Computing Minimal Distinguishing Hennessy-Milner Formulas is NP-Hard, but Variants are Tractable

Jan Martens
Eindhoven University of Technology, The Netherlands

Jan Friso Groote
Eindhoven University of Technology, The Netherlands

Abstract

We study the problem of computing minimal distinguishing formulas for non-bisimilar states in finite LTSs. We show that this is NP-hard if the size of the formula must be minimal. Similarly, the existence of a short distinguishing trace is NP-complete. However, we can provide polynomial algorithms, if minimality is formulated as the minimal number of nested modalities, and it can even be extended by recursively requiring a minimal number of nested negations. A prototype implementation shows that the generated formulas are much smaller than those generated by the method introduced by Cleaveland.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics

Keywords and phrases Distinguishing behaviour, Hennessy-Milner logic, NP-hardness

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2023.32

Related Version *Full Version*: `https://arxiv.org/abs/2307.05265`

Supplementary Material *Software*: `https://github.com/jjmartens/distinguishing-hml`
archived at `swh:1:dir:c38076d88d2e9cc0bf081739203a2474ba87b7d3`

Funding AVVA project NWO 612.001.751/TOP1.17.002.

1 Introduction

Hennessy-Milner Logic (HML) [11] can be used to explain behavioural inequivalence. If two states are not bisimilar there is a *distinguishing formula* that is valid in one state but not in the other. As the reason for the states not being bisimilar can be very subtle, such a distinguishing formula is of great help to pinpoint the cause of the inequivalence.

Cleaveland [6] introduces an efficient algorithm to calculate distinguishing formulas by back-tracking the partition refinement sequence that decides bisimilarity. He states that the formulas are minimal "in a precisely defined sense". This method is used in the mCRL2 toolset [5]. However, the generated formulas are unexpectedly large. This leads to the question in which sense distinguishing formulas are minimal and how difficult it is to obtain them. Similar questions were posed throughout the literature. Some also questioned the size of the formulas – in the setting of CTL [4], others explicitly stated that they were not minimal [25, 3], and there are even suggestions that minimisation could be NP-hard [26].

In this work we answer the question by proving that in general calculating minimal distinguishing Hennessy-Milner formulas is NP-hard. Minimality can be taken rather broadly, as having a minimal number of symbols, modalities, or logical connectives. As observed in [8] a distinguishing formula can be exponential in size. However, as was already noted in [6], when using sharing in the representation of formulas, for instance by formulating the distinguishing formula as a set of equations or a directed acyclic graph, the representation is polynomial. Calculating a minimal shared distinguishing formula is NP-complete.

34th International Conference on Concurrency Theory (CONCUR 2023).
Editors: Guillermo A. Pérez and Jean-François Raskin; Article No. 32; pp. 32:1–32:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The proof of this result uses a reduction directly from CNF-SAT and the construction is similar to the construction used by Hunt [13] where it is shown that deciding equivalence of acyclic non-deterministic automata is NP-complete. We show via the NP-hardness of deciding whether there is a distinguishing trace for an acyclic non-deterministic LTS, that computing minimal HML formulas is also NP-hard.

As distinguishing formulas are very useful, we are wondering whether a variant of minimality of distinguishing formulas exists that leads to concise formulas and that can effectively be calculated. We answer this positively by providing efficient algorithms to construct distinguishing formulas that are minimal with respect to the *observation-depth*, i.e., the number of nested modalities. Within this we can even guarantee in polynomial time that the *negation-depth*, i.e., the number of nested negations, or equivalently the number of nested alternations of box and diamond modalities, is minimal. These algorithms strictly improve upon the method by Cleaveland [6]. A prototype implementation of our algorithm shows that our formulas are indeed much smaller and more pleasant to use. In order to obtain these results we employ the notions of k-bisimilarity [19] and m-nested similarity [10].

Distinguishing formulas have been the topic of studies in many papers, more than we can mention. A recent impressive work introduces a method to find minimal distinguishing formulas for various classes of behavioural equivalences [3]. The algorithm translates the problem to determining the winning region in a reachability game. These games can grow super-exponentially in size. In the context of distinguishing deterministic finite automata, an algorithm is given that from a splitting tree finds pairwise minimal distinguishing words [23]. In a more generalized setting [25, 15] a co-algebraic method is given to generate distinguishing modal formulas. The notion of distinguishing formulas is also used in the setting with abstractions for branching bisimilarity [16, 9].

This document is structured as follows. In Section 2 the required preliminaries on LTSs and HML formulas are given. In Section 3, we show that decision problems related to finding minimal distinguishing formulas are NP-hard. Next, in Section 4 we give a procedure that generates a minimal observation- and negation-depth formula. Additionally, in this section, we give a partition refinement algorithm inspired by [23, 20] which can be used to determine minimal observation-depth distinguishing formulas. In the full version, an appendix is included containing proofs omitted here due to space constraints.

2 Preliminaries

For the numbers $i, j \in \mathbb{N}$, we define $[i, j] = \{c \in \mathbb{N} \mid i \leqslant c \leqslant j\}$, the closed interval from i to j.

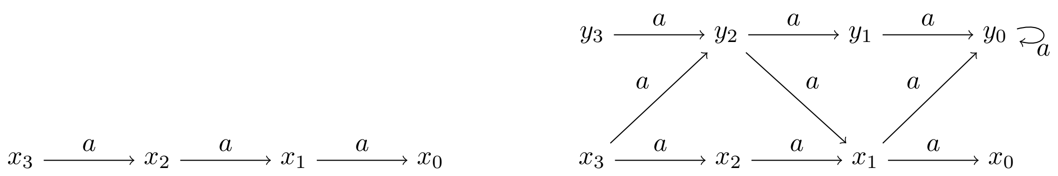
2.1 LTSs, k-bisimilarity & m-nested similarity

We use Labelled Transition Systems (LTSs) as our behavioural models. Strong bisimilarity is a widely used behavioural equivalence [19, 22], which we define in the classical inductive way.

▶ **Definition 1.** *A labelled transition system (LTS) $L = (S, Act, \rightarrow)$ is a three-tuple containing:*

- *a finite set of states S,*
- *a finite set of action labels Act, and*
- *a transition relation $\rightarrow \subseteq S \times Act \times S$.*

We write $s \xrightarrow{a} s'$ iff $(s, a, s') \in \rightarrow$. We call s' an a-derivative of s iff $s \xrightarrow{a} s'$.



(a) The LTS $\mathcal{A}_3 = (\{x_0, \ldots, x_3\}, \{a\}, \rightarrow)$.

(b) The LTS $\mathcal{B}_3 = (\{x_i, y_i \mid 0 \leqslant i \leqslant 3\}, \{a\}, \rightarrow)$.

Figure 1 Two example LTSs.

▶ **Definition 2** (k-bisimilar [19]). *Let $L = (S, Act, \rightarrow)$ be an LTS. For every $k \in \mathbb{N}$, k-bisimilarity written as $\underline{\leftrightarrow}_k$ is defined inductively:*

$$\begin{aligned} \underline{\leftrightarrow}_0 &= \{(s,t) \mid s, t \in S\}, \text{ and} \\ \underline{\leftrightarrow}_{k+1} &= \{(s,t) \mid \forall s \xrightarrow{a} s'.\exists t \xrightarrow{a} t' \text{ such that } s' \underline{\leftrightarrow}_k t', \text{ and} \\ &\qquad \forall t \xrightarrow{a} t'.\exists s \xrightarrow{a} s' \text{ such that } t' \underline{\leftrightarrow}_k s'\}. \end{aligned}$$

Bisimilarity, denoted as $\underline{\leftrightarrow}$, is defined as the intersection of all k-bisimilarity relations for all $k \in \mathbb{N}$: $\underline{\leftrightarrow} = \bigcap_{k \in \mathbb{N}} \underline{\leftrightarrow}_k$. As our transition systems are finite, and therefore finitely branching, $\underline{\leftrightarrow}$ coincides with the more general co-inductive definition of bisimulation [22]. The intuition behind $\underline{\leftrightarrow}_i$ is that within i (atomic) observations there is no distinguishing behaviour. We sketch a rather simple example that showcases this behaviour.

▶ **Example 3.** For every $n \in \mathbb{N}$, we define the LTS $\mathcal{A}_n = (S, \{a\}, \rightarrow)$ with a singleton action set, and the set of states $S = \{x_0, \ldots, x_n\}$. The transition function contains a single path $x_i \xrightarrow{a} x_{i-1}$ for all $1 \leqslant i \leqslant n$.

In Figure 1a the LTS $\mathcal{A}_3$ is shown. A state x_i can perform i a-transitions ending in a deadlock state. All states in $\mathcal{A}_3$ are behaviourally inequivalent. Intuitively, we see that distinguishing the states x_3 and x_2 takes at least 3 observations.

In general, it holds that for $n \in \mathbb{N}$, the states x_n and x_{n-1} of the LTS $\mathcal{A}_n$ are $n-1$-bisimilar but not n-bisimilar, i.e. $x_n \underline{\leftrightarrow}_{n-1} x_{n-1}$ but $x_n \not\underline{\leftrightarrow}_n x_{n-1}$. In order to distinguish these states we require n (atomic) observations. This intuition is formalized in Theorem 10.

▶ **Fact 4.** *We state these well-known facts for an LTS $L = (S, Act, \rightarrow)$, and $k \in \mathbb{N}$:*

1. *The relation $\underline{\leftrightarrow}_k$ is an equivalence relation.*
2. *If two states are k-bisimilar, they are l-bisimilar for every $l \leqslant k$.*
3. *If $\underline{\leftrightarrow}_k = \underline{\leftrightarrow}_{k+1}$ then $\underline{\leftrightarrow}_k = \underline{\leftrightarrow}_{k+u} = \underline{\leftrightarrow}$, for all $u \in \mathbb{N}$.*

For technical reasons we also define m-nested similarity [10] which uses the concept of similarity.

▶ **Definition 5** (Similarity). *Given an $L = (S, Act, \rightarrow)$, we define similarity $\underline{\rightarrow} \subseteq S \times S$ as the largest relation such that if $s \,\underline{\rightarrow}\, t$ then for all transitions $s \xrightarrow{a} s'$ there is a $t \xrightarrow{a} t'$ such that $s' \,\underline{\rightarrow}\, t'$.*

We say a state s is *simulated* by t iff $s \,\underline{\rightarrow}\, t$.

▶ **Definition 6** (cf. Def. 8.5.2. [10]). *Let $L = (S, Act, \rightarrow)$ be an LTS, and $m \in \mathbb{N}$ a number. We inductively define m-nested similarity inclusion as follows: $\underline{\rightarrow}^0 = \underline{\rightarrow}$, and for every $i \in \mathbb{N}$, the relation $\underline{\rightarrow}^{i+1} \subseteq S \times S$ is the largest relation such that for all $(s,t) \in \underline{\rightarrow}^{i+1}$ it holds that:*

- *$s \,\underline{\rightarrow}^i\, t$ and $t \,\underline{\rightarrow}^i\, s$, and*
- *if $s \xrightarrow{a} s'$ then there is a $t \xrightarrow{a} t'$ such that $s' \,\underline{\rightarrow}^{i+1}\, t'$.*

We write $\rightleftarrows^m$ as the symmetric closure of m-nested similarity inclusion, i.e. $\rightleftarrows^m = \sqsubseteq^m \cap (\sqsubseteq^m)^{-1}$, which we call *m-nested similarity*. Note that we deviate slightly from the definition in [10], where 1-nested simulation equivalence coincides with simulation equivalence.

▶ **Example 7.** For every $n \in \mathbb{N}$, we define the LTS $\mathcal{B}_n = (S, \{a\}, \rightarrow)$ with a singleton action set, the set of states $S = \{x_0, \ldots, x_n, y_0, \ldots, y_n\}$, and the transition relation containing the transition $y_0 \xrightarrow{a} y_0$ and, for every $i \in [1, n]$, the transitions:

- $y_i \xrightarrow{a} y_{i-1}$ and $x_i \xrightarrow{a} x_{i-1}$, and
- $y_i \xrightarrow{a} x_{i-1}$ if i is even, or $x_i \xrightarrow{a} y_{i-1}$ if i is odd.

In Figure 1b the LTS $\mathcal{B}_3$ is shown. We observe that x_0 is simulated by y_0, since x_0 has no outgoing transitions. So it is the case that $x_0 \sqsubseteq^0 y_0$, but $y_0 \not\sqsubseteq^0 x_0$, and hence $x_0 \not\rightleftarrows^0 y_0$. In general, for all $n \geq 1$ it holds in the LTS $\mathcal{B}_n$ that $x_n \rightleftarrows^{n-1} y_n$, but $x_n \not\rightleftarrows^n y_n$.

2.2 Hennessy-Milner logic (HML)

We use Hennessy-Milner Logic (HML) [11] to distinguish states. For some finite set of actions *Act*, the syntax of HML is defined as

$$\phi ::= tt \mid \langle a \rangle \phi \mid \neg \phi \mid \phi \wedge \phi,$$

where $a \in Act$. The logic consists of three necessary elements:

- *Observations* $\langle a \rangle \phi$, the state witnesses an observation a to a state that satisfies ϕ.
- *Negations* $\neg \phi$, the state does not satisfy ϕ.
- *Conjunctions* $\phi_1 \wedge \phi_2$, the state satisfies both ϕ_1 and ϕ_2.

The set $\mathcal{F}$ is defined to contain all HML formulas. It is common to use the abbreviations $ff = \neg tt$, $[a]\phi = \neg \langle a \rangle \neg \phi$ and $\phi_1 \vee \phi_2 = \neg(\neg \phi_1 \wedge \neg \phi_2)$.

Given an LTS $L = (S, Act, \rightarrow)$, we define the semantics of this logic $[\![-]\!]_L : \mathcal{F} \rightarrow 2^S$, inductively as follows:

$$\begin{aligned} [\![tt]\!]_L &= S, \\ [\![\langle a \rangle \phi]\!]_L &= \{s \in S \mid \exists s' \in S \text{ s.t. } s \xrightarrow{a} s' \text{ and } s' \in [\![\phi]\!]_L\}, \\ [\![\neg \phi]\!]_L &= S \setminus [\![\phi]\!]_L, \text{ and} \\ [\![\phi_1 \wedge \phi_2]\!]_L &= [\![\phi_1]\!]_L \cap [\![\phi_2]\!]_L, \end{aligned}$$

for $a \in Act$ and $\phi, \phi_1, \phi_2 \in \mathcal{F}$. This function yields for a formula $\phi \in \mathcal{F}$ the subset of S where ϕ is true. Often we omit the reference to the LTS L when it is clear from the context.

We use HML formulas to describe distinguishing behaviour. Let $L = (S, Act, \rightarrow)$ be an LTS, $s \in S$ and $t \in S$ states, and $\phi \in \mathcal{F}$ a HML formula. We write $s \sim_\phi t$ iff $s \in [\![\phi]\!] \Leftrightarrow t \in [\![\phi]\!]$, and conversely $s \not\sim_\phi t$ iff $s \in [\![\phi]\!] \Leftrightarrow t \notin [\![\phi]\!]$. Additionally, we write $s \leqslant_\phi t$ if $s \in [\![\phi]\!] \Rightarrow t \in [\![\phi]\!]$. Given a set of HML formulas $\mathcal{G}$ we write $s \sim_\mathcal{G} t$ iff for every $\psi \in \mathcal{G}$, it holds that $s \sim_\psi t$. Similarly, we write $s \leqslant_\mathcal{G} t$ iff $s \leqslant_\psi t$ for all $\psi \in \mathcal{G}$.

▶ **Definition 8.** *Given an LTS $L = (S, Act, \rightarrow)$ and two states $s, t \in S$, then a formula $\phi \in \mathcal{F}$ distinguishes s and t iff $s \not\sim_\phi t$.*

2.2.1 Metrics

To express the size of a formula we use three different metrics:

- *size* the total number of observations,
- *observation-depth* the largest number of nested observation in the formula, and

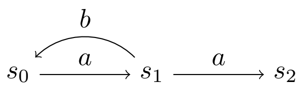


Figure 2 The LTS $M = (S, Act, \rightarrow)$, $Act = \{a, b\}$ and $S = \{s_0, s_1, s_2\}$.

- *negation-depth* the largest number of nested negations in the formula.

For these metrics we inductively define the functions $|\cdot| : \mathcal{F} \rightarrow \mathbb{N}$ for size, $d_\diamond : \mathcal{F} \rightarrow \mathbb{N}$ for observation-depth and $d_\neg : \mathcal{F} \rightarrow \mathbb{N}$ for negation-depth, as follows:

$$
\begin{array}{llllll}
|tt| & = 0, & d_\diamond(tt) & = 0, & d_\neg(tt) & = 0,\\
|\langle a\rangle\phi| & = |\phi|+1, & d_\diamond(\langle a\rangle\phi) & = d_\diamond(\phi)+1, & d_\neg(\langle a\rangle\phi) & = d_\neg(\phi),\\
|\neg\phi| & = |\phi|, & d_\diamond(\neg\phi) & = d_\diamond(\phi), & d_\neg(\neg\phi) & = d_\neg(\phi)+1,\\
|\phi_1\wedge\phi_2| & = |\phi_1|+|\phi_2|. & d_\diamond(\phi_1\wedge\phi_2) & = \max(d_\diamond(\phi_1), d_\diamond(\phi_2)). & d_\neg(\phi_1\wedge\phi_2) & = \max(d_\neg(\phi_1), d_\neg(\phi_2)).
\end{array}
$$

Given natural numbers $n, m \in \mathbb{N}$ we define the sets $\mathcal{F}_n$ and $\mathcal{F}^m$ as the fragment of HML formulas with bounded observation- and respectively negation-depth, i.e. $\mathcal{F}_n = \{\phi \mid d_\diamond(\phi) \leqslant n\}$, and $\mathcal{F}^m = \{\phi \mid d_\neg(\phi) \leqslant m\}$.

We write $\mathcal{F}_n^m$ for the set $\mathcal{F}_n^m = \mathcal{F}_n \cap \mathcal{F}^m$. Based on these metrics we define multiple notions of *minimal* distinguishing formulas.

▶ **Definition 9.** *Given an LTS $L = (S, Act, \rightarrow)$, let $\phi \in \mathcal{F}$ be an HML formula that distinguishes $s \in S$ and $t \in S$. Then in distinguishing s and t, the formula ϕ is called:*

- *to have* minimal observation-depth *iff ϕ has the least nested modalities, i.e. for all $\phi' \in \mathcal{F}$ if $s \not\sim_{\phi'} t$ then $d_\diamond(\phi) \leqslant d_\diamond(\phi')$;*
- *to have* minimal negation-depth *iff ϕ has the least nested negations, i.e., for all $\phi' \in \mathcal{F}$ if $s \not\sim_{\phi'} t$ then $d_\neg(\phi) \leqslant d_\neg(\phi')$;*
- *to be* minimal *iff ϕ has the least number of modalities, i.e., for all $\phi' \in \mathcal{F}$ if $s \not\sim_{\phi'} t$ then $|\phi| \leqslant |\phi'|$;*
- *to have* minimal observation- and negation-depth *iff it is minimal in the lexicographical order of observation and negation-depth, i.e., iff for all $\phi' \in \mathcal{F}$ if $s \not\sim_{\phi'} t$ then $d_\diamond(\phi) \leqslant d_\diamond(\phi')$ and if $d_\diamond(\phi) = d_\diamond(\phi')$ then $d_\neg(\phi) \leqslant d_\neg(\phi')$;*
- irreducible *[6, Def. 2.5] iff no ϕ' obtained by replacing a non-trivial subformula of ϕ with the formula tt distinguishes s from t.*

The first three notions correspond directly to the metrics we defined. The notion of *irreducible* distinguishing formulas corresponds to the minimality notion used in the work by Cleaveland [6]. The different notions are not comparable. This is witnessed by the LTS M pictured in Figure 2. The formula $\phi_1 = \langle a\rangle\langle a\rangle tt$ distinguishes s_0 and s_1 since $s_0 \in [\![\phi_1]\!]$ and $s_1 \notin [\![\phi_1]\!]$. Additionally, ϕ_1 is irreducible, since any formula obtained by replacing a subformula by tt is not a distinguishing formula. However, the formula ϕ_1 is not *minimal* since the formula $\phi_2 = \langle b\rangle tt$ also distinguishes s_0 and s_1.

2.2.2 Representation

A note has to be made on the representation of distinguishing formulas. It is known that distinguishing formulas can grow very large. In fact there is a family of LTSs that showcases an exponential lower bound on the size of the minimal distinguishing formula [8, 25]. This exponential lower bound is not in contradiction with the polynomial-time algorithm from Cleaveland [6] since [6] uses equations to represent the subformulas. For example the formula $\langle a\rangle\langle b\rangle\langle c\rangle tt \wedge \langle b\rangle\langle c\rangle tt$ can be represented using the equations $\phi_1 = \langle a\rangle\phi_2 \wedge \phi_2$ and $\phi_2 = \langle b\rangle\langle c\rangle tt$, or as the term in Figure 3.

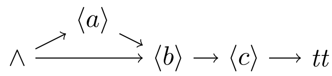


Figure 3 A HML formula represented as a shared term.

The shared representation does not change the observation-depth and the negation-depth. The size of a formula is influenced, but it does not affect the NP-hardness result.

2.2.3 Correspondences

There are strong correspondences between different fragments of HML on the one hand and m-nested similarity and bisimilarity on the other hand. We use these to obtain minimal distinguishing formulas. The first theorem states that those HML formulas that have at most k-nested observations exactly capture k-bisimilarity.

▶ **Theorem 10** ((cf. [11, Theorem 2.2])). *Given an LTS $L = (S, Act, \rightarrow)$ and two states $s, t \in S$. For every $k \in \mathbb{N}$,*

$$s \leftrightarrow_k t \iff s \sim_{\mathcal{F}_k} t.$$

In this work we are mainly interested in the contraposition of this theorem. For every $k \in \mathbb{N}$, two states $s, t \in S$ are not k-bisimilar iff there is a $\phi \in \mathcal{F}_k$ that distinguishes s and t, i.e. $s \not\sim_\phi t$. For this reason for every $k \in \mathbb{N}$ we call s and t *k-distinguishable* iff $s \not\leftrightarrow_k t$. We call the states s and t *distinguishable* iff they are k-distinguishable for some $k \in \mathbb{N}$.

▶ **Corollary 11.** *Given an LTS $L = (S, Act, \rightarrow)$ and two states $s, t \in S$. For every $k \in \mathbb{N}$,*

$$s \not\leftrightarrow_k t \iff \text{there is a formula } \phi \in \mathcal{F}_k \text{ such that } s \not\sim_\phi t.$$

In [10] it is shown that fragments of HML with bounded negation-depth allow a similar relational classification. The following theorem relates the fragment $\mathcal{F}^m$ to m-nested similarity inclusion.

▶ **Theorem 12** ((cf. [10, Corollary 8.7.6])). *Let $L = (S, Act, \rightarrow)$ be an LTS, then for all $m \in \mathbb{N}$, and states $s, t \in S$:*

$$s \sqsupseteq^m t \iff s \leqslant_{\mathcal{F}^m} t.$$

The main use for our work is that if two states are not m-nested similar, then there is a distinguishing formula with at most m nested negations.

▶ **Corollary 13.** *Let $L = (S, Act, \rightarrow)$ be an LTS, then for all $m \in \mathbb{N}$, and states $s, t \in S$:*

$$s \not\rightleftarrows^m t \iff \text{there is a formula } \phi \in \mathcal{F}^m \text{ s.t. } s \in [\![\phi]\!] \text{ and } t \notin [\![\phi]\!].$$

Let us recall the LTS $\mathcal{A}_3$ from Example 3 drawn in Figure 1a. In this LTS we see that $x_3 \leftrightarrow_2 x_2$, but $x_3 \not\leftrightarrow_3 x_2$. As a result of Corollary 11 we know that there is a formula $\phi \in \mathcal{F}_3$ that distinguishes x_3 and x_2. This is witnessed by the formula $\phi = \langle a\rangle\langle a\rangle\langle a\rangle tt \in \mathcal{F}_3$, which is a distinguishing formula, since $x_3 \in [\![\phi]\!]$ and $x_2 \notin [\![\phi]\!]$. We also see that $x_3 \sim_{\mathcal{F}_2} x_2$, hence there is no such formula in $\mathcal{F}_2$.

For the LTS $\mathcal{B}_3$ from Example 7, we aim to distinguish the states x_3 and y_3. According Corollary 13 there is a distinguishing formula $\phi \in \mathcal{F}^3$, since $x_3 \not\rightleftarrows^3 y_3$. This is witnessed by the formula $\phi = \langle a\rangle\neg\langle a\rangle\neg\langle a\rangle\neg\langle a\rangle tt$. This is a distinguishing formula as $x_3 \in [\![\phi]\!]$ and $y_3 \notin [\![\phi]\!]$. Corollary 13 also shows that this is the minimal negation-depth formula distinguishing x_3 and y_3, as $x_3 \rightleftarrows^2 y_3$.

2.2.4 Traces

Let Act be a finite set of action labels. We denote by $Act^* := \bigcup_{i \in \mathbb{N}} Act^i$ the set of all finite sequences on the action labels Act. We write ε for the empty sequence. For sequences $w, u \in Act^*$, we denote with $|w|$ its length and $w \cdot u$ the concatenation of w and u, which is sometimes also written as wu.

▶ **Definition 14.** *Given an LTS $L = (S, Act, \rightarrow)$. The set of* traces *$Tr(s) \subseteq Act^*$ of a state $s \in S$ is the smallest set satisfying:*
1. *$\varepsilon \in Tr(s)$, and*
2. *for an action $a \in Act$, and state $s' \in S$ if a trace $w \in Tr(s')$ and $s \xrightarrow{a} s'$, then $aw \in Tr(s)$.*

Inductively, we define the formula ϕ_w for every word $w \in Act^*$, such that $\phi_\varepsilon = tt$, and $\phi_{aw} = \langle a \rangle \phi_w$. We call a formula $\phi \in \mathcal{F}$ a *trace-formula* iff there is a sequence $w \in Act^*$ such that $\phi = \phi_w$.

▶ **Lemma 15.** *Let $L = (S, Act, \rightarrow)$ be an LTS, and $w \in Act^*$ a trace. Then for all $s \in S$:*

$$s \in [\![\phi_w]\!] \iff w \in Tr(s).$$

Two states $s \in S$ and $t \in S$ in an LTS $L = (S, Act, \rightarrow)$ are said to be trace-equivalent iff $Tr(s) = Tr(t)$. Bisimilarity is a more fine-grained equivalence than trace equivalence. Two states $s \in S$ and $t \in S$ can be trace-equivalent, while not being bisimilar. In this case there is a formula $\phi \in F$ such that $s \not\sim_\phi t$ and we know that ϕ is not a trace-formula. However, ϕ contains traces that are both traces of s and t. To make this more precise we define the traces of a formula by induction for formulas $\phi, \phi_1, \phi_2 \in \mathcal{F}$ as follows:

$$\begin{aligned} Tr(tt) &= \{\varepsilon\}, \\ Tr(\langle a \rangle \phi) &= \{a\} \cup \{a \cdot w \mid w \in Tr(\phi)\}, \\ Tr(\neg \phi) &= Tr(\phi), \\ Tr(\phi_1 \wedge \phi_2) &= Tr(\phi_1) \cup Tr(\phi_2). \end{aligned}$$

The traces of a formula allow us to state the correspondence between k-distinguishability and the length of shared traces. We formulate this using the minimal observation depth that, given two distinguishable states, yields the smallest $i \in \mathbb{N}$ such that the states are i-distinguishable:

▶ **Definition 16.** *Let $L = (S, Act, \rightarrow)$ be an LTS. We define the minimal observation depth $\Delta : S \times S \rightarrow \mathbb{N} \cup \{\infty\}$ by*

$$\Delta(s,t) = \begin{cases} i & \text{if } s \not\leftrightarrow_i t, \text{ and } s \leftrightarrow_{i-1} t, \\ \infty & \text{if } s \leftrightarrow t. \end{cases}$$

The next lemma says that if states have minimal observation depth i, then any distinguishing formula contains a trace of length at least i.

▶ **Lemma 17.** *Let $L = (S, Act, \rightarrow)$ be an LTS and $s, t \in S$ two distinguishable states such that $\Delta(s,t) = i$ for some $i \in \mathbb{N}$. For all $\phi \in \mathcal{F}$, if $s \not\sim_\phi t$ then there is a trace $w \in Tr(\phi)$ such that $|w| \geq i$ and $w \in Tr(s) \cup Tr(t)$.*

Proof sketch. Proven by induction on the shape of ϕ. The only interesting case is if $\phi = \langle a \rangle \phi'$ for some $a \in Act$ and $\phi' \in \mathcal{F}$. Assume without loss of generality that $s \in [\![\phi]\!]$ and $t \notin [\![\phi]\!]$. This means that there is a transition $s \xrightarrow{a} s'$ such that $s' \in [\![\phi']\!]$. Since $\Delta(s,t) = i$ there is also a $t \xrightarrow{a} t'$ such that $\Delta(s', t') = i - 1$.

Since $t \notin [\![\phi]\!]$ also $t' \notin [\![\phi']\!]$, and thus we can apply our induction hypothesis to conclude that there is a $w' \in Tr(\phi')$ such that $|w'| \geq i-1$ and $w' \in Tr(s') \cup Tr(t')$. From w' we construct aw' and observe that $aw' \in Tr(\phi)$, $aw' \in Tr(s) \cup Tr(t)$ and $|aw'| \geq i$, which finishes the proof. ◀

3 NP-hardness results

In this section we show that finding minimal distinguishing formulas is NP-hard.

We first show that the existence of a short trace is NP-complete similar to a result of Hunt [13, Sec. 2.2] on acyclic NFAs. A corollary of the construction is that finding the *minimal size* distinguishing formula is NP-hard.

We define the decision problems *TRACE-DIST* and *MIN-DIST*. Given an LTS $L = (S, Act, \rightarrow)$, two states $s, t \in S$ such that $s \not\approx_i t$ for $i = |S|$, and a number $l \in \mathbb{N}$.

TRACE-DIST: There is a trace-formula $\phi \in \mathcal{F}_i$, such that ϕ distinguishes s and t.

MIN-DIST: There is a formula $\phi \in \mathcal{F}_i$, such that ϕ distinguishes s and t, and $|\phi| \leqslant l$.

We point out that *TRACE-DIST* is not the same as deciding trace-equivalence. The problem *TRACE-DIST* decides whether there is a distinguishing trace of length i, and i is smaller than the number of states, and a minimal distinguishing trace might be super-polynomial in size [7, Sec. 5].

3.1 Reduction

We prove that *TRACE-DIST* is NP-complete and *MIN-DIST* is NP-hard by a reduction from the decision problem *CNF-SAT*. This decision problem decides whether a given propositional formula $\mathcal{C}$ in conjunctive normal form (CNF) is satisfiable. For this we define an LTS $L_{\mathcal{C}}$, based on the CNF formula $\mathcal{C}$.

▶ **Definition 18.** *Let $\mathcal{C} = C_1 \wedge \ldots \wedge C_n$ be a CNF formula over the set of proposition letters $Prop = \{p_1, \ldots, p_k\}$. We define the LTS $L_{\mathcal{C}} = (S, Act, \rightarrow)$ as follows:*

- *The set of states S is defined as*

$$S = \{\mathtt{unsat}_i^C \mid C \in \{C_1, \ldots, C_n\}, i \in [0,k]\} \cup \{\mathtt{sat}_i \mid i \in [0,k]\} \cup \{\bot_i \mid i \in [0,k]\} \cup \{s, t, \delta\}.$$

- *The set of actions Act is defined as*

$$Act = \{p, \overline{p} \mid p \in Prop\} \cup \{\mathtt{init}, \mathtt{false}\}.$$

- *The relation $\rightarrow$ contains for each $C \in \{C_1, \ldots, C_n\}$ and $i \in [1,k]$:*

$$\mathtt{unsat}_{i-1}^C \xrightarrow{p_i} \begin{cases} \mathtt{sat}_i & \text{if } p_i \text{ is a literal of } C, \\ \mathtt{unsat}_i^C & \text{otherwise,} \end{cases}$$
$$\mathtt{unsat}_{i-1}^C \xrightarrow{\overline{p_i}} \begin{cases} \mathtt{sat}_i & \text{if } \neg p_i \text{ is a literal of } C, \\ \mathtt{unsat}_i^C & \text{otherwise,} \end{cases}$$
$$\mathtt{sat}_{i-1} \xrightarrow{x} \mathtt{sat}_i \text{ for } x \in \{p_i, \overline{p_i}\}, \text{ and}$$
$$\bot_{i-1} \xrightarrow{x} \bot_i \text{ for } x \in \{p_i, \overline{p_i}\}.$$

Additionally, it contains the auxiliary transitions

$$\mathtt{unsat}_k^C \xrightarrow{\mathtt{false}} \delta \text{ for } C \in \{C_1, \ldots, C_n\},$$
$$\bot_k \xrightarrow{\mathtt{false}} \delta,$$
$$t \xrightarrow{\mathtt{init}} \mathtt{unsat}_0^C \text{ for } C \in \{C_1, \ldots, C_n\},$$
$$t \xrightarrow{\mathtt{init}} \mathtt{sat}_0,$$
$$s \xrightarrow{\mathtt{init}} \mathtt{sat}_0, \text{ and}$$
$$s \xrightarrow{\mathtt{init}} \bot_0.$$

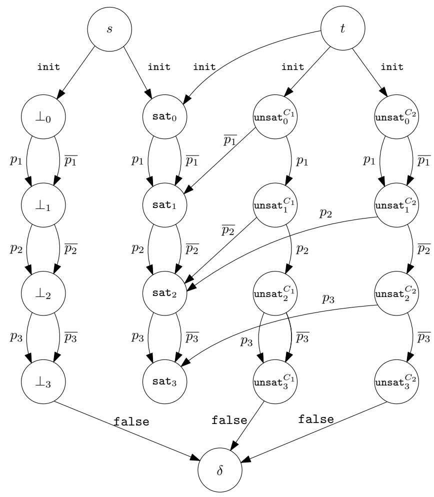


Figure 4 The LTS $L_\mathcal{C}$ for the formula $\mathcal{C} = (\neg p_1 \vee \neg p_2) \wedge (p_2 \vee p_3)$.

The LTS $L_\mathcal{C}$ for the CNF formula $\mathcal{C} = C_1 \wedge C_2$ with clauses $C_1 = \neg p_1 \vee \neg p_2$ and $C_2 = p_2 \vee p_3$ is depicted in Figure 4.

In this construction an interpretation of the propositions $Prop = \{p_1, \ldots, p_k\}$ is directly related to a word $w = a_1 \ldots a_k$, where $a_i \in \{p_i, \overline{p_i}\}$ for every $i \in [1, k]$. The set of truth assignments encoded as words is defined as:

$$\mathit{Truths} = \{a_1 \ldots a_k \mid a_i \in \{p_i, \overline{p_i}\} \text{ for all } i \in [1, k]\}.$$

Given a truth assignment $\rho : Prop \to \mathbb{B}$, we define w_ρ as $w_\rho = a_1 \ldots a_k$, where $a_i = p_i$ if $\rho(p_i) = \mathit{true}$ and $a_i = \overline{p_i}$, otherwise. Conversely, for a word $w = a_1 \ldots a_k$, a trace from *Truths*, it represents the truth assignment ρ_w defined for each $i \in [1, k]$ as:

$$\rho_w(p_i) = \begin{cases} \mathit{true} & \text{if } a_i = p_i, \\ \mathit{false} & \text{if } a_i = \overline{p}_i. \end{cases}$$

The idea of the construction of $L_\mathcal{C}$ is that it contains a $\bot$ component, a $\texttt{sat}$ component, and an $\texttt{unsat}^C$ component for every clause C. All components are deterministic and acyclic, and hence describe a finite set of traces. All the traces of these components start by a truth assignment $w \in \mathit{Truths}$. By construction, for every truth assignment $w \in \mathit{Truths}$, $w \cdot \texttt{false} \in \mathit{Tr}(\bot_0)$. In this way the $\bot$ component represents falsehood. Conversely, the state $\texttt{sat}_0$ represents a tautology, since for any truth assignment $w \in \mathit{Truths}$, $w \cdot \texttt{false} \notin \mathit{Tr}(\texttt{sat}_0)$. For every clause C, and truth assignment $w \in \mathit{Truths}$ the state $\texttt{unsat}_0^C$ contains $w \cdot \texttt{false}$ as trace iff ρ_w does not satisfy C.

▶ **Lemma 19.** *Let $L_\mathcal{C} = (S, Act, \to)$ be the LTS for a CNF formula $\mathcal{C} = C_1 \wedge \ldots \wedge C_n$ with propositions $\{p_1, \ldots, p_k\}$, then:*

$$\begin{aligned} \mathit{Tr}(\texttt{sat}_0) &= \{u \in Act^* \mid \exists w \in \mathit{Truths}.\ u \text{ is a prefix of } w\}, \\ \mathit{Tr}(\bot_0) &= \mathit{Tr}(\texttt{sat}_0) \cup \{w \cdot \texttt{false} \mid w \in \mathit{Truths}\}, \text{ and} \\ \mathit{Tr}(\texttt{unsat}_0^C) &= \mathit{Tr}(\texttt{sat}_0) \cup \{w \cdot \texttt{false} \mid w \in \mathit{Truths} \text{ and } \rho_w \text{ does not satisfy } C\}. \end{aligned}$$

This lemma is easily verified from the construction of $L_\mathcal{C}$.

▶ **Corollary 20.** *Let $w \in \mathit{Truths}$ be a trace, and $L_{\mathcal{C}}$ the LTS for the CNF formula $\mathcal{C} = C_1 \wedge \ldots \wedge C_n$. Then for any clause $C \in \{C_1, \ldots, C_n\}$:*

$$w{\cdot}\mathtt{false} \in \mathit{Tr}(\mathtt{unsat}_0^C) \iff C \text{ is not satisfied under } \rho_w.$$

The following lemma contains the main idea for the reduction of the main theorem showing *TRACE-DIST* is NP-complete.

▶ **Lemma 21.** *Given the LTS $L_{\mathcal{C}} = (S, \rightarrow, \mathit{Act})$ for a CNF formula $\mathcal{C} = C_1 \wedge \ldots \wedge C_n$, with propositions $\mathit{Prop} = \{p_1, \ldots, p_k\}$. Then there is a trace $w \in \mathit{Act}^{k+1}$ such that $w \in \mathit{Tr}(\bot_0)$, and $w \notin \mathit{Tr}(\mathtt{unsat}_0^C)$ for every $C \in \{C_1, \ldots, C_n\}$ if and only if $\mathcal{C}$ is satisfiable.*

Proof. We prove this in both directions separately.

($\Rightarrow$) As a witness, we obtain a trace $w \in \mathit{Tr}(\bot_0)$ of length at most $k{+}1$ such that $w \notin \mathit{Tr}(\mathtt{unsat}_0^C)$ for all clauses $C \in \{C_1, \ldots, C_n\}$. Since $w \in \mathit{Tr}(\bot_0)$ by Lemma 19 either $w \in \mathit{Tr}(\mathtt{sat}_0)$ or $w \in \{v \cdot \mathtt{false} \mid v \in \mathit{Truths}\}$. Since $\mathit{Tr}(\mathtt{sat}_0) \subseteq \mathit{Tr}(\mathtt{unsat}_0^C)$, and $w \notin \mathit{Tr}(\mathtt{unsat}_0^C)$, there is a trace $v \in \mathit{Truths}$ such that $w = v \cdot \mathtt{false}$. By Corollary 20 all clauses C are satisfied by ρ_w. This means ρ_w is a satisfying assignment for $\mathcal{C}$.

($\Leftarrow$) If there is a satisfying assignment ρ for $\mathcal{C}$ then we show that $w_\rho \cdot \mathtt{false}$ witnesses the implication. First observe that by definition $w_\rho \cdot \mathtt{false} \in \mathit{Tr}(\bot_0)$. Let $C \in \{C_1, \ldots, C_n\}$ be any clause. Since ρ is a satisfying assignment, C is satisfied under ρ. This means by Corollary 20 that $w_\rho \notin \mathit{Tr}(\mathtt{unsat}_0^C)$. ◀

Now we are ready to prove the main theorem of this section.

▶ **Theorem 22.** *Deciding TRACE-DIST is NP-complete.*

Proof. First we verify that TRACE-DIST is in NP. Given an LTS $L = (S, \mathit{Act}, \rightarrow)$, and two states $s, t \in S$. As a witness we get a formula $\phi \in \mathcal{F}_{|S|}$, which is a trace-formula. Since $d_\diamond(\phi) \leqslant |S|$ this is polynomial in size. It is well known that given a formula ϕ we can check in polynomial time whether $s \sim_\phi t$.

To show TRACE-DIST is NP-hard we reduce CNF-SAT to TRACE-DIST. Let $\mathcal{C} = C_1 \wedge \ldots \wedge C_n$ be a CNF formula over the propositions $\mathit{Prop} = \{p_1, \ldots, p_k\}$. Then for the LTS $L_{\mathcal{C}}$ we show there is a distinguishing trace smaller than $|S|$ for $s \in S$ and $t \in S$ if and only if $\mathcal{C}$ is satisfiable.

We begin by observing the sets $\mathit{Tr}(s)$, $\mathit{Tr}(t)$:

$$\mathit{Tr}(s) = \{\varepsilon, \mathtt{init}\} \cup \{\mathtt{init} \cdot w \mid w \in \mathit{Tr}(\bot_0) \cup \mathit{Tr}(\mathtt{sat}_0)\},$$
$$\mathit{Tr}(t) = \{\varepsilon, \mathtt{init}\} \cup \{\mathtt{init} \cdot w \mid w \in \mathit{Tr}(\mathtt{sat}_0) \cup \bigcup_{i \in [1,n]} \mathit{Tr}(\mathtt{unsat}_0^{C_i})\}.$$

Since for every $C \in \{C_1, \ldots, C_n\}$, $\mathit{Tr}(\mathtt{unsat}_0^C) \subseteq \mathit{Tr}(\bot_0)$ and $\mathit{Tr}(\mathtt{sat}_0) \subseteq \mathit{Tr}(\mathtt{unsat}_0^C)$, we know that if there is a distinguishing trace it has to be $\mathtt{init} \cdot w \in \mathit{Tr}(s)$ for a $w \in \mathit{Tr}(\bot_0)$. By Lemma 21 this trace w exists iff $\mathcal{C}$ is satisfiable. Hence, the states s and t are in *TRACE-DIST* if and only if $\mathcal{C}$ is in *CNF-SAT*. The LTS $L_{\mathcal{C}}$ can be computed in polynomial time, as it has $(n+2)(k+1)+3$ states and $2k(n+2)+2n+4$ transitions. This concludes the proof that *TRACE-DIST* is NP-complete. ◀

In the reduction a distinguishing trace is also a minimal distinguishing formula. Which means we can generalise our NP-hardness result.

▶ **Corollary 23.** *Deciding MIN-DIST is NP-hard.*

Proof. We prove this by a similar reduction as in the proof of Theorem 22. The intuition is that, given a CNF formula $\mathcal{C} = C_1 \wedge \ldots \wedge C_n$ with propositions $Prop = \{p_1, \ldots, p_k\}$, in the LTS $L_{\mathcal{C}}$ a distinguishing formula $\phi \in \mathcal{F}$ such that $|\phi| = k + 2$ necessarily is a trace-formula.

We reduce CNF-SAT to MIN-DIST. Let $\mathcal{C} = C_1 \wedge \ldots \wedge C_n$ be a CNF formula over the propositions $Prop = \{p_1, \ldots, p_k\}$. Then for the LTS $L_{\mathcal{C}}$ we show there is a distinguishing formula $\phi \in \mathcal{F}$ for $s \in S$ and $t \in S$ such that $|\phi| \leqslant k + 2$ if and only if $\mathcal{C}$ is satisfiable.

For the direction $\Rightarrow$, assume a formula $\phi \in \mathcal{F}$ exists such that $|\phi| \leqslant k + 2$ and $s \not\sim_\phi t$. We show that this means $\mathcal{C}$ is satisfiable. We observe by the deterministic behaviour that $s \leftrightarroweq_{k+1} t$. Hence, by Theorem 10 we know $d_\diamond(\phi) \geq k + 2$. Since we assume $|\phi| \leqslant k + 2$ we know that $d_\diamond(\phi) = k + 2$ and so, there are no non-trivial conjunctions, and we see that we can rewrite $\neg\neg\phi \mapsto \phi$. Hence, there is a formula $\psi = \triangle_1 \ldots \triangle_{k+2} tt$ such that for each $i \in [1, k + 2]$, $\triangle_i \in \{\langle a_i \rangle, \neg\langle a_i \rangle\}$, for some $a_1, \ldots, a_{k+2} \in Act$, such that $[\![\phi]\!] = [\![\psi]\!]$.

By Lemma 17 there is a trace $w \in Tr(\psi)$, such that $|w| \geq k + 2$ and, $w \in Tr(s) \cup Tr(t)$. The only trace of this length of s or t is in the shape $w = \mathtt{init} \cdot \hat{p}_1 \ldots \hat{p}_k \cdot \mathtt{false}$, where $\hat{p_i} \in \{p_i, \overline{p}_i\}$ for each $i \in [1, k]$. This means that $a_1 = \mathtt{init}$, $a_{j+1} = \hat{p}_j$ for each $j \in [1, k]$ and $a_{k+2} = \mathtt{false}$. We are going to show that the associated truth value $\rho = \rho_{\hat{p}_1 \ldots \hat{p}_k}$ satisfies $\mathcal{C}$ by reductio ad absurdum.

If ρ does not satisfy $\mathcal{C}$ then there is a clause C such that C is not satisfied by ρ. We claim for this clause $\mathtt{unsat}_0^C \sim_{\Delta_2 \ldots \Delta_{k+1} tt} \bot_0$, and since both s and t have a $\mathtt{init}$-transition to $\mathtt{sat}_0$ this means ψ does not distinguish any of the derivatives. Hence $s \sim_\psi t$ which is a contradiction.

For the other direction if $\mathcal{C}$ is satisfiable then by Lemma 21 there is a $w \in Act^{k+1}$ such that $w \in Tr(\bot_0)$ and $w \notin Tr(\mathtt{unsat}_0^C)$ for all clauses $C \in \{C_1, \ldots, C_n\}$. Using w we construct the distinguishing trace $w' = \mathtt{init} \cdot w$. Since $w \in Tr(\bot_0)$, $w \notin Tr(\mathtt{unsat}_0^C)$ and by construction also $w \notin Tr(\mathtt{sat}_0)$, it is the case that $w' \in Tr(s)$ and $w' \notin Tr(t)$. This means the formula $\phi_{w'}$ is a distinguishing formula and $|\phi_{w'}| = k + 2$, which finishes the second part of the proof. ◀

The problem MIN-DIST is not a member of NP since a polynomially sized witness might not exist. However, there is always a "shared" distinguishing formula of polynomial size. Since we can compute in polynomial time if a shared formula is a distinguishing formula, the decision problem MIN-DIST formulated in terms of total "shared" modalities is NP-complete.

4 Efficient algorithms

In this section we explain that despite the NP-hardness results from the previous section it is still possible to efficiently generate distinguishing formulas with minimal observation- and negation-depth. First, we introduce the method $\phi(s, t)$ listed in Algorithm 1 that generates a minimal observation-depth distinguishing formula for the states s and t. We extend $\phi(s, t)$ to the function $\psi_i(s, t)$ listed in Algorithm 2. This method computes a distinguishing formula with observation-depth of at most i and minimal negation-depth. Additionally, this procedure also prevents unnecessary conjuncts to be added. Finally, we indicate how to compute the equivalences $\leftrightarroweq_1, \ldots, \leftrightarroweq_k$, and the minimal observation- and negation-depth.

4.1 The algorithm

For every $i \in \mathbb{N}$, we define a function $\delta_i : S \times S \to 2^{Act \times S}$ that gives all distinguishing observations. More precisely, given two i-distinguishable states $s \in S$ and $t \in S$, $\delta_i(s, t)$ returns all pairs (a, s'), where $a \in Act$, $s' \in S$, such that $s \xrightarrow{a} s'$ and s' is $(i-1)$-distinguishable from all targets $t \xrightarrow{a} t'$. The definition of $\delta_i(s, t)$ is:

$$\delta_i(s, t) = \{(a, s') \mid s \xrightarrow{a} s' \text{ and } \forall t \xrightarrow{a} t'.\ \Delta(s', t') \leqslant i - 1\}.$$

Algorithm 1 Minimal-depth distinguishing formula.

input : Two states $s, t \in S$ such that $s \not\simeq_i t$
output: A formula $\phi \in \mathcal{F}$ s.t. $s \in [\![\phi]\!]$ and $t \notin [\![\phi]\!]$
Function $\phi(s,t)$ **is**
  $i := \Delta(s,t)$;
  if $\delta_i(s,t) = \emptyset$ **then**
    return $\neg\,\phi(t,s)$
  Select $(a, s') \in \delta_i(s,t)$;
  $T := \{t' \mid t \xrightarrow{a} t'\}$;
  return $\langle a \rangle \left(\bigwedge_{t' \in T} \phi(s', t') \right)$;
end

Using the function $\delta_i(s,t)$, we can compute a minimal observation-depth formula using the procedure listed as Algorithm 1. The procedure selects an action state pair $(a, s') \in \delta_i(s,t)$ and recursively distinguishes s' from all a-derivatives of t. If $\delta_i(s,t)$ is empty the negated $\phi_i(t,s)$ is calculated and in this case $\delta_i(t,s)$ is necessarily not empty.

▶ **Lemma 24.** *Given an LTS $L = (S, Act, \rightarrow)$ and two states $s, t \in S$. If $s \not\simeq_i t$ then: $\delta_i(s,t) \neq \emptyset$ or $\delta_i(t,s) \neq \emptyset$.*

Proof. As $s \not\simeq_i t$ there either is an $s \xrightarrow{a} s'$ such that $s' \not\simeq_{i-1} t'$ for all $t \xrightarrow{a} t'$, or vice-versa there is a $t \xrightarrow{a} t'$ such that $t' \not\simeq_{i-1} s'$ for all $s \xrightarrow{a} s'$. In the first case $(a, s') \in \delta_i(s,t)$, in the second case $(a, t') \in \delta_i(t,s)$. ◀

4.2 Minimal negation-depth

In order to minimize the number of negations within the minimal observation-depth formula we combine the notions of k-bisimilar and m-nested similarity inclusion.

▶ **Definition 25.** *Let $L = (S, Act \rightarrow)$ be an LTS, and $k, m \in \mathbb{N}$. We define m-nested k-similarity inclusion, denoted $\simeq_k^m$, inductively by for all $s, t \in S$, $s \simeq_0^m t$ and if $s \simeq_k^m t$ then*

1. *if $s \xrightarrow{a} s'$ there is a $t \xrightarrow{a} t'$ such that $s' \simeq_{k-1}^m t'$, and*
2. *if $m > 0$ and $t \xrightarrow{a} t'$, then there is a $s \xrightarrow{a} s'$ such that $t' \simeq_{k-1}^{m-1} s'$.*

Similarly to the original Hennessy-Milner correspondences, we observe the correspondence between the fragment $\mathcal{F}_k^m$ and the relation $\simeq_k^m$.

▶ **Theorem 26.** *Let $L = (S, Act, \rightarrow)$ be an LTS. For any $k, m \in \mathbb{N}$ and states $s, t \in S$:*

$$s \leqslant_{\mathcal{F}_k^m} t \iff s \simeq_k^m t.$$

Related to the distance measure Δ, we define the directed minimal negation-depth measure for the relation $\simeq_k^m$, for states that are not m-nested k-similar for some $k, m \in \mathbb{N}$.

▶ **Definition 27.** *Let $L = (S, Act, \rightarrow)$ be an LTS and $i \in \mathbb{N}$ be a number. We define the directed minimal negation-depth $\overrightarrow{\Delta}_i : S \times S \rightarrow \mathbb{N} \cup \{\infty\}$ by*

$$\overrightarrow{\Delta}_i(s,t) = \begin{cases} j & \text{if } s \not\simeq_i^j t, \text{ and } s \simeq_i^{j-1} t, \\ \infty & \text{if } s \Leftrightarrow_i t. \end{cases}$$

Algorithm 2 Generate a distinguishing formula with minimal observation- and negation-depth.

input : Two states $s, t \in S$ such that $s \not\equiv_i t$ for some $i \in \mathbb{N}$
output : A formula $\phi \in \mathcal{F}_i$ such that $s \in [\![\phi]\!]$ and $t \notin [\![\phi]\!]$
1 **Function** $\phi_i(s, t)$ **is**
2 $\quad j := \overrightarrow{\Delta}_i(s, t)$;
3 $\quad \mathcal{X} := \hat{\delta}_i^j(s, t)$;
4 $\quad$ **if** $\mathcal{X} = \emptyset$ **then**
5 $\quad\quad$ **return** $\neg\, \phi_i(t, s)$
6 $\quad$ Select $(a, s') \in \mathcal{X}$;
7 $\quad T := \{t' \mid t \xrightarrow{a} t'\}$;
8 $\quad$ **while** $T \neq \emptyset$ **do**
9 $\quad\quad$ Select $t_{max} \in T$ such that $\overrightarrow{\Delta}_{i-1}(s', t_{max}) \geq \overrightarrow{\Delta}_{i-1}(s', t')$ for all $t' \in T$;
10 $\quad\quad \phi_{t_{max}} := \phi_{i-1}(s', t_{max})$;
11 $\quad\quad \Phi := \Phi \cup \{\phi_{t_{max}}\}$;
12 $\quad\quad T := T \cap [\![\phi_{t_{max}}]\!]$;
13 $\quad$ **end**
14 $\quad$ **return** $\langle a \rangle \left(\bigwedge_{\phi \in \Phi} \phi \right)$
15 **end**

For every $i, j \in \mathbb{N}$ we define a function $\hat{\delta}_i^j : S \times S \to 2^{Act \times S}$ that is similar to the function δ_i. It adds an extra limitation on the number of negations needed to distinguish the pairs from all observations from t.

$$\hat{\delta}_i^j(s, t) = \{(a, s') \mid (a, s') \in \delta_i(s, t) \text{ and } \forall t \xrightarrow{a} t'.\ \overrightarrow{\Delta}_{i-1}(s', t') \leqslant j\}.$$

The next lemma guarantees that a suitable distinguishing observation exists.

▶ **Lemma 28.** *Given an LTS $L = (S, Act, \to)$ and two states $s, t \in S$. Then for all $i, j \in \mathbb{N}$, if $s \not\equiv_i^j t$ then $\hat{\delta}_i^j(s, t) \neq \emptyset$ or $\hat{\delta}_i^{j-1}(t, s) \neq \emptyset$.*

In Algorithm 2 we give the method $\psi_i(s, t)$ that given an LTS $L = (S, Act, \to)$ and i-distinguishable states $s, t \in S$ generates a formula such that $s \in [\![\psi_i(s, t)]\!]$ and $t \notin [\![\psi_i(s, t)]\!]$ with observation depth at most i and minimal negation-depth.

The algorithm attempts to find an action label $a \in Act$ and an a-derivative $s \xrightarrow{a} s'$, such that all a-derivatives t', such that $t \xrightarrow{a} t'$ are distinguishable with a formula with at most $i-1$ nested observations and j nested negations. These pairs (a, s') are given by the function $\hat{\delta}_i^j(s, t)$. In Line 6 one of these witnesses is chosen. If there is more than one suitable derivate, one is chosen at random.

The next theorem states that Algorithm 2 yields a valid distinguishing formula.

▶ **Theorem 29.** *Let $L = (S, Act, \to)$ be an LTS, and $s, t \in S$ be states. If s and t are k-distinguishable for some $k \in \mathbb{N}$ then $s \in [\![\psi_k(s, t)]\!]$ and $t \notin [\![\psi_k(s, t)]\!]$.*

The next theorem states that if $\Delta(s, t) = k$, then $\psi_k(s, t)$ yields a formula that has minimal observation-depth, and there is no formula ϕ with a smaller number of nested negations such that $s \not\leqslant_\phi t$.

▶ **Theorem 30.** *Let $L = (S, Act, \to)$ be an LTS, and $s, t \in S$ be states, such that $s \not\equiv t$ and $\Delta(s, t) = k$. Then for all $\phi \in \mathcal{F}$, if $s \not\leqslant_\phi t$ then $d_\diamond(\psi_k(s, t)) \leqslant d_\diamond(\phi)$ and if $d_\neg(\phi) < d_\neg(\psi_k(s, t))$ then $d_\diamond(\phi) > d_\diamond(\psi_k(s, t))$.*

Algorithm 3 Iterative partition refinement.

Function `Refine`(π) **is**
    $\pi' := \pi$;
    foreach $a \in Act, B' \in \pi$ **do**
        foreach $B \in \pi'$ **do**
            $C := split_a(B, B')$;
            if $C \neq B$ *and* $C \neq \emptyset$ **then**
                $\pi' := (\pi' \setminus \{B\}) \cup \{C, B \setminus C\}$;
    return π';
$i := 0$; $\pi_0 := \{S\}$;
while $\pi_i \neq$ `Refine`(π_i) **do**
    $\pi_{i+1} :=$ `Refine`(π_i);
    $i := i + 1$;

4.3 Partition refinement

In order to execute Algorithm 2, we need to compute the functions Δ and $\overrightarrow{\Delta}$. In this section we propose a simple partition refinement algorithm that does exactly this by first computing the relations $\leftrightarrows_0, \leftrightarrows_1, \ldots, \leftrightarrows_k$ iteratively. The pseudocode is listed in Algorithm 3. In contrast to the more efficient partition refinement algorithms [12, 21, 24], we guarantee that *older* blocks are used first as splitter. This method is inspired by [23] where pairwise minimal distinguishing words are computed.

Most algorithms deciding bisimilarity are so-called partition refinement algorithms [14, 21]. Our algorithms are also based on partition refinement. A *partition* π of a set S is a disjoint cover of S, i.e. a set of non-empty subsets of S and every element of S is in exactly one subset. The elements $B \in \pi$ are called *blocks*. A partition π induces the equivalence relation $\sim_\pi : S \times S$ in which the blocks are the equivalence classes, i.e. $\sim_\pi = \{(s,t) \mid \exists B \in \pi \text{ and } s, t \in B\}$. In the algorithm we filter a set of states U on a distinguishing observation with respect to a set of given states V, and an action $a \in Act$, i.e.: $split_a(U, V) = \{s \in U \mid \exists s' \in V. s \xrightarrow{a} s'\}$.

The next theorem states that the procedure listed as Algorithm 3 produces a sequence of partitions, in which the i-th partition induces i-bisimilarity.

▶ **Theorem 31.** *Given an LTS $L = (S, Act, \rightarrow)$ and partitions $\pi_0, \ldots, \pi_k$ produced by Algorithm 3. Then $\sim_{\pi_i} = \leftrightarrows_i$, for all $0 \leqslant i \leqslant k$.*

It is possible to compute the function $\overrightarrow{\Delta}_i(s,t)$ in polynomial time from the computed k-bisimilarity relations calculated in Algorithm 3. It is important to use dynamic programming such that $\overrightarrow{\Delta}_i(s,t)$ for every i, s and t is only calculated once.

4.4 Evaluation

The computation of Algorithm 2 needs to account for redundancies to guarantee a polynomial time algorithm. We use dynamic programming to achieve this. For any pair of states $s, t \in S$ if the function $\psi_i(s,t)$ is invoked, it stores the generated shared formula. Whenever the function is called again, the previously generated formula is used, with only constant extra computing and memory usage. Hence, given an LTS $L = (S, Act, \rightarrow)$ the number of recursive calls is limited to the combination of states and level $k \leqslant |S|$, i.e. $\mathcal{O}(|S|^3)$ calls.

▶ **Corollary 32.** *Given an LTS $L = (S, Act, \rightarrow)$ and a pair of distinguishable states $s, t \in S$, then the following is computable in polynomial time:*

- *A minimal observation-depth distinguishing formula,*
- *A minimal observation- and negation-depth distinguishing formula.*

A naive implementation of the algorithms requires quadratic memory. This could be a bottleneck for large state spaces. Representing the equivalences $\leftrightarrows_k$ as a splitting tree [17] is more memory efficient. In addition, an optimization is to generate only distinguishing formulas between equivalence classes of the generated equivalences, instead of individual states.

Table 1 Results from prototype implementation Algorithm 1.

Benchmark	Max						Average					
	$d_\Diamond(\phi)$		$\lvert\phi\rvert$		$d_\neg(\phi)$		$d_\Diamond(\phi)$		$\lvert\phi\rvert$		$d_\neg(\phi)$	
	Our	Cleav.	Our	Cleav.	Our	Cleav.	Our	Cleav.	Our	Cleav.	Our	Cleav.
ieee-1394-1	64	891	69	1355	0	886	64,0	247,2	69,0	373,7	0,0	243,2
ieee-1394-2	37	224	42	320	1	219	37,0	92,0	42,0	120,0	1,0	88,2
ieee-1394-3	102	698	102	1092	2	696	102,0	299,1	102,0	465,4	2,0	295,7
ieee-1394-4	76	363	83	506	2	360	76,0	196,6	80,9	276,5	2,0	194,5
ieee-1394-5	18	155	18	214	2	146	18,0	36,0	18,0	44,8	2,0	30,4

We implemented a prototype of the method introduced here. We also implemented the method proposed by Cleaveland [6] in which we decided bisimilarity by a partition refinement algorithm in which the splitter selected is the latest created block, since heuristically this has the best runtime [1, 2]. For Cleaveland's method the strategy for splitter selection matters for the size of the formulas generated. However, regardless of strategy chosen, the formulas that our method generates are always more concise in all metrics.

We post-processed the formulas to ensure both implementations resulted in formulas that are irreducible. For the benchmark we used the model from [18] containing 188.568 states and 340.607 transitions. We compared this model to 5 modified versions where we omitted one randomly chosen transition. In Table 1 the results of running the algorithms 10 times are shown. Under "Max", the worse-case of the different runs for each metric is listed for our method ("Our"), next to the result of the implementation of Cleaveland ("Cleav."). Under "Average" the average of the 10 runs is shown.

We see that our new method consistently outputs a minimal observation- and negation-depth formula, and the generated formulas only rarely deviates in size. It outperforms the method of Cleaveland in all cases. In some cases the depth is improved a factor 10.

5 Conclusions & Future work

In this work we studied the problem of computing minimal distinguishing formulas. We introduced three metrics: size, observation-depth, and negation-depth. Using a reduction directly from CNF-SAT we showed that finding a minimal sized distinguishing formula is NP-hard. However, for observation- and negation-depth, we introduce polynomial time algorithms that compute minimal formulas. A prototype demonstrates the potential improvement over the method introduced by Cleaveland [6]. A more rigorous version is implemented in the mCRL2 toolset [5].

For future work it would be interesting to extend our algorithms for equivalences beyond strong bisimilarity. For instance, a more generic coalgebraic treatment, extending [25], or computing smaller witnesses for equivalences with abstractions like branching and weak bisimilarity, improving upon the work of Korver [16].

References

1 Manuel Baclet and Claire Pagetti. Around Hopcroft's algorithm. In *Implementation and Application of Automata: 11th International Conference, CIAA 2006, Taipei, Taiwan, August 21-23, 2006. Proceedings 11*, pages 114–125. Springer, 2006.

2 Christoph Berkholz, Paul Bonsma, and Martin Grohe. Tight lower and upper bounds for the complexity of canonical colour refinement. *Theory of Computing Systems*, 60(4):581–614, 2017.

3 Benjamin Bisping, David N. Jansen, and Uwe Nestmann. Deciding all behavioral equivalences at once: A game for linear-time–branching-time spectroscopy. *Logical Methods in Computer Science*, 18(3), August 2022. doi:10.46298/lmcs-18(3:19)2022.

4 Michael C. Browne, Edmund M. Clarke, and Orna Grümberg. Characterizing finite kripke structures in propositional temporal logic. *Theoretical Computer Science*, 59(1):115–131, 1988. doi:10.1016/0304-3975(88)90098-9.

5 Olav Bunte, Jan Friso Groote, Jeroen J. A. Keiren, Maurice Laveaux, Thomas Neele, Erik P. de Vink, Wieger Wesselink, Anton Wijs, and Tim A. C. Willemse. The mCRL2 toolset for analysing concurrent systems. In Tomáš Vojnar and Lijun Zhang, editors, *Tools and Algorithms for the Construction and Analysis of Systems (TACAS '19)*, pages 21–39. Springer International Publishing, 2019. doi:10.1007/978-3-030-17465-1_2.

6 Rance Cleaveland. On automatically explaining bisimulation inequivalence. In Edmund M. Clarke and Robert P. Kurshan, editors, *Proc. Computer-Aided Verification (CAV 1990)*, pages 364–372, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg. doi:10.1007/BFb0023750.

7 Keith Ellul, Bryan Krawetz, Jeffrey Shallit, and Ming-Wei Wang. Regular expressions: New results and open problems. *J. Autom. Lang. Comb.*, 10(4):407–437, 2005.

8 Santiago Figueira and Daniel Gorín. On the size of shortest modal descriptions. In *Advances in Modal Logic*, volume 8, pages 114–132, 2010.

9 Herman Geuvers. Apartness and distinguishing formulas in Hennessy-Milner logic. In Nils Jansen, Mariëlle Stoelinga, and Petra van den Bos, editors, *A Journey from Process Algebra via Timed Automata to Model Learning*, pages 266–282. Springer Nature Switzerland, 2022. doi:10.1007/978-3-031-15629-8_14.

10 Jan Friso Groote and Frits Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, 1992. doi:10.1016/0890-5401(92)90013-6.

11 Matthew Hennessy and Robin Milner. On observing nondeterminism and concurrency. In Jaco de Bakker and Jan van Leeuwen, editors, *Automata, Languages and Programming (ICALP '1980)*, pages 299–309, Berlin, Heidelberg, 1980. Springer-Verlag. doi:10.5555/646234.758793.

12 John Hopcroft. An n log n algorithm for minimizing states in a finite automaton. In Zvi Kohavi and Azaria Paz, editors, *Theory of Machines and Computations*, pages 189–196. Academic Press, 1971. doi:10.1016/B978-0-12-417750-5.50022-1.

13 Harry B. Hunt III. *On the Time and Tape Complexity of Languages*. PhD thesis, Cornell University, 1973.

14 Paris C. Kanellakis and Scott A. Smolka. CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 86(1):43–68, 1990. doi:10.1016/0890-5401(90)90025-D.

15 Barbara König, Christina Mika-Michalski, and Lutz Schröder. Explaining non-bisimilarity in a coalgebraic approach: Games and distinguishing formulas. In Daniela Petrişan and Jurriaan Rot, editors, *Coalgebraic Methods in Computer Science (CMCS 2022)*, pages 133–154. Springer, 2020.

16 Henri Korver. Computing distinguishing formulas for branching bisimulation. In Kim G. Larsen and Arne Skou, editors, *Computer Aided Verification (CAV 1991)*, pages 13–23. Springer, 1992.

17 David Lee and Mihalis Yannakakis. Testing finite-state machines: State identification and verification. *IEEE Transactions on computers*, 43(3):306–320, 1994. doi:10.1109/12.272431.

18 Bas Luttik. *Description and formal specification of the Link Layer of P1394*. CWI report. SEN-R : software engineering. Centrum voor Wiskunde en Informatica, 1997.

19 Robin Milner. *A calculus of communicating systems*. Springer, 1980.

20 Edward F. Moore. Gedanken-experiments on sequential machines. In Claude Shannon and John McCarthy, editors, *Automata Studies*, pages 129–153. Princeton University Press, Princeton, NJ, 1956.

21 Robert Paige and Robert E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987. doi:10.1137/0216062.

22 David Park. Concurrency and automata on infinite sequences. In Peter Deussen, editor, *Theoretical Computer Science*, pages 167–183. Springer Berlin Heidelberg, 1981.

23 Rick Smetsers, Joshua Moerman, and David N Jansen. Minimal separating sequences for all pairs of states. In Adrian-Horia Dediu, Jan Janoušek, Carlos Martín-Vide, and Bianca Truthe, editors, *Language and Automata Theory and Applications (LATA 2016)*, pages 181–193. Springer, 2016.

24 Antti Valmari. Bisimilarity minimization in o(m logn) time. In Giuliana Franceschinis and Karsten Wolf, editors, *Applications and Theory of Petri Nets*, pages 123–142, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

25 Thorsten Wißmann, Stefan Milius, and Lutz Schröder. Quasilinear-time computation of generic modal witnesses for behavioural inequivalence. *Logical Methods in Computer Science*, 18(4), November 2022. doi:10.46298/lmcs-18(4:6)2022.

26 Jacob K. Wortmann, Simon R. Olesen, and Søren Enevoldsen. Caal 2.0. recursive HML, distinguishing formulae, equivalence collapses and parallel fixed-point computations. Master's thesis, Aalborg University, 2015.