Flat fragments of CTL and CTL* : separating the expressive and distinguishing powers

Eindhoven University of Technology
Department of Mathematics and Computing Science

Flat Fragments of CTL and CTL*: Separating the Expressive and Distinguishing Powers

by

Dennis Dams

98/05

ISSN 0926-4515

Reports are available at:
http://www.win.tue.nl/win/cs

Computing Science Reports 98/05
Eindhoven, March 1998

Flat Fragments of CTL and CTL*:
Separating the Expressive and Distinguishing Powers

Dennis Dams*

Abstract

We study both the expressive and the distinguishing powers of flat temporal logics. These are fragments obtained by restricting the first argument of the Until operator to propositions. Both the linear and the braching-time cases are considered.

1 Introduction

Temporal logic lies at the basis of several specification formalisms that are widely used in practice. For a large part, this acceptance stems from the availability of software tools for automated verification, that allow to prove or disprove the satisfaction of a temporal property interpreted over a modelling of the system under consideration. *Model checking* is such an approach, that has proven successful in the debugging and verification of hardware circuitry and communication protocols for example. Being based on an exhaustive inspection of the state space of the model, the scalability of model checking is limited, which is referred to as the *state explosion problem*. One way to alleviate this problem is to "collapse" the model by identifying states that are indistinguishable through properties expressed in the temporal logic being used for specification. Obviously, the lower the distinguishing power of the logic, the better reductions can be achieved. On the other hand, the expressivity of the logic should not be compomised too much.

In this article we investigate the expressive and distinguishing powers of a number of variations on CTL (Computation Tree Logic, see [CES86]). The *Until* modality is used to express that some property φ has to remain true until property ψ occurs, where φ and ψ can be arbitrary temporal formulae again, expressing properties about sequences. We consider a restriction where φ is limited to a proposition stating a property about single states only — the resulting logics are called *flat*. This fragment is of interest because it is indeed being used in the practice of specification. For example, *timing diagrams*, part of a visual specification formalism ([DJS95]), are automatically translated into temporal logic prior to model checking, and it can be shown that the resulting formulae are always flat.

The issue of expressivity is first studied for the case of the *linear-time* temporal logic LTL. We show in Section 2 that by flattening LTL, its expressive power decreases. In Section 3 we turn to variations of the branching-time logics CTL* and CTL. The result on linear-time expressivity is shown to carry over to the branching-time cases. Then, we investigate the distinguishing powers of flat versions of CTL* and CTL by linking them to "adequate" *behavioural equivalences*. These equivalences are then

*Dept. of Math. & Comp. Science, Eindhoven University of Technology, PO Box 513, 5600 MB Eindhoven. E-mail: `wsindd@win.tue.nl`. Web: `http://www.win.tue.nl/win/cs/fm/Dennis.Dams/`. Part of this research was carried out during the author's participation in the *Special Year on Logics and Algorithms*, sponsored by the Center for Discrete Mathematics & Theoretical Computer Science (DIMACS) at Rutgers University, NJ, during the summer of 1996.

compared to each other and to those induced by the non-flat versions of the logics. Section 4 concludes. Some proofs in this article have been moved into the appendix.

Comparative expressivity of CTL-like temporal logics is studied in, among others, [EH86, GK94, EW96]. Behavioural equivalences ([DN87]) induced by temporal logics are the subject of [HM80, BCG88, Sti89, Jos90, DNV90, BFG$^+$91, GKP92, vBvES94, Dam96].

2 Flat Linear-time Temporal Logic: Expressivity

Throughout this article, we assume given a nonempty set Prop of *propositions*.

2.0.1 DEFINITION *The logic* LTL *is the set of formulae* φ *defined inductively by the following grammar, where* $p \in$ Prop.

$$\varphi \; := \; p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\varphi \mid \mathsf{U}(\varphi, \varphi).$$

LTL(U) *is the fragment of* LTL *obtained by disallowing the use of the Next operator* X. flatLTL(U) *is obtained by restricting the first argument of the* U *operator to a local formula, i.e. a boolean combination of propositions. The abbreviations true, false,* $\vee$, $\rightarrow$, *etc. are defined as usual.*

Given a formula $\varphi = \mathsf{U}(\varphi_1, \varphi_2)$, φ_2 *is sometimes referred to as the eventuality of* φ, *and* φ_1 *as its initial invariant.*

2.0.2 DEFINITION *For* $\varphi \in$ LTL, $Udepth(\varphi)$ *is the maximal number of nested* U *operators in* φ. *I.e.* $Udepth(p) = 0$ *for* $p \in$ Prop, $Udepth(\neg\varphi) = Udepth(\mathsf{X}\varphi) = Udepth(\varphi)$, $Udepth(\varphi_1 \wedge \varphi_2) = \max(Udepth(\varphi_1), Udepth(\varphi_2))$, *and* $Udepth(\mathsf{U}(\varphi_1, \varphi_2)) = 1 + \max(Udepth(\varphi_1), Udepth(\varphi_2))$.

Let Σ be a set of *states* and $\mathcal{L} : \Sigma \rightarrow \mathcal{P}(\text{Prop})$ a *labelling function* indicating which propositions hold in each state. For a sequence $X = s_0 s_1 \cdots$ of states, $X(k)$ denotes the state s_k and $X(k, \ldots)$ denotes the sequence $s_k s_{k+1} \cdots$. We sometimes identify states with one-element sequences. Note that, e.g., $X(k, \ldots)(i) = X(k+i)$ and $X(k, \ldots)(i, \ldots) = X(k+i, \ldots)$. If X is finite and Y is also a sequence of states, then XY denotes their concatenation. Parentheses are used to disambiguate expressions like $XY(k)$. $|X|$ denotes the length of X, so e.g. $(AA)(|A|) = A(0)$.

We interpret LTL formulae over (single) infinite sequences of states, as follows.

2.0.3 DEFINITION *Let* A *be an infinite sequence of states,* $p \in$ Prop, *and* $\varphi \in$ LTL.

1. $A \models p$ *iff* $p \in \mathcal{L}(A(0))$.
2. $A \models \neg\varphi$ *iff* $A \not\models \varphi$.
3. $A \models \varphi_1 \wedge \varphi_2$ *iff* $A \models \varphi_1$ *and* $A \models \varphi_2$.
4. $A \models \mathsf{X}\varphi$ *iff* $A(1, \ldots) \models \varphi$.
5. $A \models \mathsf{U}(\varphi_1, \varphi_2)$ *iff* $\exists_{i \geq 0}\, A(i, \ldots) \models \varphi_2$ *and* $\forall_{0 \leq j < i}\, A(j, \ldots) \models \varphi_1$.

In this section we focus on LTL(U)[1]. While the Next operator provides the power to precisely count states in a sequence, the Until operator alone allows to count certain patterns. The following lemma states that every pattern counted requires an additional Until: it is not possible to distinguish numbers of patterns that exceed the Until depth. In this lemma, the patterns are specified by a finite sequence A, and the counting formula is restricted to flatLTL(U).

[1]See [Lam83] for a plea against the Next operator from the point of view of specification of systems.

2.0.4 LEMMA *Let A be a finite sequence of states and X an infinite sequence of states. Let $\varphi \in$ flatLTL(U). Then for every $h > Udepth(\varphi)$, and every $0 \leq k < |A|$, we have $(A^h X)(k, \dots) \models \varphi \Leftrightarrow (AA^h X)(k, \dots) \models \varphi$.*

PROOF. We prove the following, equivalent, fact: for every $h > Udepth(\varphi)$, and every $0 \leq k < |A|$, we have $(A^h AX)(k, \dots) \models \varphi \Leftrightarrow (A^h AX)(k + |A|, \dots) \models \varphi$.

By induction on the structure of φ. Consider the base case, where $\varphi \in$ Prop and hence $Udepth(\varphi) = 0$. Let $h > 0$ and $0 \leq k < |A|$. Clearly, the valuation of φ over the sequences $(A^h AX)(k, \dots)$ and $(A^h AX)(k + |A|, \dots)$ only depends on the (propositions labelling the) states $(A^h AX)(k)$ and $(A^h AX)(k + |A|)$ respectively. Because $h > 0$ and $0 \leq k \leq |A|$, we have $(A^h AX)(k) = (A^h AX)(k + |A|)$ — namely, both states are equal to $A(k)$—, and therefore $(A^h AX)(k, \dots) \models \varphi$ iff $(A^h AX)(k + |A|, \dots) \models \varphi$.

Next, we consider the induction step. We concentrate on the case that $\varphi = \mathsf{U}(q, \varphi')$; the other cases are straightforward (recall that $\varphi \in$ flatLTL(U) and hence cannot contain X's). Let $h > Udepth(\varphi)$ (so $h \geq 2$ and $0 \leq k < |A|$). We first prove the $\Rightarrow$ direction of the biimplication. Assume that $(A^h AX)(k, \dots) \models \varphi$. By definition of satisfaction, this means that we can choose $l \geq 0$ such that $(A^h AX)(k, \dots)(l, \dots) \models \varphi'$ and for every $0 \leq i < l$, $(A^h AX)(k, \dots)(i) \models q$. We consider the following 3 cases, distinguishing whether the eventuality φ' is fulfilled in some state of the first A, the second A, or later.

1. $k + l < |A|$. Then by the i.h., $(A^h AX)(k + l + |A|, \dots) \models \varphi'$, i.e. $(A^h AX)(k + |A|, \dots)(l, \dots) \models \varphi'$. By $k + l < |A|$ and the fact that $h \geq 1$, it easily follows that $(A^h AX)(k + |A| + i) = (A^h AX)(k + i)$ for every $0 \leq i < l$. Therefore we also have $(A^h AX)(k + |A|, \dots)(i) \models q$ for every $0 \leq i < l$. So $(A^h AX)(k + |A|, \dots) \models \varphi$.

2. $|A| \leq k + l < |AA|$. Then $(A^{h-1} AX)(k + l - |A|, \dots) \models \varphi'$ and therefore, by the i.h. (note that $h - 1 > Udepth(\varphi')$), $(A^{h-1} AX)(k + l, \dots) \models \varphi'$, hence $(A^h AX)(k + |A|, \dots)(l, \dots) \models \varphi'$. By $k+l < |AA|$ and the fact that $h \geq 2$, it easily follows that $(A^h AX)(k+|A|+i) = (A^h AX)(k+i)$ for every $0 \leq i < l$. Therefore we have $(A^h AX)(k + |A|, \dots)(i) \models q$ for those i. So $(A^h AX)(k + |A|, \dots) \models \varphi$.

3. $|AA| \leq k+l$. As $0 \leq k < |A|$, we have $l \geq |A|$. Therefore, we can choose $l' \geq 0$ such that $(A^h AX)(k + |A|, \dots)(l', \dots) \models \varphi'$, namely $l' := l-|A|$. From the fact that for every $0 \leq i < l$, $(A^h AX)(k, \dots)(i) \models q$ holds, it follows directly that for every $0 \leq i < l'$, $(A^h AX)(k + |A|, \dots)(i) \models q$ holds. We conclude that $(A^h AX)(k + |A|, \dots) \models \varphi$.

We proceed with the $\Leftarrow$ direction. Assume that $(A^h AX)(k + |A|, \dots) \models \varphi$. By definition of satisfaction, this means that we can choose $l \geq 0$ such that $(A^h AX)(k + |A|, \dots)(l, \dots) \models \varphi'$ and for every $0 \leq i < l$, $(A^h AX)(k + |A|, \dots)(i, \dots) \models q$. We consider the following 3 cases, distinguishing whether the eventuality is fulfilled in some state of the second A, the third A, or later.

1. $k+|A|+l < |AA|$. Then by the i.h., $(A^h AX)(k+l, \dots) \models \varphi'$, i.e. $(A^h AX)(k, \dots)(l, \dots) \models \varphi'$. Because $k + l < |A|$ and $h \geq 1$, it easily follows that $(A^h AX)(k + i) = (A^h AX)(k + |A| + i)$ for every $0 \leq i < l$. Therefore we also have $(A^h AX)(k, \dots)(i) \models q$ for every $0 \leq i < l$. So $(A^h AX)(k, \dots) \models \varphi$.

2. $|AA| \leq k + |A| + l < |AAA|$. Then $(A^{h-1} AX)(k + l, \dots) \models \varphi'$ and therefore, by the i.h. (note that $h - 1 > Udepth(\varphi')$), $(A^{h-1} AX)(k + l - |A|, \dots) \models \varphi'$, hence $(A^h AX)(k, \dots)(l, \dots) \models \varphi'$. Because $k+l < |AA|$ and $h \geq 2$, it easily follows that $(A^h AX)(k+i) = (A^h AX)(k+|A|+i)$ for every $0 \leq i < l$. Therefore we have $(A^h AX)(k, \dots)(i) \models q$ for those i. So $(A^h AX)(k, \dots) \models \varphi$.

3. $|AAA| \leq k+|A|+l$. We can choose $l' \geq 0$ such that $(A^h AX)(k, \dots)(l', \dots) \models \varphi'$, namely $l' := l+|A|$. From the fact that for every $0 \leq i < l$, $(A^h AX)(k + |A|, \dots)(i) \models q$ holds, it follows directly that for every $|A| \leq i < l'$, $(A^h AX)(k, \dots)(i) \models q$ holds. Furthermore, for $0 \leq i < |A|$, every state $(A^h AX)(k+i)$ is equal to the state $(A^h AX)(k+|A|+i)$ and as q holds for each of the latter, it also holds for the former states. We conclude that $(A^h AX)(k, \dots) \models \varphi$. □

2.1 Flattening decreases expressivity

Do we lose expressive power by flattening LTL(U)? In this section this is answered affirmatively by exposing an LTL(U) formula ψ that has no flat equivalent. We start by arguing why there are no simpler such witness formulae — this gives an impression how much expressivity is lost.

The simplest candidate is the formula $\mathsf{U}(\mathsf{U}(p, q), r)$. However, this formula may easily be rewritten into a flat equivalent using the following property.

2.1.1 PROPERTY *Let $\varphi_1, \varphi_2 \in$ LTL. The formula $\mathsf{U}(\varphi_1, \varphi_2)$ is semantically equivalent to $\mathsf{U}(true, \varphi_2) \wedge \neg\mathsf{U}(\neg\varphi_2, \neg\varphi_1 \wedge \neg\varphi_2)$.*

By applying this property, the eventuality r of the outermost Until ends up (in negated form) in the first argument. A natural next choice for a candidate witness is therefore $\mathsf{U}(\mathsf{U}(p, q), \mathsf{U}(r, s))$. However, the following property gives another way to remove an Until that occurs as the first argument of another Until. Both properties can be proven by submitting them to an automatic tautology checker such as [Jan] or [Ste] for propositional linear temporal logic.

2.1.2 PROPERTY *Let $\varphi_1, \varphi_2, \varphi_3 \in$ LTL. The formula $\mathsf{U}(\mathsf{U}(\varphi_1, \varphi_2), \varphi_3)$ is semantically equivalent to $\varphi_3 \vee \mathsf{U}(\varphi_1 \vee \varphi_2, \varphi_2 \wedge \mathsf{U}(\varphi_2, \varphi_3)) \vee \mathsf{U}(\varphi_1 \vee \varphi_2, \mathsf{U}(\varphi_1 \wedge \neg\varphi_2, \varphi_3 \wedge \mathsf{U}(\varphi_1, \varphi_2)))$.*

This suggests that the propositions r and one of p and q need to be replaced by Until formulae, leading to nesting depth 3 in both arguments. It turns out that this is as far as we need to go: we will now show that the LTL(U) formula

$$\psi = \mathsf{U}(\overbrace{r \vee \mathsf{U}(p, q \wedge \mathsf{U}(q, r))}^{\psi_1}, \overbrace{s \wedge \mathsf{U}(s, t \wedge \mathsf{U}(t, u))}^{\psi_2})$$

cannot be flattened. The following technique is used in order to prove this (cf. [EH86]). We construct two infinite sequences of models Y_i and Z_i such that (1) the LTL(U) formula ψ is true in all Y_i but false in all Z_i, and (2) for any flatLTL(U) formula $\bar{\psi}$, there is a large enough k such that $\bar{\psi}$ cannot distinguish between Y_i and Z_i for $i \geq k$.

Let p, q, r be propositions which are pairwise mutually exclusive (so no two of them can occur together in the label of a state), and similarly s, t and u. Let A be the sequence ps, qu, rt of states, i.e. A consists of 3 states the first of which has label $\{p, s\}$, the second $\{q, u\}$, and the third $\{r, t\}$. Likewise, let $B = ps, ps, rt, qu$, $C = qu, ps, rt, qu$ and D the infinite sequence $qu, qu, \ldots$. For $i \in \mathbb{N}$, define $X_i = A^i B A^i C$, and inductively define the sequences Y_i by $Y_0 = D$ and $Y_{i+1} = X_{i+1} Y_i$, and the sequences Z_i by $Z_0 = Y_0$ and $Z_{i+1} = A^{i+1} C Y_i$. See Figure 1.

$$\overbrace{\underbrace{AA\cdots A}_{i}\, B\, \overbrace{\underbrace{AA\cdots A}_{i}\, C \quad \cdots \quad \overbrace{AB\,\overbrace{ACD}^{=Z_1}}^{=Y_1}}^{=Z_i}}^{=Y_i}$$

Figure 1: Y_i and Z_i

First, we state a property that will be used frequently.

2.1.3 PROPERTY *Let φ be a boolean combination of propositions and X a sequence composed of the blocks A, B, C and D. If φ holds in every state of A, then φ holds in every state of X.*

PROOF It suffices to note that each state in B, C and D also occurs in A. □

2.1.4 LEMMA *For every $i > 0$, $Y_i \models \psi$ while $Z_i \not\models \psi$.*

PROOF Let $i \in \mathbb{N}$. It is easily seen that ψ_2 holds for any sequence that starts with prefix B; in particular it holds in BA^iCY_{i-1}. Furthermore, ψ_1 holds in all sequences that start with prefix A, with prefix $A(1)$, $A(2)$ (i.e. the last two states of A), or with state $A(2)$. We conclude that $A^iBA^iCY_{i-1} \models \psi$, i.e. $Y_i \models \psi$. On the other hand, a sequence starting with C does not satisfy ψ_1 neither ψ_2. Also, ψ_2 does not hold in any sequence that starts with prefix A, with prefix $A(1)$, $A(2)$, or with state $A(2)$. So $A^iCY_i \not\models \psi$, i.e. $Z_i \not\models \psi$. □

Below, we will show that when we restrict ourselves to the flat fragment flatLTL(U), no formula can distinguish between all Y_i on the one hand and all Z_i on the other. More precisely, we prove that a formula $\varphi \in$ flatLTL(U) evaluates the same over Y_h and Z_h for any h which exceeds the Until depth of φ. The intuition is as follows. One difference between Y_i and Z_i is that the first non-A block in Y_i is B while this is C in Z_i. It is this difference that the distinguishing formula ψ brings out: the eventuality ψ_2 differs between B and C, while the initial invariance of ψ_1 ensures that there may be only A-blocks before the first occurrence of B or C. However, flatLTL(U) cannot specify such a complex initial invariant. The first argument of an Until operator can only assert a local property, and it follows from Property 2.1.3 that any local property that is invariant over a prefix consisting of one or more A-blocks indeed also holds in any state of B and C and hence cannot prevent such blocks from occurring among the A's. The only possible way to specify in flatLTL(U) that the first non-A block in Y_i is B, is by counting down the (individual states of the) leading A-blocks until B is reached. However, as was shown in Lemma 2.0.4, the length of the formula increases with the number of A-blocks to be counted. Therefore, for every formula $\varphi \in$ flatLTL(U), there is a value of i which is large enough such that φ cannot count the leading A-blocks.

Thus, the counterpart to Lemma 2.1.4 above is as follows.

2.1.5 LEMMA *Let $\varphi \in$ flatLTL(U). For every $h > Udepth(\varphi)$, $Y_h \models \varphi \Leftrightarrow Z_h \models \varphi$.*

PROOF Directly from Lemma 2.1.7 below, taking $h = i + 1$. □

The proof uses two sublemmata. The first states that a flatLTL(U) formula φ cannot distinguish between a point along Y_{i+1} and a corresponding point along Y_i, provided that $i > Udepth(\varphi)$.

2.1.6 LEMMA *Let $\varphi \in$ flatLTL(U). Then for every $i > Udepth(\varphi)$, each of the following holds.*

1. *for every $0 \leq k < |A|$: $\quad Y_{i+1}(k, \dots) \models \varphi \Leftrightarrow Y_i(k, \dots) \models \varphi$.*
2. *for every $|A| \leq k < |A^{i+1}BA|$: $\quad Y_{i+1}(k, \dots) \models \varphi \Leftrightarrow Y_i(k - |A|, \dots) \models \varphi$.*
3. *for every $|A^{i+1}BA| \leq k < |A^{i+1}BA^{i+1}C|$: $\quad Y_{i+1}(k, \dots) \models \varphi \Leftrightarrow Y_i(k - |AA|, \dots) \models \varphi$.*

PROOF The proof can be found in the appendix. It is similar in spirit to the proof of Lemma 2.0.4. □

Finally, the following lemma relates (suffixes of) Y_i to (suffixes of) Z_i.

2.1.7 LEMMA *Let $1 \leq h \leq i+1$ and $\varphi \in$ flatLTL(U). If $Udepth(\varphi) < h$, then for every $0 \leq k < |A|$, we have $(A^hBA^{i+1}CY_i)(k, \dots) \models \varphi \Leftrightarrow (A^hCY_i)(k, \dots) \models \varphi$.*

PROOF See appendix. □

3 Branching-time Logics

Linear-time temporal logic is interpreted over sequences — a formula states a property about the temporal ordering of certain events (occurrences of propositions). Branching-time temporal logic, being interpreted over tree-like structures, in addition offers primitives to talk about *choice points* along sequences. We focus here on the family of Computation Tree Logics, that has deserved much attention in the computing science community. These logics come with quantifiers that can be used to express that a certain temporal property holds for some (or all) sequences that start from the current point in the tree. The first such logic introduced, CTL ([CES86]), bears in it a restriction on the number of temporal operators that may appear in the scope of a quantifier. This results in a model checking algorithm of low complexity. Lifting this restriction (CTL*, see [EH86]) restores the expressivity to subsume LTL, but complicates the model checking problem. In this section we turn our attention to flat versions of these branching-time temporal logics.

First, we show that the expressivity results about LTL(U) vs. flatLTL(U) of the previous section imply similar results for their branching-time brothers. Second, we focus on the relative *distinguishing powers* of the latter logics. Whereas the expressivity of a logic is determined by the classes of models that can be characterised by (single) formulae, *distinctiveness* is measured by the ability of formulae to distinguish between two given models. The distinguishing power of a logic L, interpreted over models from M, is captured by the *logical equivalence induced by* L, $\equiv_{\mathrm{L}} \subseteq M \times M$, defined by $s \equiv_{\mathrm{L}} t$ iff $\forall_{\varphi \in \mathrm{L}}\ s \models \varphi \Leftrightarrow t \models \varphi$. Distinguishing power is not to be confused with expressive power. Writing $\mathrm{L}_1 \leq \mathrm{L}_2$ to denote that L_2 is at least as expressive as L_1 (i.e. $\forall_{\varphi_1 \in \mathrm{L}_1} \exists_{\varphi_2 \in \mathrm{L}_2}\ \varphi_1$ is semantically equivalent to φ_2), we have the following relation between expressivity and distinctivity:

3.0.1 LEMMA $\mathrm{L}_1 \leq \mathrm{L}_2 \Rightarrow \equiv_{\mathrm{L}_1} \supseteq \equiv_{\mathrm{L}_2}$

PROOF Assume that (1) $\mathrm{L}_1 \leq \mathrm{L}_2$ and $s \equiv_{\mathrm{L}_2} t$, i.e. (2) $\forall_{\varphi_2 \in \mathrm{L}_2}\ s \models \varphi_2 \Leftrightarrow t \models \varphi_2$. We have to show that then $s \equiv_{\mathrm{L}_1} t$, i.e $\forall_{\varphi_1 \in \mathrm{L}_1}\ s \models \varphi_1 \Leftrightarrow t \models \varphi_1$. Let $\varphi_1 \in \mathrm{L}_1$ and assume (3) $s \models \varphi_1$. By 1, we can choose $\varphi_2 \in \mathrm{L}_2$ such that (4) φ_1 is semantically equivalent to φ_2. From 3 and 4 we have $s \models \varphi_2$, from which by 2, $t \models \varphi_2$. With 4 again, we get $t \models \varphi_1$. The other direction is symmetric. □

The other direction of the implication does not hold. The following small example[2] clarifies this. Consider the sets $\mathrm{L}_1 = \mathcal{P}(\mathbb{N})$ and $\mathrm{L}_2 = \{\mathbb{N}\backslash\{k\} \mid k \in \mathbb{N}\}$ of propositions. As models over which the propositions are interpreted, take the natural numbers, defining for $i \in \mathbb{N}$ and $\varphi \in \mathrm{L}_1, \mathrm{L}_2$: $i \models \varphi$ iff $i \in \varphi$. Clearly $\mathrm{L}_2 \lneq \mathrm{L}_1$. However, it is also easy to show that any two numbers that can be distinguished by L_1, can also be distinguished by L_2, implying that $\equiv_{\mathrm{L}_1} = \equiv_{\mathrm{L}_2}$.

Another example, more related to the topic of this article, is the comparison between CTL* and CTL. On the one hand, the star does increase the expressive power: in [EH86] it is shown that the CTL* formula $\forall\mathsf{F}(p \wedge \mathsf{X}p)$ has no equivalent in CTL. On the other hand, as shown in [BCG88], the two logics are equally distinguishing: for both of them, the induced logical equivalence coincides with *bisimulation*[3] ([Par81]). Below, these results will be extended for flat versions of the logics.

We start by defining the syntax and semantics of the various Computation Tree Logics.

[2]Thanks to Ruurd Kuiper.

[3]This correspondence is shown for finite Kripke structures in [BCG88] and can be shown to hold for image-finite Kripke structures as well — see [Dam96].

3.0.2 DEFINITION *The logic* CTL* *is the set of state formulae* φ *defined inductively by the following grammar, where* $p \in$ Prop.

$$\text{state formulae:} \quad \varphi \ := \ p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \exists\psi \tag{1}$$

$$\text{path formulae:} \quad \psi \ := \ \varphi \mid \neg\psi \mid \psi \wedge \psi \mid \mathsf{X}\psi \mid \mathsf{U}(\psi, \psi) \tag{2}$$

The abbreviations true, false, $\vee$, $\rightarrow$, $\forall$, *etc. are defined as usual. In addition, we use the following abbreviations:* $\mathsf{F}\varphi$ *stands for* $\mathsf{U}(true, \varphi)$, *and* $\mathsf{G}\varphi$ *for* $\neg\mathsf{F}\neg\varphi$. V *is used to denote the dual of* U, *so* $\mathsf{V}(\varphi_1, \varphi_2) \equiv \neg\mathsf{U}(\neg\varphi_1, \neg\varphi_2)$. *Finally,* W, *the Weak Until operator, is defined by* $\mathsf{W}(\varphi_1, \varphi_2) \equiv \mathsf{G}\varphi_1 \vee \mathsf{U}(\varphi_1, \varphi_2)$. *Note that* CTL* *formulae without quantifiers are* LTL *formulae as well.*

CTL *is the fragment of* CTL* *in which at most one occurrence of the temporal operators* X *and* U *may occur in the direct scope of any path quantifier; additional occurrences must be preceded by a new quantifier again. Formally, the defining clause, 2, of path formulae is replaced by*

$$\text{path formulae:} \quad \psi \ := \ \varphi \mid \neg\psi \mid \mathsf{X}\varphi \mid \mathsf{U}(\varphi, \varphi) \tag{3}$$

CTL*(U) *and* CTL(U) *are the fragments of* CTL* *and* CTL *respectively obtained by disallowing the use of the Next operator* X. flatCTL*(U) *and* flatCTL(U) *are obtained from these nextless versions by restricting the first argument of the* U *operator to a local formula, i.e. a boolean combination of propositions.*

A model for a CTL* formula is a *Kripke structure* $\mathcal{T} = (\Sigma, \rightarrow, \mathcal{L})$ consisting of a set Σ of states, a *transition relation* $\rightarrow$ that is assumed to be total (every state has a successor under $\rightarrow$), and a state-labelling function $\mathcal{L} : \Sigma \rightarrow \mathcal{P}(\text{Prop})$. A *path* in $\mathcal{T}$ is an *infinite* sequence $\pi = s_0 s_1 \cdots$ of states such that for every $i \in \mathbb{N}$, $s_i \rightarrow s_{i+1}$; we say that π starts in s_0. $\pi(i)$ denotes s_i. A subsequence of π, denoted $\pi_{\{i,\ldots,j\}}$ with $i \in \mathbb{N}$, $j \in \mathbb{N}\cup\{\infty\}$, and $i \leq j$, is sometimes called a *block* (in particular, π itself also is a block). A subsequence that starts in s_0 is called a *prefix* of π while one that continues infinitely is called a *suffix* of π. A *partitioning* of a (sub)sequence $\hat{s}$ is a (finite or infinite) sequence $\hat{s}_{I_0}, \hat{s}_{I_1}, \ldots$ of blocks whose concatenation is $\hat{s}$. The *length* of $\hat{s}$, denoted $length(\hat{s})$, is the number of states on it; note that the last state of $\hat{s}$ is $\hat{s}(length(\hat{s}) - 1)$ if it exists. For $s \in \Sigma$, a $(\mathcal{T}, s)$-*path* (or s-*path* when $\mathcal{T}$ is clear from the context) is a path in $\mathcal{T}$ that starts in s; similarly for prefixes. $paths(\mathcal{T}, s)$ (or simply $paths(s)$) denotes the set of all s-paths while $prefixes(\mathcal{T}, s)$ ($prefixes(s)$) contains all their prefixes. The relation $\rightarrow$ is called *image-finite* iff for every $s \in \Sigma$, the set $\{s' \mid s \rightarrow s'\}$ has finite cardinality. $\mathcal{T}$ is called *finitely branching* iff $\rightarrow$ is image-finite.

In the remainder of this section we fix a Kripke structure $\mathcal{T} = (\Sigma, \rightarrow, \mathcal{L})$. The variables s and t range over Σ unless stated otherwise.

3.0.3 DEFINITION *Let* $p \in$ Prop, $\varphi, \varphi_1, \varphi_2$ *be state formulae,* ψ *a path formula, and* π *a path in* $\mathcal{T}$.

Path formulae are interpreted along paths ($(\mathcal{T}, \pi) \models \psi$ *or* $\pi \models \psi$ *for short) and state formulae in states* ($(\mathcal{T}, s) \models \varphi$ *or* $s \models \varphi$ *for short) as defined inductively by the following rules in conjunction with the rules of Definition 2.0.3.*

1. $s \models p$ *iff* $p \in \mathcal{L}(s)$.
2. $s \models \neg\varphi$ *iff* $s \not\models \varphi$.
3. $s \models \varphi_1 \wedge \varphi_2$ *iff* $s \models \varphi_1$ *and* $s \models \varphi_2$.
4. $s \models \exists\psi$ *iff there exists an* s*-path* π *such that* $\pi \models \psi$.

5. $\pi \models \varphi$, *where* $\pi = s_0 s_1 \cdots$ *iff* $s_0 \models \varphi$.

The next definition formalises a number of "compatibility relations" between temporal logics and "behavioural" equivalences on $\mathcal{T}$.

3.0.4 DEFINITION *Let* $\equiv \subseteq \Sigma \times \Sigma$ *be an equivalence relation and* L *a logic interpreted over* Σ.

- $\equiv$ *is fine for* L *iff* $\equiv \subseteq \equiv_L$.
- $\equiv$ *is abstract for* L *iff* $\equiv \supseteq \equiv_L$.
- $\equiv$ *is adequate for* L *iff it is both fine and abstract for* $\equiv_L$.

We introduce the following notation to represent transitions that "stutter" under some notion of equivalence.

3.0.5 DEFINITION *If* $\equiv$ *is an equivalence relation on* Σ, *then the transition relation* $\xrightarrow{\equiv} \subseteq \Sigma \times \Sigma$ *is defined by* $s \xrightarrow{\equiv} t$ *iff* $s \to t \ \wedge \ s \equiv t$.

3.0.6 DEFINITION *For a sequence* $\hat{s}$ *and an equivalence relation* $\equiv$ *over* Σ, $partit_{\equiv}(\hat{s})$ *is the partitioning of* $\hat{s}$ *into maximal blocks such that within each block, all states are* $\equiv$*-equivalent.*

3.1 Expressivity

Given the results from Section 2.1, the difference in expressivity between CTL*(U) and CTL(U) on the one hand, and their flat fragments on the other hand, is easily established.

3.1.1 THEOREM CTL*(U) *is strictly more expressive than* flatCTL*(U), *and* CTL(U) *is strictly more expressive than* flatCTL(U).

PROOF First, observe that for any state formula φ in CTL*(U), and any *linear* model (i.e. infinite sequence) X, we have $X \models \varphi$ iff $X \models delquant(\varphi)$, where $delquant$ is a syntactic operation that removes all path quantifiers from φ. Now, consider the CTL(U) formula $\psi' = \exists \mathsf{U}(r \vee \exists \mathsf{U}(p, q \wedge \exists \mathsf{U}(q, r)), s \wedge \exists \mathsf{U}(s, t \wedge \exists \mathsf{U}(t, u)))$. Note that $delquant(\psi')$ is the formula ψ from Section 2.1. So, from Lemma 2.1.4 it now follows that ψ' distinguishes Y_i from Z_i, for every i. Furthermore, using the same observation, it follows from Lemma 2.1.5 that there is no flatCTL*(U) formula which distinguishes all Y_i from all Z_i. $\square$

3.2 Distinctiveness

In this section we define and compare adequate behavioural equivalences for the logics flatCTL*(U) and flatCTL(U). In order to position those results in a larger picture, we start by adapting some known results on the distinctiveness of CTL*(U) and CTL(U) to our settings.

3.2.1 CTL*(U) and CTL(U)

The equivalences induced by branching-time logics coincide with various types of *bisimulation*. As we consider logics without next-state operator, we are interested in (variations on) *stuttering equivalence* [BCG88, DNV90].

3.2.1 DEFINITION *Let* $\equiv$ *be an equivalence relation on* Σ. *We say that* $s \in \Sigma$ *has infinite* $\equiv$*-stuttering, denoted* $infstut_{\equiv}(s)$, *iff there exists an* s*-path* $\bar{s}$ *such that for all states* s' *on* $\bar{s}$, $s' \equiv s$.

3.2.2 DEFINITION *Let* $\equiv\ \subseteq \Sigma \times \Sigma$ *be a symmetric relation such that for every* $s, t \in \Sigma$, $s \equiv t$ *implies:*

1. $\mathcal{L}(s) = \mathcal{L}(t)$.
2. $infstut_{\equiv}(s)$ *iff* $infstut_{\equiv}(t)$.
3. *For*[4] *every* $s \dashrightarrow_{\equiv} \ldots \dashrightarrow_{\equiv} s_{k-1} \rightarrow s_k$ *such that* $k \geq 0$, *there exists* $t \dashrightarrow_{\equiv} \ldots \dashrightarrow_{\equiv} t_{l-1} \rightarrow t_l$ *such that* $l \geq 0$ *and* $s_k \equiv t_l$.

Then $\equiv$ *is called a (divergence sensitive) stuttering equivalence (dss-equivalence). The largest stuttering equivalence is denoted* $\equiv_{\mathrm{stut}}$.

This definition of stuttering equivalence is different from those given in [BCG88] and [DNV90], while on the side of the models, we have lifted the restriction that the Kripke structures be finite. Yet, it can be shown (see [Dam96]) that the defined equivalences coincide. Furthermore, the fineness results of [BCG88] and [DNV90] carry over to the case of infinite structures:

3.2.3 LEMMA *If* $s \equiv_{\mathrm{stut}} t$, *then* $\forall_{\varphi \in \mathrm{CTL}^*(\mathsf{U})}\ s \models \varphi \Leftrightarrow t \models \varphi$.

The converse, abstractness, only holds for Kripke structures that satisfy a certain (strong) form of finite-branchingness.

3.2.4 DEFINITION *Let* $\equiv$ *be an equivalence relation on* Σ. *We say that* $\mathcal{T}$ *is finitely branching under* $\equiv$*-stuttering iff the reflexive transitive closure* $\dashrightarrow_{\equiv}^*$ *of the relation* $\dashrightarrow_{\equiv}$ *is image-finite. For a logic* L, *"finitely branching under* L*-stuttering" abbreviates "finitely branching under* $\equiv_L$*-stuttering".*

The following property is easily proven.

3.2.5 PROPERTY *Let* $\equiv_1$ *and* $\equiv_2$ *be equivalence relations on* Σ *such that* $\equiv_1 \subseteq \equiv_2$. *If* $\mathcal{T}$ *is finitely branching under* $\equiv_2$*-stuttering then* $\mathcal{T}$ *is finitely branching under* $\equiv_1$*-stuttering.*

3.2.6 LEMMA *Assume that* $\mathcal{T}$ *is finitely branching and also finitely branching under* CTL(U)*-stuttering. If* $\forall_{\varphi \in \mathrm{CTL}(\mathsf{U})}\ s \models \varphi \Leftrightarrow t \models \varphi$, *then* $s \equiv_{\mathrm{stut}} t$.

PROOF See [Dam96]. □

Although the condition that $\mathcal{T}$ is finitely branching under CTL(U)-stuttering is the weakest condition that suffices to prove the above lemma, it may be impractical to check. Note that by Property 3.2.5, it follows that finite branchingness under CTL(U)-stuttering is implied by finite branchingness under Prop-stuttering.

As Lemma 3.2.6 immediately implies that $\equiv_{\mathrm{stut}}$ is abstract for CTL*(U), we can conclude by the following

3.2.7 COROLLARY *Assume that* $\mathcal{T}$ *is finitely branching and also finitely branching under* CTL(U)*-stuttering. Then* $\equiv_{\mathrm{stut}}$ *is adequate for both* CTL*(U) *and* CTL(U).

[4] Note that the "vice versa" is not needed because $\equiv$ is required to be symmetric.

3.2.2 flatCTL*(U)

Moving on to the flat versions of CTL*(U) and CTL(U), we adapt the stuttering equivalence as follows.

3.2.8 DEFINITION *For* $s, t \in \Sigma$, $s \equiv^0 t \Leftrightarrow \mathcal{L}(s) = \mathcal{L}(t)$.

3.2.9 DEFINITION *Let* $\equiv \subseteq \Sigma \times \Sigma$ *be a symmetric relation such that for every* $s, t \in \Sigma$, $s \equiv t$ *implies:*

1. $\mathcal{L}(s) = \mathcal{L}(t)$.
2. $infstut_{\equiv}(s)$ *iff* $infstut_{\equiv}(t)$.
3. *For every* $\hat{s} \in prefixes(s)$ *there exists* $\hat{t} \in prefixes(t)$ *such that:*
   (a) *For every* $\hat{s}_0 \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \hat{s}_{k-1} \rightarrow \hat{s}_k$ *such that* $0 \leq k < length(\hat{s})$, *there exists* $\hat{t}_0 \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \hat{t}_{l-1} \rightarrow \hat{t}_l$ *such that* $0 \leq l < length(\hat{t})$ *and* $\hat{s}_k \equiv \hat{t}_l$.
   (b) *For every* $\hat{t}_0 \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \hat{t}_{l-1} \rightarrow \hat{t}_l$ *such that* $0 \leq l < length(\hat{t})$, *there exists* $\hat{s}_0 \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \hat{s}_{k-1} \rightarrow \hat{s}_k$ *such that* $0 \leq k < length(\hat{s})$ *and* $\hat{t}_l \equiv \hat{s}_k$.

Then $\equiv$ *is called a flat star (stuttering) equivalence. The largest flat star equivalence is denoted* $\equiv_{\text{flat}^*}$.

The form of point 3 in this definition may be slightly surprising. One may wonder whether it could not be as follows.

3$'$. *For every* $s \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} s_{k-1} \rightarrow s_k$ *such that* $k \geq 0$, *there exists* $t \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} t_{l-1} \rightarrow t_l$, *such that* $l \geq 0$ *and* $s_k \equiv t_l$.

This may be clarified by considering the equivalence induced by flatCTL*(U) in game-theoretic terms. Consider states s and t and suppose that they must satisfy the same flatCTL*(U) formulae. In particular, we consider formulae of the form $\exists\psi$, where ψ is an arbitrary path formula, which may consist of a conjunction of (negations of) smaller path formulae. If t has to satisfy formulae of the same form, then Defender must have a winning strategy to the following two-phase game:

1. Phase 1: Attacker either chooses an s-path, say $\bar{s}$, which should be matched by the choice by Defender of a t-path, say $\bar{t}$, or Attacker chooses a t-path, say $\bar{t}$, which should be matched by the choice by Defender of an s-path, say $\bar{s}$.

   This phase reflects the choice that corresponds to the $\exists$ quantifier in the formula.

2. Phase 2: Attacker chooses either $\bar{s}$ or $\bar{t}$ to proceed. Denoting the result of this choice by $\bar{u}$, Attacker then chooses a state $\bar{u}(k)$ on $\bar{u}$ such that for every $0 \leq i < k$, $\bar{u}(i) \equiv^0 \bar{u}(0)$. Defender now has to proceed from the other path, call it $\bar{v}$ (so, $\bar{v} = \bar{s}$ if $\bar{u} = \bar{t}$ and $\bar{v} = \bar{t}$ if $\bar{u} = \bar{s}$). She should match the move of Attacker with the choice of a position $\bar{v}(l)$ on $\bar{v}$ such that for every $0 \leq i < l$, $\bar{v}(i) \equiv^0 \bar{v}(0)$, and also $\bar{v}(l) \equiv^0 \bar{u}(k)$ ($k = l = 0$ is possible). The game continues from $\bar{u}(k)$ and $\bar{v}(l)$.

   This second phase reflects the statement of an arbitrary path property: The fact that Attacker chooses either $\bar{s}$ or $\bar{t}$ to proceed reflects the fact that this path property may occur in positive or negated form, while the fact that all states up to $\bar{u}(k)$ have to be $\equiv^0$-equivalent carries in it the restriction of Until formulae to a propositional first argument.

This game-theoretic formulation explains the inadequacy of choosing point 3′ instead of 3. The problem is that Defender has to choose $\bar{t}$ in Phase 1, without knowing which k Attacker is going to choose in Phase 2: point 3′ only guarantees that Defender can match any move of Attacker *in which the choice for k is made at the same moment at which $\bar{s}$ is chosen.*

Fineness of $\equiv_{\mathsf{flat}^*}$ for flatCTL*(U) (and hence for flatCTL(U)) is proven by an inductive argument on the structure of the formulae. Because the inductive definition of these formulae involves path formulae, flat star equivalence is extended to paths so that the induction hypothesis can be strengthened with a part stating that any two equivalent paths satisfy the same flatCTL*(U) path formulae.

3.2.10 DEFINITION $\equiv_{\mathsf{flat}^*}$ *is extended to paths by defining* $\bar{s} \equiv_{\mathsf{flat}^*} \bar{t}$ *iff*

1. *For every* $k \geq 0$ *there exists* $l \geq 0$ *such that* $\bar{s}(k) \equiv_{\mathsf{flat}^*} \bar{t}(l)$ *and furthermore, letting* $partit_{\equiv^0}(\bar{s}_{\{0,\ldots,k-1\}}) = \bar{s}_{I_1}, \ldots, \bar{s}_{I_K}$ *and* $partit_{\equiv^0}(\bar{t}_{\{0,\ldots,l-1\}}) = \bar{t}_{J_1}, \ldots, \bar{t}_{J_L}$, $K = L$ *and for every* $0 \leq i \leq K$, *every* $s' \in \bar{s}_{I_i}$ *and* $t' \in \bar{t}_{J_i}$, *we have* $s' \equiv^0 t'$.
2. *Vice versa.*

We can now prove the following "state-path lemma".

3.2.11 LEMMA *If* $s \equiv_{\mathsf{flat}^*} t$, *then for every* $\bar{s} \in paths(s)$ *there exists* $\bar{t} \in paths(t)$ *such that* $\bar{s} \equiv_{\mathsf{flat}^*} \bar{t}$.

PROOF Let $partit_{\equiv_{\mathsf{flat}^*}}(\bar{s})$ be $B_0, B_1, \ldots$. For every $i \geq 0$ for which B_i exists, let b_i be the first state of B_i. Let $c_0 = t$. By point 3 in Definition 3.2.9, there exists a t-prefix $\hat{t}$ such that for every $0 \leq k \leq length(B_0)$ (note that by definition of B_0, all states on it are $\equiv^0$-equivalent, and that $\bar{s}(length(B_0)) = b_1$, if $length(B_0) < \omega$), there exists $0 \leq l \leq length(\hat{t})$ such that for every $0 \leq j < l$, $\hat{t}(j) \equiv^0 t$ and $\bar{s}(k) \equiv_{\mathsf{flat}^*} \hat{t}(l)$, and vice versa. Consider the shortest such t-prefix, $\hat{t}'$. Clearly, all states on $\hat{t}'$, with the exception of its last, are $\equiv^0$-equivalent. Define C_0 to be $\hat{t}'$ with its last state excepted, while c_1 (the first state of block C_1 to be defined) is defined to be the last state of $\hat{t}'$. This way, we can inductively define states c_i and blocks C_i for all $i \geq 0$ for which B_i exists. If some B_i is infinite, then point 2 in Definition 3.2.9 guarantees the existence of an appropriate C_i. It is now easily seen that for the path $\bar{t}$ formed by $C_0, C_1, \ldots$, we have $\bar{s} \equiv_{\mathsf{flat}^*} \bar{t}$. □

Fineness now follows easily.

3.2.12 LEMMA *If* $s \equiv_{\mathsf{flat}^*} t$, *then* $\forall_{\varphi \in \mathrm{flatCTL}^*(\mathsf{U})}\ s \models \varphi \Leftrightarrow t \models \varphi$.

PROOF. We prove the following two points by induction on the structure of the formula.

1. If $s \equiv_{\mathsf{flat}^*} t$, then for all state formulae $\varphi \in$ flatCTL*(U), $s \models \varphi$ iff $t \models \varphi$.
2. For paths $\bar{s}$ and $\bar{t}$: if $\bar{s} \equiv_{\mathsf{flat}^*} \bar{t}$, then for all path formulae φ that occur in flatCTL*(U) formulae, $\bar{s} \models \varphi$ iff $\bar{t} \models \varphi$.

- Base: $\varphi \in$ Prop. $s \equiv_{\mathsf{flat}^*} t$ implies that $\mathcal{L}(s) = \mathcal{L}(t)$. From this it follows that $s \models p$ iff $t \models p$ for all $p \in$ Prop.
- Induction step:
  1. The cases that φ is a conjunction or negation of state or path formulae, or a state formula interpreted over a path, are straightforward.
  2. $\varphi = \mathsf{U}(p, \varphi')$. Assume that $\bar{s} \models \varphi$. By Definition 3.0.3, this means that we can choose $k \geq 0$ such that $\bar{s}(k) \models \varphi'$ and for every $0 \leq i < k$, $\bar{s}(i) \models p$. By definition of $\bar{s} \equiv_{\mathsf{flat}^*} \bar{t}$, there exists $l \geq 0$ such that $\bar{t}(l) \equiv_{\mathsf{flat}^*} \bar{s}(k)$ and for every $0 \leq j < l$, there exists $0 \leq i < k$ such that $\bar{s}(i) \equiv^0 \bar{t}(j)$. Using the induction hypothesis, it follows that $\bar{t} \models \varphi$.

3. $\varphi = \exists\varphi'$. Straightforward using Lemma 3.2.11. $\square$

For the other direction, abstractness, we again need to impose certain forms of finite branchingness.

3.2.13 LEMMA *Assume that $\mathcal{T}$ is finitely branching and also finitely branching under* Prop*-stuttering. If $\forall_{\varphi\in\text{flatCTL}^*(\mathsf{U})}\ s \models \varphi \Leftrightarrow t \models \varphi$, then $s \equiv_{\text{flat}^*} t$.*

PROOF. Assume that $\forall_{\varphi\in\text{flatCTL}^*(\mathsf{U})}\ s \models \varphi \Leftrightarrow t \models \varphi$. We have to show that $s \equiv_{\text{flat}^*} t$. Because $\equiv_{\text{flat}^*}$ is the largest flat star equivalence, we have to show that the pair (s, t) is an element of some flat star equivalence $\equiv \subseteq \Sigma \times \Sigma$. We define this relation as follows: $u \equiv v$ iff $\forall_{\varphi\in\text{flatCTL}^*(\mathsf{U})}\ u \models \varphi \Leftrightarrow v \models \varphi$. Clearly $s \equiv t$. We show that $\equiv$ is a flat star equivalence.

1. It is trivial that $\mathcal{L}(s) = \mathcal{L}(t)$.
2. Suppose that $infstut_{\equiv}(s)$, i.e. we can choose an s-path $\bar{s}$ such that for every $i \geq 0$, $\bar{s}(i) \equiv s$. We have to show that also $infstut_{\equiv}(t)$. Suppose that this is *not* the case. Then every t-path contains a state from the set $T = \{t'' \mid t \mathrel{-_{\equiv}\!\!\to^*} t' \to t'' \wedge t' \not\equiv t''\}$. Because $\to$ is total (by assumption), T is nonempty. Because $\mathcal{T}$ is finitely branching under Prop-stuttering and also (plainly) finitely branching, this implies by Lemma 3.2.5 that T is finite, say $T = \{t_1'', \ldots, t_n''\}$. Because for every $1 \leq i \leq n$, $t_i'' \not\equiv t$ and $t \equiv s$, we have $t_i'' \not\equiv s$. Hence, by definition of $\equiv$, we can choose formulae $\varphi_i \in$ flatCTL*(U) such that $s \models \varphi_i$ and $t_i'' \not\models \varphi_i$, for every $1 \leq i \leq n$. Because all states on $\bar{s}$ are $\equiv$-equivalent to s, we have $s \models \exists\mathsf{G}(\varphi_1 \wedge \cdots \wedge \varphi_n)$, but because every t-path contains some t_i'', $t \not\models \exists\mathsf{G}(\varphi_1 \wedge \cdots \wedge \varphi_n)$, implying that $s \not\equiv t$, as $\exists\mathsf{G}(\varphi_1 \wedge \cdots \wedge \varphi_n)$ is equivalent to a flatCTL*(U) formula by Property 2.1.1. Contradiction.
3. Let $\hat{s}$ be an s-prefix. We have to show that there exists a t-prefix $\hat{t}$ such that:

   (a) For every $\hat{s}_0 \mathrel{-_{\equiv^0}\!\!\to} \cdots \mathrel{-_{\equiv^0}\!\!\to} \hat{s}_{k-1} \to \hat{s}_k$ such that $0 \leq k < length(\hat{s})$, there exists $\hat{t}_0 \mathrel{-_{\equiv^0}\!\!\to} \cdots \mathrel{-_{\equiv^0}\!\!\to} \hat{t}_{l-1} \to \hat{t}_l$ such that $0 \leq l < length(\hat{t})$ and $\hat{s}_k \equiv \hat{t}_l$.

   (b) Vice versa.

   (*) Suppose that this is *not* the case. Consider the set $T = \{\langle t', t''\rangle \mid t \mathrel{-_{\equiv^0}\!\!\to^*} t' \to t''\}$. Because $\to$ is total (by assumption), T is nonempty. Because $\mathcal{T}$ is finitely branching under Prop-stuttering and also (plainly) finitely branching, T is finite, say $T = \{\langle t_0', t_0''\rangle, \ldots, \langle t_n', t_n''\rangle\}$. We let M be the largest number such that $\hat{s}(0) \mathrel{-_{\equiv^0}\!\!\to} \hat{s}(1) \mathrel{-_{\equiv^0}\!\!\to} \cdots \mathrel{-_{\equiv^0}\!\!\to} \hat{s}(M-1) \to \hat{s}(M)$. This implies that $\hat{s}(M-1) \not\equiv^0 \hat{s}(M)$. Therefore, we can choose $p \in$ Prop such that $\hat{s}(M-1) \models p$ (and hence also $\hat{s}(i) \models p$ for every $0 \leq i \leq M-1$) and $\hat{s}(M) \not\models p$. Furthermore, for every $0 \leq k \leq M$ and $0 \leq j \leq n$, choose formulae $p_j \in$ Prop and $\varphi_{k,j} \in$ flatCTL*(U) as follows.

   - $p_j = true$ if $t_j' \equiv^0 t_j''$; otherwise, choose p_j such that $t_j' \models p_j$ (and hence $s \models p_j$) and $t_j'' \not\models p_j$, which is possible by definition of $\equiv^0$.
   - $\varphi_{k,j} = true$ if $\hat{s}(k) \equiv t_j''$; otherwise, choose $\varphi_{k,j}$ such that $\hat{s}(k) \models \varphi_{k,j}$ and $t_j'' \not\models \varphi_{k,j}$, which is possible by definition of $\equiv$.

   For $0 \leq k \leq M$, define $\psi_k = \mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_{k,1} \wedge \cdots \wedge \varphi_{k,n})$. Furthermore, for $0 \leq j \leq n$, define $\xi_j = \mathsf{U}(\varphi_{0,j} \vee \cdots \vee \varphi_{M-1,j}, \neg p)$. Define $\varphi = \exists((\bigwedge_{0\leq k\leq M}\psi_k) \wedge (\bigwedge_{0\leq j\leq n}\xi_j))$.

   Then $s \models \varphi$, as can be seen as follows. Consider an s-path $\bar{s}$ that is an extension of $\hat{s}$ (such a path exists because $\to$ is total); so $\bar{s}(k) = \hat{s}(k)$ for every $0 \leq k \leq M$. First, we show that $\bar{s} \models \psi_k$ for every $0 \leq k \leq M$. Let $0 \leq k \leq M$. Then by definition of the $\varphi_{k,j}$, for every $0 \leq j \leq n$, $\bar{s}(k) \models \varphi_{k,j}$ while for every $0 \leq i < k$, by definition of the p_j and by the fact that s is $\equiv^0$-equivalent to $\bar{s}(i)$, we have $\bar{s}(i) \models p_0 \wedge \cdots \wedge p_n$. Second, we show that $\bar{s} \models \xi_j$ for every $0 \leq j \leq n$. Let $0 \leq j \leq n$. By definition of p, we have $\bar{s}(M) \models \neg p$. Furthermore, for every $0 \leq k < M$, we have $\bar{s}(k) \models \varphi_{k,j}$ by definition of the $\varphi_{k,j}$, so $\bar{s}(k) \models \varphi_{0,j} \vee \cdots \vee \varphi_{M-1,j}$.

   Next, we show that $t \not\models \varphi$. Consider a t-path $\bar{t}$ and suppose (**) $\bar{t} \models \bigwedge_{0\leq k\leq M}\psi_k$. We show that then there exists $0 \leq j \leq n$ such that $\bar{t} \not\models \xi_j$. Our first observation is that by assumption (**), it must be the case that for every $0 \leq k \leq M$, there exists $l \geq 0$ such that for every $0 \leq j < l$, $\bar{t}(j) \equiv^0 t$ and $\hat{s}(k) \equiv \bar{t}(l)$. Namely, suppose it were *not* the case. Then we can choose $0 \leq k' \leq M$ such that for every $l \geq 0$, [there exists $0 \leq j < l$ such that $\bar{t}(j) \not\equiv^0 t$] or [$\hat{s}(k') \not\equiv \bar{t}(l)$]. In the first case, $p_0 \wedge \cdots \wedge p_n$ does not hold for all $0 \leq j < l$ and in the second, $\varphi_{k',i}$ does not hold in $\bar{t}(l)$, hence $\varphi_{k',1} \wedge \cdots \wedge \varphi_{k',n}$ does not hold in $\bar{t}(l)$. But that means that $\bar{t} \not\models \psi_{k'}$. Contradiction with assumption (**). So for every $0 \leq k \leq M$, there exists $l \geq 0$ such that for every $0 \leq j < l$, $\bar{t}(j) \equiv^0 t$ and $\hat{s}(k) \equiv \bar{t}(l)$. By the definition of M, it is also the case for such a k that for every $0 \leq i < k$, we have $\hat{s}(i) \equiv^0 s$. So, condition 3a above holds. By assumption (*), it must then be the case that condition 3b does not hold, i.e. we can choose $l' \geq 0$ such that for every $0 \leq j < l'$, $\bar{t}(j) \equiv^0 t$ and for every $k \geq 0$ such that $\hat{s}(i) \equiv^0 s$ for every $0 \leq i < k$, we have $\hat{s}(k) \not\equiv \bar{t}(l')$. Let N be the largest number such that $\bar{t}(0) \xrightarrow{\equiv^0} \bar{t}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{t}(N-1) \to \bar{t}(N)$. Clearly, we also have $l' \leq N$. We show that $\xi_{l'}$ cannot hold. Namely, if it has to hold along $\bar{t}$, then the eventuality $\neg p$ can only be fulfilled in $\bar{t}(N)$ or beyond it — this follows from the definition of p, the fact that $s \equiv t$, and the definition of N. That means that the state formula $\varphi_{0,l'} \vee \cdots \vee \varphi_{M-1,l'}$ has to hold in $\bar{t}(j)$ for every $0 \leq j < N$. However, it does not hold in $\bar{t}(l')$. This follows from the definition of the $\varphi_{k,j}$ and from the fact that by definition of N, for every $0 \leq j < N$, $\bar{t}(j)$ is equal to some t''_h $(0 \leq h \leq n)$.

We conclude that φ distinguishes between s and t. By Property 2.1.1, it follows that then there is also a formula $\varphi' \in$ flatCTL*(U) that distinguishes between s and t, from which it follows that $s \not\equiv t$. Contradiction. So assumption (*) cannot be true. □

Note that the distinguishing formula that is constructed in this proof is not in flatCTL(U): it contains several conjuncted occurrences of U in the direct scope of a single $\exists$. This situation is to be contrasted with the distinctiveness results for the non-flat versions of the logics. In Lemma 3.2.6, CTL(U)-equivalence was a sufficient condition for $\equiv_{\text{stut}}$-equivalence. In the proof of that lemma (see [Dam96]), the constructed distinguishing formula did not need to draw on the additional expressive power of the star. The obvious question is whether we could have found such a "starless" formula in the proof above as well, by being more clever. In an attempt to answer this question, we tackle the problem from the other side: in the next subsection an adequate behavioural equivalence for flatCTL(U) is defined that will be compared to $\equiv_{\text{flat}^*}$. Meanwhile, we have the following

3.2.14 COROLLARY *Assume that $\mathcal{T}$ is finitely branching and also finitely branching under* Prop-*stuttering. Then* $\equiv_{\text{flat}^*}$ *is adequate for* flatCTL*(U).

3.2.3 flatCTL(U)

In order to arrive at an adequate behavioural equivalence for flatCTL(U), let us consider the game for flatCTL(U). Consider states s and t and suppose that s satisfies a flatCTL(U) formula. Again, we concentrate on formulae of the form $\exists\psi$. Then, ψ is either $\mathsf{U}(\varphi_1, \varphi_2)$ or $\neg\mathsf{U}(\varphi_1, \varphi_2)$, where the φ_i are state formulae again[5]. If t has to satisfy formulae of the same form, then Defender must have a winning strategy to the following game:

1. Alternative 1: Attacker chooses an s-path, say $\bar{s}$, together with a state $\bar{s}(k)$ on $\bar{s}$ such that for every $0 \leq i < k$, $\bar{s}(i) \equiv^0 \bar{s}(0)$. This should be matched by the choice by Defender of a t-path, say $\bar{t}$, and a state $\bar{t}(l)$ on $\bar{t}$ such that for every $0 \leq j < l$, $\bar{t}(j) \equiv^0 \bar{t}(0)$, and also $\bar{t}(l) \equiv^0 \bar{s}(k)$. The game continues from $\bar{s}(k)$ and $\bar{t}(l)$.

   This alternative corresponds to ψ being of the form $\mathsf{U}(\varphi_1, \varphi_2)$.

2. Alternative 2:

[5] According to Definition 3.0.2, ψ may also be a state formula, in which case the $\exists$ can be eliminated and need not be considered, or ψ may start with more than one $\neg$ symbols, which can also be eliminated, in the usual way.

(a) Phase 1: Attacker chooses an s-path, say $\bar{s}$, which should be matched by the choice by Defender of a t-path, say $\bar{t}$.

This phase reflects the choice that corresponds to the $\exists$ quantifier in the formula.

(b) Phase 2: Attacker chooses a state $\bar{t}(l)$ on $\bar{t}$ such that for every $0 \leq j < l$, $\bar{t}(j) \equiv^0 \bar{t}(0)$. Defender now has to proceed from $\bar{s}$. She should match the move of Attacker with the choice of a position $\bar{s}(k)$ on $\bar{s}$ such that for every $0 \leq i < k$, $\bar{s}(i) \equiv^0 \bar{s}(0)$. The game continues from $\bar{s}(k)$ and $\bar{t}(l)$.

This phase reflects the statement of the path formula $\neg\mathsf{U}(\varphi_1, \varphi_2)$. The fact that Attacker has to choose $\bar{t}$ to proceed reflects the fact that this path property occurs in negated form.

Thus, we propose the following definition of $\equiv_{\mathrm{flat}}$.

3.2.15 DEFINITION *Let* $\equiv\ \subseteq \Sigma \times \Sigma$ *be a symmetric relation such that for every* $s, t \in \Sigma$, $s \equiv t$ *implies:*

1. $\mathcal{L}(s) = \mathcal{L}(t)$.
2. (a) *For every* $s \xrightarrow{\equiv^0} s_1 \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} s_{k-1} \to s_k$ *such that* $k \geq 0$, *there exists* $t \xrightarrow{\equiv^0} t_1 \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} t_{l-1} \to t_l$ *such that* $l \geq 0$ *and* $s_k \equiv t_l$.

   (b) *For every* $\bar{s} \in paths(s)$ *there exists* $\bar{t} \in paths(t)$ *such that: for every* $l \geq 0$ *such that* $\bar{t}(0) \xrightarrow{\equiv^0} \bar{t}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{t}(l-1) \to \bar{t}(l)$ *there exists* $k \geq 0$ *such that* $\bar{s}(0) \xrightarrow{\equiv^0} \bar{s}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{s}(k-1) \to \bar{s}(k)$ *and* $\bar{s}(k) \equiv \bar{t}(l)$.

Then $\equiv$ *is called a flat (stuttering) equivalence. The largest flat equivalence is denoted* $\equiv_{\mathrm{flat}}$.

The following lemma shows that this equivalence is fine enough to guarantee that equivalent states satisfy the same flatCTL(U) formulae. Note that we do not need to extend the definition of $\equiv_{\mathrm{flat}}$ to paths, as the inductive argument does not consider path formulae. Proofs from this subsection, being much alike those in the previous, have been moved into the appendix.

3.2.16 LEMMA *If* $s \equiv_{\mathrm{flat}} t$, *then* $\forall_{\varphi \in \mathrm{flatCTL(U)}}\ s \models \varphi \Leftrightarrow t \models \varphi$.

Reversely, flatCTL(U) can distinguish any two states that are not $\equiv_{\mathrm{flat}}$-equivalent. This follows from the following abstractness result.

3.2.17 LEMMA *Assume that* $\mathcal{T}$ *is finitely branching and also finitely branching under* Prop*-stuttering. If* $\forall_{\varphi \in \mathrm{flatCTL(U)}}\ s \models \varphi \Leftrightarrow t \models \varphi$, *then* $s \equiv_{\mathrm{flat}} t$.

3.2.18 COROLLARY *Assume that* $\mathcal{T}$ *is finitely branching and also finitely branching under* Prop*-stuttering. Then* $\equiv_{\mathrm{flat}}$ *is adequate for* flatCTL(U).

3.2.4 Separating the distinguishing powers of flatCTL*(U) and flatCTL(U)

Towards the end of Section 3.2.2 we raised the question whether two states that are not $\equiv_{\mathrm{flat}^*}$-equivalent can be distinguishing by a flat formula without drawing on the power of the star. Here, we will show that this is not the case: we present a finite Kripke structure in which states s and t are distinguishable by a flatCTL*(U) formula (and hence, by Lemma 3.2.12, not $\equiv_{\mathrm{flat}^*}$-equivalent), but $s \equiv_{\mathrm{flat}} t$ (and hence, by Lemma 3.2.16, not distinguishable by any flatCTL(U) formula).

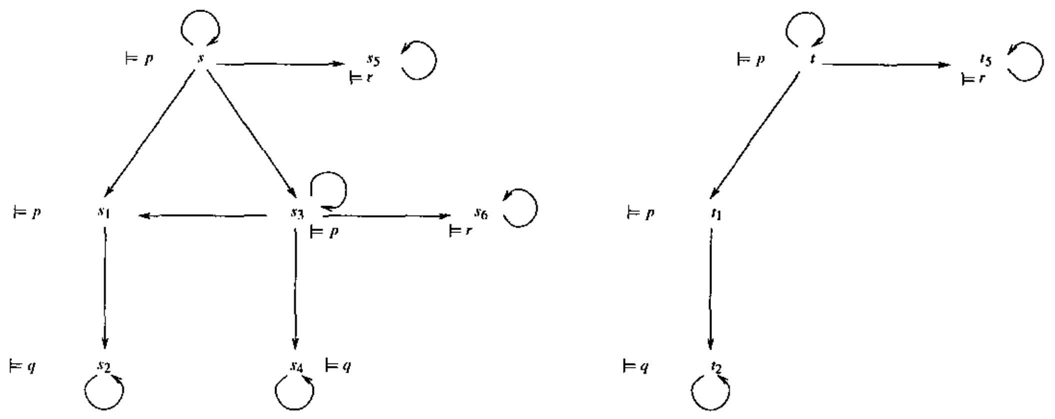

Figure 2: $s \not\equiv_{\text{flat}^*} t$ but $s \equiv_{\text{flat}} t$.

The formula in flatCTL*(U) that distinguishes s from t is $\varphi = \exists(\neg\mathsf{U}(\neg q, \neg\exists\mathsf{U}(p,r) \wedge \neg q) \wedge \mathsf{U}(\mathit{true}, q))$. In order to see this, we first translate φ into an equivalent CTL(U) formula, as follows.

$$\exists(\neg\mathsf{U}(\neg q, \neg\exists\mathsf{U}(p,r) \wedge \neg q) \wedge \mathsf{U}(\mathit{true}, q))$$

$$\equiv$$

$$\neg\forall(\mathsf{U}(\neg q, \neg\exists\mathsf{U}(p,r) \wedge \neg q) \vee \mathsf{G}\neg q)$$

$$\equiv$$

$$\neg\forall(\mathsf{W}(\neg q, \neg\exists\mathsf{U}(p,r) \wedge \neg q)$$

$$\equiv$$

$$\neg\forall\mathsf{V}(\neg\exists\mathsf{U}(p,r), \neg q)$$

$$\equiv$$

$$\exists\mathsf{U}(\exists\mathsf{U}(p,r), q)$$

So, φ expresses the property that there exists a path along which eventually q holds while in all states before that, there exists the possibility to reach an r-state via p-states only. It can easily be checked that there exists such a path starting from s, namely $s, s_3, s_4, s_4, \ldots$. On the other hand, the only path from t that eventually hits a q-state is $t, t_1, t_2, t_2, \ldots$, but from t_1 there is no possibility to reach r anymore. So, $s \models \varphi$ while $t \not\models \varphi$, and hence $s \not\equiv_{\text{flat}^*} t$.

Next, we show that $s \equiv_{\text{flat}} t$ by giving an equivalence relation $\equiv$ on states that satisfies Definition 3.2.15. $\equiv$ consists of the following pairs: $\{(s,t), (s_1,t_1), (s_2,t_2), (s_3,t), (s_4,t_2), (s_5,t_5), (s_6,t_5)\}$. The conditions of Definition 3.2.15 are obviously satisfied for the pairs (s_1,t_1), (s_2,t_2), (s_4,t_2), (s_5,t_5), and (s_6,t_5), as the corresponding subtrees are isomorphic for each pair. Next, note that s and s_3 are bisimilar. We show that the pair (s,t) satisfies the conditions of Definition 3.2.15, then this follows for (s_3,t) as well. As for (s,t), it is clear that for every prefix or path that can be taken starting from t, there exists a corresponding (in the sense of conditions 2a and 2b of Definition 3.2.15) prefix or path starting from s. Conversely, starting from s, the only non-obvious cases are the prefixes and paths going via s, s_3, s_4. If we take any such prefix that ends in s_4, then the prefix t, t_1, t_2 matches it in the sense of condition 2a (note that $s_3 \equiv^0 t_1$). If we take the (infinite) path $s, s_3, s_4, s_4, \ldots$, then the matching path in the sense of condition 2b is the path that keeps cycling in t forever.

Thus, unlike the cases CTL*/CTL and CTL*(U)/CTL(U), where in both cases the starred and unstarred versions had the same distinguishing powers, we have now identified a fragment for which these distinguishing powers are different. Other interesting comparisons are between flatCTL*(U) and CTL*(U), and between flatCTL(U) and CTL(U). In the next subsection, we consider the first of these.

3.2.5 Separating the distinguishing powers of flatCTL*(U) and CTL*(U): a surprise

It is not possible to separate $\equiv_{\mathrm{stut}}$ from $\equiv_{\mathrm{flat}^*}$: the following lemma says that they indeed coincide.

3.2.19 LEMMA $\equiv_{\mathrm{stut}} = \equiv_{\mathrm{flat}^*}$.

PROOF Because $\equiv_{\mathrm{stut}}$ is adequate for CTL*(U), $\equiv_{\mathrm{flat}^*}$ is adequate for flatCTL*(U), and flatCTL*(U) $\subseteq$ CTL*(U), we clearly have $\equiv_{\mathrm{stut}} \subseteq \equiv_{\mathrm{flat}^*}$. In order to prove $\equiv_{\mathrm{stut}} \supseteq \equiv_{\mathrm{flat}^*}$, we have to show that $\equiv_{\mathrm{flat}^*}$ satisfies the conditions in Definition 3.2.2. Points 1 and 2 are easy. As to point 3, assume that $s \xrightarrow{\equiv_{\mathrm{flat}^*}} s_1 \xrightarrow{\equiv_{\mathrm{flat}^*}} \dots \xrightarrow{\equiv_{\mathrm{flat}^*}} s_{k-1} \rightarrow s_k$ such that $k \geq 0$. *By point 3 in Definition 3.2.9 of* $\equiv_{\mathrm{flat}^*}$, *it is easy to see that there exists* $t \xrightarrow{\equiv^0} t_1 \xrightarrow{\equiv^0} \dots \xrightarrow{\equiv^0} t_{l-1} \rightarrow t_l$ such that $l \geq 0$ and $s_k \equiv_{\mathrm{flat}^*} t_l$. Next, we show that any two states t_j and $t_{j'}$ with $0 \leq j \leq j' < l$ are $\equiv_{\mathrm{flat}^*}$-equivalent. Let $0 \leq j \leq j' < l$. By point 3 in Definition 3.2.9 of $\equiv_{\mathrm{flat}^*}$, we can choose $0 \leq i \leq i' < k$ such that $t_j \equiv_{\mathrm{flat}^*} s_i$ and $t_{j'} \equiv_{\mathrm{flat}^*} s_{i'}$. By definition, $s_i \equiv_{\mathrm{flat}^*} s_{i'}$. So $t_j \equiv_{\mathrm{flat}^*} t_{j'}$. □

4 Conclusions

We have investigated the effects of flattening on the expressivity and distinctiveness of temporal logics. In Section 2 we have demonstrated an LTL(U) formula that cannot be expressed by a flat equivalent. Furthermore, we have argued that this witness formula is the shortest in terms of the number of Until operators, thereby quantifying the loss of expressivity that is caused by flattening. We expect that the restriction to the flat fragment will have no repercussions on the use of temporal logic as a specification formalism: a single Until operator nested in the initial invariant of another can still be rewritten into flat form, and more will hardly ever be needed in practice.

These results about expressivity carry over to the branching time logics CTL*(U) and CTL(U) and their flat versions, as is shown in Section 3. Figure 3 summarises the results. Relation 1 was established in [EH86] while 4 and 7 can be proven in a similar fashion. The vertical relations 2 and 3 are obvious: without Next operator one cannot count individual states. In this article, we established 5 and 6.

$$
\begin{array}{ccc}
\mathrm{CTL}^* & \overset{(1)}{\gneq} & \mathrm{CTL} \\
{}^{(2)}\rotatebox{90}{$\gneq$} & & \rotatebox{90}{$\gneq$}{}^{(3)} \\
\mathrm{CTL}^*(\mathsf{U}) & \overset{(4)}{\gneq} & \mathrm{CTL}(\mathsf{U}) \\
{}^{(5)}\rotatebox{90}{$\gneq$} & & \rotatebox{90}{$\gneq$}{}^{(6)} \\
\mathrm{flatCTL}^*(\mathsf{U}) & \overset{(7)}{\gneq} & \mathrm{flatCTL}(\mathsf{U})
\end{array}
$$

Figure 3: Expressive powers

Most of Section 3 is devoted to an investigation of adequate behavioural equivalences for flatCTL*(U) and flatCTL(U), and their comparison. It turns out that the pattern that applies for the comparisons CTL* vs. CTL and CTL*(U) vs. CTL(U), namely that the star affects the expressive but not the distinghuising powers, breaks down in the case of the flat versions: we have shown an example of two states that are distinguished by a flatCTL*(U) formula, but that are equivalent under the equivalence induced by flatCTL(U). Indeed, it turns out that flatCTL*(U) has the same distinguishing power as

CTL*(U) and CTL(U). Figure 4 puts these results into perspective. The relations $1'$ and $4'$ are based on results from [BCG88], [DNV90] and [Dam96]. The vertical relations $2'$ and $3'$ are obvious again. The somewhat surprising $5'$ and $7'$ were proven in Section 3 of this article, and $6'$ follows from these and $4'$. The adequate behavioural equivalences are shown alongside. Note that the logics are interpreted over possibly infinite Kripke structures with certain restrictions on the branching degree.

$$
\begin{array}{ccccc}
\equiv_{\mathrm{bis}} & \mathrm{CTL}^* & \overset{(1')}{=} & \mathrm{CTL} & \equiv_{\mathrm{bis}} \\
 & {}^{(2')}\rotatebox{-90}{$\subsetneq$} & & \rotatebox{-90}{$\subsetneq$}^{(3')} & \\
\equiv_{\mathrm{stut}} & \mathrm{CTL}^*(\mathsf{U}) & \overset{(4')}{=} & \mathrm{CTL}(\mathsf{U}) & \equiv_{\mathrm{stut}} \\
 & {}^{(5')}\parallel & & \rotatebox{-90}{$\subsetneq$}^{(6')} & \\
\equiv_{\mathrm{flat}^*} & \mathrm{flatCTL}^*(\mathsf{U}) & \overset{(7')}{\subsetneq} & \mathrm{flatCTL}(\mathsf{U}) & \equiv_{\mathrm{flat}}
\end{array}
$$

Figure 4: Distinguishing powers

References

[BCG88] M.C. Browne, E.M. Clarke, and O. Grumberg. Characterizing finite Kripke structures in propositional temporal logic. *Journal of Theoretical Computer Science*, 59:115–131, 1988.

[BFG+91] A. Bouajjani, J.C. Fernandez, S. Graf, C. Rodriguez, and J. Sifakis. Safety for branching time semantics. In J. Leach Albert, B. Monien, and M. Rodríguez Artalejo, editors, *Automata, Languages and Programming*, number 510 in LNCS, pages 76–92, New York, 1991. Springer-Verlag.

[CES86] E.M. Clarke, E.A. Emerson, and A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, April 1986.

[Dam96] Dennis René Dams. *Abstract Interpretation and Partition Refinement for Model Checking*. PhD thesis, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands, July 1996.

[DJS95] Werner Damm, Bernhard Josko, and Rainer Schlör. Specification and verification of VHDL-based system-level hardware designs. In Egon Börger, editor, *Specification and Validation Methods*, International Schools for Computer Scientists, pages 331–409. Oxford University Press, Oxford, 1995.

[DN87] Rocco De Nicola. Extensional equivalences for transition systems. *Acta Informatica*, 24:211–237, 1987.

[DNV90] Rocco De Nicola and Frits Vaandrager. Three logics for branching bisimulation. In *1990 IEEE Fifth Annual Symposium on Logic in Computer Science*, pages 118–129, Los Alamitos, CA, 1990. IEEE Computer Society Press. Full version available as Report CS-R9012, Centre for Math. and Comp. Sc., Amsterdam.

[EH86] E.A. Emerson and J.Y. Halpern. Sometimes and not never revisited: On branching versus linear time. *Journal of the Association for Computing Machinery*, 33(1):151–178, 1986.

[EW96] Kousha Etessami and Thomas Wilke. An until hierarchy for temporal logic. In *Eleventh Annual IEEE Symposium on Logic in Computer Science*, pages 108–117, Los Alamitos, CA, 1996. IEEE TC-MFC, IEEE Computer Society Press.

[GK94] Orna Grumberg and Robert P. Kurshan. How linear can branching-time be? In *International Conference on Temporal Logic*, number 827 in LNAI, Bonn, Germany, 1994.

[GKP92] Ursula Goltz, Ruurd Kuiper, and Wojciech Penczek. Propositional temporal logics and equivalences. In W.R. Cleaveland, editor, *CONCUR '92*, number 630 in LNCS, pages 222–236, Berlin, 1992. Springer-Verlag.

[HM80] Matthew Hennessy and Robin Milner. On observing nondeterminism and concurrency. In J.W. de Bakker and J. van Leeuwen, editors, *Proc. of the Seventh International Colloquium on Automata Languages and Programming (ICALP)*, number 85 in LNCS, pages 299–309, Berlin, 1980. Springer-Verlag.

[Jan] Geert Janssen. ptl: propositional temporal logic (PTL) tautology checker. `http://www.es.ele.tue.nl/geert/research/research_ptl.html`.

[Jos90] Bernhard Josko. A context dependent equivalence relation between Kripke structures. In E.M. Clarke and R.P. Kurshan, editors, *Computer-Aided Verification*, number 531 in LNCS, pages 204–213, New York, 1990. Springer-Verlag. An extended version appeared in B. Josko, Modular Specification and Verification of Reactive Systems, Habilitationsschrift, Carl von Ossietzky University of Oldenburg, Germany, 1993.

[Lam83] L. Lamport. What good is temporal logic? In R.E.A. Mason, editor, *Information Processing 83*, pages 657–668. IFIP, Elsevier Science Publishers B.V. (North-Holland), 1983.

[Par81] D. Park. Concurrency and automata on infinite sequences. In Peter Deussen, editor, *Theoretical Computer Science*, number 104 in LNCS, pages 167–183, Berlin, 1981. Springer-Verlag.

[Ste] Step's propositional temporal validity checker. `http://rodin.stanford.edu/ptl/index.html`.

[Sti89] Colin Stirling. Comparing linear and branching time temporal logics. In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Temporal Logic in Specification*, number 398 in LNCS, pages 1–20, Berlin, 1989. Springer-Verlag.

[vBvES94] Johan van Benthem, Jan van Eijck, and Vera Stebletsova. Modal logic, transition systems and processes. *Journal of Logic and Computation*, 4(5):811–855, 1994.

A Proofs of Section 2

PROOF OF LEMMA 2.1.6. By induction on the structure of φ. The base case is easy. As to the induction step, we concentrate on the case that $\varphi = \mathsf{U}(q', \varphi')$; the other cases are straightforward (recall that $\varphi \in$ flatLTL(U) and hence cannot contain X's). Let $i > Udepth(\varphi)$ (so $i \geq 2$). We consider separately each of the three cases of the lemma.

1. Let $0 \leq k < |A|$.

   $\Rightarrow$ direction. Assume that $Y_{i+1}(k, \dots) \models \varphi$. By definition of satisfaction, this means that we can choose $l \geq 0$ such that $Y_{i+1}(k, \dots)(l, \dots) \models \varphi'$ and that for every $0 \leq j < l$, $Y_{i+1}(k, \dots)(j) \models q'$. We consider the following cases, distinguishing in which part of Y_{i+1} the eventuality is fulfilled.

   (a) $k + l < |A|$. Then by point 1 of the i.h., we have $Y_i(k, \dots)(l, \dots) \models \varphi'$. Because $k + l < |A|$ and $i \geq 1$, it easily follows that $Y_i(k, \dots)(j) = Y_{i+1}(k, \dots)(j)$ for every $0 \leq j < l$. Therefore we also have $Y_i(k, \dots)(j) \models q'$ for every $0 \leq j < l$. So $Y_{i+1}(k, \dots) \models \varphi$.

   (b) $|A| \leq k + l < |AA|$. Then by point 2 of the i.h., we have $Y_i(k + l - |A|, \dots) \models \varphi'$. So φ' holds in some state of the first A-block of Y_i. However, this does not necessarily mean that along $Y_i(k, \dots)$ the eventuality φ' is fulfilled: namely, it may be the case that $k+l-|A| < k$, i.e. the state of Y_i where φ' holds comes before $Y_i(k)$. Therefore, we "shift φ' forward" by one A-block using Lemma 2.0.4. By that lemma, (note that $i - 1 > Udepth(\varphi')$), we have that also $Y_i(k + l, \dots) \models \varphi'$. Because $k + l < |AA|$ and $i \geq 2$, it easily follows that $Y_i(k, \dots, k + l - 1) = Y_{i+1}(k, \dots, k + l - 1)$. Therefore we also have $Y_i(k, \dots)(j) \models q'$ for every $0 \leq j < l$. So $Y_i(k, \dots) \models \varphi$.

   (c) $|AA| \leq k+l < |A^{i+1}BA|$. Then by point 2 of the i.h., we have $Y_i(k+l-|A|, \dots) \models \varphi'$. In contrast to the previous case, we do not have to shift φ' forward as $k+l-|A|$ is guaranteed to be greater than k in this case. As for the intermediate states, we have by Property 2.1.3 that $Y_i(k, \dots)(j) \models q'$ for every $0 \leq j < l$. So $Y_i(k, \dots) \models \varphi$.

   (d) $|A^{i+1}BA| \leq k+l < |A^{i+1}BA^{i+1}C|$. Then by point 3 of the i.h., we have $Y_i(k+l-|AA|, \dots) \models \varphi'$. Again, by Property 2.1.3 we have $Y_i(k, \dots)(j) \models q'$ for every $0 \leq j < l$. So $Y_i(k, \dots) \models \varphi$.

   (e) $|A^{i+1}BA^{i+1}C| \leq k + l < |A^{i+1}BA^{i+1}CA|$. This implies that $0 \leq k + l - |X^{i+1}| < |A|$ (recall that $X_{i+1} = A^{i+1}BA^{i+1}C$), i.e. the eventuality φ' is fulfilled in the first A-block of Y_i. Like in case 1b, we use Lemma 2.0.4 (note that $i - 1 > Udepth(\varphi')$) to shift φ' forward to the second A-block of Y_i: $Y_i(k + l - |X_{i+1}| + |A|, \dots) \models \varphi'$. By Property 2.1.3 it easily follows that $Y_i(k, \dots) \models \varphi$.

   (f) $|A^{i+1}BA^{i+1}CA| \leq k + l$. I.e. $|A| \leq k + l - |X_{i+1}|$, meaning that φ is fulfilled somewhere beyond the first A-block of Y_i. Using Property 2.1.3, $Y_i(k, \dots) \models \varphi$ easily follows.

   $\Leftarrow$ direction.

   (a) $k + l < |A|$. Use point 1 of the i.h.

   (b) $|A| \leq k + l < |AA|$. Use point 2 of the i.h. and Lemma 2.0.4 to shift φ' *backward* from the third to the second A-block of Y_{i+1}.

   (c) $|AA| \leq k + l$. By choosing $l' := |X_{i+1}| + l$, we have $Y_{i+1}(k, \dots)(l', \dots) \models \varphi'$. By Property 2.1.3 it follows that $Y_{i+1}(k, \dots)(j) \models q'$ for every $0 \leq j < l'$. (In the previous two cases we could not use this "direct" argument because Property 2.1.3 could not be applied there.)

2. Let $|A| \leq k < |A^{i+1}BA|$.

   $\Rightarrow$-direction. Assume that $Y_{i+1}(k, \dots) \models \varphi$. l is defined in a similar way as in the previous case.

   (a) $|A| \leq k + l < |A^{i+1}BA|$. Use point 2 of the i.h.

   (b) $|A^{i+1}BA| \leq k + l < |A^{i+1}BA^{i+1}C|$. Use point 3 of the i.h. and Property 2.1.3.

   (c) $|A^{i+1}BA^{i+1}C| \leq k + l < |A^{i+1}BA^{i+1}CA|$. Use point 1 of the i.h. for $i - 1$ (to shift φ' forward to the first A-block of Y_{i-1}) and φ' and Property 2.1.3.

(d) $|A^{i+1}BA^{i+1}CA| \leq k+l$. If $k+l-|X_{i+1}| < k$, i.e. the state of Y_i where φ' holds comes before $Y_i(k)$, then use point 2 of the i.h. for $i-1$ (to shift φ' forward to Y_{i-1}) and φ' and Property 2.1.3. Otherwise, it is "direct", only using Property 2.1.3.

$\Leftarrow$-direction. Straightforward by now.

3. $|A^{i+1}BA| \leq k < |A^{i+1}BA^{i+1}C|$. Similar to the previous case. □

PROOF OF LEMMA 2.1.7 By induction on the structure of φ. The base case is easy. As to the induction step, we concentrate on the case that $\varphi = \mathsf{U}(q', \varphi')$; the other cases are straightforward (recall that $\varphi \in$ flatLTL(U) and hence cannot contain X's). Let $1 \leq h \leq i+1$ with $h > Udepth(\varphi)$ (so $h \geq 2$). Let $0 \leq k < |A|$.

$\Rightarrow$ direction. Assume that $(A^h BA^{i+1}CY_i)(k, \ldots) \models \varphi$. By definition of satisfaction, this means that we can choose $l \geq 0$ such that $(A^h BA^{i+1}CY_i)(k, \ldots)(l, \ldots) \models \varphi'$ and for every $0 \leq j < l$, $(A^h BA^{i+1}CY_i)(k, \ldots)(j) \models q'$. We distinguish the following cases.

1. $k+l < |A|$. Then by the i.h., $(A^h CY_i)(k, \ldots)(l, \ldots) \models \varphi'$. Because $k+l < |A|$ and $h \geq 1$, $(A^h CY_i)(k, \ldots)(j) = (A^h BA^{i+1}CY_i)(k, \ldots)(j)$ for every $0 \leq j < l$. Therefore, $(A^h CY_i)(k, \ldots)(j) \models q'$ for every such j. So $(A^h CY_i)(k, \ldots) \models \varphi$.
2. $|A| \leq k+l < |AA|$. Then $(A^{h-1}BA^{i+1}CY_i)(k-|A|, \ldots)(l, \ldots) \models \varphi'$ and therefore, by the i.h. (note that $h-1 > Udepth(\varphi')$ and $k+l-|A| < |A|$), $(A^{h-1}CY_i)(k-|A|, \ldots)(l, \ldots) \models \varphi'$ and thus $(A^h CY_i)(k, \ldots)(l, \ldots) \models \varphi'$. Similar as in the previous case we have $(A^h CY_i)(k, \ldots)(j) \models q'$ for every $0 \leq j < l$. So $(A^h CY_i)(k, \ldots) \models \varphi$.
3. $|AA| \leq k+l < |A^h BA|$. In this case we have by Lemma 2.1.6, point 2, that the corresponding position of Y_i also satisfies φ': $Y_i((i-h) \cdot |A| + k + l - |A|) \models \varphi'$. So, we can choose $l' \geq 0$ such that $(A^h CY_i)(k, \ldots)(l', \ldots) \models \varphi'$, namely $l' := |A^h C| + (i-h) \cdot |A| + l - |A|$. Using Property 2.1.3, we also have that $(A^h CY_i)(k, \ldots)(j) \models q'$ for every $0 \leq j < l'$. So $(A^h CY_i)(k, \ldots) \models \varphi$.
4. $|A^h BA| \leq k+l \leq |A^h BA^{i+1}C|$. In this case we have by Lemma 2.1.6, point 3, that the corresponding position of Y_i also satisfies φ': $Y_i((i-h) \cdot |A| + k + l - |AA|) \models \varphi'$. So, we can choose $l' \geq 0$ such that $(A^h CY_i)(k, \ldots)(l', \ldots) \models \varphi'$, namely $l' := |A^h C| + (i-h) \cdot |A| + l - |AA|$. Using Property 2.1.3, we also have that $(A^h CY_i)(k, \ldots)(j) \models q'$ for every $0 \leq j < l'$. So $(A^h CY_i)(k, \ldots) \models \varphi$.
5. $|A^h BA^{i+1}C| \leq k+l$. "Direct", only using property Property 2.1.3.

$\Leftarrow$ direction. The cases $k+l < |A|$ and $|A| \leq k+l < |AA|$ are similar to the corresponding cases for the $\Rightarrow$ direction. The case $|AA| \leq k+l$ is "direct", only using Property 2.1.3. □

B Proofs of Section 3

PROOF OF LEMMA 3.2.16. By induction on the structure of the formula.

- Base: $\varphi \in$ Prop. $s \equiv_{\text{flat}} t$ implies that $\mathcal{L}(s) = \mathcal{L}(t)$. From this it follows that $s \models p$ iff $t \models p$ for all $p \in$ Prop.
- Induction step:
  1. The cases that φ is a negation or conjunction are straightforward.
  2. $\varphi = \exists\mathsf{U}(p, \varphi')$. Assume that $s \models \varphi$. By Definition 3.0.3, this means that we can choose an s-path $\bar{s}$ and $n \geq 0$ such that $\bar{s}(n) \models \varphi'$ and for every $0 \leq i < n$, $\bar{s}(i) \models p$. From $s \equiv_{\text{flat}} t$ and clause 2a in Definition 3.2.15, we can prove that there exists a t-path $\bar{t}$ and $m \geq 0$ such that $\bar{s}(n) \equiv_{\text{flat}} \bar{t}(m)$ and for every $0 \leq j < m$, there exists $0 \leq i < n$ such that $\bar{t}(j) \equiv^0 \bar{s}(i)$. By the induction hypothesis, it follows from $\bar{s}(n) \models \varphi'$ and $\bar{s}(n) \equiv_{\text{flat}} \bar{t}(m)$ that $\bar{t}(m) \models \varphi'$. Because for every $0 \leq j < m$, $\bar{t}(j)$ is $\equiv^0$-equivalent to some $\bar{s}(i)$ with $0 \leq i < n$, and for every such i we have $\bar{s}(i) \models p$, we also have $\bar{t}(j) \models p$ for every $0 \leq j < m$. Hence, $t \models \exists\mathsf{U}(p, \varphi')$.

3. $\varphi = \exists\neg\mathsf{U}(p, \varphi')$. Assume that $s \models \varphi$. By Definition 3.0.3, this means that we can choose an s-path $\bar{s}$ such that $\bar{s} \not\models \mathsf{U}(p, \varphi')$. We have to show that there exists a t-path $\bar{t}$ such that $\bar{t} \not\models \mathsf{U}(p, \varphi')$. Suppose (*) that this is *not* the case; we will derive a contradiction. Because $s \equiv_{\text{flat}} t$, by clause 2b in Definition 3.2.15, we can choose a t-path $\bar{t}$ such that: for every $l \geq 0$ such that $\bar{t}(0) \xrightarrow{\equiv^0} \bar{t}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{t}(l-1) \to \bar{t}(l)$ there exists $k \geq 0$ such that $\bar{s}(0) \xrightarrow{\equiv^0} \bar{s}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{s}(k-1) \to \bar{s}(k)$ and $\bar{s}(k) \equiv \bar{t}(l)$. By our assumption (*), $\bar{t} \models \mathsf{U}(p, \varphi')$, i.e. we can choose $m \geq 0$ such that $\bar{t}(m) \models \varphi'$ and for all $0 \leq j < m$, $\bar{t}(j) \models p$. From clause 2b in Definition 3.2.15, we can prove that we can choose $k' \geq 0$ such that $\bar{s}(k') \equiv_{\text{flat}} \bar{t}(m)$ and for every $0 \leq i < k'$, there exists $0 \leq j < m$ such that $\bar{s}(i) \equiv^0 \bar{t}(j)$. By the induction hypothesis, it follows from $\bar{t}(m) \models \varphi'$ and $\bar{s}(k') \equiv_{\text{flat}} \bar{t}(m)$ that $\bar{s}(k') \models \varphi'$. Because for every $0 \leq i < k'$, $\bar{s}(i)$ is $\equiv^0$-equivalent to some $\bar{t}(j)$ with $0 \leq j < m$, and for every such j we have $\bar{t}(j) \models p$, we also have $\bar{s}(i) \models p$ for every $0 \leq i < k'$. Hence, $\bar{s} \models \mathsf{U}(p, \varphi')$. Contradiction.

PROOF OF LEMMA 3.2.17. Assume that $\forall_{\varphi \in \text{flatCTL}(\mathsf{U})}\ s \models \varphi \Leftrightarrow t \models \varphi$. We have to show that $s \equiv_{\text{flat}} t$. Because $\equiv_{\text{flat}}$ is the largest flat equivalence, we have to show that the pair (s, t) is an element of some flat equivalence $\equiv \subseteq \Sigma \times \Sigma$. We define this relation as follows: $u \equiv v$ iff $\forall_{\varphi \in \text{flatCTL}(\mathsf{U})}\ u \models \varphi \Leftrightarrow v \models \varphi$. Clearly $s \equiv t$. We show that $\equiv$ is a flat equivalence.

1. It is trivial that $\mathcal{L}(s) = \mathcal{L}(t)$.
2. (a) Suppose that $s \xrightarrow{\equiv^0} s_1 \xrightarrow{\equiv^0} \ldots \xrightarrow{\equiv^0} s_{k-1} \to s_k$ with $k \geq 0$. We have to show that there exists $t \xrightarrow{\equiv^0} t_1 \xrightarrow{\equiv^0} \ldots \xrightarrow{\equiv^0} t_{l-1} \to t_l$ with $l \geq 0$ and $s_k \equiv t_l$. Suppose (*) that this is *not* the case. Consider the set $T = \{\langle t', t''\rangle \mid t \xrightarrow{\equiv^0}{}^* t' \to t''\}$. Because $\to$ is total (by assumption), T is nonempty. Because T is finitely branching under Prop-stuttering and also (plainly) finitely branching, T is finite, say $T = \{\langle t_1', t_1''\rangle, \ldots, \langle t_n', t_n''\rangle\}$. For every $1 \leq i \leq n$, choose formulae $p_i \in \text{Prop}$ and $\varphi_i \in \text{flatCTL}(\mathsf{U})$ as follows.
   - $p_i = \mathit{true}$ if $t_i' \equiv^0 t_i''$; otherwise, choose p_i such that $t_i' \models p_i$ (and hence $s \models p_i$) and $t_i'' \not\models p_i$, which is possible by definition of $\equiv^0$.
   - choose φ_i such that $s_k \models \varphi_i$ and $t_i'' \not\models \varphi_i$ — this is possible by our assumption (*).

   Define $\varphi = \exists\mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_1 \wedge \cdots \wedge \varphi_n)$. Then clearly $s \models \varphi$. Next, we show that $t \not\models \varphi$. Suppose that, conversely, $t \models \varphi$, i.e. we can choose a t-path $\bar{t}$ such that $\bar{t} \models \mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_1 \wedge \cdots \wedge \varphi_n)$. This means that we can choose $l \geq 0$ such that $\bar{t}(l) \models \varphi_1 \wedge \cdots \wedge \varphi_n$ while for every $0 \leq j < l$, $\bar{t}(j) \models p_1 \wedge \cdots \wedge p_n$. On the other hand, every t-path contains one of the t_i''; in particular, by definition of T, we can choose i such that either $\bar{t}(l)$ is t_i'', or we can choose $0 \leq j < l$ such that $\bar{t}(j)$ is t_i''. But in the first case, we have $\bar{t}(l) \not\models \varphi_i$, implying $\bar{t}(l) \not\models \varphi_1 \wedge \cdots \wedge \varphi_n$ and in the second case we have $\bar{t}(j) \not\models p_i$, implying $\bar{t}(j) \not\models p_1 \wedge \cdots \wedge p_n$. In both cases we have a contradiction. So we conclude that $t \not\models \varphi$. But then $s \not\equiv t$, as $\varphi \in \text{flatCTL}(\mathsf{U})$. Contradiction.

   (b) Let $\bar{s} \in \mathit{paths}(s)$. We have to show that there exists $\bar{t} \in \mathit{paths}(t)$ such that: for every $l \geq 0$ such that $\bar{t}(0) \xrightarrow{\equiv^0} \bar{t}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{t}(l-1) \to \bar{t}(l)$ there exists $k \geq 0$ such that $\bar{s}(0) \xrightarrow{\equiv^0} \bar{s}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{s}(k-1) \to \bar{s}(k)$ and $\bar{s}(k) \equiv \bar{t}(l)$. Suppose (*) that this is *not* the case. This means that for every $\bar{t} \in \mathit{paths}(t)$, we can choose $l \geq 0$ such that: $\bar{t}(0) \xrightarrow{\equiv^0} \bar{t}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{t}(l-1) \to \bar{t}(l)$ and for every $k \geq 0$, it is *not* the case that $\bar{s}(0) \xrightarrow{\equiv^0} \bar{s}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{s}(k-1) \to \bar{s}(k)$ or it is *not* the case that $\bar{s}(k) \equiv \bar{t}(l)$. Consider the set T of pairs $(\bar{t}(l-1), \bar{t}(l))$ for all such l (if $l = 0$, then take the pair $(\bar{t}(l), \bar{t}(l))$). Because T is finitely branching under Prop-stuttering and also (plainly) finitely branching, T is finite, say $T = \{\langle t_1', t_1''\rangle, \ldots, \langle t_n', t_n''\rangle\}$. For every $1 \leq i \leq n$, choose formulae $p_i \in \text{Prop}$ and $\varphi_i \in \text{flatCTL}(\mathsf{U})$ as follows.
   - choose φ_i such that $t_i'' \models \varphi_i$ and for every $k \geq 0$ such that $\bar{s}(0) \xrightarrow{\equiv^0} \bar{s}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{s}(k-1) \to \bar{s}(k)$, we have $s_k \not\models \varphi_i$ — this is possible by our assumption (*).
   - if for every $0 \leq k$, $\bar{s}(k) \not\models \varphi_i$, then define $p_i = \mathit{true}$; otherwise, if $\bar{s}(k) \models \varphi_i$ for some $0 \leq k$, then by the definition of the φ_i in the previous point there exists $0 \leq i \leq k-2(?)$ such that $\bar{s}(i) \not\equiv^0 \bar{s}(i+1)$; hence we can define p_i such that $\bar{s}(i) \models p_i$ and $\bar{s}(i+1) \not\models p_i$:(and hence $s \models p_i$) and $t_i'' \not\models p_i$.

Define $\varphi = \exists\neg\mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_1 \vee \cdots \vee \varphi_n)$. Then, by the definition of the p_i and φ_i, we have for every $\bar{t} \in \mathit{paths}(t)$: $\bar{t} \models \mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_1 \vee \cdots \vee \varphi_n)$, and thus $t \not\models \varphi$. Next, we show that $s \models \varphi$, by showing that $\bar{s} \models \neg\mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_1 \vee \cdots \vee \varphi_n)$. Namely, suppose that, conversely, $\bar{s} \models \mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_1 \vee \cdots \vee \varphi_n)$, i.e. we can choose $k \geq 0$ and $1 \leq i \leq n$ such that $\bar{s}(k) \models \varphi_i$ while for every $0 \leq i' < k$, $\bar{s}(i') \models p_1 \wedge \cdots \wedge p_n$. But the fact that $\bar{s}(k) \models \varphi_i$ means, by definition of the φ_i, that it cannot be the case that $\bar{s}(0) \xrightarrow{\equiv^0} \bar{s}(1) \xrightarrow{\equiv^0} \cdots \xrightarrow{\equiv^0} \bar{s}(k-1) \rightarrow \bar{s}(k)$. Hence, by the definition of the p_i, there must be $0 \leq i \leq k-2$ such that $\bar{s}(i+1) \not\models p_i$. This contradicts the fact that for every $0 \leq i' < k$, $\bar{s}(i') \models p_1 \wedge \cdots \wedge p_n$. Therefore, we conclude that $\bar{s} \models \neg\mathsf{U}(p_1 \wedge \cdots \wedge p_n, \varphi_1 \vee \cdots \vee \varphi_n)$, i.e., $s \models \varphi$. But then $s \not\equiv t$, as $\varphi \in$ flatCTL(U). Contradiction. $\square$

Computing Science Reports | **Department of Mathematics and Computing Science**
Eindhoven University of Technology

In this series appeared:

96/01	M. Voorhoeve and T. Basten	Process Algebra with Autonomous Actions, p. 12.
96/02	P. de Bra and A. Aerts	Multi-User Publishing in the Web: DreSS, A Document Repository Service Station, p. 12
96/03	W.M.P. van der Aalst	Parallel Computation of Reachable Dead States in a Free-choice Petri Net, p. 26.
96/04	S. Mauw	Example specifications in phi-SDL.
96/05	T. Basten and W.M.P. v.d. Aalst	A Process-Algebraic Approach to Life-Cycle Inheritance Inheritance = Encapsulation + Abstraction, p. 15.
96/06	W.M.P. van der Aalst and T. Basten	Life-Cycle Inheritance A Petri-Net-Based Approach, p. 18.
96/07	M. Voorhoeve	Structural Petri Net Equivalence, p. 16.
96/08	A.T.M. Aerts, P.M.E. De Bra, J.T. de Munk	OODB Support for WWW Applications: Disclosing the internal structure of Hyperdocuments, p. 14.
96/09	F. Dignum, H. Weigand, E. Verharen	A Formal Specification of Deadlines using Dynamic Deontic Logic, p. 18.
96/10	R. Bloo, H. Geuvers	Explicit Substitution: on the Edge of Strong Normalisation, p. 13.
96/11	T. Laan	AUTOMATH and Pure Type Systems, p. 30.
96/12	F. Kamareddine and T. Laan	A Correspondence between Nuprl and the Ramified Theory of Types, p. 12.
96/13	T. Borghuis	Priorean Tense Logics in Modal Pure Type Systems, p. 61
96/14	S.H.J. Bos and M.A. Reniers	The I^2C-bus in Discrete-Time Process Algebra, p. 25.
96/15	M.A. Reniers and J.J. Vereijken	Completeness in Discrete-Time Process Algebra, p. 139.
96/17	E. Boiten and P. Hoogendijk	Nested collections and polytypism, p. 11.
96/18	P.D.V. van der Stok	Real-Time Distributed Concurrency Control Algorithms with mixed time constraints, p. 71.
96/19	M.A. Reniers	Static Semantics of Message Sequence Charts, p. 71
96/20	L. Feijs	Algebraic Specification and Simulation of Lazy Functional Programs in a concurrent Environment, p. 27.
96/21	L. Bijlsma and R. Nederpelt	Predicate calculus: concepts and misconceptions, p. 26.
96/22	M.C.A. van de Graaf and G.J. Houben	Designing Effective Workflow Management Processes, p. 22.
96/23	W.M.P. van der Aalst	Structural Characterizations of sound workflow nets, p. 22.
96/24	M. Voorhoeve and W. van der Aalst	Conservative Adaption of Workflow, p.22
96/25	M. Vaccari and R.C. Backhouse	Deriving a systolic regular language recognizer, p. 28
97/01	B. Knaack and R. Gerth	A Discretisation Method for Asynchronous Timed Systems.
97/02	J. Hooman and O. v. Roosmalen	A Programming-Language Extension for Distributed Real-Time Systems, p. 50.
97/03	J. Blanco and A. v. Deursen	Basic Conditional Process Algebra, p. 20.
97/04	J.C.M. Baeten and J.A. Bergstra	Discrete Time Process Algebra: Absolute Time, Relative Time and Parametric Time, p. 26.
97/05	J.C.M. Baeten and J.J. Vereijken	Discrete-Time Process Algebra with Empty Process, p. 51.
97/06	M. Franssen	Tools for the Construction of Correct Programs: an Overview, p. 33.
97/07	J.C.M. Baeten and J.A. Bergstra	Bounded Stacks, Bags and Queues, p. 15.

97/08	P. Hoogendijk and R.C. Backhouse	When do datatypes commute? p. 35.
97/09	Proceedings of the Second International Workshop on Communication Modeling, Veldhoven, The Netherlands, 9-10 June, 1997.	Communication Modeling- The Language/Action Perspective, p. 147.
97/10	P.C.N. v. Gorp, E.J. Luit, D.K. Hammer E.H.L. Aarts	Distributed real-time systems: a survey of applications and a general design model, p. 31.
97/11	A. Engels, S. Mauw and M.A. Reniers	A Hierarchy of Communication Models for Message Sequence Charts, p. 30.
97/12	D. Hauschildt, E. Verbeek and W. van der Aalst	WOFLAN: A Petri-net-based Workflow Analyzer, p. 30.
97/13	W.M.P. van der Aalst	Exploring the Process Dimension of Workflow Management, p. 56.
97/14	J.F. Groote, F. Monin and J. Springintveld	A computer checked algebraic verification of a distributed summation algorithm, p. 28
97/15	M. Franssen	λP-: A Pure Type System for First Order Loginc with Automated Theorem Proving, p.35.
97/16	W.M.P. van der Aalst	On the verification of Inter-organizational workflows, p. 23
97/17	M. Vaccari and R.C. Backhouse	Calculating a Round-Robin Scheduler, p. 23.
97/18	Werkgemeenschap Informatiewetenschap redactie: P.M.E. De Bra	Informatiewetenschap 1997 Wetenschappelijke bijdragen aan de Vijfde Interdisciplinaire Conferentie Informatiewetenschap, p. 60.
98/01	W. Van der Aalst	Formalization and Verification of Event-driven Process Chains, p. 26.
98/02	M. Voorhoeve	State / Event Net Equivalence.
98/03	J.C.M. Baeten and J.A. Bergstra	Deadlock Behaviour in Split and ST Bisimulation Semantics, p. 15.
98/04	R.C. Backhouse	Pair Algebras and Galois Connections, p. 14