Decidability of Model Checking Multi-Agent Systems with Regular Expressions against Epistemic HS Specifications

Jakub Michaliszyn, Piotr Witkowski
University of Wrocław, Poland
{jmi,pwit}@cs.uni.wroc.pl

Abstract

Epistemic Halpern-Shoham logic (EHS) is an interval temporal logic defined to verify properties of Multi-Agent Systems. In this paper we show that model checking Multi-Agent Systems with regular expressions against EHS specifications is decidable. We achieve this by reducing the model checking problem to the satisfiability problem of Monadic Second-Order Logic on trees.

1 Introduction

Model checking is the leading technique in verification. Despite its advantages, the number of commercial applications of this technique is still relatively small. This is often explained by the high entry level for model checking specialists – writing appropriate specifications requires a lot of knowledge and, at the same time, a lot of precision. Therefore, for many companies, it is much easier and cost-efficient to perform tests rather than to verify the software. This means that one of the most important challenges for the verification community is to lower the difficulty of performing formal verification. Among many approaches to this problem, one is to use a specification language that treats intervals (processes), rather than time points, as primary objects. It is believed that many important properties are significantly easier to express in terms of intervals and their temporal correspondence than using time points [Lomuscio and Michaliszyn, 2013].

Early attempts to interval-focused verification include the Moszkowski's Interval Temporal Logic [Moszkowski, 1983] and the Propositional Dynamic Logic [Fischer and Ladner, 1979]. The former, however, uses quite a complicated language (one of the main operators allows to split a given interval into two parts, which is quite unreadable), and the latter, while allows for actions that may be seen as intervals, is still essentially point based.

Recently, two lines of research emerged to define a logic for the model checking problem in which intervals are first-class citizens. First, in [Lomuscio and Michaliszyn, 2013], the Epistemic Halpern-Shoham (EHS) logic was introduced. Not much later, [Montanari *et al.*, 2014] employed the logic HS, later extended with regular expressions [Bozzelli *et al.*, 2017]. Both EHS and HS are based on the Halpern-Shoham logic [Halpern and Shoham, 1991], but work on different assumptions.

The most significant difference, we believe, is the way that both logics treat past events. To exemplify this, consider the following property: if the server is offline, then there was an interval earlier where a warning message was sent. In the EHS setting, every property is considered in an interval with a *history*, i.e., a computation from the initial state to the beginning of this interval. So, the backward modality "there was an interval earlier" refers to an event that really happened during this computation. In the HS case, on the other hand, there are no histories, therefore this means "it was possible to reach this interval by a computation containing sending a warning". This makes the semantics of HS easier, but also less intuitive.

The other difference between EHS and HS is less fundamental but easier to notice: EHS is defined for multi-agent systems while HS typically considers a single-agent system. Therefore, EHS contains also *epistemic* modalities, typical for multi-agent systems: K_i ("agent i knows...") and C_Γ ("it is common knowledge among the group of agents Γ...").

A lot of research was performed to establish the decidability and complexity status of the model checking problem with HS [Molinari *et al.*, 2015; Bozzelli *et al.*, 2016a; Molinari *et al.*, 2016; Bozzelli *et al.*, 2018a; Bozzelli *et al.*, 2018b; Molinari *et al.*, 2018], as well as its expressibility [Bozzelli *et al.*, 2016b; Goranko *et al.*, 2004]; perhaps the best summary is the Alberto Molinari PhD thesis [Molinari, 2019]. For the whole logic, the model checking problem is decidable, but no elementary upper bound is known. The best known lower bound, on the other hand, is the EXPSPACE lower bound that works for any fragment containing two modalities: "begins with" and "ends with". On the other hand, the decidability status of the model checking against EHS specifications was not known. Here, we show that the model checking problem for EHS is decidable. The obtained complexity is nonelementary, which matches the best known upper bound for a (simpler) logic HS.

To show the decidability, we reduce the problem to the satisfiability problem of the monadic second-order logic on trees (SkS), which is known to be non-elementary decidable. To deal with epistemic modalities, we consider a variant of SkS in which the trees are infinite in both directions, i.e., every node has a predecessor and successors.

To complete the study on the decidability status, we also consider EHS in an LTL-like scenario, where the question of interest is whether all the tracks of a given model satisfy a given EHS formula (that is, once a track is selected, the modalities refer only to intervals within this track). Under such assumptions, the model checking problem is undecidabile even if only epistemic modalities are allowed.

Related work A large number of point-based logics is used in the model-checking scenario, including LTL, CTL, ATL and so on; in some restricted cases, their expressive power matches the expressive power of the logic HS [Bozzelli *et al.*, 2016b]. However, none of them is equivalent (in terms of the expressive power) to the logic EHS.

Two extensions of the Propositional Dynamic Logic are seemingly close to EHS. First one is E-PDL [van Benthem *et al.*, 2006], which adds epistemic modalities to PDL. These modalities, however, are interpreted over points rather than over intervals. The other one is called "PDL with all extras" [Lange, 2006], and allows to use conjunction and complementation with PDL actions. However, the semantics of these operations is still point-based: for example, the action $\langle a_1 \wedge a_2 \rangle$ means "there is a path satisfying a_1 and there is a path satisfying a_2... ", rather than "there is a path satisfying a_1 and a_2... ", so it cannot be used to express conjunctions of properties of intervals.

2 Interval-Based Interpreted Systems

We assume a set of agents $\mathbf{A} = \{0, \ldots, m\}$ to be fixed for the rest of this paper; the agent 0 is often called *the environment*.

The models we use for verification are called *interpreted systems with regular labellings* (ISRL). These models are similar to standard Multi-Agent Systems, except that the labelling function is defined on sequences of states rather than single states by means of regular expressions.

Given a family of finite alphabets $X = \{X_0, \ldots, X_m\}$, the set of regular expressions over X, denoted by RE_X, is defined by the following BNF expression:

$$e ::= \epsilon \mid s \mid e \cdot e \mid e + e \mid e^*$$

where s is a Boolean combination of conditions $i\colon l$, where $i \in \{0, \ldots, m\}$ and $l \in X_i$. We allow parentheses for grouping and often omit the concatenation symbol "$\cdot$".

A regular expression over X is interpreted over a word whose letters are from the product $X_0 \times \cdots \times X_m$. The semantics of regular expressions over X is defined in the usual manner, i.e., by a function Lang that for a regular expression returns a language of words accepted by this expression. The only difference is the case of s, where we define $\mathrm{Lang}(s)$ to be the set of letters $(x_0, \ldots, x_m)$ such that s is satisfied when we replace each "$i\colon$ " by "$x_i =$".

The languages definable with such regular expressions are regular, and therefore can be recognized by finite automata.

Example 1. *The regular expression* $(0\colon a \vee 1\colon b)^*$ *over* $\{\{a, b\}, \{a, b\}\}$ *defines the language* $\{(c_0, d_0) \ldots (c_m, d_m) \mid \forall i. c_i = a \vee d_i = b\}$.

The models we study are defined as follows.

Definition 2 (ISRL, [Lomuscio and Michaliszyn, 2016])**.** *An* interpreted system with regular labellings *(ISRL) is a tuple* $IS = (\{L_i, l_i^I, ACT_i, P_i, t_i\}_{i \in \mathbf{A}}, \lambda)$, *where for each* $i \in \mathbf{A}$:

- L_i *is a finite* set of local states *for agent* i,
- $l_i^I \in L_i$ *is the* initial state *for agent* i,
- ACT_i *is a finite set of* local actions *available to agent* i,
- $P_i\colon L_i \to 2^{ACT_i}$ *is a* local protocol function *for agent* i, *returning the set of possible actions in a local state,*
- $t_i \subseteq L_i \times ACT \times L_i$, *where* $ACT = ACT_0 \times \cdots \times ACT_m$, *is a* local transition relation *for agent* i *returning the next local state when a joint action is performed by all agents on a given local state.*

Furthermore, $\lambda\colon Var \to RE_{\{L_0, \ldots, L_m\}}$ *is a labelling function, where* Var *is a finite set of propositional variables.*

Paths For an ISRL as above, we define the set of global configurations G as $L_0 \times \cdots \times L_m$, and the *initial global configuration* l_G^I as $(l_0^I, \ldots, l_m^I)$. The transition relation t_G between global configurations is induced by IS as follows: $t_G((l_0, \ldots, l_m), (l'_0, \ldots, l'_m))$ iff there exists a joint action $(a_0, \ldots, a_m) \in ACT$ such that for all i we have $a_i \in P_i(l_i)$ and $t_i(l_i, (a_0, \ldots, a_m), l'_i)$.

A *path* is a finite sequence $g_1 g_2 \ldots g_k$ of elements of G such that for each $i < k$ we have $t_G(g_i, g_{i+1})$.

For each global configuration $g = (l'_0, \ldots, l'_m) \in G$ and an agent i, by $l_i(g)$ we denote the local state l'_i.

Definition 3 (Model)**.** *Given an ISRL* $IS = (\{L_i, l_i^I, ACT_i, P_i, t_i\}_{i \in \mathbf{A}}, \lambda)$, *the* model of IS *is a tuple* $M = (S, s^I, t, \{\sim_i\}_{i \in A}, \lambda)$, *where*

- S *is the* set of paths *of* IS *starting in* l_G^I,
- $s^I = l_G^I$ *is the (single-state path containing the)* initial state *of the system,*
- t *is the* transition relation *such that* $t(s_1, s_2)$ *iff there is a global state* g *such that* $s_1 g = s_2$ *(i.e., the path* s_1 *is the prefix of* s_2 *of length* $|s_2| - 1$*),*
- $\sim_i \subseteq S^2$ *is the* epistemic equivalence relation for agent i *such that* $g_1 \ldots g_k \sim_i g'_1 \ldots g'_l$ *iff* $l_i(g_k) = l_i(g'_l)$.
- λ *is the* labelling function *(same as in* IS*).*

We assume that the system is deadlock-free, i.e., for every state there is at least one possible joint action of agents. This assumption is only to make the presentation easier: any system can be easily amended to meet this requirement, e.g., by adding a "sink state". Therefore, the models are labelled infinite unordered trees with the set of nodes S, the root s_I, the set of edges t, the labelling λ and epistemic relations $\sim_i$.

For paths p_1, p_2, we write $p_1 \sqsubset p_2$ if p_1 is a proper prefix of p_2 and $p_1 \sqsubseteq p_2$ if $p_1 \sqsubset p_2$ or $p_1 = p_2$. An *interval* is a pair $[s, t]$, such that $s, t \in S$ and $s \sqsubseteq t$. An interval represents a path in a system augmented with a history of states that lead to this path from the initial state of the system. It is worth to note that in a model, $s \sqsubset t$ means that t is a descendant of s.

Comment. The definition presented here is essentially equivalent to the one in [Lomuscio and Michaliszyn, 2016]. The main difference is that they defined intervals as pairs (p, p_h) such that $p_h p$ is a path from a starting configuration. In our notion, this is represented as $[p_h g, p_h p]$, where g is the first element of p. We decided to use an alternative representation to make the translation from EHS to MSO on trees easier to present.

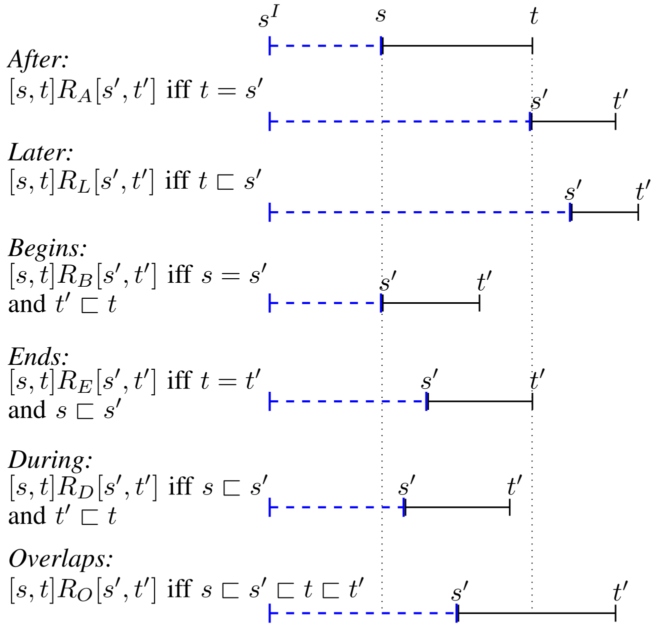
s^I
s
t
After:
$[s,t]R_A[s',t']$ iff $t = s'$
s'
t'
Later:
$[s,t]R_L[s',t']$ iff $t \sqsubset s'$
s'
t'
Begins:
$[s,t]R_B[s',t']$ iff $s = s'$ and $t' \sqsubset t$
s'
t'
Ends:
$[s,t]R_E[s',t']$ iff $t = t'$ and $s \sqsubset s'$
s'
t'
During:
$[s,t]R_D[s',t']$ iff $s \sqsubset s'$ and $t' \sqsubset t$
s'
t'
Overlaps:
$[s,t]R_O[s',t']$ iff $s \sqsubset s' \sqsubset t \sqsubset t'$
s'
t'

Figure 1: Basic temporal relations. Dashed lines represent the histories of intervals.

3 The Logic EHS

EHS uses twelve temporal operators, defined as in [Halpern and Shoham, 1991]. Six of these relations are presented in Figure 1. The remaining six relations are defined as their inverses: $R_{\bar{X}} = {R_X}^{-1}$. EHS allows also for two kinds of epistemic modalities: K_i for an agent i ("agent i knows") and C_Γ for a set of agents Γ ("it is common knowledge in Γ").

Definition 4 (Syntax of EHS)**.** *The syntax of EHS is defined by the following BNF.*

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid C_\Gamma\varphi \mid \langle X\rangle\varphi$$

where $p \in$ Var is a propositional variable, $i \in \boldsymbol{A}$ is an agent, $\Gamma \subseteq \boldsymbol{A}$ is a set of agents, and X is one of the HS modalities: A, $\bar{A}$, B, $\bar{B}$, D, $\bar{D}$, E, $\bar{E}$, L, $\bar{L}$, O or $\bar{O}$.

We use standard abbreviations, including $[X]\varphi$ for $\neg\langle X\rangle\neg\varphi$ (universal temporal operator) and the usual Boolean connectives $\vee$, $\Rightarrow$, $\Leftrightarrow$ and the constants $\top$, $\bot$.

Two intervals I, I' are indistinguishable for an agent i, denoted as $I \sim_i I'$, if they have the same length and the states on corresponding positions are indistinguishable [Lomuscio and Michaliszyn, 2013]. Formally, $[s, sg_1 \dots g_k] \sim_i [s', s'g'_1 \dots g'_l]$ iff $k = l$, $s \sim_i s'$ and $sg_1 \dots g_j \sim_i s'g'_1 \dots g'_j$ for each $j \leq k$. Similarly, for a group of agents Γ, we define $\sim_\Gamma$ on states as the transitive closure of $(\bigcup_{i\in\Gamma} \sim_i)$, and then we extend this to intervals in the same manner.

We now define the satisfaction relation.

Definition 5 (Semantics of EHS)**.** *Given an EHS formula φ, an ISRL IS, its model $M = (S, s^I, t, \{\sim_i\}_{i\in A}, \lambda)$, and an interval $I = [g_1 \dots g_k, g_1 \dots g_l]$, the formula φ holds in the interval I, denoted $M, I \models \varphi$, iff one of the following holds:*

- *$\varphi = p$ and $g_k \dots g_l \in$ Lang$(\lambda(p))$,*
- *$\varphi = \neg\varphi'$ and it is not the case that $M, I \models \varphi'$,*
- *$\varphi = \varphi_1 \wedge \varphi_2$ and $M, I \models \varphi_1$ and $M, I \models \varphi_2$,*
- *$\varphi = K_i\varphi$, $i \in \boldsymbol{A}$, and $M, I' \models \varphi$ for all $I' \sim_i I$,*
- *$\varphi = C_\Gamma\varphi$, $\Gamma \subseteq \boldsymbol{A}$, and $M, I' \models \varphi$ for all $I' \sim_\Gamma I$,*
- *$\varphi = \langle X\rangle\varphi$ and there is an interval I' such that IR_XI' and $M, I' \models \varphi$, where R_X is a temporal relation.*

We write $IS, I \models \varphi$ if $M, I \models \varphi$, where M is the model of IS, and $IS \models \varphi$ if $IS, [s^I, s^I] \models \varphi$.

The problem of model checking ISRL against EHS specification is defined as follows: given an ISRL IS and an EHS specification φ, decide whether $IS \models \varphi$.

3.1 Example

Consider a bit transmission protocol, where an agent *Sender* wants to deliver a single bit message to an agent *Receiver* via a faulty communication channel, modelled using *Environment*. Assume that the Sender first computes the bit (which is modelled as a non-deterministic choice of the bit), and then repeatedly tries to send the bit. Sending the bit requires three actions (not necessarily happening immediately one after the other): opening the communication channel, sending the bit and closing the communication channel. For each $i \in \{0, 1\}$ being the possible value of the bit, we consider two labellings: $\lambda(s_i)$ matching the intervals starting with opening the communication channel, ending with closing the communication channel and sending the bit i during the interval, and $\lambda(r_i)$ matching in every state (point interval) where the bit to be sent is chosen and its value is i.

In this scenario, we can verify whether, for example:

- $[G](K_{Receiver}r_0 \to \langle\bar{L}\rangle s_0)$ – if Receiver knows that the value of the bit is 0, then it means that Sender has sent 0.
- $[G](s_0 \vee s_1) \Rightarrow [O](\neg s_0 \wedge \neg s_1)$ – sending actions do not overlap.
- $[G]\langle L\rangle K_{Receiver}(r_0 \vee r_1)$ – it is always possible that Receiver will eventually know the value of the bit.

where $[G]\psi = [A]\psi \wedge [L]\psi$, interpreted in $[s^I, s^I]$, means "globally". For more examples, please refer to [Lomuscio and Michaliszyn, 2013; Lomuscio and Michaliszyn, 2014] and [Lomuscio and Michaliszyn, 2016].

4 Decidability of Model Checking

The *infinite tree of degree* k is, as usual, a connected structure with a single root such that each node has exactly k (ordered) successors, and every node has a single predecessor except for the root that has no predecessors. The *doubly-infinite tree of degree* k is a connected structure such that each node has exactly k (ordered) successors, and every node has a single predecessor (e.g., the set of the integers with the successor relation is the doubly-infinite tree of degree 1).

To show the decidability, we reduce the model checking problem to the satisfiability problem of SkS$^\infty$ (where k is the degree of the models of a given IS), the Monadic Second-Order Logic on the doubly-infinite tree of degree k. To show that SkS$^\infty$ is decidable, we use the decidability of SkS, the Monadic Second-Order logic on infinite trees of degree k. Below we define SkS$^\infty$ and SkS.

Syntax Terms of SkS are formed out of individual (first-order) variables x, y, z, etc., the empty string ϵ, and right concatenation with $1 \dots k$. Atomic formulas are of the form $w = w'$, $w < w'$ and $w \in X$, where w, w' are terms and X is a (second-order) variable. Formulas are built from atomic formulas using the logical connectives $\wedge$, $\vee$, $\Rightarrow$, $\neg$ and the quantifiers $\exists, \forall$ of both individual and second-order variables. The syntax of SkS^∞ is the same.

Semantics SkS is interpreted over the infinite tree of degree k; individual variables are interpreted as nodes of the tree and second-order variables are interpreted as sets of nodes. The concatenation xi, for $i \in \{1, \dots, k\}$ is interpreted as the ith successor of the interpretation of x. We define a relation $succ(x, y)$ that holds if y is a successor of x; then, the inequality $<$ is interpreted as the transitive closure of $succ$. Inclusion and equality of sets, are definable in SkS in an obvious way. The semantics of SkS^∞ is the same except that it is interpreted over the doubly-infinite tree of degree k. This semantics can be extended to formulas with free variables and labelled trees in the usual manner.

The decidability (of the satisfiability problem) for SkS was shown in [Rabin, 1977]. Here we show that SkS^∞ is also decidable with the same complexity.

Theorem 6. *There are polynomial time reductions from* SkS *to* SkS^∞ *and from* SkS^∞ *to* $\mathrm{S(k+1)S}$.

The reduction from SkS to SkS^∞ is straightforward — we can express that there exists a second-order variable *Root* with exactly one element and *relativise* the formula so that each quantifier ignores the nodes above this element. The other reduction uses the $k + 1$th successor of the root to simulate its predecessor, and then repeats it for the predecessor and so on. Then, another *relativisation* is needed to remove unused $k + 1$th successors. It turns out that the successor relations and the relation $<$ can be expressed in this encoding.

4.1 Proof Overview

For a given ISRL IS, its model M and an EHS formula φ, our goal is to construct an SkS^∞ formula Ψ such that Ψ is satisfiable iff $IS \models \varphi$. Since checking the former is decidable, the decidability of the latter follows.

We will present the construction of the formula Ψ in three steps. Here we give a simplified description of these steps:

Step 1 We construct a formula Φ_{IS} whose models, viewed as unordered trees, have a clearly-marked substructure isomorphic with M without the relations $\sim_i$. For any model of Φ_{IS}, let h be the isomorphism from this substructure to M. Notice that we cannot express the relations $\sim_i$ because SkS^∞ with binary predicates is undecidable.

Step 2 We show that the formulas of EHS without epistemic modalities can be translated to SkS^∞: for every EHS formula φ, we construct an SkS^∞ formula $\Psi(x, y)$ such that $\Phi_{IS} \wedge \Psi(v, v')$ holds for some nodes v, v' if $IS, [h(v), h(v')] \models \varphi$ (this part is the reason for the alternative definition of the semantics, as commented in Section 2).

Step 3 We extend the translation to cover epistemic modalities. This is more technical, so we discuss this using an example. Assume that the upper tree in Figure 2 represents a model of IS and that $s_0 \sim_1 s_4$, $s_1 \sim_1 s_5$, $s_2 \sim_1 s_6$ and $s_3 \sim_1 s_7$. Consider the formula $K_1\varphi$. A naive approach is to try to write an SkS^∞formula with variables x, y that states "for all x', y' such that the paths from x to y and from x' to y' have the same length and their corresponding states are indistinguishable for agent 1, φ (translated to SkS^∞) holds". The logic SkS^∞ is too weak to express this.

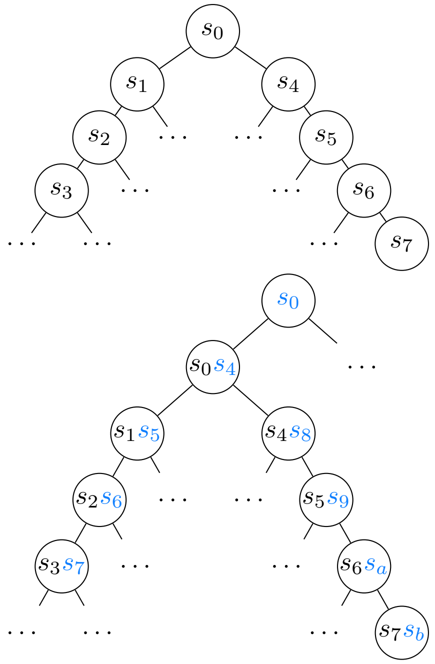

Figure 2: An example of translating epistemic modalities.

To perform Step 3, we actually need a stronger version of the constructions mentioned in Steps 1-2 that allows us to "re-root" the tree. We assume that the formulas are parametrised by a "current root", which defines a subtree isomorphic with the model of IS. Then, to deal with an epistemic modality, we allow to move the root and relabel the whole tree (using fresh second-order variables) such that in the relabelled tree, an indistinguishable interval ($[s_4, s_7]$ in our example) is in the same place as the interval $[x, y,]$ ($[s_0, s_3]$ in our example). The construction guarantees that both intervals have the same length and their relative states are indistinguishable as required. By using universal quantifiers of SkS^∞ we can check all relevant intervals.

Notice that indistinguishable intervals can have very long histories; this is why we use doubly-infinite trees – to make sure that we have room to include the whole history. Alternative approaches to this problem are possible, but we found this solution the most elegant.

4.2 Representing a Model

Fix an ISRL $IS = (\{L_i, l_i^I, ACT_i, P_i, t_i\}_{i \in \mathbf{A}}, \lambda)$ and its model M. Assume that the maximal number of successors of a global state in M is k. Let G and t_G be as above and let $G = \{g_0, \dots, g_n\}$ be an enumeration of all global states. We treat each g_i as a second-order predicate. By G-substructure of a model N we denote the substructure of N

consisting of all the elements satisfying any of g_i. We define an SkS^∞ formula $\mathbf{model}_{IS}$, whose G-substructure, viewed as an unranked tree, is isomorphic with M. The formula $\mathbf{model}_{IS}(r, g_0, \ldots, g_n)$ (where r is an individual variable and g_i are second-order variables) is the conjunction of the following formulas.

- $g_0(r)$ — the root r is the initial state of M.
- $\forall_x(\bigvee_{g \in G} g(x) \wedge x \neq r) \Rightarrow x > r \wedge \forall_y.succ(y, x) \Rightarrow \bigvee_{g \in G} g(y)$ — the G-substructure is a tree rooted in r.
- $\forall_x \bigwedge_{g,g' \in G | g \neq g'} \neg(g(x) \wedge g'(x))$ — no node is labelled by two different gs
- $\forall_{x,y,z}.succ(x, y) \wedge succ(x, z) \wedge y \neq z \Rightarrow \bigwedge_{g \in G}(g(y) \Rightarrow \neg g(z))$ — no node has two successors labelled by the same g.
- $\bigwedge_{(g,g') \notin t_G} \forall_x \forall_y.succ(x, y) \Rightarrow \neg g(x) \vee \neg g'(y)$ and $\bigwedge_{(g,g') \in t_G} \forall_x.g(x) \Rightarrow \exists_y(succ(x, y) \wedge g'(y))$ — the successors in the G-substructure of a state satisfying g are exactly $\{g' \mid t_G(g, g')\}$, i.e., the successors of g.

The construction guarantees the following.

Lemma 7. *Let IS be an ISRL and M be its model of degree k and global states $\{g_0, \ldots, g_n\}$. The* SkS^∞ *formula* $\mathbf{model}_{IS}(r, g_0, \ldots, g_n)$ *is satisfiable, and if T is its model, then the $\{g_0, \ldots, g_n\}$-substructure of T viewed as an unranked tree is isomorphic with M.*

4.3 Translation of Formulas

Hereafter we abbreviate $g_0, \ldots, g_n$ to $\vec{g}$. We inductively define an "EHS to SkS^∞" translation $\mathbf{ets}$.

We start by defining the translation function for p being a propositional letter. This can be done in a standard way — we transform the regular expression $\lambda(p)$ into an automaton, and then write a formula that labels the path from x to y with the states of the automaton, starting from an initial one and ending with a final one.

Conjunction and negation are translated directly. For any EHS formulas φ, φ', we define:

- $\mathbf{ets}_{\neg\varphi}(x, y, r, \vec{g}) = \neg\mathbf{ets}_\varphi(x, y, r, \vec{g})$
- $\mathbf{ets}_{\varphi \wedge \varphi'}(x, y, r, \vec{g}) = \mathbf{ets}_\varphi(x, y, r, \vec{g}) \wedge \mathbf{ets}_{\varphi'}(x, y, r, \vec{g})$

We now proceed to definition of translation function for temporal relations. Define $\mathbf{ets}_{\langle X \rangle\varphi}(x, y, r, \vec{g})$ as:

- $\exists_z y \leq z \wedge \mathbf{ets}_\varphi(y, z, r, \vec{g})$ if $X = A$,
- $\exists_z r \leq z \leq x \wedge \mathbf{ets}_\varphi(z, x, r, \vec{g})$ if $X = \bar{A}$,
- $\exists_z x \leq z < y \wedge \mathbf{ets}_\varphi(x, z, r, \vec{g})$ if $X = B$,
- $\exists_z y < z \wedge \mathbf{ets}_\varphi(x, z, r, \vec{g})$ if $X = \bar{B}$,
- $\exists_z x < z \leq y \wedge \mathbf{ets}_\varphi(z, y, r, \vec{g})$ if $X = E$,
- $\exists_z r \leq z < x \wedge \mathbf{ets}_\varphi(z, y, r, \vec{g})$ if $X = \bar{E}$.

The translation for remaining temporal operators is defined in a similar way. Notice that so far we never modify r and $\vec{g}$, and therefore the construction would work also in SkS.

Now we discuss epistemic modalities (Step 3). For every $i \in \mathbf{A}$ define $\sim_i \subseteq G^2$ as $g \sim_i g'$ iff $l_i(g) = l_i(g')$. Similarly, for every $\Gamma \subseteq \mathbf{A}$ define $\sim_\Gamma \subseteq G^2$ as the transitive closure of $\bigcup_{i \in \Gamma} \sim_i$. Since $\sim_i$ is just $\sim_{\{i\}}$, it is enough to define the transition function for epistemic modalities of the form C_Γ, where $\Gamma \subseteq \mathbf{A}$.

Define $\mathbf{ets}_{C_\Gamma\varphi}(x, y, r, g_0, \ldots, g_n)$ as

$$\forall_{r'} \forall_{\vec{g'}}.\mathbf{equiv}_\Gamma(x, y, r, \vec{g}, r', \vec{g'}) \Rightarrow \mathbf{ets}_\varphi(x, y, r', \vec{g'})$$

where $\mathbf{equiv}_\mathbf{\Gamma}$ states that $r', \vec{g'}$ give a model in which the interval $[x, y]$ is indistinguishable for Γ with the interval $[x, y]$ in the model given by $r, \vec{g}$. Formally, $\mathbf{equiv}_\Gamma(x, y, r, \vec{g}, r', \vec{g'})$ is defined as:

$$\mathbf{model}_{IS}(r', \vec{g'}) \wedge \forall_z.x \leq z \leq y \Rightarrow \bigvee_{i,j | (g_i, g_j) \in \sim_\Gamma} (g_i(z) \wedge g'_j(z))$$

4.4 Correctness of the Construction

We show that the construction is correct.

Lemma 8. *Let IS be an ISRL with the model M and φ be an EHS formula. Let N be a model of* $\mathbf{model}_{IS}(r, \vec{g})$ *and h be the isomorphism from M to $\{g_0, \ldots, g_n\}$-substructure of N. For every interval $[s, t]$ in M we have $M, [s, t] \models \varphi$ iff* $\mathbf{ets}_\varphi(h^{-1}(s), h^{-1}(t), r, \vec{g})$ *holds in N.*

Recall that the isomorphism exists due to Lemma 7. The proof goes by induction and is omitted due to the page limit.

The main result follows:

Theorem 9. *Model checking ISRL against EHS specifications is decidable.*

Proof. For an ISRL IS and an EHS formula φ, the model checking amounts to constructing the formula

$$\exists_r \exists_{\vec{g}}.\mathbf{model}_{IS}(r, \vec{g}) \wedge \mathbf{ets}_\varphi(r, r, r, \vec{g})$$

and checking whether this formula is satisfiable (which is equivalent to being true as it has no free variables). It follows from Lemma 8 that this formula is satisfiable iff $M, [s^I, s^I] \models \varphi$, because $h(s^I) = r$ for any isomorphism h defined as in Lemma 7. □

The resulting complexity is non-elementary. However, SkS and SkS^∞ are elementary when the number of quantifiers alternations is bounded. Therefore, a better complexity can be obtained, if needed, by considering EHS formulas with bounded alternation depth.

5 LTL-like Semantics Leads to Undecidability

In a sense, the way that the intervals are quantified in EHS is similar to CTL: a single formula may depend on different branches of computation. An alternative, LTL-like, approach would be to consider the specification over each possible computation separately. More precisely, for a given ISRL IS and its model M, a *trace* is an infinite path in the tree M starting in the root. By $traces(IS)$ we denote the set of all such traces.

We adjust the definition of the semantics of EHS (Definition 5) to traces in a straightforward way: each time the semantics refers to an interval, it has to be an interval in this particular trace. We define $IS \models_{LTL} \varphi$ iff for all

$$\begin{pmatrix}1s\\(q_0,B)\\a\end{pmatrix}\begin{pmatrix}1\\B\\(q_1,B)\end{pmatrix}\begin{pmatrix}1\\B\\B\end{pmatrix}\begin{pmatrix}1e\\B\\B\end{pmatrix}\begin{pmatrix}2s\\a\\a\end{pmatrix}\begin{pmatrix}2\\b\\(q_1,B)\end{pmatrix}\begin{pmatrix}2\\(q_2,B)\\B\end{pmatrix}\begin{pmatrix}2e\\B\\B\end{pmatrix}\begin{pmatrix}1s\\a\\a\end{pmatrix}\begin{pmatrix}1\\b\\(q_f,b)\end{pmatrix}\begin{pmatrix}1\\(q_2,B)\\c\end{pmatrix}\begin{pmatrix}1e\\B\\B\end{pmatrix}\begin{pmatrix}-\\-\\-\end{pmatrix}\begin{pmatrix}-\\-\\-\end{pmatrix}\dots$$

Figure 3: An example of an interval encoding a computation of a Turing machine (states of agent 3 are omitted for readability).

$T \in traces(IS)$ and all the intervals $[s^I, t]$ (i.e., starting from the initial state) we have $T, [s^I, t] \models \varphi$.

Just like LTL is computationally harder than CTL, the LTL-like semantics makes EHS harder.

Theorem 10. *For an ISRL IS and an EHS formula φ, checking whether $IS \models_{LTL} \varphi$ is undecidable, even if φ contains only epistemic modalities.*

Proof. We reduce the boundedness problem for Turing machines. We assume a Turing machine M with the alphabet Σ including the letter B (blank), set of states Q, a single initial state q_0, and a transition function $\delta : \Sigma \times Q \to Q \times \Sigma \times \{L, R\}$ such that M starts with the empty tape. A machine M is *bounded* iff M visits only finitely many configurations during its computations. It is well known that checking if such a machine is bounded is undecidable.

We construct an ISRL IS and a formula φ such that there is a trace T of IS and t such that $T, (s^I, t) \models \varphi$ iff M is bounded. This implies that $IS \models_{LTL} \neg\varphi$ iff M is not bounded, thus the undecidability.

We use an abbreviation $\widehat{K}_i$ defined, for each agent i, as $\widehat{K}_i\psi = \neg K_i \neg\psi$. Notice that $\widehat{K}_i\psi$ means "there is an interval indistinguishable for i from the current one that satisfies ψ".

We use four agents: 0 with states $\{1_s, 1, 1_e, 2_s, 2, 2_e, -\}$, $1, 2$ with states $S = \Sigma \cup (Q \times \Sigma) \cup \{-\}$, and 3 with a single state $-$. All the agents have a single action a that is allowed in every state and allows to move from any state except $-$ to any state; from $-$ an agent can move only to $-$.

The idea is that the interval that satisfies φ will encode the initial configuration of M, and we will require that for each configuration, there is an interval (somewhere in the trace, not necessary below the current interval) encoding the next configuration. We will also stipulate that there are only finitely many configurations, which means that M is bounded.

First, we define the labelling $\lambda(I) = (1\colon (q_0, B))(1\colon B)^*$ – it labels the intervals where the states of the agent 1 encode the initial configuration. We will also employ $\lambda(a_i)$ defined as $(0\colon i_s)(0\colon i)^*(0\colon i_e)$ for $i \in \{1, 2\}$. We will call intervals matching $\lambda(a_i)$ *i-blocks*; blocks will be used to encode configurations (e.g., Figure 3 contains three blocks). Furthermore, let $\lambda(end)$ be defined by $\top^*(0\colon - \wedge 1\colon - \wedge 2\colon -)\top^*$.

So far, we can express the following: the interval encodes the initial configuration in the states of agent 1 and there are only finitely many configurations: $a_1 \wedge I \wedge \widehat{K}_3 end$.

It remains to express that if there is an interval expressing a configuration, then there is an interval expressing the next configuration. To do this, we need two more labellings.

Let $\bar{1}$=2 and $\bar{2}$=1. For $i \in \{1, 2\}$, let the labelling $\lambda(next_i)$ match the intervals matched by a_i where the states of agent 1 and the states of agent 2 describe a configuration (contain exactly one pair (q, a) in each case) and the configuration described by the states of $\bar{i}$ is the configuration following the configuration described by the states of i. This can be expressed as a regular expression in standard way.

Let $\psi_i = a_i \Rightarrow \widehat{K}_{\bar{i}}(a_{\bar{i}} \wedge next_{\bar{i}})$ be a formula for every i-block there is an $\bar{i}$-block I' such that the configuration stored by agent $\bar{i}$ is the same in both cases. Moreover, because of $next_{\bar{i}}$, the states of i in I' encode the next configuration.

Finally, we define $\varphi = a_1 \wedge I \wedge \widehat{K}_3 end \wedge K_3(\psi_1 \wedge \psi_2)$.

If M visits only a bounded number of configurations, then we can easily construct a trace satisfying φ that contains all these configurations and then all the states are triples $(-, -, -)$. On the other hand, if there are a trace T and an interval I that satisfy φ, then T contains finitely many configurations including the initial one. By ψ_1 and ψ_2, T contains all the configurations that M visits, meaning that M visits only a bounded number of configurations. □

A similar question: does $T, [s^I, s^I] \models \varphi$ for all $T \in traces(IS)$ (as in the definition of $IS \models \varphi$), is also undecidable, but only if we also have at least one modality among $\{A, \bar{A}, \bar{B}, \bar{D}, \bar{E}, L, \bar{L}\}$. Otherwise, the model checking is decidable, as only point intervals are reachable from the initial interval. Interestingly, LTL-like model checking EHS without epistemic modalities is decidable [Bozzelli *et al.*, 2016b].

6 Conclusions

We showed that the model checking problem for EHS is decidable in non-elementary time. The only known lower bound is EXPSPACE. The problem of establishing the precise complexity, however, remains open and is hard to predict. On the one hand, the non-emptiness problem of regular expressions with complementation is TOWER-complete, and EHS can express regular expressions and negation; however, the proof of the non-elementary lower bound for regular expressions with complementation alternates complementation and Kleene star, and it does not seem to be possible to do this in EHS. On the other hand, even for a simpler logic, HS, no elementary algorithm is known.

We also showed that LTL-like semantics combined with epistemic modalities on intervals lead to undecidability. The undecidability of the satisfiability problem for EHS without temporal modalities can be shown in the same way.

Acknowledgements

The first author was supported by Polish National Science Centre grant 2014/15/D/ST6/00719. The second author was supported by Polish National Science Centre grant 2016/21/B/ST6/01444.

References

[Bozzelli *et al.*, 2016a] Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Interval temporal logic model checking: The border between good and bad HS fragments. In *Automated Reasoning - 8th International Joint Conference, IJCAR 2016, Proceedings*, volume 9706 of *Lecture Notes in Computer Science*. Springer, 2016.

[Bozzelli *et al.*, 2016b] Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Interval vs. point temporal logic model checking: an expressiveness comparison. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016*, pages 26:1–26:14, 2016.

[Bozzelli *et al.*, 2017] Laura Bozzelli, Alberto Molinari, Angelo Montanari, and Adriano Peron. An in-depth investigation of interval temporal logic model checking with regular expressions. In *Software Engineering and Formal Methods - 15th International Conference, SEFM 2017*, pages 104–119, 2017.

[Bozzelli *et al.*, 2018a] Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Model checking for fragments of the interval temporal logic HS at the low levels of the polynomial time hierarchy. *Inf. Comput.*, 262(Part):241–264, 2018.

[Bozzelli *et al.*, 2018b] Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Which fragments of the interval temporal logic HS are tractable in model checking? *Theoretical Computer Science*, 2018.

[Fischer and Ladner, 1979] Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.*, 18(2):194–211, 1979.

[Goranko *et al.*, 2004] Valentin Goranko, Angelo Montanari, and Guido Sciavicco. A road map of interval temporal logics and duration calculi. *Journal of Applied Non-Classical Logics*, 14(1-2):9–54, 2004.

[Halpern and Shoham, 1991] J.Y. Halpern and Y. Shoham. A propositional modal logic of time intervals. *Journal of The ACM*, 38:935–962, 1991.

[Lange, 2006] Martin Lange. Model checking propositional dynamic logic with all extras. *J. Applied Logic*, 4(1):39–49, 2006.

[Lomuscio and Michaliszyn, 2013] Alessio Lomuscio and Jakub Michaliszyn. An epistemic Halpern-Shoham logic. In *IJCAI 13*, pages 1010–1016, 2013.

[Lomuscio and Michaliszyn, 2014] Alessio Lomuscio and Jakub Michaliszyn. Decidability of model checking multi-agent systems against a class of EHS specifications. In *Proceedings of the Twenty-first European Conference on Artificial Intelligence*, pages 543–548. IOS Press, 2014.

[Lomuscio and Michaliszyn, 2016] Alessio Lomuscio and Jakub Michaliszyn. Model checking multi-agent systems against epistemic HS specifications with regular expressions. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016*, pages 298–308. AAAI Press, 2016.

[Molinari *et al.*, 2015] Alberto Molinari, Angelo Montanari, and Adriano Peron. Complexity of ITL model checking: Some well-behaved fragments of the interval logic HS. In *22nd International Symposium on Temporal Representation and Reasoning, TIME 2015*, pages 90–100. IEEE Computer Society, 2015.

[Molinari *et al.*, 2016] Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Model checking well-behaved fragments of HS: the (almost) final picture. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR.*, pages 473–483, 2016.

[Molinari *et al.*, 2018] Alberto Molinari, Angelo Montanari, and Adriano Peron. Model checking for fragments of halpern and shoham's interval temporal logic based on track representatives. *Inf. Comput.*, 259(3):412–443, 2018.

[Molinari, 2019] Alberto Molinari. Model checking: the interval way. *PHD Thesis, CoRR*, abs/1901.03880, 2019.

[Montanari *et al.*, 2014] Angelo Montanari, Aniello Murano, Giuseppe Perelli, and Adriano Peron. Checking interval properties of computations. In *Temporal Representation and Reasoning (TIME), 2014 21st International Symposium on*, pages 59–68. IEEE, 2014.

[Moszkowski, 1983] B. C. Moszkowski. *Reasoning about digital circuits*. PhD thesis, Stanford University, Stanford, CA, USA, 1983.

[Rabin, 1977] Michael O. Rabin. Decidable theories. In Jon Barwise, editor, *HANDBOOK OF MATHEMATICAL LOGIC*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, pages 595 – 629. Elsevier, 1977.

[van Benthem *et al.*, 2006] Johan van Benthem, Jan van Eijck, and Barteld P. Kooi. Logics of communication and change. *Inf. Comput.*, 204(11):1620–1662, 2006.