is undecidable under \((\leq, S, Dis)\). We will use the well known result that the problem of checking whether a Turing machine diverges on an empty input, denoted by BlankNon-Halting, is undecidable [12]. Given a Turing machine \( M \) we will construct an HS\(_{core}\)-formula which is satisfiable under \((\leq, S, Dis)\) if and only if \( M \) diverges with empty input, which implies undecidability of HS\(_{core}\) under \((\leq, S, Dis)\).

We will extensively use the alternating sequence of unit\(_1\) and unit\(_2\) encoded by the formula \( \varphi_{unit 1, 2} \) (see the previous sub-section). To make a reduction from BlankNon-Halting we will use a technique known from the literature [5, Theorem 4.4] which was used to prove undecidability of HS\(_{core}\) under irreflexive semantics. The approach is based on (i) encoding a Cantor-like enumeration of contents of Turing machine cells in consequent steps of computation (see Figure 3) by means of ’up-pointers’ whose intended placement is depicted in Figure 4 and then (ii) using the obtained placement of ’up-pointers’ to transfer information according to the transition function of a Turing machine. Under the reflexive we will obtain (ii) by a simple adaptation of the known technique but (i) will require non-trivial modifications, which make the proof relatively complex.
Recall, that by \( C_n(m) \in \Gamma^+ \) we denote the content of the \( m \)'th cell in the \( n \)'th step of \( M \)'s computation when starting with empty input. In order to refer to a particular cell in a given step of computation we introduce horizontal and vertical axes containing ordered positive natural numbers – as depicted in Figure 3. Then, a pair of natural numbers, namely \( x \)-coordinate and \( y \)-coordinate, enables us to refer to any cell of \( M \) in any step of computation, e.g., \((1, 1)\) refers to \( C_1(1)\).

Figure 3: Horizontal lines depict the subsequent contents of \( M \)'s tape. It suffices to consider cell’s \((x, y)\) for \( x \leq y \), which are enumerated and marked with grey background.

To encode the computation of \( M \) with empty input it suffices to consider cells denoted by pairs \((x, y) \in \mathbb{N}_+ \times \mathbb{N}_+\) such that \( x \leq y \), where \( \mathbb{N}_+ \) is the set of all positive natural numbers, i.e., without 0. Let us denote the set of coordinates referring to these cells by \( S \):

\[
S := \{(x, y) \mid x, y \in \mathbb{N}_+, \text{ and } x \leq y\} \cup (1, 0),
\]

where \((1,0)\) is an auxiliary element which makes the encoding more convenient. Let \( \text{enum} \) be the enumeration of \( S \) as depicted with dashed arrows in Figure 3, i.e., \( \text{enum} : \mathbb{N} \rightarrow S \) is a bijective function defined recursively as follows:

\[
\begin{align*}
\text{(enum1)} & \quad \text{enum}(0) = (1, 0); \\
\text{(enum2)} & \quad \text{If } \text{enum}(n) = (x, y) \text{ for } x < y, \text{ then } \text{enum}(n+1) = (x+1, y); \\
\text{(enum3)} & \quad \text{If } \text{enum}(n) = (x, x) \text{ for some } x \in \mathbb{N}, \text{ then } \text{enum}(n+1) = (0, x+1);
\end{align*}
\]

where \( n \) is any natural number. We introduce the relations \( \text{wall}, \text{diag}, \text{up}, \) and \( \text{line}_k \)'s for \( k \in \mathbb{N} \) in a similar way as it was done in [5], namely:

- \( \text{wall} \subseteq \mathbb{N} \times \{0, 1\} \) intends to determine numbers lying in the first column of the enumeration \( \text{enum} \). Hence, the intended definition of \( \text{wall} \) is that for any \( n \in \mathbb{N} \) it holds that:

\[
\text{wall}(n) = 1 \text{ whenever } \text{enum}(n) = (0, y) \text{ for some } y \in \mathbb{N};
\]
• \( \text{diag} \subseteq \mathbb{N} \times \{0, 1\} \) intends to determine numbers lying on the diagonal in the enumeration \( \text{enum} \). The intended definition of \( \text{diag} \) is that for any \( n \in \mathbb{N} \):

\[
\text{diag}(n) = 1 \quad \text{whenever} \quad \text{enum}(n) = (x, x) \quad \text{for some} \quad x \in \mathbb{N}; \tag{24}
\]

• \( \text{up} \subseteq \mathbb{N} \times \mathbb{N} \) intends to assign to each natural number its direct neighbour from above in the enumeration \( \text{enum} \). The intended definition of \( \text{up} \) is such that that for any \( m, n \in \mathbb{N} \):

\[
\text{up}(m) = n \quad \text{whenever} \quad \text{enum}(m) = (x, y) \quad \text{and} \quad \text{enum}(n) = (x, y + 1)
\]

for any \( x, y \in \mathbb{N} \); \tag{25}

• \( \text{line}_k \subseteq \mathbb{N} \times \{0, 1\} \) for any \( k \in \mathbb{N} \) intends to determine numbers lying in the \( k \)-th row in the enumeration \( \text{enum} \). The intended meaning of \( \text{line}_k \) is such that for any \( n \in \mathbb{N} \):

\[
\text{line}_k(n) = 1 \quad \text{whenever} \quad \text{enum}(n) = (x, k) \quad \text{for some} \quad x \in \mathbb{N}
\]

such that \( x \leq k \). \tag{26}

Next, we show conditions similar to the ones introduced in [5, Theorem 4.4], which imposed on the relations \( \text{wall} \), \( \text{diag} \), \( \text{up} \), and \( \text{line}_k \)'s force them to obtain the intended meaning. Define:

(c1) \( \text{line}_0(0) = 1 \) and \( \text{up}(0) = 1 \);

(c2) For each \( n \) there is exactly one \( k \) such that \( \text{line}_k(n) = 1 \);

(c3) If \( m < n \), \( \text{line}_k(m) = 1 \), and \( \text{line}_k(n) = 1 \), then for all \( o \) such that \( m < o < n \) it holds that \( \text{line}_k(o) = 1 \);

(c4) \( \text{wall}(n) = 1 \) whenever for some \( k \) we have \( \text{line}_k(n) = 1 \) and \( \text{line}_k(n - 1) = 0 \). \( \text{diag}(n) = 1 \) whenever for some \( k \) we have \( \text{line}_k(n) = 1 \) and \( \text{line}_k(n + 1) = 0 \). Moreover, if \( \text{line}_k(n) = 1 \) and \( \text{diag}(n) = 1 \), then \( \text{line}_{k+1}(n + 1) = 1 \);

(c5) If \( \text{diag}(n) = 0 \), then there is \( m \) such that \( \text{up}(m) = n \);

(c6) \( \text{up} : \mathbb{N} \rightarrow \mathbb{N} \) is an injective function;

(c7) If \( m < n \) then \( \text{up}(m) < \text{up}(n) \);

(c8) If \( \text{line}_k(m) = 1 \) and \( \text{up}(m) = n \), then \( \text{line}_{k+1}(n) = 1 \).

where \( m, n, o, k \) are any natural numbers. The approach used in [5] can be used to show that (c1)–(c8) force (23)–(26):

**Lemma 3.** Let \( \text{wall} \subseteq \mathbb{N} \times \{0, 1\} \), \( \text{diag} \subseteq \mathbb{N} \times \{0, 1\} \), \( \text{up} \subseteq \mathbb{N} \times \mathbb{N} \), and \( \text{line}_k \subseteq \mathbb{N} \times \mathbb{N} \) for any \( k \in \mathbb{N} \) be any relations. If they satisfy conditions (c1)–(c8), then it holds that (23)–(26).

In what follows we will use the implication tricks introduced in [5], which occur to work under \((\leq, S, \text{Dis})\) semantics as well. For any \( \text{HS} \) positive temporal literals \( \lambda_1, \lambda_2, \) and \( \lambda_3 \), define:

\[
\begin{align*}
\lambda_1 \land \lambda_2 \Rightarrow_H \lambda_3 & := [\mathcal{U}(\lambda_1 \rightarrow (\mathcal{A})p_1) \land [\mathcal{U}(\lambda_2 \rightarrow (\mathcal{A})p_2) \land [\mathcal{U}(p_2 \rightarrow \neg(\mathcal{B})p_1) \\
& \land [\mathcal{U}(p_1 \rightarrow p_3) \land [\mathcal{U}(p_1 \rightarrow (\mathcal{B})p_3) \land [\mathcal{U}(p_2 \rightarrow p_3) \land [\mathcal{U}(p_2 \rightarrow (\mathcal{B})p_3) \land [\mathcal{U}((\mathcal{A})p_3 \rightarrow \lambda_3); \tag{27} \\
\lambda_1 \land \lambda_2 \Rightarrow_V \lambda_3 & := [\mathcal{U}(\lambda_1 \rightarrow (\mathcal{A})p_1) \land [\mathcal{U}(\lambda_2 \rightarrow (\mathcal{A})p_2) \land [\mathcal{U}(p_2 \rightarrow \neg(\mathcal{E})p_1) \\
& \land [\mathcal{U}(p_1 \rightarrow p_3) \land [\mathcal{U}(p_1 \rightarrow (\mathcal{E})p_3) \land [\mathcal{U}(p_2 \rightarrow p_3) \land [\mathcal{U}(p_2 \rightarrow (\mathcal{E})p_3) \land [\mathcal{U}((\mathcal{A})p_3 \rightarrow \lambda_3). \tag{28}
\end{align*}
\]
Lemma 4 ([5, Claim 4.1]). Let $\mathcal{M}$ be any HS-model and let $\lambda_1$, $\lambda_2$, $\lambda_3$ be any HS positive temporal literals. The following hold:

- If $\mathcal{M} \models [\lambda_1 \land \lambda_2 \Rightarrow_H \lambda_3]$, $\mathcal{M}, \langle x_1, y \rangle \models \lambda_1$, and $\mathcal{M}, \langle x_2, y \rangle \models \lambda_2$ for any $x_1$, $x_2$, $y$, then for all $x$ we have $\mathcal{M}, \langle x, y \rangle \models \lambda_3$.
- If $\mathcal{M} \models [\lambda_1 \land \lambda_2 \Rightarrow_V \lambda_3]$, $\mathcal{M}, \langle x, y_1 \rangle \models \lambda_1$, and $\mathcal{M}, \langle x, y_2 \rangle \models \lambda_2$ for any $x$, $y_1$, $y_2$, then for all $y$ we have $\mathcal{M}, \langle x, y \rangle \models \lambda_3$.

Notice that if $[\lambda_1 \land \lambda_2 \Rightarrow_H \lambda_3]$ forces $\lambda_3$ to be satisfied in some interval $\langle x, y \rangle$, then it forces $\lambda_3$ to be satisfied in all intervals $\langle x', y \rangle$ for any $x'$ (i.e., $\lambda_3$ is horizontally stable [5]). Similarly, $[\lambda_1 \land \lambda_2 \Rightarrow_V \lambda_3]$ forces $\lambda_3$ to be vertically stable.

The most important and innovative part of the proof is to construct an HS$_{core}$-formula which encodes $enum$, i.e., conditions (c1)–(c8). The formula will force a specific placement of propositional variables $up_1$ and $up_2$, as depicted in Figure 4. Intuitively, $up_1 \lor up_2$ being satisfied in $\langle u_m, u_n \rangle$ represents that $up(m) = n - 1$, i.e., in the enumeration $enum$ the direct up neighbor of $m$ is $n - 1$.

![Figure 4: Intended placement of propositional variables, where ‘○’ denotes an interval in which unit is satisfied, ‘●’ an interval in which line1 or line2 is satisfied, ‘□’ an interval in which $up_1$ or $up_2$ is satisfied, whereas ‘above →’, ‘above →’, ‘← up1’, and ‘← up2’ denote lines in which above, above, $up_1$, and $up_2$ are true, respectively.](image)

We start encoding $enum$ by forcing placement of alternating propositional variables $unit_1$ and...
Lemma 6. Let \( \mathcal{M} \) be any HS-model under \((\leq,S,\text{Dis})\) semantics. Assume that \( \mathcal{M}, \langle u_0, u_1 \rangle \models \varphi_{\text{unit}1,2} \land \varphi_{\text{unit}} \) for some interval \( \langle u_0, u_1 \rangle \) and let \( u_1 < u_2 < \ldots \) be the infinite sequence of immediate \( \langle \cdot, \cdot \rangle \)-successors. Then, for any interval \( \langle x, y \rangle \) the following conditions are equivalent:

1. \( \mathcal{M}, \langle x, y \rangle \models \text{unit} \);

2. \( x = u_i \) and \( y = u_{i+1} \) for some \( i \in \mathbb{N} \).

Proof sketch. By (27) we have that \( \text{unit}1 \lor \text{unit}2 \) implies \( \text{unit} \). It remains to show the reverse implication. By (9) and (28) \( \text{unit} \) cannot be satisfied in any \( \langle x, y \rangle \) such that \( x < u_0 \). Then, (29)–(31) disallows \( \text{unit} \) to be satisfied in any \( \langle u_i, u_j \rangle \) such that \( i + 1 < j \). It follows that \( \text{unit} \) implies \( \text{unit}1 \lor \text{unit}2 \). \( \square \)

Now, we introduce propositional variables \( \text{line}_1 \) and \( \text{line}_2 \) which will enable us to distinguish horizontal ‘lines’ depicted in Figure 3. The intended placement of these variables is such that \( \text{line}_1 \lor \text{line}_2 \) is satisfied in an interval \( \langle u_m, u_n \rangle \) if all numbers from \( \{m, m + 1, \ldots, n\} \) belong to the same \( \text{line}_k \) from Figure 3. Such a placement is forced by the following formula:

\[
\varphi_{\text{line}} := [U](\text{line}_1 \rightarrow [A] \text{line}_2) \land [U](\text{line}_2 \rightarrow [A] \text{line}_1) \land \\
[A][B][\text{line} \land [U](\text{line}_k \land \text{line} \rightarrow \bot) \land \\
\text{line}_1 \land \\
[U](\text{line}_k \land [\overline{A}][\overline{B}] \text{line}_2 \rightarrow \bot) \land [U](\text{line}_k \land [A][E] \text{line}_1 \rightarrow \bot) \land \\
[U](\text{line}_1 \land [B][\overline{E}] \text{line}_2 \rightarrow \bot) \land \\
[U](\text{line}_2 \land [B][\overline{E}] \text{line}_1 \rightarrow \bot),
\]

where \( k \in \{1, 2\} \), and \( \text{line}_1 \), \( \text{line}_2 \), and \( \text{line} \) are propositional variables.

Lemma 6. Let \( \mathcal{M} \) be any HS-model under \((\leq,S,\text{Dis})\) semantics. Assume that \( \mathcal{M}, \langle u_0, u_1 \rangle \models \varphi_{\text{unit}1,2} \land \varphi_{\text{unit}} \land \varphi_{\text{line}} \) for some interval \( \langle u_0, u_1 \rangle \). Then, for any interval \( \langle x, y \rangle \) the following conditions are satisfied:

1) If \( \mathcal{M}, \langle x, y \rangle \models \text{line}_1 \), then there is \( z \) such that \( \mathcal{M}, \langle y, z \rangle \models \text{line}_2 \);

2) If \( \mathcal{M}, \langle x, y \rangle \models \text{line}_2 \), then there is \( z \) such that \( \mathcal{M}, \langle y, z \rangle \models \text{line}_1 \);

3) If \( \mathcal{M}, \langle x, y \rangle \models \text{unit} \), then there is exactly one interval \( \langle u, w \rangle \) such that \( u \leq x, w \geq y \), and \( \mathcal{M}, \langle u, w \rangle \models \text{line}_1 \lor \text{line}_2 \). Moreover, \( \langle u, w \rangle \) is such that either \( \mathcal{M}, \langle u, w \rangle \models \text{line}_1 \) or \( \mathcal{M}, \langle u, w \rangle \models \text{line}_2 \).
Proof sketch. The conditions 1) and 2) follow directly from (32). Hence, there is a sequence \( \langle u_{k0}, u_{k1}, u_{k2}, \ldots \rangle \) with alternating line\(_1\) and line\(_2\). By (34) line\(_1\) is satisfied in \( \langle u_0, u_1 \rangle \) and by (33) line\(_1\) and line\(_2\) are not satisfied in any interval \( (x, y) \) such that \( x < u_0 \). Then, by (35)–(37) we obtain that line\(_1\) and line\(_2\) cannot be satisfied anywhere accept this alternating sequence, which implies the condition 3).

Finally, we force the intended placement of up\(_1\) and up\(_2\), i.e., we want to force up\(_1\) \( \lor \) up\(_2\) to hold in \( \langle u_m, u_n \rangle \) whenever up\((m) = n - 1\), i.e., \( n - 1 \) is the direct up neighbor of \( m \) in the enumeration enum. The particularly important part is to ensure that these 'up-pointers' are functional and injective in a sense that for each \( u_m \) there is exactly one \( u_n \) pointed by up\(_1\) or up\(_2\), and each \( u_n \) is pointed by up\(_1\) or up\(_2\) from at most one \( u_m \). We encode this condition by a quite complex interplay between various propositional variables – mainly between up\(_1\), up\(_2\), and line\(_1\), line\(_2\). We achieve it by means of the following formula:

\[
\varphi_{up} := (A)0_{up} \land [U](0_{up} \rightarrow \text{unit}) \land [U](0_{up} \rightarrow \text{above}_1) \land \]
\[
[U](0_{up} \rightarrow \text{above}_1) \land [U](0_{up} \rightarrow \text{E} \text{above}) \land [above_k \land \text{above} \Rightarrow_H (E)(E)up_k] \land \]
\[
[U](0_{up} \rightarrow \overline{up}_1) \land [A][B][B]\overline{up}_k \land [U](up_k \land \overline{up}_k \rightarrow \perp) \land \]
\[
[E][B]\overline{above} \land \]
\[
[U][(E)(E)line_k \rightarrow \overline{above}] \land \]
\[
[U](above \land \overline{above} \rightarrow \perp) \land \]
\[
[(D)line_k \land (A)[E]line_k \Rightarrow_H \overline{above} \land \]
\[
[U](up_1 \rightarrow (A)\overline{above}_2) \land [U](up_2 \rightarrow (A)\overline{above}_1) \land \]
\[
[U](\overline{above}_k \rightarrow \text{unit}) \land \]
\[
[above_k \land \overline{above} \Rightarrow_H (A)\overline{above}_k] \land \]
\[
[U](\text{unit}_k \rightarrow (B)up_k) \land \]
\[
[U](\text{unit}_1 \rightarrow (B)\overline{up}_2) \land [U](\text{unit}_2 \rightarrow (B)\overline{up}_1) \land \]
\[
[U](\overline{above} \land up_k \rightarrow \perp) \land \]
\[
[U](\overline{up}_1 \land (D)\overline{up}_2 \rightarrow \perp) \land [U](\overline{up}_2 \land (D)\overline{up}_1 \rightarrow \perp) \land \]
\[
[U](up_k \land (D)line_k \rightarrow \perp) \land \]
\[
[U](line_k \rightarrow (A)r) \land [U](r \rightarrow \text{unit}) \land [U](r \rightarrow \overline{E} \text{above}) \land \]
\[
[up_k \land (D)line_i \Rightarrow_V (B)(B)line_i], \]
where \( k, l \in \{1, 2\} \), and up\(_1\), up\(_2\), \( \overline{up}_1\), \( \overline{up}_2\), above\(_1\), above\(_2\), above, and \( \overline{above} \) are propositional variables.

Lemma 7. Let \( M = (D, I(D), \{ \text{rel} \}_R \in \text{HS}_w, V) \) be an HS-model under \( (\leq, S, \text{Dis}) \) and \( u_0 < u_1 < u_2 < \ldots \) an infinite sequence of immediate \( \prec \)-successors in D. Assume that \( M, (u_0, u_1) \models \varphi_{\text{unit}_{1,2}} \land \varphi_{\text{unit}} \land \varphi_{\text{line}} \land \varphi_{up} \). Then, for all \( (x, y) \in I(D) \) the following conditions are equivalent:

1. \( M, (x, y) \models up_1 \lor up_2; \)

2. \( x = u_i \) and \( y = u_j \) for \( i, j \in \mathbb{N} \) such that we have enum\((i) = (u, w)\) and \( \text{enum}(j - 1) = (u, w + 1) \) for some \( u, w \in \mathbb{N} \) (in other words up\((i) = j \) for up defined by (25)).
Proof sketch. As already mentioned, the main part of the proof is to show that \( u_p \) and \( u_q \) are functional and injective. By (49) each \( u_m \) points with an ‘up-pointer’ to some \( u_n \) and by (51) each \( u_n \) can be pointed by at most one ‘up-pointer’. Hence, it remains to show that no \( u_m \) can ‘up-point’ two \( u_n \)’s.

First, for each \( n \) which is not on a diagonal in enumeration \( enum \) we mark by means of (45) the interval \( \langle u_n, u_{n+1} \rangle \) with a propositional variable \( \text{above} \). Then, by (40) and a proper placement of \( \text{above}_1 \) and \( \text{above}_2 \) we force that each non diagonal \( n \) is ‘up-pointed’. The condition that no \( u_m \) can ‘up-point’ two \( u_n \)’s is forced by (53) and (55), which encode the interplay between \( \text{line}_k \)’s and \( \text{up}_k \)’s.

We have shown the hard part of the proof, i.e., forcing the placement of propositional variables as presented in Figure 4. Using this placement we can adapt slightly modified formulas from [5, Theorem 4.4] to encode a computation of a Turing machine with empty input. First, we represent transition function \( \delta \) by means of \( \text{triples to cells} \) function \( \tau \). Let:

\[
\Sigma := \Gamma - \{\triangleright, \sqcup\}; \quad Q^- = Q - \{q_{\text{halt}}\}; \quad LEnd = \{\triangleright\} \cup \{Q^- \times \{\triangleright\}\}.
\]

Then, define:

\[
W := \{Q^- \times \Sigma \times \Sigma\} \cup \{\Sigma \times \{Q^- \times \Sigma\} \times \Sigma\} \cup \{\Sigma \times \Sigma \times \{Q^- \times \Sigma\}\} \cup \\
\{\text{LEnd} \times \Sigma \times \Sigma\} \cup \{\{\sqcup\} \times \text{LEnd} \times \Sigma\} \cup \{\Sigma \times \{\sqcup\} \times \text{LEnd}\} \cup \\
\{((q_{\text{start}}, \triangleright), \sqcup, \triangleright)\}.
\]

\( \tau : W \rightarrow \Gamma^+ \) is such that for any \( (x, y, z) \in W \) we have:

\[
\tau(x, y, z) = \begin{cases} 
(q', y) & \text{if either } x \in \{Q^- \times \{q_{\text{halt}}\}\} \times \{\Sigma \cup \{\triangleright\}\} \text{ and } \delta(x) = (q', R), \text{ or } z \in \{Q^- \times \{q_{\text{halt}}\}\} \times \{\Sigma \cup \{\sqcup\}\} \text{ and } \delta(z) = (q', L); \\
(q', y') & \text{if } y \in \{Q^- \times \{q_{\text{halt}}\}\} \times \Gamma \text{ and } \delta(y) = (q', y'); \\
y' & \text{if } y = (q, y') \text{ and } \delta(y) = (q', L) \text{ or } \delta(y) = (q', R); \\
y & \text{otherwise.}
\end{cases}
\]

\( \tau \) determines computation of \( M \) in a sense that for any \( n, m \in \mathbb{N}_+ \) such that \( m \leq n \), we have:

\[
C_n(m) = \begin{cases} 
\tau(\sqcup, C_{n-1}(1), C_{n-1}(2)) & \text{if } m = 1; \\
\tau(C_{n-1}(m-1), C_{n-1}(m), C_{n-1}(m+1)) & \text{if } 1 < m < n; \\
\tau(C_{n-1}(n-1), \sqcup, C_n(1)) & \text{if } m = n.
\end{cases}
\]

For a given Turing machine \( M = (\Gamma, Q, q_{\text{start}}, q_{\text{halt}}, \delta) \) we encode \( M \)’s computation with empty
input as follows:

$$\varphi_M := \langle A \rangle (q_{\text{start}}, \triangleright) \land$$

$$[U](\text{line}_k \rightarrow \langle E \rangle \cup) \land$$

$$\bigwedge_{x \in \Gamma} [U](\langle q_{\text{halt}}, x \rangle \rightarrow \bot) \land$$

$$\bigwedge_{x \in \Gamma^+} [U](x \rightarrow \text{unit}) \land$$

$$\bigwedge_{x,y \in \Gamma^+} [U](x \land y \rightarrow \bot) \land$$

$$\bigwedge_{(x,y,z) \in W} \left( [y \land \langle A \rangle z \Rightarrow_H \langle E \rangle(y,z)] \land \right.$$

$$\left. [U]((y, z) \rightarrow \text{unit}) \land \right$$

$$\left. [(y, z) \land \langle A \rangle x \Rightarrow_H \langle E \rangle(x, y, z)] \land \right$$

$$\left. [U]((x, y, z) \rightarrow \text{unit}) \land \right$$

$$\left. [(x, y, z) \land \text{unit}_k \Rightarrow_V \langle B \rangle(x, y, z)_k] \land \right$$

$$\left. [U]((x, y, z)_k \rightarrow \text{up}_k) \land \right$$

$$\left. [U]((x, y, z)_k \rightarrow \langle E \rangle \tau(x, y, z)) \right).$$

where $k \in \{1, 2\}$. Since the above encoding is a quite straightforward adaptation of the one in [5, Theorem 4.4] we leave the following lemma without a proof and direct the reader to [5].

**Lemma 8.** The following conditions are equivalent for any deterministic Turing machine $M$:

1. $\varphi_{\text{unit}1,2} \land \varphi_{\text{unit}} \land \varphi_{\text{line}} \land \varphi_{\text{up}} \land \varphi_M$ is HS-satisfiable under $(\leq, S, \text{Dis})$;

2. $M$ diverges with empty input.

Notice that $\varphi_{\text{unit}1,2} \land \varphi_{\text{unit}} \land \varphi_{\text{line}} \land \varphi_{\text{up}} \land \varphi_{\text{horn}} M$ is an HS$_{\text{core}}$-formula, so the undecidability of HS$_{\text{core}}$-satisfiability under $(\leq, S, \text{Dis})$ follows.

**Theorem 2.** HS$_{\text{core}}$-satisfiability under $(\leq, S, \text{Dis})$ is undecidable.

## 4 Conclusions

In the paper we have studied computational complexity of HS-fragments under reflexive semantics in which punctual intervals are forbidden, and a time-line is discrete, denoted by $(\leq, S, \text{Dis})$. As we have showed, disallowing punctual intervals in reflexive semantics allows us to retain some expressive power which was lost by weakening the semantics (i.e., choosing reflexive rather than irreflexive semantics). The additional expressive power allows us to show new complexity results, namely we have proved that under $(\leq, S, \text{Dis})$ the satisfiability problem in HS$_{\text{horn}}^D$ is PSPACE-hard and in HS$_{\text{core}}$ it is undecidable. In contrast, we recall that if punctual intervals are allowed or the time-line is dense, then under reflexive semantics the satisfiability problem in HS$_{\text{horn}}^D$ is P-complete, and its decidability in HS$_{\text{core}}$ is an open problem.

One of the important properties of the semantics $(\leq, S, \text{Dis})$ is that the accessibility relations $\text{rel}_A$ and $\text{rel}_R$ is irreflexive. As a result, we were able to introduce (quite sophisticated) formulas...
simulating a computation of a Turing machine. Among the technical aspects of the proofs a particularly important idea we have developed is to introduce propositional variables with two indices and formulas which force these variables to be satisfied alternately (for an example see Figure 4). This method turned out to be especially important and allowed us to overcome serious problems caused by low expressive power of reflexive semantics.

Among the interesting open problems concerning complexity of $\text{HS}$-fragments under the semantics $(\leq, S, \text{Dis})$ we plan to study the following:

- Is $\text{HS}_{korn}^\Box$ under $(\leq, S, \text{Dis})$ decidable? If so, what is its computational complexity?
- What is the computational complexity of $\text{HS}_{core}^\Box$ under $(\leq, S, \text{Dis})$? It does not seem that our proofs may be adapted for $\text{HS}_{core}^\Box$.

Acknowledgements

This work was supported by the NCN grant 2016/23/N/HS1/02168 and by the Foundation for Polish Science (FNP).

References