Branching-Depth Hierarchies

Shoham Shamir Orna Kupferman Eli Shamir

School of Engineering and Computer Science
The Hebrew University
Jerusalem 91904, Israel

Abstract

We study the distinguishing and expressive power of branching temporal logics with bounded nesting depth of path quantifiers. We define the fragments $\mathrm{CTL}_i^\star$ and CTL_i of $\mathrm{CTL}^\star$ and CTL, where at most i nestings of path quantifiers are allowed. path quantifiers in $\mathrm{CTL}^\star$ and CTL formulas, respectively. We show that for all $i \geq 1$, the logic $\mathrm{CTL}_{i+1}^\star$ has more distinguishing and expressive power than $\mathrm{CTL}_i^\star$; thus the branching-depth hierarchy is strict. We describe equivalence relations H_i that capture $\mathrm{CTL}_i^\star$: two states in a Kripke structure are H_i-equivalent iff they agree on exactly all $\mathrm{CTL}_i^\star$ formulas. While H_1 corresponds to trace equivalence, the limit of the sequence $H_1, H_2, \ldots$ is Milner's bisimulation. These results are not surprising, but they give rise to several interesting observations and problems. In particular, while $\mathrm{CTL}^\star$ and CTL have the same distinguishing power, this is not the case for $\mathrm{CTL}_i^\star$ and CTL_i. We define the branching depth of a structure as the minimal index i for which $H_{i+1} = H_i$. The branching depth indicates on the possibility of using bisimulation instead of trace equivalence (and similarly for simulation and trace containment). We show that while bisimulation can be calculated in polynomial time, the problem of finding the branching depth is PSPACE-complete.

1 Introduction

Temporal logics, which are modal logics geared towards the description of the temporal ordering of events, have been adopted as a powerful tool for specifying and verifying concurrent programs [16]. One of the most significant developments in this area is the discovery of algorithmic methods for verifying temporal logic properties of *finite-state* programs [2,11,17]. This derives its significance both from the fact that many synchronization and communication protocols can be modeled as finite-state programs, as well as from the great ease of use of fully algorithmic methods. Finite-state programs can be modeled by transition systems where each state has a bounded description, and hence can be characterized by a fixed number of Boolean atomic propositions. This means that a finite-state program can be viewed as a finite propositional *Kripke structure* and that its properties can be specified using propositional

This is a preliminary version. The final version can be accessed at
URL: http://www.elsevier.nl/locate/entcs/volume39.html

temporal logic. Thus, to verify the correctness of the program with respect to a desired behavior, one only has to check that the program, modeled as a finite Kripke structure, satisfies (is a model of) the propositional temporal logic formula that specifies that behavior. Hence the name *model checking* for the verification methods derived from this viewpoint [3].

We distinguish between two types of temporal logics: linear and branching [9]. In linear temporal logics, each moment in time has a unique possible future, while in branching temporal logics, each moment in time may split into several possible futures. Linear temporal logic enables us to express properties of paths in Kripke structures. In branching temporal logics, we add the ability to quantify over paths that branch from a state in the structure. In the branching temporal logic CTL*, this is done using the *path quantifiers* E (exists) and A (for all). The CTL* formula $E\varphi$ is satisfied in a state s of a Kripke structure if there exists a path that starts in s and satisfies the formula φ. Similarly, the formula $A\varphi$ is satisfied in a state s of a Kripke structure if all the paths that start in s satisfy the formula φ. The branching temporal logic is CTL, which is a sub logic of CTL* is restricted to formulas in which every temporal quantifier is immediately preceded by a path quantifier.

Sub-logics of CTL* have interesting properties. For example, while the model-checking problem for CTL* is PSPACE-complete, the model-checking problem for CTL can be solved in linear time [18,2]. So, it may be beneficial to explore other sub-logics of CTL* and compare them to CTL and to each other. One way of comparing two logics is by examining their expressive power. We say that two formulas *agree* on a Kripke structure K if K satisfies both formulas or K does not satisfy both of them. Two CTL* formulas are *equivalent* if they agree on all Kripke structures. We say that a logic $\mathcal{L}_1$ has more *expressive power* than another logic $\mathcal{L}_2$ if there exists a formula in $\mathcal{L}_1$ that has no equivalent formula in $\mathcal{L}_2$. For example, it has shown in [5] that CTL* has more expressive power than CTL. [1] The logic $\mathcal{L}_1$ has more *distinguishing power* then $\mathcal{L}_2$ if there exists a formula φ in $\mathcal{L}_1$, and two Kripke structures K_1 and K_2, such that φ distinguishes between K_1 and K_2 (i.e. φ is satisfied by one and only one of the two structures) and there is no formula in $\mathcal{L}_2$ that distinguishes between K_1 and K_2. Clearly if $\mathcal{L}_1$ has more distinguishing power than $\mathcal{L}_2$, then $\mathcal{L}_1$ also has more expressive power than $\mathcal{L}_2$. The opposite direction is not true. For example it is proved in [1] that CTL* and CTL have the same distinguishing power: both logics can distinguish between K_1 and K_2 if and only if K_1 and K_2 are not *bisimilar*.

A bisimulation relation between K_1 and K_2 relates states of K_1 with states of K_2 so that two related states w_1 and w_2 agree on the propositions that hold in them, every successor of w_1 is related to some successor of w_2, and every successor of w_2 is related to some successor of w_1 [14]. Bisimulation is helpful when comparing two systems. Two bisimilar states of the same structure

[1] Note that in our terminology, it may be that both $\mathcal{L}_1$ has more expressive power than $\mathcal{L}_2$ and $\mathcal{L}_2$ has more expressive power than $\mathcal{L}_1$, in which case $\mathcal{L}_2$ and $\mathcal{L}_2$ are incomparable.

can be merged to a single state, leading to smaller and more manageable systems. Also, the *simulation* relation between structures, which releases the third requirement of bisimulation (that is, successors of w_1 in K_1 should be matched by successors of w_2 in K_2, but not vice-versa), is useful to check that an implementation is correct with respect to a specification. Indeed, if an implementation K_1 is simulated by a specification K_2, then K_1 has less behaviors than K_2, thus all its behaviors are allowed, and it is correct.

In this work we examine the sub-logics of CTL* obtained by bounding the nesting depth of the path quantifiers E and A. We define the sequence CTL$^\star_1$, CTL$^\star_2$, CTL$^\star_3$,... of logics, where formulas of CTL$^\star_i$ are allowed to nest at most i path quantifiers. In particular CTL$^\star_1$ coincides with the linear temporal logic LTL. We show that the logics in the sequence constitute a strict distinguishing-power hierarchy: formulas with more nested path quantifiers have more distinguishing power. This implies that the logics also induce a strict expressive-power hierarchy [2]. We define equivalence relations H_i over states of a Kripke structure that correspond to the distinguishing power of CTL$^\star_i$, and describe a polynomial-space algorithm for calculating these relations [3]. In particular, H_1 corresponds to trace equivalence. Milner's bisimulation is then the limit of the sequence $H_1, H_2, \ldots$. We further show that the *stuttering* version of the logics (that is, their restriction to formulas that do not contain the X (next) temporal operator does still retains a strict distinguishing-power hierarchy.

The above results are not too surprising or exciting. They do lead, however, to several interesting observations and problems we discuss and study. Let CTL$_1$, CTL$_2$, CTL$_3$, ... be the sub-logics of CTL defined in a similar fashion. Thus, formulas of CTL$_i$ are allowed to nest at most i path quantifiers. Equivalently, CTL$_i$ = CTL $\cap$ CTL$^\star_i$. Recall that CTL* and CTL have the same distinguishing power. What about CTL$^\star_i$ and CTL$_i$? We show that CTL$^\star_i$ has more distinguishing power than CTL$_i$, for all $i \geq 1$. In fact, we show that CTL$^\star_1$ has more distinguishing power than CTL$_i$, for all $i \geq 1$. There is an interesting phenomenon here: the situation is nice when the H_i relations reach their limit: CTL* and CTL have the same distinguishing power, they both correspond to bisimulation, and bisimulation can be calculated in polynomial time. On the other hand, the situation in the intermediate H_i is less nice: CTL$^\star_i$ and CTL$_i$ have different distinguishing power, and finding the relation H_i induced by CTL$^\star_i$ required polynomial space. For a Kripke structure K, let the *branching depth* of K be the minimal index i such that $H_{i+1} = H_i$. In particular, the branching depth of K is 0 if for all pairs w and w' of states in K, the states w and w' are bisimilar iff they agree on the set of traces that start in them. We use this observation in order to prove that the problem of

[2] In [6], a different expressiveness hierarchy for temporal logics is presented, showing that LTL formulas with more nested Until temporal operators have greater expressive power.

[3] We mention also [?], in which a strict hierarchy of sub-logics of the Hennessy-Milner logic was introduced, where each sub-logic allows the use of one more negation, and a series of simulation relations that corresponds to the sub-logics hierarchy was shown.

finding the branching depth of a Kripke structure is PSPACE-complete. This refutes the hope for a more efficient calculation of the relations H_i.

2 Preliminaries

Formulas of the branching temporal logic CTL* are constructed from some set AP of atomic propositions using the usual boolean operators, the temporal operators X (next time) and U (until), and the path quantifier E (exists). The logic CTL* has two types of formulas: *state formulas* and *path formulas*. A CTL* state formula is either:

- $true, false$ or p, for $p \in AP$.
- $\neg\varphi_1$ or $\varphi_1 \vee \varphi_2$, where φ_1 and φ_2 are CTL* state formulas.
- $E\psi$, where ψ is a CTL* path formula.

A CTL* path formula is either:

- A CTL* state formula.
- $\neg\psi_1, \psi_1 \vee \psi_2, X\psi_1$ or $\psi_1 U \psi_2$, where ψ_1 and ψ_2 are CTL* path formulas.

Note that the syntax we describe does not contain the A path quantifier, which can be expressed by dualizing E; i.e., $A\varphi = \neg E \neg\varphi$. A CTL* formula is a CTL* state formula.

The semantics of CTL* is defined with respect to a Kripke structure $K = (AP, W, W_0, R, L)$, where AP is the set of atomic propositions, W is a set of states, W_0 is a set of initial states, R is a total transition relation, and $L : W \rightarrow 2^{\mathrm{AP}}$ labels each state with the set of atomic propositions that hold in it. We assume that W is finite[4].

When $R(w, w')$, we say that w' is a *successor* of w. A *path* in K is an infinite sequence of states $\pi = w_1, w_2, \ldots$ such that for all $i \geq 1$, we have $R(w_i, w_{i+1})$. The i'th state in the path π is denoted by $\pi(i)$, with $\pi(1)$ being the first state in the path. The prefix of length i of the path π is denoted by $\pi[i] = \pi(1), \pi(2), \ldots, \pi(i)$ and the suffix of π that begins from the i'th state is $\pi^i = \pi(i), \pi(i+1), \ldots$. A *finite path* is a prefix of an infinite path. We use $|\rho|$ to denote the length of a finite path ρ. We extend the labeling function L to paths, thus $L(\pi) = L(\pi(1)) \cdot L(\pi(2)) \cdots$. For a state w, the *trace set* of w, denoted $\mathcal{T}(w)$, is the set of infinite words that corresponds to paths starting at w. Formally, $\mathcal{T}(w) = \{L(\pi) : \pi(1) = w\}$. Note that $\mathcal{T}(w) \subseteq (2^{\mathrm{AP}})^\omega$. For a set $W' \subseteq W$, the trace set of W' is the union of the trace sets of the members of W'. In particular, the *language* of K is $\mathcal{T}(W_0) = \{L(\pi) : \pi(1) \in W_0\}$. Finally, we say that K is *deterministic* if for every states w and set $\sigma \subseteq AP$ of atomic propositions, w has at most one successor w' with $L(w') = \sigma$.

[4] Our results hold also for infinite Kripke structures, only that then, the sequence of relations defined in Definition 3.2 may be infinite, in which case the fixed-point calculation of its limit does not terminate – just as is the case for the standard Milner algorithm for calculating bisimulation [15].

We use $w \models \varphi$ to indicate that a state formula holds at state w in K. Similarly, for a path π in K and a path formula ψ the notation $\pi \models \psi$ indicates that ψ holds along the path π. The relation $\models$ is defined inductively below, with φ_1 and φ_2 being state formulas, ψ_1 and ψ_2 being path formulas, w being a state in K, and π being a path in K.

- $w \models true$ and $w \not\models false$.
- $w \models \neg\varphi_1$ iff $w \not\models \varphi_1$.
- $w \models \varphi_1 \vee \varphi_2$ iff $w \models \varphi_1$ or $w \models \varphi_2$.
- $w \models E\psi_1$ iff there exists a path π such that $\pi(1) = w$ and $\pi \models \psi$.
- $\pi \models \varphi_1$ iff $\pi(1) \models \varphi_1$.
- $\pi \models \neg\psi_1$ iff $\pi \not\models \psi_1$.
- $\pi \models \psi_1 \vee \psi_2$ iff $\pi \models \psi_1$ or $w \models \psi_2$.
- $\pi \models X\psi_1$ iff $\pi^2 \models \psi_1$.
- $\pi \models \psi_1 U \psi_2$ iff there exists a position $j \geq 1$ such that $\pi^j \models \psi_2$ and for all $1 \leq i < j$ we have that $\pi^i \models \psi_1$.

For a Kripke structure K, we say that K satisfies φ, denoted $K \models \varphi$ iff $w_0 \models \varphi$, for all $w_0 \in W_0$.

The logic CTL is a subset of CTL* in which the temporal operators X, U, and their negations are immediately preceded by the path quantifier E. Formally, a state formula in CTL is either:

- $true, false$ or p, for $p \in AP$.
- $\neg\varphi_1$ or $\varphi_1 \vee \varphi_2$, where φ_1 and φ_2 are CTL state formulas.
- $E\psi$, where ψ is a CTL path formula.

A path formula in CTL is either $X\varphi_1, \varphi_1 U \varphi_2, \neg X\varphi_1$, or $\neg\varphi_1 U \varphi_2$, where φ_1 and φ_2 are CTL state formulas.

Consider a Kripke structure $K = (AP, W, W_0, R, L)$. A relation $H \subseteq W \times W$ is a *bisimulation* relation if for all pairs w and w' of states with $H(w, w')$, the following hold:

- $L(w) = L(w')$.
- For every states s such that $R(w, s)$, there exists a state s' such that $R(w', s')$ and $H(s, s')$
- For every states s' such that $R(w', s')$, there exists a state s such that $R(w, s)$ and $H(s, s')$

Every Kripke structure has a maximal bisimulation relation $H_{\max}$ that contains all other bisimulation relations. We say that two states w and w' are bisimilar iff $H_{\max}(w, w')$. It is well known that two states are bisimilar iff they agree on all CTL* formulas [1].

3 Branching Depth

Formulas in CTL* can be classified according to the maximal number of nested path quantifiers they contain. We will see that the more nested path quantifiers we allow, the more expressive power we get. Moreover, given i, we can determine if two states in a Kripke structure agree on all CTL* formulas that have at most i nested path quantifiers.

We define CTL$_i^\star$ to be the sub-logic of CTL* that allows only formulas with at most i path quantifiers. The formal definition is given below.

Definition 3.1 Given a set AP of atomic propositions, we define the logics CTL$_1^\star$, CTL$_2^\star$,... inductively. We begin with the logic CTL$_0^\star$, defined as follows:

- A state formula in CTL$_0^\star$ is $p, \neg\varphi, \varphi \vee \psi, true, false$, where $p \in AP$, and φ and ψ are state formulas in CTL$_0^\star$. Thus a state formula in CTL$_0^\star$ is a boolean combination of atomic propositions.
- A path formula in CTL$_0^\star$ is a state formula in CTL$_0^\star$ or $X\varphi, \varphi U \psi, \neg\varphi, \varphi \vee \psi$, where φ and ψ are path formulas in CTL$_0^\star$.

For $i \geq 0$, we define CTL$_{i+1}^\star$ as follows.

- A CTL$_{i+1}^\star$ state formula is one of the following.
  · a CTL$_i^\star$ state formula,
  · $E\theta$, where θ is a CTL$_i^\star$ path formula.
  · $\neg\varphi$ or $\varphi \vee \psi$, where φ and ψ are CTL$_{i+1}^\star$ state formulas.
- A CTL$_{i+1}^\star$ path formula is one of the following.
  · a CTL$_i^\star$ path formula,
  · a CTL$_{i+1}^\star$ state formula,
  · $\neg\varphi$, $\varphi \vee \psi$, $X\varphi$, or $\varphi U \psi$, where φ and ψ are CTL$_{i+1}^\star$ path formulas.

A CTL$_i^\star$ formula is a CTL$_i^\star$ state formula. Note that for all i, the logic CTL$_{i+1}^\star$ subsumes CTL$_i^\star$. Note also that states formulas of CTL$_0^\star$ are boolean combinations of the linear temporal logic LTL.

Every sub-logic of CTL* induces a natural equivalence relation on the states of a Kripke structure: two states are equivalent if and only if they agree on all the formulas of the sub-logic. Below we define equivalence relations $H_0, H_1, H_2, \ldots$ on states of a Kripke structure. In the following theorems we show that for every i, the relation H_i is the equivalence relation induced by the logic CTL$_i^\star$.

Definition 3.2 Given a Kripke structure $K = (AP, W, W_0, R, L)$, we define the sequence $H_0, H_1, H_2, \ldots$ of relations on the states of K, and the sequence $P_0, P_1, P_2, \ldots$ of relations on finite paths in K.

We start by defining the relation H_0. Two states w and w' are H_0 equivalent if and only if $L(w) = L(w')$. Then, for all $i \geq 0$, we define P_i as follows. Two finite paths ρ and ρ' are P_i equivalent, denoted $P_i(\rho, \rho')$, if and only if $|\rho| = |\rho'|$ and for every $j \geq 1$, we have $H_i(\rho(j), \rho'(j))$. Finally, for all $i \geq 0$, we define H_{i+1} as follows. Two states w and w' are H_{i+1} equivalent, denoted $H_{i+1}(w, w')$, if and only if the following hold:

- For every finite path ρ that starts in w, there exists a finite path ρ' that starts in w' and $P_i(\rho, \rho')$.
- For every finite path ρ' that starts in w', there exists a finite path ρ that starts in w and $P_i(\rho, \rho')$.

We refer to the relations H_i as the *path-quantification state relations of* K, (PQS relations of K, for short) and we refer to the relations P_i as the *path-quantification path relations of* K (PQP relations of K, for short).

Note that $H_{i+1} \subseteq H_i$, and $P_{i+1} \subseteq P_i$. Since W is finite, there must be a fixed point in the sequence of these relations; we call this fixed point H. Obviously, $H = H_j$ for some $j \leq |W|$.

We extend the PQP relations to infinite paths. For $i \geq 0$ and two infinite paths π and π', the relation $P_i(\pi, \pi')$ holds iff for all $j \geq 0$, we have $H_i(\pi(j), \pi'(j))$. The following lemma now follows from the definition of H_i and König's Lemma.

Lemma 3.3 *For all $i \geq 1$, the relation $H_i(w, w')$ holds iff for every path π that starts in w there exists a path π' that starts in w' such that $P_{i-1}(\pi, \pi')$, and for every path π' that starts in w' there exists a path π that starts in w such that $P_{i-1}(\pi, \pi')$.*

Lemma 3.3 gives us a way for calculating the relations H_i. Indeed, the transition from H_i to H_{i+1} is reduced to checking the equivalence of the trace sets of states, with L being extended to atomic propositions that indicate the equivalence classes of P_{i-1}. Since checking equivalence of trace sets can be done in polynomial space [**?**], and since there are at most polynomially many checks to make, finding each H_i can be done in polynomial space.

Example 3.4 Consider the Kripke structures M_1 and N_1 in Figure 3. The language of both structures is $a \cdot b \cdot c^\omega + a \cdot b \cdot d^\omega$. Accordingly, the initial states of M_1 and N_1 are H_1 equivalent. On the other hand, since the state labeled b in M_1 branches to both $b \cdot c^\omega$ and $b \cdot d^\omega$ while the states labeled b in N_1 each branches to either $b \cdot c^\omega$ or $b \cdot d^\omega$, these states, as well as the initial states, are not H_2 equivalent.

Proof. From the definition of H_i we know that for every $j \geq 1$, there exists a finite path ρ'_j of length j that begins in w' such that $P_{i-1}(\pi[j], \rho'_j)$. We define π' by induction, first we define $\pi'(1)$ to be w'. Assume we defined up to $\pi(k)$ and that there is an infinite number of $j \geq k$ such that $\rho'_j[k] = \pi[k]$. Since the number of states in the Kripke structure is finite, then there must be a state s such that for an infinite number of $j \geq k+1$ it holds that $\rho'_j(k+1) = s$ and $\rho'_j[k] = \pi[k]$. Define $\pi'(k+1)$ to be s, we have that there exists an infinite number of $j \geq k+1$ such that $\rho'_j[k+1] = \pi[k+1]$.

Thus π' is defined for all k, all that is left is to prove that $P_{i-1}(\pi, \pi')$. For every k there exists some $j \geq k$ such that $\rho'_j[k] = \pi[k]$. Since ρ'_j and $\pi[j]$ are P_{i-1} equivalent, then $H_{i-1}(\pi(k), \rho'_j(k))$. Since $\rho'_j(k) = \pi(k)$, we have that $H_{i-1}(\pi(k), \pi'(k))$ for all k. So from the definition of P_{i-1} we have that π and π' are P_{i-1} equivalent. □

We now show that the logics $\mathrm{CTL}^{\star}_i$ characterize the distinguishing power of the PQS relations.

Theorem 3.5 *Let $K = (AP, W, W_0, R, L)$ be a Kripke structure, let $H_0, H_1, H_2, \ldots$ be the PQS relations of K, and let $P_0, P_1, P_2, \ldots$ be the PQP relations of K. For all $i \geq 0$, for every state formula φ in $CTL^{\star}_i$, and for every path formula ψ in $CTL^{\star}_i$, the following hold.*

- *If w and w' are two states in K such that $H_i(w, w')$, then $w \models \varphi$ iff $w' \models \varphi$.*
- *If π and π' are two infinite paths in K such that $P_i(\pi, \pi')$, then $\pi \models \psi$ iff $\pi' \models \psi$.*

The proof is not hard, and proceeds by induction on i for both path and state formulas, where the induction step for path formulas is proved by induction on the structure of path formulas in $\mathrm{CTL}^{\star}_{i+1}$.

Proof. The proof proceeds by induction on i for both path and state formulas, where the induction step for path formulas is proved by induction on the structure of path formulas in $\mathrm{CTL}^{\star}_{i+1}$.

The base is $i = 0$, where we need to prove that if states w and w' are such that $H_0(w, w')$, then they agree on all state formulas in $\mathrm{CTL}^{\star}_0$. Since $H_0(w, w')$ iff $L(w) = L(w')$ the base is proved for state formulas.

Assume that we proved the claim up to i. We now wish to prove the claim for state formulas in $\mathrm{CTL}^{\star}_{i+1}$. Let w and w' be two states such that $H_{i+1}(w, w')$, and let φ be a state formula in $\mathrm{CTL}^{\star}_{i+1}$. The formula φ can have in one of the following forms:

- φ is a state formula in $\mathrm{CTL}^{\star}_i$. Since w and w' are H_{i+1} equivalent, and $H_i \subseteq H_{i+1}$, we can deduce that w and w' are also H_i equivalent. So, from the induction hypothesis, they agree on φ.
- $\varphi = E\theta$ for some path formula θ in $\mathrm{CTL}^{\star}_i$. In this, case if $w \models \varphi$ then there exists a path π that begins in w and $\pi \models \theta$. Since $H_{i+1}(w, w')$, there is a path π' that begins in w' and $P_i(\pi, \pi')$. From the induction hypothesis it follows that $\pi' \models \theta$, therefore $w' \models \varphi$. If $w' \models \varphi$ we get that $w \models \varphi$ by similar considerations.
- $\varphi = \neg\psi$ or $\varphi = \psi \vee \theta$ for some state formulas ψ and θ in $\mathrm{CTL}^{\star}_{i+1}$. This case can be easily proved by induction on the structure of φ using the previous cases as the basis for the induction.

All that is left to prove is the base of the induction and induction step for path formulas, we shall prove them together in the following. Assume that we proved the claim up to i, and also assume that we proved the claim for state formulas in $\mathrm{CTL}^\star_{i+1}$. Let φ be a path formula in $\mathrm{CTL}^\star_{i+1}$ and let π and π' be two paths in K such that $P_{i+1}(\pi, \pi')$. We shall use induction on the structure of φ to prove that $\pi \models \psi$ if and only if $\pi' \models \psi$. The base for the induction is given in the following cases:

- φ is a path formula in $\mathrm{CTL}^\star_i$. Obviously $P_i(\pi, \pi')$, so from the induction hypothesis this case is proved.
- φ is a state formula in $\mathrm{CTL}^\star_{i+1}$. Let w be the first state in π and let w' be the first state in π'. From the definition of the relation P_{i+1} it follows that $H_{i+1}(w, w')$, therefore w and w' agree on φ.

For the induction step on the structure of φ we have the following cases:

- $\varphi = X\psi$ for some path formula ψ in $\mathrm{CTL}^\star_{i+1}$. Obviously the paths π^2 and π'^2 are P_{i+1} equivalent. From the induction hypothesis on the structure of φ we have that $\pi^2 \models \psi$ iff $\pi'^2 \models \psi$. Since $\pi \models \varphi$ iff $\pi^2 \models \psi$ and since $\pi' \models \varphi$ iff $\pi^2 \models \psi$, then this case is proved.
- $\varphi = \psi U \theta$ with ψ and θ being path formulas in $\mathrm{CTL}^\star_{i+1}$. Assume that $\pi \models \varphi$, then there exists a number $j \geq 1$ such that $\pi^j \models \theta$ and for all $1 \leq l < j$ we have that $\pi^l \models \psi$. Since for all n we have that $P_{i+1}(\pi^n, \pi'^n)$, then from the induction hypothesis on the structure of φ we get that for all $l < j$ $\pi'^l \models \psi$ and $\pi'^j \models \theta$ therefore $\pi' \models \varphi$. If we assume that $\pi' \models \varphi$ we get that $\pi \models \varphi$ in a similar fashion.
- $\varphi = \neg\psi$ or $\varphi = \psi \vee \theta$ for some path formulas ψ and θ in $\mathrm{CTL}^\star_{i+1}$. The proof for this case is obvious.

□

We now prove the opposite direction; i.e., that if two states agree on all $\mathrm{CTL}^\star_i$ formulas then they are H_i equivalent.

Theorem 3.6 *Let $K = (AP, W, W_0, R, L)$ be a Kripke structure, let $H_0, H_1, H_2, \ldots$ be the PQS relations of K, and let $P_0, P_1, P_2, \ldots$ be the PQP relations of K. Then, for all $i \geq 0$, the following hold.*

- *If w and w' are two states in K that are not H_i equivalent then there exists a $CTL^\star_i$ state formula that distinguishes between them.*
- *If π and π' are two paths in K that are not P_i equivalent then there exists a $CTL^\star_i$ path formula that distinguishes between them.*

Proof. As before we prove this theorem by induction on i. For the base $i = 0$ of the induction, let w and w' be two states in K that are not H_0 equivalent, therefore $L(w) \neq L(w')$. Without loss of generality we can assume that there exists an atomic proposition $p \in L(w)$ such that $p \notin L(w')$. Thus the formula p distinguishes between w and w'.

Assume that we proved the claim for i. First we prove the induction step for states, let w and w' be two states that are not H_{i+1} equivalent. Without loss of generality we can assume that there is a finite path ρ of length n that begins in w that has no P_i equivalent path that begins in w'. Let $\rho'_1, \ldots, \rho'_m$ be all the finite paths of length n that begin in w'. Since ρ and ρ'_j are not P_i equivalent, then for every $1 \leq j \leq m$ there exists a position k_j such that the states $\rho(k_j)$ and $\rho'_j(k_j)$ are not H_i equivalent. From the induction hypothesis we have that there exists a CTL$_i^\star$ state formula φ_j that distinguishes between $\rho(k_j)$ and $\rho'_j(k_j)$, without loss of generality we can assume that $\rho(k_j) \models \varphi_j$ and $\rho'_j(k_j) \models \neg\varphi_j$. So the CTL$_i^\star$ path formula $\psi_j = X^{k_j-1}p$ distinguishes between ρ and ρ'_j. The formula $\varphi = E \bigwedge_{j=1,..,m} \psi_j$ is a CTL$_{i+1}^\star$ state formula that distinguishes between w and w'.

All that is left to prove is the induction step for paths. Let π and π' be two paths in K that are not P_{i+1} equivalent. Then there exists a position j such that $\pi(j)$ and $\pi'(j)$ are not H_{i+1} equivalent. From what we proved above we have the CTL$_{i+1}^\star$ state formula φ that distinguishes between $\pi(j)$ and $\pi'(j)$. So the CTL$_{i+1}^\star$ path formula $\psi = X^{j-1}\varphi$ distinguishes between π and π'. □

Note that in the proof of the last theorem the only temporal operator we used is the X operator. The results of this section can be summed up in the following corollaries.

Corollary 3.7 *Given a Kripke structure K, let $H_0, H_1, H_2, \ldots$ be the PQS relations of K. Two states in K are H_i equivalent if and only if they agree on all the state formulas in CTL$_i^\star$.*

Corollary 3.8 *Two states in a Kripke structure are H equivalent if and only if they agree on all CTL* formulas.*

It follows that the limit H of $H_0, H_1, \ldots$ is the bisimulation relation. Note that the intermediate PQS relations H_i are different from both the *observation equivalence* relations and the *limited observation equivalence* relations studied in [15,8], although all three relations converge to the bisimulation relation. The observation equivalence relations places a much weaker requirement on the paths than the PQS relations. In the limited observation equivalence relations (called *strong equivalence* in [15]) the i'th iteration involves a "look ahead" of at most i states, in the PQS relations the i'th iteration involves at most i branches.

We now show that formulas with more nesting depth have greater distinguishing and expressive power. Thus, the expressiveness hierarchy they induce is strict.

Theorem 3.9 *Consider the logics CTL$_0^\star$, CTL$_1^\star$, ... For all $i \geq 0$, the logic CTL$_{i+1}^\star$ is more distinguishing and more expressive than CTL$_i^\star$.*

Proof. Clearly, CTL$_1^\star$ is more expressive than CTL$_0^\star$, which can just specify propositional assertions. Consider the sequences $M_1, M_2, \ldots$ and $N_1, N_2, \ldots$ of Kripke structures presented in Figure 3. By Definition 3.2, it is not hard to see

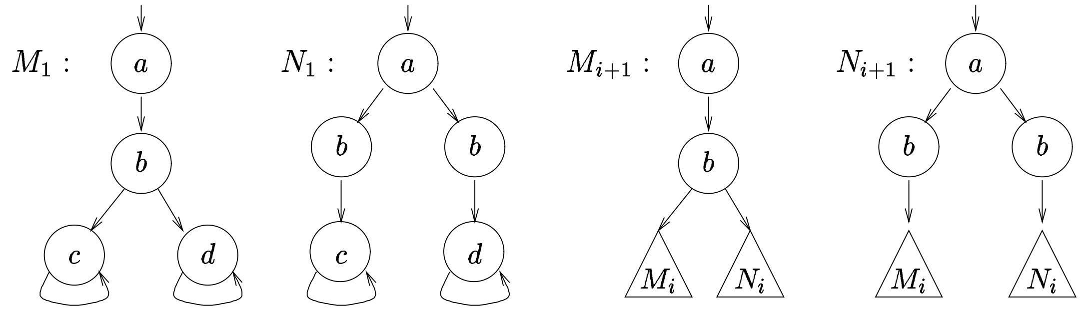

Fig. 1. The sequences $M_1, M_2, \ldots$ and $N_2, N_2, \ldots$

that the initial states of M_i and N_i are H_i equivalent. Hence, by Corollary 3.7, no CTL$_i^\star$ formula can distinguish between M_i and N_i. We describe a CTL$_{i+1}^\star$ formula that distinguishes between the states. Let $\psi_1 = X(EXc \wedge EXd)$, and let $\psi_{i+1} = X(EX\psi_i \wedge EX\neg\psi_i)$. The formula ψ_i is a CTL$_i^\star$ path formula. Now, the CTL$_{i+1}^\star$ formula $\varphi_{i+1} = E\psi_i$ is such that $M_i \models \varphi_{i+1}$ and $M_i \not\models \varphi_{i+1}$. It follows that CTL$_{i+1}^\star$ is more distinguishing, and hence also more expressive, than CTL$_i^\star$. □

4 Branching Depth with Stuttering

Another relation that we can explore in the same manner is the stuttering relation. Intuitively, stuttering equivalence of two infinite words means that the letters in these words appear in the same order, ignoring repetitions of the same letter. In this section we extend the results of the previous section to *stuttering*.

A *partition* of a path π is an infinite sequence $B_1, B_2, \ldots$ of nonempty disjoint finite sets of states such that $\pi(1) \in B_1$ and $\pi(i+1) \in B_j$ iff $\pi(i) \in B_j$ or $\pi(i) \in B_{j-1}$. We give here the definition for equivalence with respect to stuttering.

Definition 4.1 For a Kripke structure $K = (AP, W, W_0, R, L)$, define the following relations $H_0^{\text{stut}}, H_1^{\text{stut}}, \ldots \subseteq W \times W$ inductively as follows.

- $H_0^{\text{stut}}(w, w')$ iff $L(w) = L(w')$.
- $H_{n+1}^{\text{stut}}(w, w')$ iff:

(i) For every path π in K with $\pi(1) = w$, there exist a path π' in K with $\pi'(1) = w'$, a partition $B_1, B_2, \ldots$ of π, and a partition $B'_1, B'_2, \ldots$ of π', such that for all $j > 0$, $s \in B_j$, and $s' \in B'_j$, we have $H_n^{\text{stut}}(s, s')$.

(ii) For every path π' in K with $\pi'(1) = w'$, there exist a path π in K with $\pi(1) = w$, a partition $B_1, B_2, \ldots$ of π, and a partition $B'_1, B'_2, \ldots$ of π', such that for all $j > 0$, $s \in B_j$, and $s' \in B'_j$, we have $H_n^{\text{stut}}(s, s')$.

The relation $H^{\text{stut}} \subseteq W \times W'$ is defined as follows: $H^{\text{stut}}(w, w')$ iff $H_i^{\text{stut}}(w, w')$ for all i. Finally, we say that two states w and w' are *equivalent with respect to* *stuttering* if $H^{\text{stut}}(w, w')$. Note that each relation H^{stut}_{i+1} is contained in H^{stut}_{i}, therefore there exists a fixed point in the sequence of these relations. Clearly this fixed point is equal to the relation H^{stut} defined above.

Our definition is the expected extension of Definition 3.2 to the stuttering case. Interestingly, the same definition appears in [1] as the definition of stuttering bisimulation with no reference to its branching-depth structure. So, while for the stuttering case, the natural definition for bisimulation is the fixed point of the stuttering PQS relations, for the non-stuttering case, the natural definition is the local one, of [15], which is simpler than the fixed point of the PQS relations.

For a logic D let $D \setminus \{X\}$ denote the set of all formulas ψ in D such that ψ does not contain the X quantifier. In particular, we refer to the logics LTL $\setminus\{X\}$, $\text{CTL}^\star_1\setminus\{X\}$, $\text{CTL}^\star_2\setminus\{X\}$, ...

Consider the logics $\text{CTL}^\star_i\setminus\{X\}$ and the relations H^{stut}_i above. Using the same considerations as in the previous section, one can prove the following theorems.

Theorem 4.2 *Consider a Kripke structure $K = (AP, W, W_0, R, L)$ and the equivalence relations $H^{\text{stut}}_0, H^{\text{stut}}_1, \ldots$ on the states of K. For every $i > 0$, two states in K are H^{stut}_i equivalent if and only if they agree on all the state formulas in $CTL^\star_i\setminus\{X\}$.*

Note that while in the proof of Theorem 3.6 we use only the X temporal operator, in Theorem 4.2 the only temporal operator we use is the U operator. The limit case of Theorem 4.2 is proved in [1], and our results here provide an alternative proof, based on the observation that H^{stut} is the limit of the relations H^{stut}_i.

Theorem 4.3 [1] *Two states in a Kripke structure are H^{stut} equivalent if and only if they agree on all $CTL^\star \setminus\{X\}$ formulas.*

5 Branching Depth in CTL

In this section, we examine the restriction the logics $\text{CTL}^\star_i$, defined in Section 3, to CTL. Formally, we define the sequence CTL_1, CTL_2, CTL_3, ... of logics, where $\text{CTL}_i = \text{CTL} \cap \text{CTL}^\star_i$. We show that while $\text{CTL}^\star$ and CTL have the same distinguishing power, $\text{CTL}^\star_i$ has more distinguishing power then CTL_i, for all $i \geq 1$.

In [5] it was proved that $\text{CTL}^\star$ has more expressive power than CTL. Emerson and Halpern proved that the formula $\varphi = AF(p \wedge Xp)$ has no equivalent CTL formula. This was done by defining two sequences $M_1, M_2, M_3, \ldots$ and $N_1, N_2, N_3, \ldots$ of Kripke structures for which the following hold:

(i) For all i, $M_i \models \varphi$ and $N_i \not\models \varphi$.

(ii) If ψ is a CTL formula of length at most i, then $M_i \models \psi$ iff $N_i \models \psi$.

It follows that no CTL formula is equivalent to φ; thus CTL* has more expressive power than CTL. A careful analysis of the inductive argument in proof in [5] shows that Emerson and Halpern actually prove a stronger claim:

Lemma 5.1 *If ψ is a CTL$_i$ formula, then $M_i \models \psi$ iff $N_i \models \psi$.*

Now, since the CTL$_1^\star$ formula $\varphi' = \neg EG(\neg(p \wedge Xp))$ is equivalent to φ, it follows that for all i, CTL$_i$ has no formula equivalent to φ'. Thus, CTL$_1^\star$ has more distinguishing and expressive power than CTL$_i$, for all i.

6 The Branching Depth of Kripke Structures

For a Kripke structure K, let the *branching depth* of K be the minimal index i such that there are CTL$_i^\star$ formulas that distinguish between states in K but no CTL$_{i-1}$ formula can distinguish between states in K. In other words, the PQS relations that correspond to K are such that $H_{i+1} = H_i$. In particular, the branching depth of K is 0 if for all pairs w and w' of states in K, the states w and w' are bisimilar iff they agree on the set of traces that start in them. Since bisimulation can be checked in polynomial time, having branching level 0 is a very helpful property of a structure [5].

In [7], Grumberg and Kurshan introduce the notion of *equi-linearity* for CTL* formulas. A CTL* formula ψ is equi-linear if it cannot distinguish between states with the same trace sets. For example, the CTL* formula $AGAFp$ has no equivalent LTL formula, but is equi-linear (with respect to finite Kripke structures). Our definition of branching depth can be viewed as the structural analogue. Accordingly, we say that a Kripke structure is equi-linear iff it has branching depth 0. Note that a structure is equi-linear if all CTL* formulas cannot distinguish states with the same trace set.

Lemma 6.1 *All deterministic structures are equi-linear.*

Proof. Follows immediately from the fact that for deterministic structures bisimulation coincides with trace equivalence. □

Deciding whether a Kripke structure is deterministic can be done in polynomial time. On the other hand, as we show below, deciding whether the structure is equi-linear is much harder.

Theorem 6.2 *Deciding whether a Kripke structure is equi-linear is PSPACE-complete.*

[5] The computational advantage is so compelling as to make simulation useful also to researchers that favor the linear approach to specification: in automatic verification, simulation is widely used as a sufficient condition for trace containment [4]; in manual verification, trace containment is most naturally proved by exhibiting local witnesses such as simulation relations or refinement mappings (a restricted form of simulation relations) [10,13,12].

Proof. Consider a Kripke structure $K = (AP, W, W_0, R, L)$. Since bisimulation implies trace equivalence, we only have to check that for every two states w and w' with $\mathcal{T}_K(w) = \mathcal{T}_K(w')$, the states w and w' are bisimilar. Let $H_{\max}$ be the maximal bisimulation of K. We can find $H_{\max}$ in polynomial time. For a given pair $\langle w, w' \rangle$, deciding whether $\mathcal{T}_K(w) = \mathcal{T}_K(w')$ can be done in polynomial space. So, a naive algorithm that checks whether all the pairs $\langle w, w' \rangle$ not in $H_{\max}$ are such that $\mathcal{T}_K(w) \neq \mathcal{T}_K(w')$, requires polynomial space.

For the lower bound, we first claim that the problem of deciding whether a Kripke structure contains two states w and w' such that $\mathcal{T}_K(w) = \mathcal{T}_K(w')$ is PSPACE-hard (note that this is different than the problem of deciding whether $\mathcal{T}_K(w) = \mathcal{T}_K(w')$, for given w and w', which is known to be PSPACE-hard). To see this, recall that the problem of deciding whether a Kripke structure is universal (that is, $\mathcal{T}(W_0) = \Sigma^\omega$) is PSPACE-hard. The proof described in [19] reduces the problem of deciding whether a deterministic polynomial space Turing machine T accepts an input word x to the universality problem by defining a Kripke structure $K_{T,x}$ such that the trace set of the initial set of $K_{T,x}$ contains all words that do not encode a legal computation of T on x or encode a legal rejecting computation. The initial set of $K_{T,x}$ is then universal iff T rejects x. The structure $K_{T,x}$ contains an accepting sink (that is, a clique with $2^{|\mathrm{AP}|}$ states, corresponding to all the possible subsets of AP). Once $K_{T,x}$ observes that a word does not encode a legal computation or that a rejecting computation has been reached, it goes to the accepting sink. The structure $K_{T,x}$ can be defined so that all states w and w' have different trace sets, except possibly for an initial state w_σ and a state w'_σ in the accepting sink, both labeled with σ. In fact, $K_{T,x}$ is universal iff for all $\sigma \subseteq AP$, there is an initial state w_σ labeled σ such that w_σ is universal (that is, $L(w_\sigma) = \sigma \cdot (2^{\mathrm{AP}})^\omega$). So, if we add to $K_{T,x}$ two states s and s', labeled with the same letter, such that s has as successors exactly all the initial states and s' has as successors exactly all the states in the accepting sink, we have that T rejects x iff the obtained structure contains two states with the same trace sets. Hence, the latter problem is PSPACE-hard.

We now do a reduction from the problem of deciding whether a Kripke structure contains two states with the same trace set to the problem of deciding whether the structure is equi-linear. Given a Kripke structure K with state space $\{w_1, \ldots, w_n\}$, let $K_1, \ldots, K_n$ be Kripke structures over a set AP' of atomic propositions for which $AP \cap AP' = \emptyset$, such that all K_i's have the same language but are pairwise not bisimilar (that is, for all i and j, K_i and K_j are not bisimilar). We can define K_i with $O(n)$ states and a single initial state (for example, using the same considerations as in the structures in Figure 3). Now, let K' be K where each state w_i also has a branch to the initial state of K_i. Since no two states in K' are bisimilar, it follows that K' is equi-linear iff no two states in K' have the same trace set, and we are done. □

We use Theorem 6.2 in order to prove that the problem of finding the branching depth of a Kripke structure is PSPACE-complete. This refutes the hope for a more efficient calculation of quantities related to the relations H_i.

Theorem 6.3 *The problem of finding the branching depth of a Kripke structure is PSPACE-complete.*

Proof. Consider a Kripke structure $K = (AP, W, W_0, R, L)$. In order to find the branching depth of K, we can calculate the PQS relations H_i of K. The branching depth of K is the fixed point. Since the transitions from H_i to H_{i+1} can be done in polynomial space, and a fixed-point is obtained within at most $|W|$ iterations, membership in PSPACE follows.

For the lower bound, we do a reduction from the problem of deciding whether K is equi-linear. Indeed, K is equi-linear iff its branching depth is 0. □

References

[1] Browne, M., E. Clarke and O. Grumberg, *Characterizing finite Kripke structures in propositional temporal logic*, Theoretical Computer Science **59** (1988), pp. 115–131.

[2] Clarke, E., E. Emerson and A. Sistla, *Automatic verification of finite-state concurrent systems using temporal logic specifications*, ACM Transactions on Programming Languages and Systems **8** (1986), pp. 244–263.

[3] Clarke, E., O. Grumberg and D. Peled, "Model Checking," MIT Press, 1999.

[4] Cleaveland, R., J. Parrow and B. Steffen, *The concurrency workbench: A semantics-based tool for the verification of concurrent systems*, ACM Trans. on Programming Languages and Systems **15** (1993), pp. 36–72.

[5] Emerson, E. and J. Halpern, *Sometimes and not never revisited: On branching versus linear time*, Journal of the ACM **33** (1986), pp. 151–178.

[6] Etessami, K. and T. Wilke, *An until hierarchy for temporal logic*, in: *Proc. 11th IEEE Symposium on Logic in Computer Science*, DIMACS, 1996, pp. 108–117.

[7] Grumberg, O. and R. Kurshan, *How linear can branching-time be*, in: *Proc. First International Conference on Temporal Logic*, Lecture Notes in Artificial Intelligence **827** (1994), pp. 180–194.

[8] Kanellakis, P. and S. Smolka, *CCS expressions, finite state processes and three problems of equivalence*, in: *Proc. Second ACM Symposium on Principles of Distributed Computing*, Montreal, Quebec, 1983, pp. 228–240.

[9] Lamport, L., *Sometimes is sometimes "not never" - on the temporal logic of programs*, in: *Proc. 7th ACM Symposium on Principles of Programming Languages*, 1980, pp. 174–185.

[10] Lamport, L., *Specifying concurrent program modules*, ACM Trans. on Programming Languages and Systenms **5** (1983), pp. 190–222.

[11] Lichtenstein, O. and A. Pnueli, *Checking that finite state concurrent programs satisfy their linear specification*, in: *Proc. 12th ACM Symposium on Principles of Programming Languages*, New Orleans, 1985, pp. 97–107.

[12] Lynch, N., "Distributed algorithms," Morgan Kaufmann, 1996.

[13] Lynch, N. A. and M. Tuttle, *Hierarchical correctness proofs for distributed algorithms*, in: *Proc. 6th ACM Symp. on Principles of Distributed Computing*, 1987, pp. 137–151.

[14] Milner, R., *An algebraic definition of simulation between programs*, in: *Proc. 2nd International Joint Conference on Artificial Intelligence* (1971), pp. 481–489.

[15] Milner, R., "A Calculus of Communicating Systems," Lecture Notes in Computer Science **92**, Springer Verlag, Berlin, 1980.

[16] Pnueli, A., *The temporal semantics of concurrent programs*, Theoretical Computer Science **13** (1981), pp. 45–60.

[17] Queille, J. and J. Sifakis, *Specification and verification of concurrent systems in Cesar*, in: *Proc. 5th International Symp. on Programming*, Lecture Notes in Computer Science **137** (1981), pp. 337–351.

[18] Sistla, A. and E. Clarke, *The complexity of propositional linear temporal logic*, Journal ACM **32** (1985), pp. 733–749.

[19] Wolper, P., "Synthesis of Communicating Processes from Temporal Logic Specifications," Ph.D. thesis, Stanford University (1982).