Master's Thesis

V.R. Sathiyanarayana
Chennai Mathematical Institute, India

Abstract

My master's thesis!

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

Keywords and phrases Logic, Modal and temporal logic

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Acknowledgements I want to thank my advisors Dr. Paul Gastin and Dr. Aiswarya Cyriac for helping me through this project

1 Introduction

The separation property, invented by Dov Gabbay, is a strangely influential consequence of the design of popular temporal languages. Simply put, it requires all formulas in the language to be equivalent to a variant made up of formulas purely concerned with one *region* of the flow of time. Surprisingly, this property is linked to expressive completeness: a sufficiently expressive temporal language with the separation property can express any first-order property. In the next few sections, we will detail the separation property over linear time, its consequences on functional completeness, and discuss the generalization described in [3].

2 Preliminaries

Before discussing separation, we define some standard notions. A flow of time is simply a non-empty set T partially ordered by the binary relation $<$. We symbolically refer to these flows by the pair $(T, <)$. Examples include $(\mathbb{N}, <)$ and $(\mathbb{R}, <)$ with their natural ordering, unordered trees with the descendant relation, and Mazurkiewicz traces. We will consider the truth values of propositions (from a fixed set $\mathcal{P}$) at points on these flows.

The first-order vocabulary over these structures contains the ordering relation $<$ and a collection of *monadic* relations $Q_1, Q_2, \cdots$ that match the propositions $q_1, q_2, \cdots$ in $\mathcal{P}$. An assignment h of atoms in a time flow $(T, <)$ assigns to each Q_i a subset of T where the atom q_i is true. Augmented with the assignment, the triplet $(T, <, h)$ is called a *temporal structure*. First-order formulas are evaluated over these structures in the usual way. In this discussion, we pay special attention to first-order formulas with a single free-variable; they quite naturally mirror temporal formulas.

Instead of free variables and quantification, temporal languages employ *connectives* to reason through time. Popular connectives include F, P, G, H, U, and S, known as *future*, *past*, *globally*, *history*, *until* and *since* respectively. In this paper, we will limit our discussion to connectives that are definable by monadic first-order formulas.

Temporal formulas are evaluated at points in time. In a temporal structure $\mathcal{M} = (T, <, h)$, atoms are evaluated as

$$\mathcal{M}, t \vDash p \iff (T, <, h[x \mapsto t]) \vDash p(x) \iff t \in h(p)$$

As per the standard notation, the assignment $h[x \mapsto t]$ assigns the time point t to the first-order variable x. To simplify the presentation, we use $\mathcal{M}, t \vDash \varphi(t)$ to mean $(T, <, h[x \mapsto t]) \vDash \varphi(x)$.

For a generic connective $\sharp$ of arity n, let $\varphi_\sharp(t, X_1, \cdots X_n)$ be the monadic first-order formula defining it. Here, t is the point in time that the connective is evaluated at, and the X_i are monadic (second-order) variables. These variables expect a single-variable first-order formula, as shown below

$$\mathcal{M}, t \vDash \sharp(A_1, \cdots A_n) \Longleftrightarrow \mathcal{M}, t \vDash \varphi_\sharp(t, \alpha_{A_1}, \cdots \alpha_{A_n})$$

Here, A_i are temporal formulas and α_{A_i} are their first-order translations. Notably, $\varphi_\sharp$ can only quantify over elements in the domain T; it cannot use second order quantifiers.

We illustrate this behaviour with an example. The connective F is defined by the formula

$$\varphi_F(t, X) \triangleq \exists x. (t < x) \wedge X(x)$$

Hence, we have

$$\mathcal{M}, t \vDash Fp \Longleftrightarrow \mathcal{M}, t \vDash \exists y. (t < y) \wedge p(y)$$

We similarly define the other main connectives

$$\begin{aligned}
\varphi_P(t, X) &\triangleq \exists x. (x < t) \wedge X(x) \\
\varphi_G(t, X) &\triangleq \forall x. (t < x) \wedge X(x) \\
\varphi_H(t, X) &\triangleq \forall x. (x < t) \wedge X(x) \\
\varphi_U(t, X_1, X_2) &\triangleq \exists x. \ [(t < x) \wedge X_1(x) \wedge \forall y \, ((t < y < x) \rightarrow X_2(y))] \\
\varphi_S(t, X_1, X_2) &\triangleq \exists x. \ [(x < t) \wedge X_1(x) \wedge \forall y \, ((x < y < t) \rightarrow X_2(y))]
\end{aligned}$$

Note that, unlike the typical definition of U, φ_U doesn't rely on the present point t. Such an until is referred to in the literature by either the *strict* until (see [2]) or the *strong* until (see [1]). This particular behaviour makes observing separation much easier.

▶ **Definition 2.1** (Expressive Completeness). *A temporal language is **first-order expressively complete** over a class of time flows iff there exists a temporal formula A for any first-order formula with one free variable $\varphi(t)$ such that*

$$\mathcal{M}, t \vDash A \Longleftrightarrow \mathcal{M}[x \mapsto t] \vDash \varphi(x)$$

for any flow $\mathcal{M}$ in the class.

On a related note, a flow of time $(T, <)$ is termed to be expressively complete if there exists an expressively complete temporal language over it.

3 Linear Flows

In [3], Gabbay showed how the temporal language $\mathbf{L}$ with the strict until U and since S connectives satisfies the separation property over the integer time flow $(\mathbb{Z}, <)$.

To discuss this further, we need the notion of *regions* and *pure formulas*. Informally, the flow of time $(T, <)$ is partitioned into a set of regions. The positions of these regions depends on the position of the time point t where the temporal formula is being evaluated. For linear flows, Gabbay selected three regions:

- The *past* of t, formally defined as $\{x \mid x \in \mathbb{Z} \wedge x < t\}$.
- The *present*, which is simply $\{t\}$.
- The *future* of t, which naturally is $\{x \mid x \in \mathbb{Z} \wedge t < x\}$

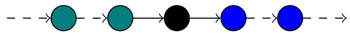

Figure 1 Regions for linear separation. The present is black, the past is green, and the future is blue.

Note that these regions are disjoint, and that the union of these regions produces the entire flow. Also, notice that these regions are first-order definable.

Now, we define *pure formulas*. For any flow $(T, <)$, we denote two assignments h and h' to be in *agreement* over a region $R \subset T$ *iff* for any atom $q \in \mathcal{P}$ and any point $s \in R$,

$$s \in h(q) \iff s \in h'(q)$$

Now, call a temporal formula A *pure* with respect to a region R *iff* for any two assignments h and h' that agree on R,

$$(T, <, h), t \vDash A \iff (T, <, h'), t \vDash A$$

In other words, A is true on h' *iff* A is true on h. We use the terms *pure past*, *pure present*, or *pure future* to denote pure formulas in the past, present, and future regions respectively.

It's easy to see that formulas that don't use the S and U connectives are pure-present. In a similar vein, formulas that are rooted by a S connective and don't use S connectives are pure-past. We formalize this understanding using the notion of *syntactically pure* formulas. To simplify presentation, we refer to formulas rooted by a U (or an S) as U-formulas (or S-formulas, respectively).

▶ **Definition 3.1.** *A temporal formula φ in the temporal language of S and U is*
1. *syntactically pure-present iff it doesn't use the S and U connectives.*
2. *syntactically pure-past iff it is a boolean combination of S-formulas that don't use the U connective.*
3. *syntactically pure-future iff it is a boolean combination of U-formulas that don't use the S connective.*

Finally, call a formula A **separated** if it is a boolean combination of pure formulas. Now, we can state the separation property

▶ **Theorem 3.2** (Separation Property for linear flows)**.** *Every temporal formula A in the language of S and U over linear time can be equivalently represented by a separated formula.*

The proof of this theorem is quite involved, and is presented in full detail in [3]. In the next few sections, I'll give a high-level overview of Gabbay et. al.'s scheme. To mirror their notation, I'll write U formulas as $U(p, q)$ instead of $q\,\mathcal{U}\,p$.

3.1 Separating S and U over linear time

As a reminder, we restate the definitions of U and S

$$\mathcal{M}, t \vDash U(p, q) \iff \mathcal{M}, t \vDash \exists x.\,(t < x) \wedge p(x) \wedge \forall y\,(t < y < x \to q\,(y))$$
$$\mathcal{M}, t \vDash S(p, q) \iff \mathcal{M}, t \vDash \exists x.\,(x < t) \wedge p(x) \wedge \forall y\,(x < y < t \to q\,(y))$$

For convenience, we refer to the left condition (p) in $U(p, q)$ as the *target* condition and the right condition (q) as the *path* condition. Observe that, over linear time, a formula composed only of Us is a pure future formula, a formula composed of Ss is a pure past formula. The task, therefore, is to transform formulas with both Us and Ses.

Over the integer time flow $(\mathbb{Z}, <)$, these connectives naturally possess the following properties

$$\begin{aligned} &U(\alpha \vee \beta, \gamma) \equiv U(\alpha, \gamma) \vee U(\beta, \gamma) \\ &U(\alpha, \beta \wedge \gamma) \equiv U(\alpha, \beta) \wedge U(\alpha, \gamma) \end{aligned} \tag{1}$$

In addition, their negations can be usefully rewritten as

$$\begin{aligned} \neg U(\alpha, \beta) &\equiv G(\neg\alpha) \vee U(\neg\alpha \wedge \neg\beta, \neg\alpha) \\ \neg S(\alpha, \beta) &\equiv H(\neg\alpha) \vee S(\neg\alpha \wedge \neg\beta, \neg\alpha) \end{aligned}$$

where the semantics of G and H are

$$\begin{aligned} \mathcal{M}, t \vDash G(\alpha) &\iff \mathcal{M}, t \vDash \forall t'.\, t' > t \rightarrow \varphi_\alpha(t') \\ \mathcal{M}, t \vDash H(\alpha) &\iff \mathcal{M}, t \vDash \forall t'.\, t' < t \rightarrow \varphi_\alpha(t') \end{aligned}$$

Here, φ_α is the first-order translation of α.

Our strategy involves *pulling-out* Us from inside S and vice versa. We accomplish this by writing all temporal formulas in a standard notation, and then applying a sequence of *elimination* rules. In the next section, we describe these rules.

3.1.1 Eliminations

Let α, β, φ and ψ be boolean combinations of propositional atoms. In the following subsections, we pull out a $U(\varphi, \psi)$ from inside a S under a variety of minimal configurations. In later sections, we show that these configurations suffice.

$S(\alpha \wedge U(\varphi, \psi), \beta)$

This formula requires $U(\varphi, \psi)$ to be true at a point t' in the past of t. This in turn implies φ at some point t'' ahead of t'. This naturally breaks down into three cases: $t'' > t$, $t'' = t$, and $t' < t'' < t$. The translation is

$$\begin{aligned} & S(\varphi \wedge \beta \wedge S(\alpha, \psi \wedge \beta), \beta) \\ \vee \quad & (S(\alpha, \psi \wedge \beta) \wedge (\varphi \vee (\psi \wedge U(\varphi, \psi)))) \end{aligned}$$

$S(\alpha \wedge \neg U(\varphi, \psi), \beta)$

In this case, we immediately rewrite $\neg U(\varphi, \psi)$ as $G(\neg\alpha) \vee U(\neg\alpha \wedge \neg\beta, \neg\alpha)$. This gives us

$$\begin{aligned} & S(\alpha \wedge \neg U(\varphi, \psi), \beta) \equiv \\ & \qquad S(\alpha \wedge G(\neg\varphi), \beta) \\ & \quad \vee \quad S(\alpha \wedge U(\neg\varphi \wedge \neg\psi, \neg\varphi), \beta) \end{aligned}$$

where each individual case can be translated using the ideas used to rewrite $S(\alpha \wedge U(\varphi, \psi), \beta)$.

$S(\alpha, U(\varphi, \psi))$

It's instructive to recognize how $S(\alpha, U(\varphi, \psi))$ could be translated. Unlike the previous cases, the Until fragment needs to be true at each point in the path to α. This could involve multiple segments in this path where ψ is true till φ is true. Wonderfully, this is *indistinguishable* from the case where, at each point in the path, either φ or ψ is true. This formula is translated to

$$\begin{array}{ll} & S(\alpha, \bot) \\ \vee & S(\alpha, \varphi \vee \psi) \wedge [\varphi \vee (\psi \wedge U(\varphi, \psi))] \end{array}$$

Here, $S(\alpha, \bot)$ can only be true if α is true at the previous point. Otherwise, we'll need $U(\varphi, \psi)$ to be satisfied at the previous location, hence the $\varphi \vee (\psi \wedge U(\varphi, \psi))$ at the present. At each point t' in the path to α, if $t' + 1 \vDash \varphi$, $t' \vDash U(\varphi, \psi)$. Otherwise, $t' + 1 \vDash \psi$. At this point, we can use an inductive argument, starting from the previous point, to prove the correctness of this translation.

$S(\alpha, \beta \vee U(\varphi, \psi))$

The idea is to attempt to enforce $U(\varphi, \psi)$ at each point in the path *iff* we can detect an earlier point in the path which needed to satisfy it. A simple way to detect these points is to look for the moment where $\neg\beta$ was true, and check whether, along the way to that point, $\neg\varphi$ was true at each step. Accordingly, $S(\neg\beta \wedge \neg\alpha, \neg\varphi \wedge \neg\alpha)$ does the trick. Here, the $\neg\alpha$ is to ensure that we specifically look for points in the future of α, the leftmost point in our consideration.

It's important to recognize that we are capable of recognizing such points at each step of the path to α. This means that, if we recognized such a point that's 3 steps away, we recognized it at 2 and 1 step away too. This allows us a simple fix: $S(\neg\beta, \neg\varphi \wedge \neg\alpha) \rightarrow \varphi \vee \psi$. If φ was true, we will not see this point in our next search. Otherwise, ψ would be true, allowing for the possibility of enforcement in the future.

The overall translation now is

$$\begin{array}{ll} & S(\alpha, \neg\alpha \wedge (S(\neg\beta \wedge \neg\alpha, \neg\varphi \wedge \neg\alpha) \rightarrow \varphi \vee \psi)) \\ \wedge & S(\neg\beta \wedge \neg\alpha, \neg\varphi \wedge \neg\alpha) \rightarrow (\varphi \vee (\psi \wedge U(\varphi, \psi))) \end{array}$$

$S(\alpha, \beta \vee \neg U(\varphi, \psi))$

This case is very similar to the previous case. The points we search for must be in danger of satisfying $U(\varphi, \psi)$; hence, we look for $S(\neg\beta \wedge \neg\alpha, \psi \wedge \neg\alpha)$. We fix these points by requiring φ to be false. In the worst-case, we've dragged on the possible *until* to the present, at which point we can extinguish all hope. This gives us the overall translation:

$$\begin{array}{ll} & S(\alpha, \neg\alpha \wedge (S(\neg\beta \wedge \neg\alpha, \psi \wedge \neg\alpha) \rightarrow \not\varphi)) \\ \wedge & S(\neg\beta \wedge \neg\alpha, \psi \wedge \neg\alpha) \rightarrow ((\neg\psi \wedge \neg\varphi) \vee (\neg U(\varphi, \psi))) \end{array}$$

$S(\alpha \wedge U(\varphi, \psi), \beta \vee U(\varphi, \psi))$

This is a neat combination of $S(\alpha \wedge U(\varphi, \psi), \beta)$ and $S(\alpha, \beta \vee U(\varphi, \psi))$. The translation is simple. *I believe Gabbay made a typo in this particular example. [4] mentions this.*

$$\begin{array}{lll} & & S(\alpha, \psi) \wedge (\varphi \vee (\psi \wedge U(\varphi, \psi))) \\ \vee & & S(\varphi \wedge S(\alpha, \psi), S(\neg\beta, \neg\varphi) \rightarrow \varphi \vee \psi) \\ & \wedge & S(\neg\beta, \neg\varphi) \rightarrow (\varphi \vee (\psi \wedge U(\varphi, \psi))) \end{array}$$

3.1.2 Putting it all together

The eliminations presented in the previous section lend credence to the idea of separation. Amazingly, Gabbay presents a neat induction scheme that builds on these rules to separate *any* temporal formula in the language. In this section, we present an overview of his arguments (presented in more detail in [3]).

▶ **Lemma 3.3.** *Let φ and ψ be pure-present formulas and α and β be formulas such that the only appearance of a U in either of them is $U(\varphi, \psi)$, and that U isn't nested inside a S. Then $S(\alpha, \beta)$ can be written as a syntactically separated formula where the only appearance of U is $U(\varphi, \psi)$.*

Proof. We start by writing α and β in their conjunctive and disjunctive normal forms respectively. During this transformation, we treat all top-level instances of U and S in them as atomic propositions. This gives us

$$\alpha \equiv \bigvee_i (\alpha_{i,1} \wedge \alpha_{i,2} \wedge \cdots \wedge \alpha_{i,m_i})$$
$$\beta \equiv \bigwedge_j (\beta_{j,1} \vee \beta_{j,2} \vee \cdots \vee \beta_{j,n_j})$$

Here, the literals $\alpha_{i,k}$ and $\beta_{i,k}$ are composed of propositional atoms, S formulas, and $U(\varphi, \psi)$. We use the above and equation (1) to write $S(\alpha, \beta)$ as

$$\begin{aligned} S(\alpha, \beta) &\longmapsto S\left(\bigvee_i (\alpha_{i,1} \wedge \cdots \wedge \alpha_{i,m_i}), \beta\right) \\ &\longmapsto \bigvee_i S(\alpha_{i,1} \wedge \cdots \wedge \alpha_{i,m_i}, \beta) \\ &\longmapsto \bigvee_i S\left(\alpha_{i,1} \wedge \cdots \wedge \alpha_{i,m_i}, \bigwedge_j (\beta_{j,1} \vee \cdots \vee \beta_{j,n_i})\right) \\ &\longmapsto \bigvee_i \bigwedge_j S\left(\alpha_{i,1} \wedge \cdots \wedge \alpha_{i,m_i}, \beta_{j,1} \vee \cdots \vee \beta_{j,n_i}\right) \end{aligned} \tag{2}$$

In the resulting formula, the target of each top-level S is a conjunction of literals, and the path condition is a disjunction of literals. Notably, if $U(\varphi, \psi)$ doesn't appear in the target and the path of a top-level S formula, that subformula is a pure-past formula.

Hence, we focus our attention on the top-level S formulas containing $U(\varphi, \psi)$. In one such formula, let α' be the conjunction of all literals in the target that aren't $U(\varphi, \psi)$ or its negation. Similarly, let β' be the disjunction of all literals in the path that aren't $U(\varphi, \psi)$ or its negation. This lets us write that formula as one of the following

$$S(\alpha' \wedge \pm U(\varphi, \psi), \beta')$$
$$S(\alpha', \beta' \vee \pm U(\varphi, \psi))$$
$$S(\alpha' \wedge \pm U(\varphi, \psi), \beta' \vee \pm U(\varphi, \psi))$$

Clearly, the eliminations we explored in the previous section can separate this formula! Additionally, note that the only U formula in the RHS of the eliminations is $U(\varphi, \psi)$, satisfying the condition specified in the beginning of the lemma.

Applying these elimination rules to each top-level S containing a $U(\varphi, \psi)$ produces a separated formula equivalent to $S(\varphi, \psi)$. This completes the proof. ◀

The second step of the induction scheme is to consider cases where $U(\varphi, \psi)$ is nested under multiple levels of S.

▶ **Lemma 3.4.** *Let φ and ψ be pure-present formulas, and let γ be a formula such that the only appearance of a U in α is $U(\varphi, \psi)$. Then, γ can be written as a syntactically separated formula where the only appearance of a U is $U(\varphi, \psi)$.*

Proof. We show this lemma by inducting on the pair (n_1, n_2), where n_1 is the maximum number of nested Ses above a $U(\varphi, \psi)$ and n_2 is the number of $U(\varphi, \psi)$ nested inside n_1 Ss.
Base case. Here, $n_1 = 0$, and γ is already separated.
Induction step. Pick the most deeply nested subformula $S(\alpha, \beta)$ of γ such that all instances of $U(\varphi, \psi)$ in α and β are not nested inside a S. Applying lemma 3.3 to $S(\alpha, \beta)$ strictly reduces (n_1, n_2), allowing us to use the induction hypothesis. Remember, lemma 3.3 only generates formulas where then only appearance of U is $U(\varphi, \psi)$, which is required to use the induction hypothesis.
This completes the proof. ◀

The next step generalizes this approach to different (basic) until subformulas.

▶ **Lemma 3.5.** *Let $\varphi_1, \varphi_2, \dots \varphi_n$ and $\psi_1, \psi_2, \dots \psi_n$ be pure present formulas and γ be a formula such that all appearances of U in γ are of the form $U(\varphi_i, \psi_i)$ for some $i \in \{1, 2, \dots n\}$. Then, γ can be written as a syntactically separated formula.*

Proof. Predictably, we induct on n.
Base case. This is $n = 1$, identical to lemma 3.4.
Induction case. Introduce new propositional atoms $p_1, p_2, \dots p_{n-1}$. For each $i \in \{1, \dots n-1\}$, replace each occurrence of $U(\varphi_i, \psi_i)$ in γ with p_i to produce γ'. We can apply lemma 3.4 to γ' to produce its separated equivalent, γ''. Replace each instance of p_i in γ'' with $U(\varphi_i, \psi_i)$ to produce γ'''. Finally, apply the induction hypothesis on γ''' to separate γ.
This proves the lemma.

▶ Remark. It isn't difficult to see that we cannot use lemma 3.4 if we introduce a single atom p_n to represent $U(\varphi_n, \psi_n)$. Introducing more atoms is essential to the overall induction structure.

◀

We can now finally consider the case of nested Us.

▶ **Lemma 3.6.** *Let γ be a formula that doesn't contain Ss nested inside a U. Then, γ can be separated.*

Proof. We cleverly induct on the maximum nesting depths of Us under a S. Let n be the maximum U-nesting depth of γ.
Base case. This is $n = 1$, which is lemma 3.5.
Induction step. Suppose there are m subformulas rooted at a U that aren't under a U and are under an S. Introduce $2m$ atoms $\{p_1, \dots p_{2m}\}$ and replace the target and path conditions of these m subformulas with these atoms. This produces a new formula γ' that is amenable to lemma 3.5. Applying the lemma produces a separated formula γ'' that uses the atoms $\{p_1, \dots p_{2m}\}$. These atoms may appear under a S in the separated formula γ''. Now, replace each of these atoms by the target/path condition they substituted earlier. This produces γ''', a formula with the maximum U-nesting depth under a S strictly $< n$. Applying the induction hypothesis on γ''' proves this lemma.

▶ Remark. We don't need to consider the value m in our induction hypothesis, as required in the proof of lemma 3.4. ◀

Before we finally prove the separation theorem, notice that, since U and S are duals of each other, the eliminations in section 3.1.1 and lemmas 3.3, 3.4, 3.5 and 3.6 hold when the U and S are swapped.

▶ **Theorem 3.7** (Separation Property for Linear Time). *Any formula γ that uses S and U can be separated.*

Proof. We induct over the *junction depth* of the input formula. Define this depth as follows.

▶ **Definition 3.8** (Junction Depth). *The junction depth of a temporal formula γ is the length of the longest sequence of subformulas $\alpha_1, \alpha_2, \dots \alpha_n$ of γ such that*

1. *The root of all α_i is either a U or a S.*
2. *α_{i+1} is a subformula of α_i.*
3. *If α_i is rooted by a U (or a S), then α_{i+1} is rooted by a S (or a U, respectively).*
4. *There is no subformula β of γ such that*
   a. *β is a strict subformula of α_i.*
   b. *α_{i+1} is a strict subformula of β.*
   c. *β and α_{i+1} are rooted by the same connective.*

Note that condition 4 isn't necessary to compute the junction depth. However, I will use it in my proof.

As an illustration, observe that the junction depth of the formula $U\left(a, S\left(U(c,d), U(e,f)\right)\right)$ is 3, and there are two possible sequences:

- $U\left(a, S\left(U(c,d), U(e,f)\right)\right), S(U(c,d), U(e,f)), U(c,d)$.
- $U\left(a, S\left(U(c,d), U(e,f)\right)\right), S(U(c,d), U(e,f)), U(e,f)$.

Let the junction depth of γ be n.

Base case 1: $n = 1$. The formula is already separated.

Base case 2: $n = 2$. In this case, apply lemma 3.6 to separate γ.

Induction step: $n \geq 3$. Let there be m sequences of subformulas that witness the junction depth n. Form a set A of all subformulas at position 3 of these m sequences; the size of A can be less than m. Note that condition 4 makes these subformulas maximal; i.e., no formula in A is a subformula of another. This maximality allows us to substitute each formula in A with a newly introduced atom from the set $\{p_1, \dots p_{|A|}\}$.

Call the resulting formula γ'. It isn't difficult to argue that this formula has a junction depth of strictly $< n$, allowing us to apply the induction hypothesis. This produces a separated formula γ'' with $|A|$ new atoms. Substitute the subformulas in A at the corresponding atoms in γ'' to produce γ'''.

Now, all subformulas in A have a junction depth of $n-2$. If all of these appear in the pure-present segment of γ'', the new junction depth of γ''' grows to at-most $n-2$. Similarly, if one of these substitutions occurs inside a pure-past / pure-future segment of γ'', the junction depth grows to at-most $n-1$. This allows us to apply the induction hypothesis again, producing the fully separated formula γ'''.

This proves the separation theorem over linear time. ◀

3.2 Implying Expressive-Completeness

In this section, we provide an overview of the proof that Theorem 3.7 implies expressive completeness. We start by proving an auxiliary result.

▶ **Lemma 3.9.** *Every formula of $m+1$ variables $\varphi(t, x_1, \ldots, x_m)$ in the first-order monadic logic of order with the monadic relations $\{Q_1, Q_2, \cdots Q_k\}$ can be written in the form*

$$\bigvee_i \beta_i(t) \wedge \alpha_i(t, x_1, \ldots, x_m)$$

for some i where $t, x_1, \ldots, x_m$ are the $m+1$ free variables in φ, $\beta_i(t)$ is quantifier free, and no atomic formula of the form $Q(t)$ appears in $\alpha_i(t, x)$ for any $Q \in \{Q_1, \ldots Q_k\}$.

Proof. We show this lemma by inducting on the quantifier depth of φ.

Base case. φ is quantifier free. In this case, it's easy to see that the DNF form of φ is what we need.

Induction case. Suppose the lemma holds for all formulas of quantifier depth $< n$, and φ has quantifier depth n. Begin by writing all subformulas of the form $\forall y.\, \alpha$ as $\neg\exists y.\, \neg\alpha$. After this transformation, φ is a boolean combination of atomic formulas of the form $Q(y)$, $y < z$ ($y, z \in \{t, x_1, \ldots, x_m\}$) and quantified subformulas $\exists y.\, \psi$ for some bound variable y.

Observe that each ψ in the previous form has a quantifier depth of $n-1$. Applying the induction hypothesis on ψ gives us an equivalent formula

$$\psi \equiv \bigvee_i \beta_i(t) \wedge \alpha_i(t, x_1, \ldots, x_m, y)$$

Now, we simply push the existential quantifier deeper inside $\exists y.\, \psi$:

$$\begin{aligned}\exists y.\, \psi &\longmapsto \exists y.\, \left(\bigvee_i \beta_i(t) \wedge \alpha_i(t, x_1, \ldots, x_m, y)\right)\\ &\longmapsto \bigvee_i \exists y.\, (\beta_i(t) \wedge \alpha_i(t, x_1, \ldots, x_m, y))\\ &\longmapsto \bigvee_i \beta_i(t) \wedge \exists y.\, \alpha_i(t, x_1, \ldots, x_m, y)\end{aligned}$$

where all $\beta_i(t)$ remain quantifier free and no $Q(t)$ appears in any α_i.

After writing each $\exists y.\, \psi$ in this form, φ becomes a boolean combination of $Q(y)$, $y < z$ (again, $y, z \in \{t, x_1, \ldots, x_m\}$), and $\exists y.\, \alpha(t, x_1, \ldots x_m, y)$. We can now take the DNF form of this formula by treating each $\exists y.\, \alpha$ as though it were an atom. It isn't difficult to see that this final formula is what we need, proving the lemma. ◀

Notice that the primary arguments in the proof of Lemma 3.9 are quite general. These arguments can be reused to show similar results in the case of more complex first-order relational vocabularies. For now, consider a useful corollary.

▶ **Corollary 3.10.** *Every single-variable formula $\varphi(t)$ in the first-order monadic order of logic can be written in the form*

$$\bigvee_i \left(\beta_i(t) \wedge \bigwedge_j (\pm\exists y.\, \alpha_{i,j}(t, y))\right)$$

where $\beta_i(t)$ is quantifier-free and $Q(t)$ doesn't appear in α.

Proof. This can easily be observed by realizing that, in the case of a single-variable formula, all α_i in Lemma 3.9 must be boolean combination of formulas of the form $\exists y.\, \psi$. Considering each of these as atoms and writing the DNF form of the resulting formula gives us what we need. ◀

Finally, we consider the separation theorem.

▶ **Theorem 3.11** (Separation Theorem for Linear Time). *Every single-variable formula $\varphi(t)$ of the first-order monadic logic of order with the monadic relations $\{Q_1, \ldots, Q_m\}$ evaluated over linear time can be expressed by a formula in the temporal logic of the strict S and U over linear flows of time.*

Proof. Before we begin the proof, note that, as per the established norms, the temporal language has access to the monadic relations $\{Q_1, \ldots, Q_m\}$ through the use of propositional atoms $\{q_1, \ldots, q_m\}$.

We induct on the quantifier depth of φ.

Base case. $\varphi(t)$ is quantifier free. Construct a temporal formula by replacing all instances of $Q_i(t)$ in φ with the propositional atom q_i. This resulting formula is clearly equivalent to φ when evaluated at any time point t. Notably, it's a *pure-present* formula.

Induction case. Suppose $\varphi(t)$ has quantifier depth n. Write $\varphi(t)$ in the form presented in Corollary 3.10.

$$\varphi(t) \equiv \bigvee_i \left(\beta_i(t) \wedge \bigwedge_j \left(\pm \exists y.\, \alpha_{i,j}\left(t, y\right)\right) \right) \tag{3}$$

Observe that one can easily construct a pure-present temporal formula ρ_i for each $\beta_i(t)$. Hence, we focus our attention on the $\exists y.\, \alpha(t, y)$. We start by getting rid of all instances of the variable t in α by introducing a few new monadic relations.

Introduce three new monadic symbols $R_<$, $R_=$, and $R_>$. In each α, substitute all atomic formulas that involve t in the following way.

$$\begin{aligned} x < t &\longmapsto R_<(x) \\ x = t &\longmapsto R_=(x) \\ t < x &\longmapsto R_>(x) \end{aligned}$$

By Corollary 3.10, these are the only instances of t in α. Call the resulting formula α'. Transforming each α in φ in this way produces the formula φ', defined below

$$\varphi'(t) \triangleq \bigvee_i \left(\beta_i(t) \wedge \bigwedge_j \left(\pm \exists y.\, \alpha'_{i,j}\left(y\right)\right) \right) \tag{4}$$

Observe that each α' in φ' satisfies the following properties.

(a) Their quantifier depth is at-most $n-1$.
(b) They have a single free-variable (y).
(c) They are equivalent to α *if* $R_<$, $R_=$, and $R_>$ are modelled appropriately (which we'll discuss in a later part of the proof).

These conditions allow us to use the induction hypothesis on α' to produce a temporal formula γ. Notably, since the increased pool of monads has created the new propositional atoms $r_>$, $r_=$, and $r_<$. γ may contain these atoms.

Now, observe that the existential quantifier in $\exists y.\, \alpha'(y)$ can be expressed in the temporal language as $\diamond\gamma$, where $\diamond$ is shorthand for *at some point in time.* $\diamond$ can be expressed with S and U as follows:

$$\diamond\gamma \equiv \gamma \vee U(\gamma, \top) \vee S(\gamma, \top)$$

We proceed to construct temporal formulas $\gamma_{i,j}$ for each $\alpha'_{i,j}$ in (4). This allows us to construct the monolithic temporal formula ψ:

$$\psi \triangleq \bigvee_i \left(\rho_i \wedge \bigwedge_j (\pm \diamond \gamma_{i,j}) \right)$$

Again, if $r_<$, $r_=$, and $r_>$ are appropriately modelled, ψ is equivalent to $\varphi(t)$.

Using Theorem 3.7, we now *separate* ψ into a boolean combination of pure-past, present, and future formulas. We write the separated formula as follows:

$$\psi \equiv \mathbf{B}(\psi_{<,1}, \ldots, \psi_{<.m_<}, \psi_{=,1}, \ldots, \psi_{<,m_=}, \psi_{>,1}, \ldots \psi_{>,m_>})$$

where $\mathbf{B}$ abstracts the boolean combinations and the $\psi_{<,i}$, $\psi_{=,i}$, and $\psi_{>,i}$ are pure-past, present, and future formulas.

We earlier stated that if $R_<$, $R_=$, and $R_>$ are appropriately modelled, ψ is equivalent to $\varphi(t)$. The correct values of $R_<$, $R_=$, and $R_>$ are, quite naturally,

$$R_< = \{s \mid s < t\}$$
$$R_= = \{t\}$$
$$R_> = \{s \mid t < s\}$$

Let the partial assignment of atoms over the flow of time h represent this model.

Now, consider a partial assignment $h_<$ that agrees with h on all atoms but $r_<$, $r_=$, and $r_>$. $h_<$ models $r_<$ to $\top$ everywhere, and $r_>$ and $r_=$ to $\bot$ everywhere. This assignment, by its definition, agrees with h on the past of t, and consequently, for each pure-past $\psi_{<,i}$,

$$h \vDash \psi_{<,i} \longleftrightarrow h_< \vDash \psi_{<,i}$$

Construct the formula $\psi'_{<,i}$ by substituting all instances of $r_<$ in $\psi_{<,i}$ by $\top$ and all instances of $r_=$ and $r_>$ by $\bot$, i.e.,

$$\psi'_{<,i} \triangleq \psi_{<,i} \begin{bmatrix} r_< \mapsto \top \\ r_= \mapsto \bot \\ r_> \mapsto \bot \end{bmatrix}$$

It's easy to see that

$$h_< \vDash \psi_{<,i} \longleftrightarrow h_< \vDash \psi'_{<,i}$$

Hence,

$$h \vDash \psi_{<,i} \longleftrightarrow h_< \vDash \psi_{<,i} \longleftrightarrow h_< \vDash \psi'_{<,i}$$

Observe that $\psi'_{<,i}$ no longer uses the additional atoms! And since h and $h_<$ agree on all other atoms,

$$h \vDash \psi_{<,i} \longleftrightarrow h \vDash \psi'_{<,i}$$

We can similarly substitute the atoms in the $\psi_{>,i}$ and $\psi_{=,i}$ to get rid of $r_<$, $r_=$, and $r_>$ in ψ. Call this new formula ψ'.

$$\psi' \triangleq \mathbf{B}(\psi'_{<,1}, \ldots, \psi'_{<.m_<}, \psi'_{=,1}, \ldots, \psi'_{<,m_=}, \psi'_{>,1}, \ldots \psi'_{>,m_>})$$

We claim that ψ' is equivalent to $\varphi(t)$. To see why, take any linear temporal structure $\mathcal{M} = (T, <, h)$ and a point $t \in T$ in the flow. Let h' be the extension of h with the appropriate valuations of $R_<$, $R_=$ and $R_>$ and $\mathcal{M}'$ be $(T, <, h')$. It's easy to see that the following double-implications immediately hold:

$$\mathcal{M}, t \vDash \varphi(t) \iff \mathcal{M}', t \vDash \varphi'(t) \iff \mathcal{M}', t \vDash \psi \iff \mathcal{M}, t \vDash \psi'$$

This proves the separation theorem.

◀

4 Ordered Trees

Flows of time can be more complicated than the linear structures we've seen so far. The notion of branching time, where the flow resembles a tree, is a well known example. While temporal languages over unordered trees have been studied quite extensively (see [6]), in this work we look at ordered trees.

In addition to the descendent order (which corresponds to the natural forward flow of time), ordered trees use a *sibling* order. Correspondingly, the first-order vocabulary includes two binary relations: $<$ and $\prec$, where $x < y$ indicates y is a descendant of x and $x \prec y$ indicates y comes after x in the sibling order. All immediate children of a node are totally ordered by $\prec$.

In [4], Marx introduced the temporal language $\mathcal{X}_{until}$ over ordered trees. This language has the same expressive power as *Conditional XPath*, which Marx proved to be expressively complete in [5]. It defines four connectives that are similar to the strict U and S Gabbay defines for linear time. These are $\Leftarrow$, $\Rightarrow$, $\Uparrow$, and $\Downarrow$, defined by the following monadic first-order formulas.

$$\begin{aligned}
\varphi_{\Downarrow}(t, X_1, X_2) &\triangleq \exists x.\ [(t < x) \land X_1(x) \land \forall y\,((t < y < x) \to X_2(y))] \\
\varphi_{\Uparrow}(t, X_1, X_2) &\triangleq \exists x.\ [(x < t) \land X_1(x) \land \forall y\,((x < y < t) \to X_2(y))] \\
\varphi_{\Rightarrow}(t, X_1, X_2) &\triangleq \exists x.\ [(t \prec x) \land X_1(x) \land \forall y\,((t \prec y \prec x) \to X_2(y))] \\
\varphi_{\Leftarrow}(t, X_1, X_2) &\triangleq \exists x.\ [(x \prec t) \land X_1(x) \land \forall y\,((x \prec y \prec t) \to X_2(y))]
\end{aligned}$$

Marx suggested a separation property for this temporal language over ordered trees. The regions he proposed, with respect to an arbitrary point t in the flow, were

- The *present* point, which we call t.
- The *future*, defined as $\{x \mid t < x\}$.
- The *left* of t, defined as $\{x \mid x \prec t \lor \exists y.\, y \prec t \land y < x\}$.
- The *right* of t, defined analogously as $\{x \mid t \prec x \lor \exists y.\, t \prec y \land y < x\}$.
- The *past*, which consists of all points not claimed by other regions.

Figure 2 shows how these regions partition the tree.

Unfortunately, Marx's proof of separation in [4] is incorrect. He fails to take into consideration that the $\Downarrow$ modality is non-deterministic. This indicates that one cannot extend equation (1) to $\Downarrow$, as

$$\Downarrow(a, b \land c) \not\equiv \Downarrow(a, b) \land \Downarrow(a, c)$$

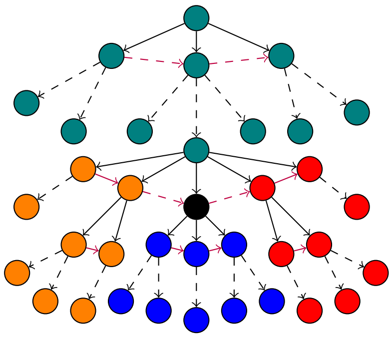

Figure 2 Marx's regions. The black node is the present, the orange nodes belong to the *left* region, the red nodes to the *right*, the blue nodes are the *future*, and the green nodes are the *past*. The descendant relation is given by the black lines and the sibling order is denoted by the red lines. Dashed lines indicate potential intermediate nodes.

In the next few sections, we show that a certain class of formulas in $\mathcal{X}_{until}$ can be separated. We also show a few results that show that certain formulas can never be separated into these regions.

4.1 Partial Separation of $\mathcal{X}_{until}$

As with linear flows, we can define a notion of syntactically pure formulas in $\mathcal{X}_{until}$. Again, to simplify presentation, we refer to formulas rooted by a π for $\pi \in \{\Leftarrow, \Rightarrow, \Uparrow, \Downarrow\}$ as a π-formula.

▶ **Definition 4.1.** *A temporal formula φ in $\mathcal{X}_{until}$ is*
1. *syntactically pure-present if it doesn't use any connectives.*
2. *syntactically pure-future if it's a boolean combination of $\Downarrow$-formulas that don't use the $\Uparrow$ connective.*
3. *syntactically pure-left if it's a boolean combination of $\Leftarrow$-formulas that contain pure-present, pure-future and/or smaller $\Leftarrow$-formulas of this form.*
4. *syntactically pure-right if it's a boolean combination of $\Rightarrow$-formulas that contain pure-present, pure-future, and/or smaller $\Rightarrow$-formulas of this form.*
5. *syntactically pure-past if it's a boolean combination of $\Uparrow$-formulas that contain pure-present, pure-left, pure-right, and/or smaller $\Uparrow$-formulas of this form.*

It's simple enough to observe that all syntactically pure formulas are semantically pure. To simplify our arguments, we also introduce the notion of the *pure π formula* for $\pi \in \{\Leftarrow, \Rightarrow, \Downarrow, \Uparrow\}$. Simply put, a formula is pure π *iff* the only connective it uses is π.

In this subsection, we aim to show the following theorem.

▶ **Theorem 4.2** (Partial Separation of $\mathcal{X}_{until}$)**.** *Let φ be a formula in $\mathcal{X}_{until}$ such that all $\Downarrow$-subformulas of φ are syntactically pure-future. Then, φ can be separated.*

This theorem can be seen as a consequence of two facts. First, (1) is valid for the modalities $\pi \in \{\Uparrow, \Leftarrow, \Rightarrow\}$. Second, the eliminations presented in [4] prove that formulas of the form $\Uparrow(a \wedge \pm\Downarrow(p, q), b \vee \pm\Downarrow(p, q))$ can be separated.

4.1.1 Separating $\Leftarrow$, $\Rightarrow$ and $\Uparrow$

Before we prove Theorem 4.2, we prove a few lemmas.

▶ **Lemma 4.3.** *Let φ be a formula in $\mathcal{X}_{until}$ that doesn't use the $\Uparrow$ and $\Downarrow$ connectives. Then, φ can be separated.*

Proof. This is a simple consequence of Theorem 3.7. Observe that φ can only use the $\Leftarrow$ and $\Rightarrow$ connectives, and that the directions of these connectives prevent φ from probing nodes that aren't siblings of the present point t. Moreover, their operation mirrors that of S and U over linear time. Since the sibling order $\prec$ over all siblings of a node produces a total-order, we're justified in applying Theorem 3.7. ◀

Next, we lift a few eliminations from [4].

▶ **Lemma 4.4.** *The following eliminations are valid.*

- $\Leftarrow(\alpha \wedge \Uparrow(\varphi,\psi),\beta) \equiv \Leftarrow(\alpha,\beta) \wedge \Uparrow(\varphi,\psi)$
- $\Leftarrow(\alpha \wedge \neg\Uparrow(\varphi,\psi),\beta) \equiv \Leftarrow(\alpha,\beta) \wedge \neg\Uparrow(\varphi,\psi)$
- $\Leftarrow(\alpha,\beta \vee \Uparrow(\varphi,\psi)) \equiv \Leftarrow(\alpha,\beta) \vee (\Leftarrow(\alpha,\top) \wedge \Uparrow(\varphi,\psi))$
- $\Leftarrow(\alpha,\beta \vee \neg\Uparrow(\varphi,\psi)) \equiv \Leftarrow(\alpha,\beta) \vee (\Leftarrow(\alpha,\top) \wedge \neg\Uparrow(\varphi,\psi))$

Proof. These equivalences follow from the fact that the truth of $\pm\Uparrow(\varphi,\psi)$ at a sibling of t implies $\pm\Uparrow(\varphi,\psi)$ at t. The eliminations merely explicate this fact. ◀

▶ Note. These eliminations *don't* proliferate the parameters of the $\Uparrow$ connective. The eliminations we observed in Section 3.1.1 don't have this property.

We now implement these eliminations in a more general setting.

▶ **Lemma 4.5.** *Let α and β be formulas in $\mathcal{X}_{until}$ that (1) don't use the $\Leftarrow$, $\Rightarrow$ and $\Downarrow$ connectives, and (2) only use the $\Uparrow$ connective to insert the subformula $\Uparrow(\varphi,\psi)$ for some $\mathcal{X}_{until}$ formulas φ and ψ. Then, $\Leftarrow(\alpha,\beta)$ can be written as a boolean combination of pure-$\Leftarrow$ formulas and $\Uparrow(\varphi,\psi)$.*

▶ Remark. This lemma only yields a separated formula if $\Uparrow(\varphi,\psi)$ is a pure-past formula.

Proof. Since $\Leftarrow$ and $\Rightarrow$ satisfy a version of (1), we can follow the procedure outlined in Lemma 3.3 and write $\Leftarrow(\alpha,\beta)$ in the manner of (2) to get

$$\Leftarrow(\alpha,\beta) \equiv \bigvee_i \bigwedge_j \Leftarrow(\alpha_{i,1} \wedge \cdots \wedge \alpha_{i,m_i}, \beta_{j,1} \vee \cdots \vee \beta_{j,n_i})$$

As in Lemma 3.3, the $\Leftarrow$-formula for each $\{i,j\}$ can be considered to be in the form of

$$\Leftarrow(\alpha' \wedge \pm\Uparrow(\varphi,\psi),\beta')$$
$$\Leftarrow(\alpha',\beta' \vee \pm\Uparrow(\varphi,\psi))$$
$$\Leftarrow(\alpha' \wedge \pm\Uparrow(\varphi,\psi),\beta' \vee \pm\Uparrow(\varphi,\psi))$$

We can apply the eliminations detailed in Lemma 4.4 to complete the proof. ◀

We now generalize this lemma a little further.

▶ **Lemma 4.6.** *Take an expanded set of atoms $\mathcal{P}' \triangleq \mathcal{P} \cup \{\Uparrow(\varphi,\psi)\}$. Suppose γ is a formula in $\mathcal{X}_{until}$ that only uses the $\Leftarrow$ connective and takes atoms from $\mathcal{P}'$. Then, γ is equivalent to a boolean combination of pure-$\Leftarrow$ formulas, atoms in $\mathcal{P}$, and the formula $\Uparrow(\varphi,\psi)$.*

Proof. Despite the stricter wording, this lemma is the counterpart of Lemma 3.4 with the connectives S and U substituted by $\Leftarrow$ and $\Uparrow$. As with that lemma, we induct on (n_1,n_2), where n_1 is the highest $\Leftarrow$-depth at which a $\Uparrow(\varphi,\psi)$ appears in γ and n_2 is the number of instances of $\Uparrow(\varphi,\psi)$ at depth n_1.

Base case. $(n_1, n_2) = (1, 1)$. This is equivalent to Lemma 4.5.

Induction step. Take a subformula $\Leftarrow(\alpha', \beta')$ at $\Leftarrow$-depth $(n-1)$ such that (1) $\Leftarrow$ doesn't appear in α' and β', and (2) $\Uparrow(\varphi, \psi)$ appears in at-least one of α' and β'. It's easy to see that applying Lemma 4.5 to $\Leftarrow(\alpha', \beta')$ strictly reduces (n_1, n_2), allowing us to apply the induction hypothesis.

▶ Remark. Since φ and ψ aren't proliferated in Lemma 4.4, they don't enter the pure-$\Leftarrow$ formulas in this lemma. ◀

We now consider the case of multiple $\Uparrow$-formulas inside a $\Leftarrow$-formula.

▶ **Lemma 4.7.** *Take an expanded set of atoms $\mathcal{P}' \triangleq \mathcal{P} \cup \{\Uparrow(\varphi_1, \psi_1), \ldots, \Uparrow(\varphi_n, \psi_n)\}$. Suppose γ is a formula in $\mathcal{X}_{until}$ that only uses the $\Leftarrow$ connective and takes atoms from $\mathcal{P}'$. Then, γ is equivalent to a boolean combination of pure-$\Leftarrow$ formulas, atoms in $\mathcal{P}$, and the $\Uparrow(\varphi_i, \psi_i)$ formulas.*

Proof. This is the counterpart of Lemma 3.5. Predictably, we prove this by inducting on n. The details are left to the reader.

▶ Remark. Again, the structure of the $\Uparrow$-formulas is maintained during the transformation. ◀

Before we proceed with the next result, note that Lemmas 4.5, 4.6, and 4.7 are valid when the $\Leftarrow$ is replaced by a $\Rightarrow$.

▶ **Lemma 4.8.** *Take an expanded set of atoms $\mathcal{P}' \triangleq \mathcal{P} \cup \{\Uparrow(\varphi_1, \psi_1), \ldots, \Uparrow(\varphi_n, \psi_n)\}$. Suppose γ is a formula in $\mathcal{X}_{until}$ that only uses the $\Leftarrow$ and $\Rightarrow$ connectives and takes atoms from $\mathcal{P}'$. Then, γ is equivalent to a boolean combination of pure-$\Leftarrow$ formulas, pure-$\Rightarrow$ formulas, atoms in $\mathcal{P}$, and the $\Uparrow(\varphi_i, \psi_i)$.*

Proof. Introduce new atoms $r_1, \ldots, r_n$ to $\mathcal{P}$ to produce $\mathcal{P}''$. In γ, replace each instance of $\Uparrow(\varphi_i, \psi_i)$ with r_i to produce the formula γ'. Notice that γ' has no $\Uparrow$-subformulas. Separate γ' according to Lemma 4.3 to get

$$\gamma' \equiv \mathbf{B}(\gamma_{\Leftarrow,1}, \ldots, \gamma_{\Leftarrow,n_\Leftarrow}, \gamma_{\Rightarrow,1}, \ldots, \gamma_{\Rightarrow,n_\Rightarrow}, \gamma_{=,1}, \ldots \gamma_{=,n_=})$$

where all $\gamma_{\Leftarrow,i}$ formulas only use the $\Leftarrow$ connective, all $\gamma_{\Rightarrow,j}$ formulas only use the $\Rightarrow$ connective, and all $\gamma_{=,k}$ use no connectives.. At this stage, replace all instances of r_i by $\Uparrow(\varphi_i, \psi_i)$ in the $\gamma_{\Leftarrow,i}$, $\gamma_{\Rightarrow,j}$, and $\gamma_{=,k}$ to produce $\gamma'_{\Leftarrow,i}$, $\gamma'_{\Rightarrow,j}$ and $\gamma'_{=,k}$. The $\gamma'_{=,k}$ already satisfy our condition; hence, we only apply Lemma 4.7 to the others to prove this lemma. ◀

We can now state an interesting corollary.

▶ **Corollary 4.9.** *Let γ be a formula in $\mathcal{X}_{until}$ that doesn't use the $\Downarrow$ connective. Then, γ can be separated.*

Proof. This is a simple matter of noticing that all $\Uparrow$-formulas that don't use the $\Downarrow$ connective are syntactically pure-past. ◀

4.1.2 Separating $\Downarrow$, $\Rightarrow$, and $\Leftarrow$

At this stage, we begin considering pure-future $\Downarrow$-formulas. It's simple to notice that any formula that only uses the $\Leftarrow$ (or $\Rightarrow$) and $\Downarrow$ connectives is immediately syntactically separated; it is a boolean combination of syntactically pure-left (or pure-right) and pure-future formulas. It's easy to extend this observation to show the following lemma.

▶ **Lemma 4.10.** *Let γ be a formula in $\mathcal{X}_{until}$ that doesn't use the $\Uparrow$ connective. Then, γ can be separated.*

Proof. Let $\Downarrow(\varphi_1, \psi_1), \ldots, \Downarrow(\varphi_n, \psi_n)$ be γ's $\Downarrow$-subformulas that don't appear under ths scope of a $\Downarrow$ (i.e., they're the *top-level* $\Downarrow$-subformulas). Introduce new atoms $\{r_1, \ldots, r_n\}$ and for each $i \in \{1, \ldots, n\}$, replace the instance of the subformula $\Downarrow(\varphi_i, \psi_i)$ in γ with r_i to produce the formula γ'.

Observe that γ' only uses the $\Leftarrow$ and $\Rightarrow$ connectives. This allows us to use Lemma 4.3 separate γ'. In the separated formula, substitute all instances of r_i with $\Downarrow_i (\varphi_i, \psi_i)$. It's easy to see that, after substitution, we get a syntactically separated formula. ◀

We can extend this reasoning further by reusing the method used to prove Lemma 4.8.

▶ **Lemma 4.11.** *Let γ be a $\mathcal{X}_{until}$ formula such that all $\Downarrow$ subformulas are syntactically pure-future and all $\Uparrow$-subformulas are syntactically pure-past. Then, γ can be separated.*

Proof. Let $\Downarrow(\varphi_1, \psi_1), \ldots, \Downarrow(\varphi_n, \psi_n)$ be all subformulas of γ that (1) don't appear under a $\Downarrow$ and (2) don't appear under a $\Uparrow$. Note that, as all $\Uparrow$-subformulas are syntactically pure-past, any $\Downarrow$-subformula that appears under a $\Uparrow$ must have an $\Leftarrow$ or $\Rightarrow$ between it and the $\Uparrow$. Also, note that each $\Downarrow(\varphi_i, \psi_i)$ are syntactically pure-future.

Introduce n new atoms $r_1, \ldots r_n$ and substitute each $\Downarrow(\varphi_i, \psi_i)$ in γ by r_i. Note that no r_i is embedded under a $\Uparrow$. Call this new formula γ'. It's easy to observe that one can apply Lemma 4.8 to separate γ'. Let the separated formula be γ''. Since no r_i is under a $\Uparrow$ in γ', no r_i is under a $\Uparrow$ in γ''.

Hence, all r_i must occur as a pure-present atom or inside a pure-$\Leftarrow$ formula or a pure-$\Rightarrow$ formula in γ''. Substituting $\Downarrow(\varphi_i, \psi_i)$ for each r_i keeps the formula separated, proving the lemma. ◀

4.1.3 Normal Form

We've reached another checkpoint in our proof of Theorem 4.2. At this stage, it's clear that only the case where the pure-future $\Downarrow$-formula is inside an $\Uparrow$-formula remains. To facilitate progress, we consider a *standard form* for pure-past $\Uparrow$-formulas.

▶ **Definition 4.12.** *Let $\gamma = \Uparrow(\varphi, \psi)$ be a formula in $\mathcal{X}_{until}$ such that φ and ψ are of the form*

$$\varphi \triangleq \bigwedge_i \pm\varphi_i \quad \psi \triangleq \bigvee_j \pm\psi_j$$

where φ_i and ψ_j are either atoms, pure-left $\Leftarrow$-formulas, pure-right $\Rightarrow$-formulas, or smaller $\Uparrow$-formulas in standard form. Then, γ is a $\Uparrow$-formula in standard form.

It's easy to see that all standard-form $\Uparrow$-formulas are syntactically pure-past. However, the next result makes these more useful.

▶ **Lemma 4.13.** *Let γ be a syntactically pure-past formula in $\mathcal{X}_{until}$. Then, γ can be written as a boolean combination of $\Uparrow$formulas in standard form.*

Proof. Since γ is a boolean combination of syntactically pure-past $\Uparrow$-formulas, we limit our discussion to the case where γ is a $\Uparrow$ formula. Hence,

$$\gamma \triangleq \Uparrow(\varphi, \psi)$$

We begin by inducting on the $\Uparrow$-depth of γ.

Base case. The $\Uparrow$-depth of γ is 1. This means φ and ψ only use the $\Leftarrow$, $\Rightarrow$, and $\Downarrow$ connectives. We hence use Lemma 4.10 to separate them to φ' and ψ'. Since γ is a syntactically pure-past formula, φ' and ψ' will be boolean combinations of atoms, pure-left $\Leftarrow$-formulas, pure-right $\Rightarrow$-formulas. Writing φ' and ψ' in DNF and CNF form gives us

$$\varphi' \equiv \bigvee_i \bigwedge_j \pm\varphi'_{i,j}$$
$$\psi' \equiv \bigwedge_i \bigvee_j \pm\psi'_{i,j}$$

where each $\varphi'_{i,j}$ and $\psi'_{i,j}$ are pure-left $\Leftarrow$-formulas, pure-right $\Rightarrow$-formulas, or atoms. Hence, by token of the procedure used in (2), we have

$$\Uparrow(\varphi, \psi) \equiv \bigvee_i \bigwedge_j \Uparrow(\pm\varphi'_{i,1} \wedge \cdots \wedge \pm\varphi'_{i,n_i}, \pm\psi'_{j,1} \vee \cdots \vee \pm\psi'_{j,m_j})$$

This final form is clearly in standard form.

Induction step. Let $\Uparrow(\alpha, \beta), \ldots, \Uparrow(\alpha, \beta)$ be all of φ's and ψ's $\Uparrow$-subformulas that aren't under the scope of a $\Uparrow$. Since these subformulas have a strictly lower $\Uparrow$-depth than γ, we can apply the induction hypothesis to produce equivalent boolean combinations of $\Uparrow$-formulas in standard form for each $\Uparrow(\alpha_i, \beta_i)$. Simply substitute these formulas for $\Uparrow(\alpha_i, \beta_i)$ in φ and ψ to produce formulas φ' and ψ' in which each $\Uparrow$-subformula is in standard form. This gives us the new formula γ' equivalent to γ:

$$\gamma' \triangleq \Uparrow(\varphi', \psi')$$

Notice that the separation procedure described in the proof of Lemma 4.11 maintains the structure of the pure-past $\Uparrow$-subformulas in the separated form. Directly applying that lemma on φ' and ψ' produces the separated formulas φ'' and ψ''. As in the base case, we can write φ'' in DNF and ψ'' in CNF to complete the proof.

◀

4.1.4 Pulling out $\Downarrow$ from $\Uparrow$

We now restate some of the eliminations justified in the appendix of [4]. These eliminations make vital use of the formula θ, defined as

$$\theta \triangleq \Leftarrow(\varphi \vee (\psi \wedge \Downarrow(\varphi, \psi)), \top) \vee \Rightarrow(\varphi \vee (\psi \wedge \Downarrow(\varphi, \psi)), \top) \tag{5}$$

θ merely states that "my parent satisfies $\Downarrow(\varphi, \psi)$ because of a sibling of mine." Hence, $\varphi \vee \theta$ implies $\Downarrow(\varphi, \psi)$ at the parent. Similarly, $\neg\theta$ states that "no sibling of mine is responsible for my parent satisfying $\Downarrow(\varphi, \psi)$." Consequently, $\neg\varphi \wedge \neg\psi \wedge \neg\theta$ implies $\neg\Downarrow(\varphi, \psi)$ at the parent. Notably, θ is a disjunction of a pure-left and a pure-right formula, and hence can appear inside the scope of a $\Uparrow$ in a pure-past formula.

Now, we move onto the eliminations.

$\Uparrow(\alpha \wedge \Downarrow(\varphi, \psi), \beta)$

This formula requires $\Downarrow(\varphi, \psi)$ to be true at the ancestor node that satisfies α. The path taken by this $\Downarrow$-formula can deviate from the ancestral path to α at any point. With θ, we can measure when it deviates. Hence, this formula is equivalent to

$$\begin{aligned} & \Uparrow(\beta \wedge (\theta \vee \varphi) \wedge \Uparrow(\alpha, \beta \wedge \psi), \beta) \\ \vee \quad & \Uparrow(\alpha, \beta \wedge \psi) \wedge (\theta \vee \varphi \vee (\psi \wedge \Downarrow(\varphi, \psi))) \end{aligned}$$

$\Uparrow(\alpha \wedge \neg\Downarrow(\varphi, \psi), \beta)$

We again have an ancestral path to α, and at that point $\Downarrow(\varphi, \psi)$ cannot be true. This indicates that $\Downarrow(\varphi, \psi)$ cannot be true along this ancestral path as well, indicating that, we must either hit a point on the path where $\neg\varphi \wedge \neg\psi$ is true, or we need to force $\neg\Downarrow(\varphi, \psi)$ at the present. This can be observed in the following separated formula.

$$\begin{aligned} & \Uparrow(\neg\theta \wedge \neg\varphi \wedge \neg\psi \wedge \beta \wedge \Uparrow(\alpha, \beta \wedge \neg\varphi \wedge \neg\theta), \beta) \\ \vee \quad & \Uparrow(\alpha, \beta \wedge \neg\varphi \wedge \neg\theta) \wedge \neg\theta \wedge ((\neg\varphi \wedge \neg\psi) \vee (\neg\varphi \wedge \neg\Downarrow(\varphi, \psi))) \end{aligned}$$

$\Uparrow(\alpha, \beta \vee \Downarrow(\varphi, \psi))$

As in Section 3.1.1, we use the idea of the *unfulfilled* point. This time, such points are detected by noticing that $\neg\varphi \wedge \neg\theta$ are true along the ancestral path to a $\neg\beta$. We attempt to fulfil the point by ensuring $(\varphi \vee \theta) \vee \psi$. This gives us the entire formula

$$\begin{aligned} & \Uparrow(\alpha, \neg\alpha \wedge (\Uparrow(\neg\beta \wedge \neg\alpha, \neg\alpha \wedge \neg\varphi \wedge \neg\theta) \to (\psi \vee \varphi \vee \theta))) \\ \wedge \quad & \Uparrow(\neg\beta \wedge \neg\alpha, \neg\alpha \wedge \neg\varphi \wedge \neg\theta) \to (\varphi \vee \theta \vee (\psi \wedge \Downarrow(\varphi, \psi))) \end{aligned}$$

$\Uparrow(\alpha, \beta \vee \neg\Downarrow(\varphi, \psi))$

Again, as in Section 3.1.1, we use the idea of a dangerous point. We look for ancestral paths to a $\neg\beta$ that satisfy ψ at each point. If we find such a path, we enforce $\neg\varphi \wedge \neg\theta$. This gives the formula

$$\begin{aligned} & \Uparrow(\alpha, \neg\alpha \wedge (\Uparrow(\neg\alpha \wedge \neg\beta, \psi \wedge \neg\alpha) \to (\neg\varphi \wedge \neg\theta))) \\ \wedge \quad & \Uparrow(\neg\alpha \wedge \neg\beta, \psi \wedge \neg\alpha) \to (\neg\theta \wedge \neg\varphi \wedge (\neg\psi \vee \neg\Downarrow(\varphi, \psi))) \end{aligned}$$

In the same vein, we can separate any combination of $\Uparrow(\alpha \pm \Downarrow(\varphi, \psi), \beta \vee \pm\Downarrow(\varphi, \psi))$. We refer to [4] for the full arguments. Notably, if α, β, φ, and ψ were replaced by atoms, the formulas on the right would be syntactically separated. And if θ was also replaced by an atom, the formulas on the right only use the $\Uparrow$ and $\Downarrow$ connectives. Observe that the only instance of $\Downarrow$ in all cases would be $\Downarrow(\varphi, \psi)$.

Unfortunately, unlike the eliminations in Lemma 4.4, the parameters of the $\Downarrow$ (the φ and ψ) appear outside of the $\Downarrow$ in the separated formula. This complicates our proof.

4.1.5 Final steps

We begin with the following lemma.

▶ **Lemma 4.14.** *Let φ and ψ be boolean combinations of atoms, pure-$\Leftarrow$, and pure-$\Rightarrow$ formulas. Take the expanded set of atoms $\mathcal{P}' = \mathcal{P} \cup \{\Downarrow(\varphi, \psi)\}$. Let α and β be boolean combinations of atoms from $\mathcal{P}'$, pure-left $\Leftarrow$-formulas, pure-right $\Rightarrow$-formulas, and pure-past $\Uparrow$-formulas. Then, $\gamma \triangleq \Uparrow(\alpha, \beta)$ can be separated.*

▶ Note. Replacing $\Downarrow(\varphi,\psi)$ by a new atom r in γ produces a pure-past $\Uparrow$-formula.

Proof. We start by writing α and β in their DNF and CNF forms respectively. This allows us to write γ as

$$\gamma \equiv \bigvee_i \bigwedge_j \Uparrow(\pm\alpha_{i,1} \wedge \cdots \wedge \pm\alpha_{i,n_i}, \pm\beta_{j,1} \vee \cdots \vee \pm\beta_{j,n_j})$$

where the α_{i,k_i} and β_{j,k_j} are atoms, instances of $\Downarrow(\varphi,\psi)$, and semantically pure π-formulas for $\pi \in \{\Leftarrow, \Rightarrow, \Uparrow\}$. For each $\{i,j\}$, we can write the corresponding $\Uparrow$-formula in γ as

$$\gamma_{i,j} \triangleq \Uparrow(\alpha' \wedge \pm\Downarrow(\varphi,\psi), \beta' \vee \pm\Downarrow(\varphi,\psi))$$

where α' and β' are conjunctions and disjunctions of semantically pure past, left, present, and right formulas. As we've done many times before, we employ the eliminations detailed earlier for each $\{i,j\}$ to produce the formula $\gamma_{i,j}$.

Let $A = \{\alpha', \beta', \varphi, \psi\}$ and $B = A \cup \{\theta\}$. θ here is defined by Equation (5). From our discussion on pulling out the $\Downarrow$ from $\Uparrow$, we know that the eliminations produce syntactically separated formulas in which the pure-past and pure-present subformulas connect the formulas in B using boolean combinations and the $\Uparrow$ connective.

By the nature of the formulas in A, any formula that connects them using boolean operations and the $\Uparrow$ connective is syntactically separated. Unfortunately, the formula θ doesn't share these properties, as it isn't necessarily a separated formula.

As θ only uses the $\Leftarrow$, $\Rightarrow$ and $\Downarrow$ connectives, we can produce a separated equivalent θ' using Lemma 4.10. We must now argue that θ' contains no pure-future segment. We do so using the following claim.

▷ Claim. Take two temporal structures $\mathcal{M} = (T, <, \prec, h)$ and $\mathcal{M}' = (T, <, \prec, h')$ and points $t \in T$ such that h and h' agree on t, the left of t, and the right of t, but disagree on the future of t. We claim that

$$\mathcal{M}, t \vDash \theta \longleftrightarrow \mathcal{M}', t' \vDash \theta$$

Proof. We prove this by contradiction. Suppose, without loss of generality, that $\mathcal{M}, t \vDash \theta$ and $\mathcal{M}', t' \nvDash \theta$. Further suppose that

$$\mathcal{M}, t \vDash \Leftarrow(\varphi \vee (\psi \wedge \Downarrow(\varphi,\psi)), \top)$$
$$\mathcal{M}', t \nvDash \Leftarrow(\varphi \vee (\psi \wedge \Downarrow(\varphi,\psi)), \top)$$

We choose this possibility because the argument for the alternative is similar.

Suppose $\mathcal{M}, t \vDash \Leftarrow(\varphi, \top)$. Since φ only uses the $\Leftarrow$ and $\Rightarrow$ connectives, the truth of φ at t only depends on the assignments of atoms at the siblings of t. Similarly, the truth of $\Leftarrow(\varphi, \top)$ at t only depends on the its siblings. Since $\mathcal{M}$ and $\mathcal{M}'$ agree on the siblings of t, we must have $\mathcal{M}', t \vDash \Leftarrow(\varphi, \top)$.

Hence, we must have $\mathcal{M}, t \vDash \Leftarrow(\psi \wedge \Downarrow(\varphi,\psi), \top)$. Take the left sibling $s \in T$ such that $\mathcal{M}, s \vDash \psi \wedge \Downarrow(\varphi,\psi)$. It's easy to see that $\mathcal{M}$ and $\mathcal{M}'$ agree on the siblings of s and the future of s. Hence, since ψ is only concerned with siblings and $\Downarrow(\varphi,\psi)$ is pure-future, we must have that $\mathcal{M}', s \vDash \psi \wedge \Downarrow(\varphi,\psi)$. Hence, $\mathcal{M}', t \nvDash \Leftarrow(\varphi \vee (\psi \wedge \Downarrow(\varphi,\psi)), \top)$, forming a contradiction. ◁

We now confidently replace all top-level *(i.e., not under a connective)* pure-future $\Downarrow$-formulas (if any exist) in θ' with $\bot$. This produces θ'', a boolean combination of pure left, present, and right formulas. We further replace θ with θ'' in each $\gamma_{i,j}$ to produce $\gamma'_{i,j}$.

Let $B' = A \cup \{\theta''\}$. Observe that the past and present segments of $\gamma'_{i,j}$ are now formulas that connect the formulas in B' with boolean combinations and the $\Uparrow$ formula. Since all formulas in B' are boolean combinations syntactically pure-past, pure-present, pure-left and pure-right formulas, $\gamma'_{i,j}$ is separated. Hence, the formula γ'

$$\gamma' \triangleq \bigwedge_i \bigvee_j \gamma'_{i,j}$$

and is equivalent to γ, proving the lemma. ◀

References

1 Michael Benedikt and Clemens Ley. Limiting until in ordered tree query languages. *ACM Trans. Comput. Logic*, 17(2), mar 2016. doi:10.1145/2856104.

2 Volker Diekert and Paul Gastin. Pure future local temporal logics are expressively complete for mazurkiewicz traces. In Martín Farach-Colton, editor, *LATIN 2004: Theoretical Informatics*, pages 232–241, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

3 Dov M. Gabbay, Ian Hodkinson, and Mark Reynolds. *Temporal Logic (Vol. 1): Mathematical Foundations and Computational Aspects*. Oxford University Press, Inc., USA, 1994.

4 Maarten Marx. Conditional xpath, the first order complete xpath dialect. In *Proceedings of the Twenty-Third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '04, page 13–22, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/1055558.1055562.

5 Maarten Marx. Conditional xpath. *ACM Transactions on Database Systems (TODS)*, 30(4):929–959, 2005.

6 Alexander Rabinovich and Shahar Maoz. Why so many temporal logics climb up the trees? In Mogens Nielsen and Branislav Rovan, editors, *Mathematical Foundations of Computer Science 2000*, pages 629–639, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.