Fundamenta Informaticae 80 (2007) 169–197
IOS Press

Uniform satisfiability in PSPACE for local temporal logics over Mazurkiewicz traces*

Paul Gastin
LSV, ENS Cachan & CNRS, France

Dietrich Kuske
Institut für Informatik, Universität Leipzig, Germany

Abstract. We study the complexity of temporal logics over concurrent systems that can be described by Mazurkiewicz traces. We develop a general method to prove that the uniform satisfiability problem of local temporal logics is in PSPACE. We also demonstrate that this method applies to all known local temporal logics.

1. Introduction

Antoni Mazurkiewicz introduced the notion of *trace* to describe the behaviors of concurrent systems [11, 12]. This had a major influence in the studies of distributed systems. Since the pioneering work of Mazurkiewicz, trace theory has been developed by numerous researchers and is certainly one of the most extensively studied models of concurrency, see e.g. [5].

Temporal logics over traces have been introduced to specify the expected behaviors of concurrent systems. Indeed, for practical applications, it is of foremost importance to have specification languages with low complexity for the model checking or the satisfiability problem. Mazurkiewicz traces are labeled partial orders where the ordering describes the causality between events in the trace. This is exactly what is needed to reason about concurrent systems but the prefix structure of traces is rather complex. Due to that, global temporal logics [9, 14, 19, 2] which describe properties of global configurations have a very high complexity. The satisfiability problem is undecidable when the logic is based on an *existential* until [15] or non elementary when a universal until is used [20].

Local temporal logics specify properties of local events in the trace and not of global configurations. Still, local temporal logics have a good expressive power since the simplest one based on (existential)

*Work partly supported by the DAAD-PROCOPE project Temporal and Quantitative Analysis of Distributed Systems.

next and (universal) until has the same expressive power as first order logic over traces [4]. Moreover, local temporal logics have usually a low complexity, i.e., satisfiability can be solved in PSPACE. We cannot expect a lower complexity since already the classical temporal logic LTL over sequences is PSPACE-complete and LTL over words is a special case of most local temporal logics over traces.

Several local temporal logics were introduced [18, 1, 8, 3] and each time the complexity was proved to be in PSPACE or EXPTIME. Whenever a new local temporal logic was introduced, a new proof of the complexity was needed. To circumvent this need, a general framework to study the complexity of local temporal logics was introduced in [6] were it was shown that all local temporal logics where the modalities are definable in monadic second order logic (MSO) are decidable in PSPACE. In this result, we assumed that the architecture of the system is not part of the input which consists of the formula only.

Since the complexity also depends on the architecture of the system, it is important to study the *uniform* satisfiability problem where the input is formed by the formula and the architecture of the system. For systems described by Mazurkiewicz traces, the architecture is given by the dependence alphabet, i.e., the set of actions the system might perform together with the dependency relation between these actions. A more concrete view of the architecture is a set of processes and a mapping from each action to the set of processes involved in this action. Here, two actions are dependent if they share a common process and conversely any dependence alphabet can be described with this more concrete view based on processes.

The uniform satisfiability problem was studied in [7] for general modalities that can be described by MSO formulas. The complexity depends on the number of alternations of set quantifiers in these formulas. Unfortunately, any alternation in the set quantifiers adds an exponent to the space complexity. Fortunately, most local temporal logics that have been studied [18, 1, 4] can be defined without quantifier alternation. Hence, from the general result of [7] we obtain a 2-EXPSPACE upper bound for the uniform satisfiability of these logics.

In the present paper, we improve this result by 2 exponents for the usual temporal logics. More precisely, we prove that the uniform satisfiability problem for the usual temporal logics is in PSPACE. For this, we introduce a general method which is inspired from the proof technique used in [6]. More precisely, we say that a modality is PSPACE-effective if there is a PSPACE algorithm that can compute a Büchi automaton for the modality, given the set of processes that defines the architecture. Then, we show that the uniform satisfiability problem is in PSPACE for all local temporal logics based on PSPACE-effective modalities.

In Section 2 we recall some definitions on Mazurkiewicz traces and in the next section we introduce local temporal logics over traces. The uniform satisfiability problem is defined in Section 4 and we give a general method to prove that this problem is in PSPACE when the modalities are PSPACE-effective.

In Section 8 we show that all modalities introduced in the classical local temporal logics [18, 1, 4] are PSPACE-effective. Some of these results are based on the interesting new notions of *general and special variance* of a Büchi automaton introduced in Section 7. More precisely, assume that we are given a (nondeterministic) Büchi automaton $\mathcal{A}$ for a formula $\varphi(x)$ with one individual free variable x. We want to construct Büchi automata for the formulas $\forall x\ \varphi$ and $\forall x\ \neg\varphi$. The usual construction which is based on $\forall x\ \varphi = \neg\exists x\ \neg\varphi$ uses two complement operations for the former and one complement operation for the latter and therefore increases the size of the automaton by two or one exponents, respectively. Instead, we show that 'universal' automata for the formulas $\forall x\ \varphi$ and $\forall x\ \neg\varphi$ can be constructed efficiently in space $O(m \log |\mathcal{A}|)$ (and have therefore at most $|\mathcal{A}|^{O(m)}$ many states) when the general or special variance of $\mathcal{A}$ is m. We apply these results to Büchi-automata of logarithmic general or special variance in which case this approach improves the usual construction by almost two (by one, resp.) exponent.

This paper uses the process-based approach to Mazurkiewicz traces where the atomic actions are identified with the set of processes involved. The alternative action-based approach starts from a set of atomic actions and declares some of them dependent and some independent. In Section 5, we obtain similar results in this setting.

A question related to the uniform satisfiability problem is the general satisfiability problem. It asks whether a property (expressed by some formula) can occur at all, i.e., whether there exists a set of processes such that the formula becomes satisfiable. In Section 6, we show this problem undecidable for a rather restricted local temporal logic.

2. Traces

We only give very few definitions on Mazurkiewicz traces, those that are needed in this paper. We refer the reader to [12, 5] for more details on the theory of traces.

A *dependence alphabet* is a pair (Σ, D) where Σ is finite alphabet of actions and $D \subseteq \Sigma^2$ is a reflexive and symmetric relation on Σ called *dependence relation*. A *trace* over (Σ, D) is (an isomorphism class of) a labeled, at most countably infinite partial order $t = (V, \preceq, \lambda)$ such that $(V, \preceq)$ is a partial order and $\lambda : V \to \Sigma$ is the labeling function satisfying for all $x, y \in V$

- $\downarrow x = \{z \in V \mid z \preceq x\}$ is finite
- $(\lambda(x), \lambda(y)) \in D$ implies $x \preceq y$ or $y \preceq x$
- $x \prec y$ implies $(\lambda(x), \lambda(y)) \in D$,

where $\prec = \prec \setminus \prec^2$ is the immediate successor relation. The alphabet of the trace t is $\mathrm{alph}(t) = \lambda(V)$. The set $\mathbb{R}(\Sigma, D)$ contains all finite or infinite traces over the dependence alphabet (Σ, D).

A *linearization* of a trace $t = (V, \preceq, \lambda)$ is a linear order $\leq$ on V that extends the partial order $\preceq$ and is at most of order type ω (i.e., also with respect to $\leq$, any node of V dominates only finitely many other nodes). Such a linearization can naturally be identified with a finite or infinite word over Σ. For any linearization $w = a_0 a_1 \dots$ of t, the trace t is isomorphic to $[w] = (V', E^*, \lambda)$ with $V' = \{i \in \mathbb{N} \mid 0 \leq i < |w|\}$, $\lambda(i) = a_i$, and $E = \{(i, j) \in V'^2 \mid i < j \text{ and } (a_i, a_j) \in D\}$.

For $m \in \mathbb{N}$, an *m-extended trace* over (Σ, D) is a trace $(V, \preceq, \lambda)$ together with m sets of positions $X_1, \dots, X_m \subseteq V$. The set of m-extended traces is denoted $\mathbb{R}_m(\Sigma, D)$. If $w = a_0 a_1 a_2 \cdots \in \Sigma^\infty$ is a finite or infinite word and $X_1, \dots, X_m \subseteq \{i \in \mathbb{N} \mid 0 \leq i < |w|\}$, then we denote by $([w], X_1, \dots, X_m)$ the corresponding m-extended trace.

Alternatively, the dependence alphabet can be defined with the more concrete notion of processes. Let Π be a finite set of *process names*. The dependence alphabet induced by Π is (Σ, D) where Σ is the set of nonempty subsets of Π and the dependence relation D is defined by $(a, b) \in D$ iff $a \cap b \neq \emptyset$. We denote by $\mathbb{R}(\Pi)$ the set of finite or infinite traces over the dependence alphabet (Σ, D) induced by Π. We also write $\mathbb{R}_m(\Pi)$ for the set of m-extended traces over Π.

We are interested in the complexity of problems where the architecture, i.e., the dependence alphabet, is part of the input. Using Π instead of the induced dependence alphabet (Σ, D) may allow an exponentially more concise description of the architecture and therefore yields stronger results. Hence, we state and prove our results with the architecture described by Π. Indeed, they also hold when the architecture is presented by an arbitrary dependence alphabet (Σ, D) as explained in Section 5.

3. Local temporal logics

We fix a countably infinite set $\mathcal{P}$ of *process names*. The syntax of a local temporal logic $\mathrm{TL}(B)$ is given by a set B of *modality names* with associated arities. Then the syntax of the logic $\mathrm{TL}(B)$ is defined by the grammar

$$\varphi ::= M(\underbrace{\varphi, \ldots, \varphi}_{\mathrm{arity}(M)}) \mid p$$

where M ranges over B and p over the infinite alphabet $\mathcal{P}$. The size $|\varphi|$ of a formula φ is the number of its subformulas, so, e.g., $|M(p,p)| = 2$ since the only subformulas are p and the formula itself.

To define the semantics of a temporal logic, we associate with any modality name M of arity m and any finite set of processes Π a set of $(m+1)$-extended traces $[\![M]\!]_\Pi \subseteq \mathbb{R}_{m+1}(\Pi)$ over Π. When there is no ambiguity, we simply write $[\![M]\!]$ for $[\![M]\!]_\Pi$.

Let $t = (V, \preceq, \lambda)$ be a trace over some set of processes Π and φ be a formula of $\mathrm{TL}(B)$. The semantics φ^t of φ in t is the set of positions in V where φ holds. The inductive definition is as follows. If $\varphi = p \in \mathcal{P}$, then $\varphi^t = \{v \in V \mid p \in \lambda(v)\}$. If $\varphi = M(\varphi_1, \ldots, \varphi_m)$ where $M \in B$ is of arity $m \geq 0$, then

$$\varphi^t = \{v \in V \mid (t, \varphi_1^t, \ldots, \varphi_m^t, \{v\}) \in [\![M]\!]_\Pi\}.$$

We also write $t, v \models \varphi$ for $v \in \varphi^t$.

Boolean connectives The simplest modalities allow to model Boolean connectives: for $\Pi \subseteq \mathcal{P}$ finite, set

$$[\![\vee]\!]_\Pi = \{(V, \preceq, \lambda, X, Y, \{z\}) \in \mathbb{R}_3(\Pi) \mid z \in X \cup Y\}$$
$$[\![\neg]\!]_\Pi = \{(V, \preceq, \lambda, X, \{y\}) \in \mathbb{R}_2(\Pi) \mid y \notin X\} .$$

Then $(\varphi \vee \psi)^t$ is the set of positions that satisfy φ or ψ. Similarly, $(\neg\varphi)^t$ is the set of positions in t that do not satisfy φ.

Strict universal until. The simplest logic $\mathrm{TL}(\vee, \neg, \mathsf{SU})$ studied in [4] uses, apart from Boolean connectives, only one modality SU of arity 2. The strict universal until $\varphi\ \mathsf{SU}\ \psi$ claims the existence of a vertex y in the proper future of the current one z such that ψ holds at y and φ holds for all vertices properly between z and y. This intuition is captured by the following definition of the language $[\![\mathsf{SU}]\!]_\Pi \subseteq \mathbb{R}_3(\Pi)$:

$$[\![\mathsf{SU}]\!]_\Pi = \{(V, \preceq, \lambda, X, Y, \{z\}) \in \mathbb{R}_3(\Pi) \mid \exists y \in Y : z \prec y \wedge \forall x : z \prec x \prec y \to x \in X\} .$$

Clearly, this is a first-order definition and it was proved in [4] that $\mathrm{TL}(\vee, \neg, \mathsf{SU})$ and first-order logic for traces are equally expressive.

From the strict universal until, we can derive several interesting modalities. Intuitively, $\mathsf{EX}\,\varphi$ (*exists-next*) means that there is an immediate successor of the current vertex where φ holds. Therefore, we have $\mathsf{EX}\,\varphi = \bot\ \mathsf{SU}\ \varphi$ (where $\bot$ means false) and the semantics of EX is inherited from the semantics of SU. It can also be given directly by

$$[\![\mathsf{EX}]\!]_\Pi = \{(V, \preceq, \lambda, X, \{y\}) \in \mathbb{R}_2(\Pi) \mid \exists x \in X : y \prec x\} .$$

Universal until. $\varphi \,\mathsf{U}\, \psi$ is another modality which can be defined as an abbreviation for the formula $\psi \vee (\varphi \wedge (\varphi \mathsf{SU} \psi))$. Alternatively, in our framework, it is given by the following language $[\![\mathsf{U}]\!]_\Pi \subseteq \mathbb{R}_3(\Pi)$:

$$[\![\mathsf{U}]\!]_\Pi = \{(V, \preceq, \lambda, X, Y, \{z\}) \in \mathbb{R}_3(\Pi) \mid \exists y \in Y : z \preceq y \wedge \forall x : z \preceq x \prec y \to x \in X\} .$$

Even though the logic $\mathrm{TL}(\vee, \neg, \mathsf{EX}, \mathsf{U})$ is as expressive as $\mathrm{TL}(\vee, \neg, \mathsf{SU})$ (see [4]), we do not know any direct way to express SU with EX and U.

The classical modalities *eventually* and *always* are obtained from the universal until by $\mathsf{F}\,\varphi = \top \,\mathsf{U}\, \varphi$ and $\mathsf{G}\,\varphi = \neg\,\mathsf{F}\,\neg\varphi$.

Existential until. The temporal logic for causality TLC was introduced in [1]. In our framework, it can be defined by $\mathrm{TL}(\neg, \vee, \mathsf{EX}, \mathsf{EY}, \mathsf{Eco}, \mathsf{EG}, \mathsf{EU}, \mathsf{ES})$. Intuitively, $\mathsf{Eco}\,\varphi$ claims that φ holds for some vertex concurrent to the current one. The formula $\varphi \,\mathsf{EU}\, \psi$ holds if there is a path in the Hasse-diagram of the trace starting in the current vertex such that φ holds along the path until ψ holds (and ψ holds somewhere along this path). Similarly, $\mathsf{EG}\,\varphi$ claims the existence of a maximal such path, starting from the current vertex, where φ always holds. Finally, EY and ES are the past versions of EX and EU, resp. Then the semantics of TLC is obtained with the following modalities

$$\begin{aligned}
[\![\mathsf{EY}]\!]_\Pi &= \{(V, \preceq, \lambda, X, \{y\}) \in \mathbb{R}_2(\Pi) \mid \exists x \in X : x \prec y\} \\
[\![\mathsf{Eco}]\!]_\Pi &= \{(V, \preceq, \lambda, X, \{y\}) \in \mathbb{R}_2(\Pi) \mid \exists x \in X : \neg(x \preceq y \vee y \preceq x)\} \\
[\![\mathsf{EU}]\!]_\Pi &= \{(V, \preceq, \lambda, X, Y, \{z\}) \in \mathbb{R}_3(\Pi) \mid \exists n \geq 0, \exists x_0 \prec x_1 \prec \cdots \prec x_n : \\
&\qquad\qquad z = x_0 \wedge x_0, x_1, \ldots, x_{n-1} \in X \wedge x_n \in Y\} \\
[\![\mathsf{ES}]\!]_\Pi &= \{(V, \leq, \lambda, X, Y, \{z\}) \in \mathbb{R}_3(\Pi) \mid \exists n \geq 0, \exists x_0 \succ x_1 \succ \cdots \succ x_n : \\
&\qquad\qquad z = x_0 \wedge x_0, x_1, \ldots, x_{n-1} \in X \wedge x_n \in Y\} \\
[\![\mathsf{EG}]\!]_\Pi &= \{(V, \preceq, \lambda, X, \{y\}) \in \mathbb{R}_3(\Pi) \mid \exists P \subseteq X : P \text{ maximal path in } (V, \prec) \text{ starting in } y\}
\end{aligned}$$

For cograph dependence alphabets, TLC has the same expressive power as first-order logic [3], but due to the claim of the existence of a path in the modalities EU, ES or EG it can express properties that are not expressible in first-order logic for some other dependence alphabets.

Process-based modalities. We conclude the section by considering temporal logics where the modalities are linked to processes. The first such logic was introduced by Thiagarajan [18] but this logic is not *pure future* and we still do not know its expressive power. An alternative was given in [4] and shown to be expressively complete for FO. It is based on the modalities X_p and U_p for $p \in \mathcal{P}$. Intuitively, $\mathsf{X}_p\,\varphi$ claims that φ holds on the first vertex of process p which is *strictly above* the current one. Hence, we have $\mathsf{X}_p\,\varphi = (\neg p)\ \mathsf{SU}\ (p \wedge \varphi)$. Similarly, $\varphi \,\mathsf{U}_p\, \psi$ says that the sequence of vertices of process p which are *above* the current one satisfy φ until ψ. Therefore, $\varphi \,\mathsf{U}_p\, \psi = (p \to \varphi)\ \mathsf{U}\ (p \wedge \psi)$.

Finally, we show that the temporal logic over traces TrPTL introduced by Thiagarajan [18] can also be dealt with in our framework. It is based on modalities $\mathcal{O}_p$ and $\mathcal{U}_p$ ($p \in \mathcal{P}$) of arity 1 and 2 respectively.

The semantics given in [18] is that of a global temporal logic. Hence it may come as a surprise that we can deal with it in our framework. But actually, apart initially, formulas are evaluated at *prime* configurations, i.e., configurations having exactly one maximal element. By identifying a prime configuration with its maximal vertex we see that the logic is actually local. Intuitively, $\mathcal{O}_p\,\varphi$ means that φ

holds at the first vertex of process p which is *not below* the current one. Similarly, $\varphi\,\mathcal{U}_p\,\psi$ means that we have φ until ψ on the sequence of vertices located on process p and that are *not below* the current vertex (actually, it is slightly more complex since the sequence includes the last vertex of process p which is below the current one if it exists). Formally, the semantics is defined as follows:

$$\begin{aligned}
[\![\mathcal{O}_p]\!]_\Pi = \{(V,\preceq,\lambda,X,\{y\}) \in \mathbb{R}_2(\Pi) \mid \exists x \in X :\\
p \in \lambda(x) \wedge x \not\preceq y \wedge \forall z : (z \prec x \wedge p \in \lambda(z)) \to z \preceq y\}\\
[\![\mathcal{U}_p]\!]_\Pi = \{(V,\preceq,\lambda,X,Y,\{z\}) \in \mathbb{R}_3(\Pi) \mid \exists y \in Y : p \in \lambda(y)\\
\wedge \forall x : (p \in \lambda(x) \wedge x \preceq z) \to x \preceq y\\
\wedge \forall x : (p \in \lambda(x) \wedge x \prec y \wedge \neg\exists x' : (p \in \lambda(x') \wedge x \prec x' \preceq z)) \to x \in X\}
\end{aligned}$$

Since the logic TrPTL is defined by FO-formulas, it is contained in FO but the precise expressive power of TrPTL is still unknown.

4. Uniform satisfiability problem for local temporal logics

Let $\mathrm{TL}(B)$ be a local temporal logic. The uniform satisfiability problem for $\mathrm{TL}(B)$ is the following:

input: a finite set of processes Π and a formula φ of $\mathrm{TL}(B)$

question: Is there a trace $t \in \mathbb{R}(\Pi)$ and a vertex v in t with $t, v \models \varphi$?

For an alphabet Σ and $m \in \mathbb{N}$, we will denote $\Sigma_m = \Sigma \times \{0,1\}^m$. Let $w = a_0 a_1 \cdots \in \Sigma^\infty$ be a word over Σ and $X_i \subseteq \{j \mid 0 \le j < |w|\}$ be sets for $1 \le i \le m$. Then $(w, X_1, \cdots, X_m)$ denotes the word $b_0 b_1 \ldots$ over Σ_m with $b_i = (a_i, x_i^1, x_i^2, \ldots, x_i^m)$ and $x_i^j = 1$ iff $i \in X_j$.

In order to decide this satisfiability problem, we need some effectiveness assumptions on the modalities from B. Here, we assume that the semantics of each modality can be described by a finite automaton which can be constructed in PSPACE.

We use automata $\mathcal{B} = (Q, \Gamma, I, T, F, R)$ accepting both finite and infinite words. Here Q is the finite set of states, Γ the input alphabet, $I \subseteq Q$ the subset of initial states, $T \subseteq Q \times \Gamma \times Q$ the (nondeterministic) transition relation, $F \subseteq Q$ defines the acceptance condition for finite runs and $R \subseteq Q$ defines the Büchi acceptance condition for infinite runs. We simply call them Büchi automata.

Definition 4.1. A modality M of arity m is PSPACE-effective if there exists a PSPACE algorithm with the following specification

input: a finite set of processes Π

output: a Büchi-automaton $\mathcal{C}_{M,\Pi}$ that accepts the word language over Σ_{m+1} (with Σ the set of nonempty subsets of Π) defined by

$$\{(w, X_1, \ldots, X_{m+1}) \in (\Sigma_{m+1})^\infty \mid \forall x : x \in X_{m+1} \leftrightarrow ([w], X_1, \ldots, X_m, \{x\}) \in [\![M]\!]_\Pi\}\ .$$

A temporal logic $\mathrm{TL}(B)$ is PSPACE-*effective* if its modalities are uniformly PSPACE-effective (i.e., the automata $\mathcal{C}_{M,\Pi}$ can be constructed in PSPACE on input M and Π).

Note that, since the automaton $\mathcal{C}_{M,\Pi}$ can be constructed in polynomial space, it can have at most $2^{\mathtt{poly}(|\Pi|)}$ many states.

The atomic propositions $p \in \mathcal{P}$ and the Boolean connectives are easy to deal with. More precisely, for each $p \in \mathcal{P}$, there is a one state automaton $\mathcal{C}_{p,\Pi}$ accepting the words $(w, X) \in \Sigma_1^\infty$ such that $X = p^{[w]} = \{x \mid 0 \le x < |w| \text{ and } p \in \lambda(x)\}$. Also, there is a one state automaton $\mathcal{C}_{\neg,\Pi}$ accepting the words $(w, X, Y) \in \Sigma_2^\infty$ such that $Y = \{x \mid 0 \le x < |w|\} \setminus X$ and there is a one state automaton $\mathcal{C}_{\vee,\Pi}$ accepting the words $(w, X, Y, Z) \in \Sigma_3^\infty$ such that $Z = X \cup Y$.

Although Definition 4.1 might look rather restricted, as it turns out, all the temporal modalities mentioned in Section 3 fall into this setting. We show this in Section 8 using some general results that we prove in Section 7.

Here, we describe the general method, inspired from [6], to solve the uniform satisfiability problem of the logic $\mathrm{TL}(B)$ when automata $\mathcal{C}_{M,\Pi}$ can be computed for each modality $M \in B$.

Let $\mathrm{TL}(B)$ be some PSPACE-effective temporal logic and let Π be some finite set of processes. Since Π is fixed throughout the construction, we will abbreviate $\mathcal{C}_{M,\Pi}$ by $\mathcal{C}_M$ for any modality name $M \in B$. We still denote by Σ the set of nonempty subsets of Π. For formulas φ and ψ, we write $\varphi \le \psi$ if φ is a subformula of ψ (this includes the case $\varphi = \psi$). Let ξ be a formula from $\mathrm{TL}(B)$ and let $\mathrm{Sub}(\xi) = \{\varphi \in \mathrm{TL}(B) \mid \varphi \le \xi\}$. Let $w \in \Sigma^\infty$ and, for $\varphi \le \xi$, let X_φ be sets of positions in w. As explained above, the tuple $(w, (X_\varphi)_{\varphi \le \xi})$ can be considered as a word $\overline{w}$ over the alphabet $\overline{\Sigma} = \Sigma \times \{0,1\}^{\mathrm{Sub}(\xi)}$. For $\psi = M(\varphi_1, \ldots, \varphi_m) \le \xi$, let $\overline{w}{\restriction}\psi = (w, X_{\varphi_1}, \ldots, X_{\varphi_m}, X_\psi) \in (\Sigma_{m+1})^\infty$.

The construction. For a formula $\varphi \in \mathrm{TL}(B)$, let $\mathrm{top}(\varphi)$ be the outermost modality name of φ. Formally, we set $\mathrm{top}(p) = p$ for $p \in \mathcal{P}$ and $\mathrm{top}(M(\varphi_1, \ldots, \varphi_m)) = M$. Let $Q = \prod_{\varphi \le \xi} Q_{\mathrm{top}(\varphi)}$ be the set of states of the automaton $\mathcal{A}_\xi$ where $Q_{\mathrm{top}(\varphi)}$ is the set of states of the Büchi-automaton $\mathcal{C}_{\mathrm{top}(\varphi)}$. The alphabet of $\mathcal{A}_\xi$ is $\overline{\Sigma}$. For a letter $\overline{a} \in \overline{\Sigma}$ and states $p = (p_\varphi)_{\varphi \le \xi}$ and $q = (q_\varphi)_{\varphi \le \xi}$, we have a transition $p \xrightarrow{\overline{a}} q$ in $\mathcal{A}_\xi$ if and only if, for all $\varphi \le \xi$, we have $p_\varphi \xrightarrow{\overline{a}{\restriction}\varphi} q_\varphi$ in the automaton $\mathcal{C}_{\mathrm{top}(\varphi)}$. Note that a sequence of states $p^0, p^1, \ldots$ defines a run of $\mathcal{A}_\xi$ for a word $\overline{w} \in \overline{\Sigma}^\infty$ if and only if for each $\varphi \le \xi$, its projection $p^0_\varphi, p^1_\varphi, \ldots$ on φ is a run of $\mathcal{C}_{\mathrm{top}(\varphi)}$ for the word $\overline{w}{\restriction}\varphi$. A run of $\mathcal{A}_\xi$ is accepting if and only if for each $\varphi \le \xi$, its projection on $\mathcal{C}_{\mathrm{top}(\varphi)}$ is accepting (here we use a generalized Büchi acceptance condition).

Lemma 4.1. Let $\overline{w} = (w, (X_\varphi)_{\varphi \le \xi}) \in \overline{\Sigma}^\infty$. Then, $\overline{w} \in \mathcal{L}(\mathcal{A}_\xi)$ if and only if for each $\varphi \le \xi$ we have $X_\varphi = \varphi^{[w]} = \{x \mid [w], x \models \varphi\}$.

Proof:
Assume $\overline{w} \in \mathcal{L}(\mathcal{A}_\xi)$. We show that $X_\varphi = \varphi^{[w]}$ by structural induction on $\varphi \le \xi$. This is clear for $\varphi = p \in \mathcal{P}$. So let $\varphi = M(\varphi_1, \ldots, \varphi_m) \le \xi$. Assume by induction that $\varphi_i^{[w]} = X_{\varphi_i}$ holds for $1 \le i \le m$. Since $\overline{w}$ is accepted by the automaton $\mathcal{A}_\xi$, the word $\overline{w}{\restriction}\varphi = (w, X_{\varphi_1}, \ldots, X_{\varphi_m}, X_\varphi)$ is accepted by $\mathcal{C}_M$. Hence, using the definition of $\mathcal{C}_M$ and the hypothesis we get

$$X_\varphi = \{x \mid ([w], X_{\varphi_1}, \ldots, X_{\varphi_m}, \{x\}) \in [\![M]\!]_\Pi\} = \varphi^{[w]}.$$

For the other direction, assume that $\varphi^{[w]} = X_\varphi$ for all $\varphi \le \xi$. Clearly, for $\varphi = p \in \mathcal{P}$, the word $\overline{w}{\restriction}\varphi \in \Sigma_1^\infty$ is accepted by $\mathcal{C}_p$. Let $\varphi = M(\varphi_1, \ldots, \varphi_m) \le \xi$. Then $X_{\varphi_i} = \varphi_i^{[w]}$ and therefore

$\varphi^{[w]} = \{x \mid ([w], \varphi_1^{[w]}, \ldots, \varphi_m^{[w]}, \{x\}) \in [\![M]\!]_\Pi\} = \{x \mid ([w], X_{\varphi_1}, \ldots, X_{\varphi_m}, \{x\}) \in [\![M]\!]_\Pi\} = X_\varphi$. Since $\overline{w}{\restriction}\varphi = (w, X_{\varphi_1}, \ldots, X_{\varphi_m}, X_\varphi)$ we deduce from the definition of $\mathcal{C}_M$ that $\overline{w}{\restriction}\varphi$ is accepted by $\mathcal{C}_M$. Since this holds for each $\varphi \leq \xi$ we obtain $\overline{w} \in \mathcal{L}(\mathcal{A}_\xi)$. □

Proposition 4.1. The formula $\xi \in \mathrm{TL}(B)$ is satisfiable by some trace over Π if and only if there exists $\overline{w} = (w, (X_\varphi)_{\varphi \leq \xi}) \in \mathcal{L}(\mathcal{A}_\xi)$ with $X_\xi \neq \emptyset$.

Proof:
Assume that ξ is satisfiable by some trace t. Consider any linearization $w \in \Sigma^\infty$ of t and a position x in w with $[w], x \models \xi$. Let $\overline{w} = (w, (\varphi^{[w]})_{\varphi \leq \xi}) \in \overline{\Sigma}^\infty$. By Lemma 4.1 we get $\overline{w} \in \mathcal{L}(\mathcal{A}_\xi)$. Moreover, we have $x \in \xi^{[w]} = X_\xi \neq \emptyset$.

Conversely let $\overline{w} = (w, (X_\varphi)_{\varphi \leq \xi}) \in \mathcal{L}(\mathcal{A}_\xi)$ with $X_\xi \neq \emptyset$. By Lemma 4.1 we get $\emptyset \neq X_\xi = \xi^{[w]} = \{x \mid [w], x \models \xi\}$. Therefore, ξ is satisfiable by the trace $[w]$. □

Theorem 4.1. The uniform satisfiability problem for any PSPACE-effective temporal logic $\mathrm{TL}(B)$ is in PSPACE.

Proof:
Let ξ be some formula from $\mathrm{TL}(B)$ whose satisfiability in $\mathbb{R}(\Pi)$ we want to check. By Proposition 4.1, we have to decide whether $\mathcal{A}_\xi$ accepts some word $\overline{w} = (w, (X_\varphi)_{\varphi \leq \xi})$ with $X_\xi \neq \emptyset$. In order to do so, we have to store in memory a bounded number of states of $\mathcal{A}_\xi$ and to decide whether there is a transition between two such states.

Since the temporal logic $\mathrm{TL}(B)$ is PSPACE-effective, the number of states of any of the automata $\mathcal{C}_M$ is in $2^{\mathtt{poly}(|\Pi|)}$. Recall that the states of $\mathcal{A}_\xi$ are $|\xi|$-tuples of states from the automata $\mathcal{C}_M$. Hence, a state of $\mathcal{A}_\xi$ can be stored in space $|\xi| \cdot \log(2^{\mathtt{poly}(|\Pi|)})$ hence in $\mathtt{poly}(|\xi| + |\Pi|)$. Also, the transition function of $\mathcal{C}_M$ can be checked in space $\mathtt{poly}(|\Pi|)$ and we deduce that the transition function of $\mathcal{A}_\xi$ can also be checked in space $\mathtt{poly}(|\xi| + |\Pi|)$. □

5. Action-based temporal logics

We explain now the slight changes that arise when the architecture is presented by an arbitrary dependence alphabet (Γ, D) instead of a set of processes Π and its induced dependence alphabet. In this action-based approach, we fix a countably infinite set A of *action names*. The syntax of the local temporal logic $\mathrm{TL}_{\mathrm{act}}(B)$ is defined by the grammar

$$\varphi ::= M(\underbrace{\varphi, \ldots, \varphi}_{\mathrm{arity}(M)}) \mid a$$

where M ranges over the set B of modality names and a over the infinite set A of action names. With any modality name M of arity m and any dependence alphabet (Γ, D), we associate a set $[\![M]\!]_{(\Gamma,D)} \subseteq \mathbb{R}_{m+1}(\Gamma, D)$ of $(m+1)$-extended traces over (Γ, D). Then φ^t is defined as before for formulas $\varphi \in \mathrm{TL}_{\mathrm{act}}(B)$ and traces $t = (V, \preceq, \lambda) \in \mathbb{R}(\Gamma, D)$. The only difference is for constants $a \in A$ where we let $a^t = \{v \in V \mid \lambda(v) = a\}$.

The uniform satisfiability problem for the temporal logic $\mathrm{TL}_{\mathrm{act}}(B)$ now becomes:

input: a dependence alphabet (Γ, D) and a formula φ of $\mathrm{TL}_{\mathrm{act}}(B)$
question: Is there a trace $t \in \mathbb{R}(\Gamma, D)$ and a vertex v in t with $t, v \models \varphi$?

To solve this problem efficiently in this context, we adopt the notion of a PSPACE-effective temporal logic as follows: A modality M of arity m is PSPACE-effective if there exists a PSPACE algorithm with the following specification

input: a dependence alphabet (Γ, D)
output: a Büchi-automaton $\mathcal{C}_{M,\Gamma,D}$ that accepts the language over $\Gamma_{m+1} = \Gamma \times \{0,1\}^{m+1}$ defined by

$$\{(w, X_1, \ldots, X_{m+1}) \in (\Gamma_{m+1})^\infty \mid \forall x : x \in X_{m+1} \leftrightarrow ([w], X_1, \ldots, X_m, \{x\}) \in [\![M]\!]_{(\Gamma,D)}\} .$$

A temporal logic $\mathrm{TL}_{\mathrm{act}}(B)$ is PSPACE-*effective* if its modalities are uniformly PSPACE-effective, i.e., the automata $\mathcal{C}_{M,\Gamma,D}$ can be constructed in PSPACE on input M and (Γ, D).

We will show that the uniform satisfiability problem for any PSPACE-effective action-based temporal logic can be solved in polynomial space. This is achieved by a reduction to Theorem 4.1.

First, the set of process names associated with the set of action names A is $\mathcal{P} = \{\{a, b\} \mid a, b \in A\}$. A dependence alphabet (Γ, D) uniquely defines a finite set of processes $\Pi = \{\{a, b\} \mid (a, b) \in D\}$. Note that not all finite subsets of $\mathcal{P}$ are induced by some dependence alphabet. Let, as before, Σ be the set of nonempty subsets of Π. Identifying $c \in \Gamma$ with the set $\{p \in \Pi \mid c \in p\} \in \Sigma$, we obtain $\Gamma \subseteq \Sigma$ and $\mathbb{R}(\Gamma, D) \subseteq \mathbb{R}(\Pi)$. Furthermore, a trace $t = (V, \preceq, \lambda) \in \mathbb{R}(\Pi)$ is in $\mathbb{R}(\Gamma, D)$ iff $\lambda(v) \in \Gamma$ for all $v \in V$.

Now, for each modality M, we infer its *process* semantics from its *action* semantics: $[\![M]\!]_\Pi = [\![M]\!]_{(\Gamma,D)}$ if Π is defined by some dependence alphabet (Γ, D), and $[\![M]\!]_\Pi = \emptyset$ otherwise.

Now let $\varphi \in \mathrm{TL}_{\mathrm{act}}(B)$ be a formula. Then φ may contain subformulas of the form a with $a \in A$. For $a \notin \Gamma$, replace any of these occurrences by $\bot$, otherwise, replace them by $\bigwedge_{p \in a} p \wedge \bigwedge_{p \in \Pi \setminus a} \neg p$. These replacements result in a process-based formula $\overline{\varphi} \in \mathrm{TL}(B)$. Then it is an easy exercise to prove that for all $t \in \mathbb{R}(\Gamma, D)$ and v in t we have $t, v \models \varphi$ (where the modalities M are evaluated by $[\![M]\!]_{(\Gamma,D)}$) iff $t, v \models \overline{\varphi}$ (where the modalities M are evaluated by $[\![M]\!]_\Pi$).

Consider now the PSPACE-effective unary modality *everywhere* whose semantics is given by

$$[\![\mathsf{E}]\!]_{(\Gamma,D)} = \{(V, \preceq, \lambda, X, \{y\}) \in \mathbb{R}_2(\Gamma, D) \mid \forall x : x \in X\}.$$

Then, φ is satisfiable over $\mathbb{R}(\Gamma, D)$ iff $\varphi \wedge \mathsf{E} \bigvee_{a \in \Gamma} a$ is satisfiable over $\mathbb{R}(\Gamma, D)$ iff $\overline{\varphi \wedge \mathsf{E} \bigvee_{a \in \Gamma} a}$ is satisfiable over $\mathbb{R}(\Pi)$. Thus, we reduced the instance (φ, Γ, D) of the uniform satisfiability problem of $\mathrm{TL}_{\mathrm{act}}(B)$ to the instance $(\overline{\varphi \wedge \mathsf{E} \bigvee_{a \in \Gamma} a}, \Pi)$ of the uniform satisfiability problem of $\mathrm{TL}(B \cup \{\mathsf{E}\})$. By Theorem 4.1, the latter can be decided in space polynomial in $|\Pi| + |\overline{\varphi \wedge \mathsf{E} \bigvee_{a \in \Gamma} a}|$. Since $|\Pi| \leq |\Gamma|^2$, we proved

Corollary 5.1. The uniform satisfiability problem of any PSPACE-effective temporal logic $\mathrm{TL}_{\mathrm{act}}(B)$ is in PSPACE.

6. General satisfiability

Let $\mathrm{TL}(B)$ be a local temporal logic. The general satisfiability problem for $\mathrm{TL}(B)$ is the following:

Figure 1. Shape of pointed trace (t, v)

input: a formula φ of $\mathrm{TL}(B)$
question: Is there a finite set of processes Π, a trace $t \in \mathbb{R}(\Pi)$ and a vertex v in t with $t, v \models \varphi$?

We show that this general satisfiability problem of the simple temporal logic $\mathrm{TL}(\vee, \neg, \mathsf{SU}, \mathsf{EY})$ is undecidable. Recall that formulas of this logic can also use the derived modalities universal until U, always G, and existential next EX. For this undecidability to hold, it is important that there is no bound on the size of the set Π.

To prove this undecidability, we reduce the halting problem (with empty input) of Turing machines to the general satisfiability problem. So let $\mathcal{M}$ be a Turing machine with sets of states Q, of tape symbols Γ, and let \$ be an additional symbol. Furthermore, fix two symbols ℓ and r. Now define the following sets of processes

$$\begin{aligned}\Pi_\ell &= \{\ell\} \times (Q \cup \Gamma \cup \{\$\}),\\ \Pi_r &= \{r\} \times (Q \cup \Gamma \cup \{\$\}), \text{ and}\\ \Pi_0 &= \{\ell, r\} \cup \Pi_\ell \cup \Pi_r\ .\end{aligned}$$

Consider the following formula

$$\varphi_0 = r \wedge \mathsf{G}\Big((\ell \leftrightarrow \neg r) \wedge \mathsf{EX}(r \wedge \mathsf{EY}\,\ell) \wedge \mathsf{EX}(\ell \wedge \mathsf{EY}\,r)\Big)\ .$$

Let Π be some set of processes, let $t = (V, \preceq, \lambda) \in \mathbb{R}(\Pi)$ and $v \in V$. Then $t, v \models \varphi_0$ if and only if $\{\ell, r\} \subseteq \Pi$ and the pointed trace (t, v) has the shape indicated in Fig. 1. In that figure, solid arrows denote the covering relation and dotted arrows its transitive closure, i.e., the strict order. Furthermore, all the nodes in the first row belong to process ℓ and all those in the second to process r.

Now, consider the formula

$$\varphi_1 = \mathsf{G}\left[\begin{array}{ll} & \ell \to \bigvee_{p \in \Pi_\ell}(p \wedge \bigwedge_{q \in \Pi_r \cup \Pi_\ell \setminus \{p\}} \neg q)\\ \wedge & r \to \bigvee_{p \in \Pi_r}(p \wedge \bigwedge_{q \in \Pi_\ell \cup \Pi_r \setminus \{p\}} \neg q)\end{array}\right]$$

Let $t = (V, \preceq, \lambda) \in \mathbb{R}(\Pi)$ and $v \in V$ such that $t, v \models \varphi_0 \wedge \varphi_1$. The pointed trace (t, v) has the form described above. Moreover, the formula φ_1 expresses that the events in the first row of t (i.e., the events on process ℓ) that are in the future of v encode some word from Π_ℓ^ω and therefore some word u_ℓ from $(Q \cup \Gamma \cup \{\$\})^\omega$. Similarly, the events from process r that are above v encode some word u_r from $(Q \cup \Gamma \cup \{\$\})^\omega$.

Next, consider the following formula

$$\varphi_2 = (r, \$) \wedge \mathsf{G}\Big(\big((r, \$) \to \mathsf{EX}(\ell, \$)\big) \wedge \big((\ell, \$) \to \mathsf{EX}(r, \$)\big)\Big) \wedge \neg\big((\neg\ell)\ \mathsf{SU}\ (r, \$)\big)\ .$$

The first two conjuncts express that the events that are marked by filled circles belong to process $(\ell, \$)$ and $(r, \$)$, resp. By the last conjunct, none of the events between v and the next filled event on process r belongs to $(r, \$)$. Hence, again by the second conjunct, the filled events are precisely those that belong to $(\ell, \$)$ and $(r, \$)$, respectively. Hence the two infinite words u_ℓ and u_r over $Q \cup \Gamma \cup \{\$\}$ can be written as

$$u_\ell = \$u_\ell^0\$u_\ell^1\$u_\ell^2 \dots \text{ and } u_r = \$u_r^0\$u_r^1\$u_r^2 \dots$$

with $u_\ell^i, u_r^i \in (Q \cup \Gamma)^*$ and such that $|u_\ell^i| = |v_r^j|$ for all $i, j \geq 0$.

It remains to express by a formula $\varphi_{\mathcal{M}}$ that

(1) u_r^0 is the initial configuration of the Turing machine $\mathcal{M}$ on the empty word,
(2) $u_r^i = u_\ell^i$,
(3) $u_\ell^i \vdash_{\mathcal{M}} u_r^{i+1}$ or $u_\ell^i = u_r^{i+1}$, and
(4) u_r contains an accepting state.

Since all this is rather standard, we leave to the interested reader the task of writing the formula $\varphi_{\mathcal{M}}$.

Let $\varphi = \varphi_0 \wedge \varphi_1 \wedge \varphi_2 \wedge \varphi_{\mathcal{M}}$. We show that φ is *generally satisfiable* iff the Turing machine $\mathcal{M}$ accepts the empty word.

Assume first that $\mathcal{M}$ accepts the empty word and let n be larger than the maximal size used by the tape during the accepting computation of $\mathcal{M}$ starting from the empty word. Let $u^0, u^1, \dots, u^m \in (Q \cup \Gamma)^n$ be words encoding the accepting computation: u^0 is the initial configuration on the empty word, $u^i \vdash u^{i+1}$ for $0 \leq i < m$ and u^m contains the accepting state. Let $\Pi = \Pi_0 \cup \{p_0, p_1, \dots, p_n\}$. Then, there exists a pointed trace (t, v) over Π whose shape is described by Figure 1 and where the words encoded on process ℓ and r are

$$u_\ell = u_r = \$u^0\$u^1\$ \cdots \$u^m\$u^m\$u^m\$ \cdots$$

Note that we need the processes $p_0, p_1, \dots, p_n$ to get the slanted arrows in Figure 1: the k-th vertices on process ℓ or r after a filled node belong to process p_k. By construction, we have $t, v \models \varphi$, hence φ is generally satisfiable.

Conversely, if there exists a finite set of processes Π, a trace $t = (V, \preceq, \lambda) \in \mathbb{R}(\Pi)$, and a node $v \in V$ such that $t, v \models \varphi$ then we show easily that the Turing machine $\mathcal{M}$ accepts the empty word.

Since the formula φ can be constructed from $\mathcal{M}$, we showed

Theorem 6.1. The general satisfiability problem for the local temporal logic $\mathrm{TL}(\{\mathsf{SU}, \mathsf{EY}, \wedge, \neg\})$ is undecidable.

7. Universal first order quantification and Büchi automata

Let $\mathcal{B}$ be a Büchi-automaton over the alphabet $\Sigma_1 = \Sigma \times \{0, 1\}$. It is the aim of this section to construct a "small" automaton for the language

$$\{(w, X) \in \Sigma_1^\infty \mid \forall x : x \in X \leftrightarrow (w, \{x\}) \in \mathcal{L}(\mathcal{B})\} .$$

We show in Section 7.4 that this is useful to prove that a modality M is PSPACE-effective. Indeed, if we start with an automaton $\mathcal{B}_{M,\Pi}$ accepting the language $[\![M]\!]_\Pi$ then we obtain the automaton $\mathcal{C}_{M,\Pi}$ as defined in Definition 4.1.

We first show how to construct a "small" automaton $\mathcal{C}$ for the *universal* language of $\mathcal{B}$ defined as $\mathcal{L}_\forall(\mathcal{B}) = \{w \in \Sigma^\infty \mid \forall x : (w, \{x\}) \in \mathcal{L}(\mathcal{B})\}$. The standard approach would use the definition of the universal quantifier $\forall = \neg\exists\neg$. Hence, the number of states of the resulting automaton $\mathcal{C}$ could be doubly exponential in that of $\mathcal{B}$. Here, we will show that a single exponential suffices in general.

We are also interested in an automaton $\overline{\mathcal{C}}$ for the universal language of the complement of $\mathcal{B}$: $\mathcal{L}_\forall(\overline{\mathcal{B}}) = \{w \in \Sigma^\infty \mid \forall x : (w, \{x\}) \notin \mathcal{L}(\mathcal{B})\}$. The standard approach yields an automaton $\overline{\mathcal{C}}$ with exponentially many states.

Moreover, we show that if the pebble x has only little influence (in two related senses to be made precise below) on the behavior of $\mathcal{B}$, then we can build even smaller automata $\mathcal{C}$ and $\overline{\mathcal{C}}$.

7.1. Complementation of Büchi automata

We first revisit the complementation construction for Büchi automata in order to infer precise bounds on the space complexity and the number of states obtained.

Let $\mathcal{B} = (Q, \Sigma, I, T, F, R)$ be a Büchi-automaton. For $w \in \Sigma^*$ and $p \in Q$, let $p \cdot w$ denote the set of states $q \in Q$ with $p \xrightarrow{w} q$ in $\mathcal{B}$. Also, let $P \cdot w = \bigcup_{p \in P} p \cdot w$ for $P \subseteq Q$.

Proposition 7.1. Let $\mathcal{B} = (Q, \Sigma, I, T, F, R)$ be a Büchi-automaton such that $|I \cdot w| \leq m$ for any $w \in \Sigma^*$. Then, in space $O(m \log |Q|)$, one can compute a Büchi-automaton $\mathcal{C}$ over Σ such that $\mathcal{L}(\mathcal{C}) = \Sigma^\infty \setminus \mathcal{L}(\mathcal{B})$.

Proof:
This complexity result can be obtained by a careful inspection of several constructions for the complement of Büchi automata. For finite runs, we simply use the classical subset construction yielding a deterministic automaton. By the hypothesis, each reachable subset contains at most m states from Q and therefore can be encoded with $m \log |Q|$ bits. Hence, the subset construction can be carried out in space $O(m \log |Q|)$.

For infinite runs, our first proof was based on alternating automata following the constructions of [10, 13]. More precisely, the non-depterministic Büchi automaton $\mathcal{B}$ yields immediately an alternating co-Büchi-automaton $\mathcal{B}_1$ for the complement. By our hypothesis, the number of distinct states that appear at a level i in a run-tree of $\mathcal{B}_1$ is bounded by m. Then, $\mathcal{B}_1$ is transformed into an equivalent weak alternating automaton $\mathcal{B}_2$ [10]. The key point to obtain the complexity is that we can restrict to run-trees of $\mathcal{B}_2$ such that the number of distinct states that appear at some level i is also bounded by m. Then, the translation of $\mathcal{B}_2$ to a Büchi automaton $\mathcal{C}$ [13] yields $|Q|^{O(m)}$ many states and can be performed in space $O(m \log |Q|)$.

Another possibility is to use Safra's determinization construction as suggested by an anonymous referee. Following the construction described, e.g., in [16, Chap. I, Sec. 9], we obtain a deterministic Rabin automaton $\mathcal{C}_1$ whose states are labeled trees. Here the key observation is that, by our hypothesis, the set of states that label the root of a tree is of size at most m. It follows that the Safra-trees have at most m nodes that can be choosen from a set V of size $2m$. For $X \subseteq Q$, let T_X be the set of Safra-trees labeled X at the root. Then, the set of Safra-trees in $\mathcal{C}_1$ is the union of all T_X with $|X| \leq m$. Next, following the proof of [16, Chap. I, Prop. 10.4] we get $|T_X| \leq (4m)^{2m}$. The number of subsets X of Q with at most m elements is bounded by $|Q|^m$. Hence the number of Safra-trees in $\mathcal{C}_1$ is at most $|Q|^m \cdot (4m)^{2m}$. The space needed to store such Safra-trees is therefore $O(m \log |Q|)$ and the construction of $\mathcal{C}_1$ can be done in space $O(m \log |Q|)$. We still have to complement the acceptance condition and to turn it into a Büchi

condition. The Rabin automaton $\mathcal{C}_1$ has $2m$ pairs, one for each node from V. The pair associated with node $v \in V$ claims that v is marked infinitely often and that it is ultimately present in all Safra-trees of the run. To check the complement, we guess a subset $U \subseteq V$ and we check that ultimately only nodes in U are marked and that each node in U is infinitely often not present in the Safra-trees of the run. This multiplies the number of states by $(m+1)2^{m+1}$ and yields a Büchi automaton $\mathcal{C}$ which can still be constructed in space $O(m \log |Q|)$. □

7.2. Universal language and general variance

Now let $\mathcal{B} = (Q, \Sigma_1, I, T, F, R)$ be a Büchi-automaton. We aim at a small Büchi-automaton for the universal language of the complement of $\mathcal{B}$

$$\begin{aligned}\mathcal{L}_\forall(\overline{\mathcal{B}}) &= \{w \in \Sigma^\infty \mid \forall x : (w, \{x\}) \notin \mathcal{L}(\mathcal{B})\} \\ &= \Sigma^\infty \setminus \{w \in \Sigma^\infty \mid \exists x : (w, \{x\}) \in \mathcal{L}(\mathcal{B})\}.\end{aligned}$$

The standard approach first projects to Σ^∞ the language $\mathcal{L}(\mathcal{B})$ restricted to the words (w, X) where X is a singleton and then complements the resulting automaton. Hence, the language in question can be accepted by a Büchi-automaton with $|Q|^{O(|Q|)}$ many states. The following criterion on the Büchi-automaton $\mathcal{B}$ allows to avoid this exponential blow-up.

The *general variance* of $\mathcal{B}$, denoted $\mathrm{GenVar}(\mathcal{B})$, is the maximal size of a set

$$I \cdot (w, \emptyset) \cup \bigcup_{0 \le x < |w|} I \cdot (w, \{x\})$$

for $w \in \Sigma^*$. In other words, it is the maximal number of states one can reach reading $w \in \Sigma^*$ independently from the position of the pebble x (and this pebble need not be placed in w at all).

Proposition 7.2. Let $\mathcal{B} = (Q, \Sigma_1, I, T, F, R)$ be a Büchi-automaton over the alphabet Σ_1 with general variance $\mathrm{GenVar}(\mathcal{B}) \le m$. Then one can construct a Büchi-automaton $\overline{\mathcal{C}}$ with $\mathcal{L}(\overline{\mathcal{C}}) = \mathcal{L}_\forall(\overline{\mathcal{B}})$ in space $O(m \log |Q|)$.

Proof:
Doubling the number of states if necessary, we can transform $\mathcal{B}$ so that it has no run on a word $(w, X) \in \Sigma_1^\infty$ with $|X| \ge 2$ and it only accepts words $(w, X) \in \Sigma_1^\infty$ where X is a singleton. Note that the general variance is not changed by this transformation.

Let $\mathcal{B}' = (Q, \Sigma, I, T', F, R)$ be the projection of the automaton $\mathcal{B}$ to the alphabet Σ, i.e., $T' = \{(p, a, q) \mid (p, (a, 0), q) \in T \text{ or } (p, (a, 1), q) \in T\}$. We have $\mathcal{L}(\mathcal{B}') = \{w \in \Sigma^\infty \mid \exists x : (w, \{x\}) \in \mathcal{L}(\mathcal{B})\}$. Since $\mathcal{B}$ does not allow any run on a word (w, X) with $|X| \ge 2$, the set $I \cdot w$ in $\mathcal{B}'$ equals $I \cdot (w, \emptyset) \cup \bigcup_{0 \le x < |w|} I \cdot (w, \{x\})$ in the old automaton $\mathcal{B}$, i.e., $I \cdot w$ contains at most m elements. Hence the result follows from Proposition 7.1. □

7.3. Universal language and special variance

We still assume that $\mathcal{B} = (Q, \Sigma_1, I, T, F, R)$ is a Büchi-automaton. Here, we want to build a small Büchi-automaton for the universal language $\mathcal{L}_\forall(\mathcal{B}) = \{w \in \Sigma^\infty \mid \forall x : (w, \{x\}) \in \mathcal{L}(\mathcal{B})\}$ of $\mathcal{B}$ itself.

Let $w \in \Sigma^\infty$, u be a prefix of w of length i, and $p \in Q$. Provided $\mathcal{B}$ is complete, $p \in I \cdot (u, \emptyset) \cup \bigcup_{0 \le x < |u|} I \cdot (u, \{x\})$ iff $\mathcal{B}$ can reach p after i steps in some run on $(w, \{x\})$ for some position x in w. The set $\mathrm{states}(\mathcal{B}, w, i) \subseteq Q$ is defined analogously disregarding all non-successful runs, i.e., $p \in \mathrm{states}(\mathcal{B}, w, i)$ iff $\mathcal{B}$ can reach p after i steps in some *successful* run on $(w, \{x\})$ for some position x in w. The *special variance* of $\mathcal{B}$, denoted $\mathrm{SpeVar}(\mathcal{B})$, is the maximum of all values $|\mathrm{states}(\mathcal{B}, w, i)|$ for $w \in \Sigma^\infty$ and $i \in \mathbb{N}$. Note that the special variance is always bounded by the general variance.

Proposition 7.3. Let $\mathcal{B} = (Q, \Sigma_1, I, T, F, R)$ be a Büchi-automaton with $\mathrm{SpeVar}(\mathcal{B}) \le m$. Then, in space $O(m \log |Q|)$, one can compute a Büchi-automaton $\mathcal{C}$ over Σ such that $\mathcal{L}(\mathcal{C}) = \mathcal{L}_\forall(\mathcal{B})$.

The proof of this proposition, that uses the following two lemmas, can be found on page 184.

For simplicity, we write $\Sigma(i) = \Sigma \times \{i\}$ for $i = 0, 1$ such that $\Sigma_1 = \Sigma(0) \cup \Sigma(1)$. The canonical projection from Σ_1^∞ onto Σ^∞ is denoted π. Doubling the number of states of $\mathcal{B}$ if necessary, we can assume that if $(w, X) \in \mathcal{L}(\mathcal{B})$ then X is a singleton. Hence, $\mathcal{L}(\mathcal{B}) \subseteq \Sigma(0)^*\Sigma(1)\Sigma(0)^\infty$. A word $w \in \Sigma^\infty$ belongs to $\mathcal{L}_\forall(\mathcal{B})$ iff each word $v \in \Sigma(0)^*\Sigma(1)\Sigma(0)^\infty$ with $\pi(v) = w$ is accepted by $\mathcal{B}$. To accept $\mathcal{L}_\forall(\mathcal{B})$, we first construct an alternating automaton[1] as follows

- The set of states Q' equals $Q \uplus \{B \subseteq Q \mid 0 \le |B| \le m\}$
- The initial condition is $\bigvee\{J \subseteq I \mid 0 \le |J| \le m\}$
- $F' = F \cup \{\emptyset\}$ and $R' = R \cup \{B \subseteq Q \mid 1 \le |B| \le m\}$
- For $p \in Q$ and $a \in \Sigma$, we have $\delta'(p, a) = \bigvee\{q \in Q \mid (p, (a, 0), q) \in T\}$
- For $A \subseteq Q$ with $1 \le |A| \le m$ and $a \in \Sigma$, we set

$$\begin{aligned}\delta'(A, a) = \quad & \bigvee\{B \subseteq Q \mid 1 \le |B| \le m \text{ and } \forall q \in B,\ \exists p \in A : (p, (a, 0), q) \in T\} \\ & \wedge \bigvee\{q \in Q \mid \exists p \in A : (p, (a, 1), q) \in T\}\end{aligned}$$

- Finally, for $a \in \Sigma$ we set $\delta(\emptyset, a) = \bot$.

This finishes the construction of the alternating automaton $\mathcal{B}' = (Q', \iota', \delta', F', R')$.

Lemma 7.1. $\mathcal{L}_\forall(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{B}')$

Proof:
Let $w = a_0a_1a_2\cdots \in \mathcal{L}_\forall(\mathcal{B})$. We call a word $v \in \Sigma(0)^*\Sigma(1)\Sigma(0)^\infty$ *relevant* if $\pi(v) = w$. Since $w \in \mathcal{L}_\forall(\mathcal{B})$, any relevant word is accepted by $\mathcal{B}$, i.e., for any relevant word $v = b_0b_1b_2\ldots$, there exists a successful run $q_0^v \xrightarrow{b_0} q_1^v \xrightarrow{b_1} \ldots$ of $\mathcal{B}$ on v. Using these runs, we define a successful run tree of $\mathcal{B}'$ on w:

- the set of nodes is $V = \{u \in \Sigma(0)^* \cup \Sigma(0)^*\Sigma(1)\Sigma(0)^* \mid \pi(u) \text{ is a prefix of } w\}$.
- the set of edges E is given by $E = \{(u, ub) \mid ub \in V \text{ and } b \in \Sigma_1\}$.

[1] Similarly to our Büchi-automata that accept finite and infinite words, our (Büchi-)alternating automata have two sets of accepting states, one for infinite runs and one for finite runs.

- to define the labeling $\rho : V \to Q'$, let $u \in V$. First assume that $u \in \Sigma(0)^*\Sigma(1)\Sigma(0)^*$. Then u can be extended uniquely to some relevant word v. Recall that $q_0^v, q_1^v \dots$ is a successful run of $\mathcal{B}$ on v. We set $\rho(u) = q_{|u|}^v$.

  Now assume that $u \in \Sigma(0)^*$. If $|u| < |w|$ then u can be extended to some relevant word, but this time, the extension may not be unique. On the other hand, if $|u| = |w|$ then u is not a prefix of any relevant word. So let $\rho(u) = \{q_{|u|}^v \mid v \text{ is a relevant extension of } u\}$. In particular, $\rho(u)$ is a set of states that occur as state number $|u|$ in some successful run of $\mathcal{B}$ on some $(w, \{x\})$ with $|u| \le x < |w|$. Hence, by the assumption on $\mathcal{B}$, the set $\rho(u)$ contains at most m elements and is therefore a state of the alternating automaton $\mathcal{B}'$.

We first prove that this is indeed a run tree. To this aim, let $u \in V$ be an inner node with $n = |u| < |w|$. We have to show

$$\{\rho(ub) \mid ub \in V \text{ and } b \in \Sigma_1\} \models \delta'(\rho(u), a_{n+1}) \,. \tag{1}$$

First consider the case $u \in \Sigma(0)^*\Sigma(1)\Sigma(0)^*$. Then, the unique successor of u in the tree (V, E) is $u' = u(a_n, 0)$. Let v be the unique extension of u' to a relevant word. Since v is also the unique extension of u to a relevant word, we have $\rho(u) = q_n^v$ and $\rho(u') = q_{n+1}^v$. Since the sequence of states q_i^v forms a run of $\mathcal{B}$ on the word v, this implies $(\rho(u), (a_n, 0), \rho(u')) \in T$. Hence (1) follows.

Next suppose $u \in \Sigma(0)^*$ with $|u| = n$. Then the successors of u in the tree (V, E) are the words $u_0 = u\,(a_n, 0) \in \Sigma(0)^*$ and $u_1 = u\,(a_n, 1) \in \Sigma(0)^*\Sigma(1)$. Then we have

$$\begin{aligned} \rho(u) &= \{q_n^v \mid v \text{ is a relevant extension of } u\} \\ \rho(u_0) &= \{q_{n+1}^v \mid v \text{ is a relevant extension of } u_0\} \end{aligned}$$

Since any relevant extension v of u_0 is also a relevant extension of u, for all $q \in \rho(u_0)$ we find $p \in \rho(u)$ such that $(p, (a_n, 0), q) \in T$. Let v be the unique relevant extension of u_1 so that $q_{n+1}^v = \rho(u_1)$. As above, since v is also a relevant extension of u, we have $q_n^v \in \rho(u)$ and $(q_n^v, (a_n, 1), q_{n+1}^v) \in T$. Thus, (1) follows.

It remains to be shown that the run tree (V, E, ρ) is successful. Clearly, its root ε is labeled $\rho(\varepsilon) \subseteq I$ since all the successful runs on relevant words start in I. Since $\rho(\varepsilon) \in Q'$, this implies $1 \le |\rho(\varepsilon)| \le m$ and therefore $\rho(\varepsilon)$ satisfies the initial condition of $\mathcal{B}'$. Now consider a maximal branch in (V, E). Assume first that all its nodes belong to $\Sigma(0)^*$. If the branch is finite then its last label is $\emptyset \in F'$. If it is infinite then all its labels belong to $\{B \subseteq Q \mid 1 \le |B| \le m\}$. Hence the branch is accepting. Alternatively, there exists a relevant word $v \in \Sigma(0)^*\Sigma(1)\Sigma(0)^\infty$ such that the nodes of the branch are the finite prefixes of v. Let n be the position of the letter from $\Sigma(1)$ in v. The sequence of labels of the branch ends with $q_{n+1}^v, q_{n+2}^v, \dots$. Since this is a suffix of a successful run on v, the branch is accepting. □

Lemma 7.2. $\mathcal{L}(\mathcal{B}') \subseteq \mathcal{L}_\forall(\mathcal{B})$

Proof:
Let (V, E, ρ) be a successful run tree of the alternating automaton $\mathcal{B}'$ on the word $w = a_0a_1a_2\cdots \in \Sigma^\infty$. Furthermore, let $n \ge 0$ be some position in w. We have to prove that the relevant word

$$v = (a_0, 0)\,(a_1, 0)\,\dots(a_{n-1}, 0)\,(a_n, 1)\,(a_{n+1}, 0)\,(a_{n+2}, 0)\dots$$

is accepted by the Büchi-automaton $\mathcal{B}$.

Inductively, we construct a maximal branch $x_0, x_1, \ldots$ such that $\emptyset \neq \rho(x_i) \subseteq Q$ for $0 \leq i \leq n$ and $\rho(x_i) \in Q$ for $i > n$. The node x_0 is the root of the run tree (V, E, ρ). Now suppose that $x_i$with $i < |w|$ has been chosen. Then

$$\{\rho(x) \in Q' \mid (x_i, x) \in E\} \models \delta'(\rho(x_i), a_i) \; . \tag{2}$$

Choosing x_{i+1}, we distinguish three cases.

1. Suppose $i < n$. Because of $\emptyset \neq \rho(x_i) \subseteq Q$ and (2), there exists a node $x_{i+1} \in V$ with $(x_i, x_{i+1}) \in E$ and $\rho(x_{i+1}) \subseteq \{q \in Q \mid \exists p \in \rho(x_i) : (p, (a_i, 0), q) \in T\}$. Since $i + 1 \leq n < |w|$ the node x_{i+1} is not a leaf. Using $\delta(\emptyset, a_n) = \bot$, we deduce that $\rho(x_{i+1}) \neq \emptyset$.

2. Now suppose $i = n$. Because of $\emptyset \neq \rho(x_i) \subseteq Q$ and (2), there exist a node $x_{i+1} \in V$ with $(x_i, x_{i+1}) \in E$ and a state $p \in \rho(x_i)$ such that the triple $(p, (a_i, 1), \rho(x_{i+1}))$ is a transition from T.

3. Finally, suppose $i > n$. Because of $\rho(x_i) \in Q$ and (2), there exists $x_{i+1} \in V$ with $(x_i, x_{i+1}) \in E$ and $(\rho(x_i), (a_i, 0), \rho(x_{i+1})) \in T$.

To obtain a successful run of the Büchi-automaton $\mathcal{B}$ on v, we first set $q_i = \rho(x_i)$ for $i > n$. By construction of x_{n+1}, there exists $q_n \in \rho(x_n)$ with $(q_n, (a_n, 1), q_{n+1}) \in T$. Now, if $i < n$ and $q_{i+1} \in \rho(x_{i+1})$ has been chosen, there exists $q_i \in \rho(x_i)$ with $(q_i, (a_i, 0), q_{i+1}) \in T$ by construction of x_{i+1}. This defines a run $q_0, q_1, \ldots$ of $\mathcal{B}$ on v with $q_i \in \rho(x_i)$ for $i \leq n$ and $q_i = \rho(x_i)$ for $n < i \leq |w|$. With $i = 0$, we obtain $q_0 \in \rho(x_0) \subseteq I$, i.e., the run starts in some initial state of $\mathcal{B}$. Since the maximal branch $x_0, x_1, x_2 \ldots$ is accepting and ultimately labelled by states in Q, the run is successful as well. □

Proof of Proposition 7.3:
By Lemmas 7.1 and 7.2, $\mathcal{L}_\forall(\mathcal{B})$ can be accepted by an alternating automaton with state set $Q' = Q \uplus \{B \subseteq Q \mid 0 \leq |B| \leq m\}$. Let (V, E, ρ) be some minimal (with respect to set inclusion) accepting run tree of the alternating automaton $\mathcal{B}'$ on $w \in \Sigma^\infty$. Consider level n in this run tree. First observe that this level contains exactly one node x with $\rho(x) \subseteq Q$. Now let x be some node on level n with $\rho(x) \in Q$. Consider some maximal branch of the run tree that contains x. As we saw in the proof of Lemma 7.2, $\rho(x)$ is state number n in some accepting run of $\mathcal{B}$ on some relevant word. This shows that the set $\{\rho(x) \mid x \text{ is some node on level } n \text{ of the run tree}\}$ equals $\{B\} \cup C$ for some $B, C \subseteq Q$ with $|B|, |C| \leq m$. For infinite runs, adopting the proof from [13], one can construct an equivalent Büchi-automaton $\mathcal{C}$ whose states consist of such sets $\{B\} \cup C$ together with an $(m+1)$-tuple of binary values $\{0, 1\}$. To store one element of Q, space $\log |Q|$ suffices, hence any state of $\mathcal{C}$ can be stored in space $O(m \log |Q|)$. For finite runs, the situation is even simpler since we do not need the $(m+1)$-tuple of binary values. □

7.4. Polynomial variance and PSPACE-effectiveness

Let $\mathcal{B}$ be a Büchi-automaton over the alphabet Σ_1. As announced at the beginning of Section 7, we show here how to construct a "small" automaton for the language

$$\{(w, X) \in \Sigma_1^\infty \mid \forall x : x \in X \leftrightarrow (w, \{x\}) \in \mathcal{L}(\mathcal{B})\} \; .$$

Since this property can be expressed in monadic second order logic, such an automaton $\mathcal{C}$ exists, but its number of states is in general doubly exponential. Using the notion of general and special variance, we present two special cases where this increase can be avoided.

Lemma 7.3. Let $\mathcal{B}$ be a Büchi-automaton over the alphabet Σ_1 with n states and general variance m. Then we can construct in space $O(m \log n)$ a Büchi-automaton $\mathcal{C}$ over the alphabet Σ_1 such that

$$\mathcal{L}(\mathcal{C}) = \{(w, X) \in \Sigma_1^\infty \mid \forall x : x \in X \leftrightarrow (w, \{x\}) \in \mathcal{L}(\mathcal{B})\}.$$

Proof:
There is a Büchi-automaton $\mathcal{B}_1$ over Σ_2 with $2n + 1$ states such that

$$\mathcal{L}(\mathcal{B}_1) = \{(w, X, \{x\}) \in \Sigma_2^\infty \mid x \in X \to (w, \{x\}) \in \mathcal{L}(\mathcal{B})\} .$$

The automaton checks whether $(w, \{x\}) \in \mathcal{L}(\mathcal{B})$ and also whether the second set is a singleton (this requires doubling the number of states of $\mathcal{B}$) and goes into a new accepting state if the second set is not contained in the first, otherwise, it accepts if $\mathcal{B}$ accepts $(w, \{x\})$. The general variance (and therefore the special variance) of this automaton is at most $m + 1$. By Proposition 7.3, we can construct in space $O(m \log n)$ a Büchi automaton $\mathcal{C}_1$ such that $\mathcal{L}(\mathcal{C}_1) = \mathcal{L}_\forall(\mathcal{B}_1)$.

There is also a Büchi-automaton $\mathcal{B}_2$ with $2n$ states and general variance m such that

$$\mathcal{L}(\mathcal{B}_2) = \{(w, X, \{x\}) \in \Sigma_2^\infty \mid x \notin X \wedge (w, \{x\}) \in \mathcal{L}(\mathcal{B})\} .$$

By Proposition 7.2, we can construct in space $O(m \log n)$ a Büchi automaton $\mathcal{C}_2$ such that $\mathcal{L}(\mathcal{C}_2) = \mathcal{L}_\forall(\overline{\mathcal{B}_2}) = \{(w, X) \in \Sigma_1^\infty \mid \forall x : x \notin X \to (w, \{x\}) \notin \mathcal{L}(\mathcal{B})\}$. Still in space $O(m \log n)$ we can construct the automaton $\mathcal{C}$ accepting the intersection $\mathcal{L}(\mathcal{C}_1) \cap \mathcal{L}(\mathcal{C}_2) = \{(w, X) \in \Sigma_1^\infty \mid \forall x : x \in X \leftrightarrow (w, \{x\}) \in \mathcal{L}(\mathcal{B})\}$. □

As a corollary, we obtain a sufficient condition based on the *general* variance to ensure PSPACE-effectiveness of a modality.

Proposition 7.4. Let M be a modality of arity m. Assume that there exists a PSPACE algorithm which, given a finite set of processes Π, computes a Büchi-automaton $\mathcal{B}_{M,\Pi}$ with $\mathrm{GenVar}(\mathcal{B}_{M,\Pi}) \in \mathtt{poly}(|\Pi|)$ accepting the language

$$\mathcal{L}(\mathcal{B}_{M,\Pi}) = \{(w, X_1, \ldots, X_m, \{x\}) \in \Sigma_{m+1}^\infty \mid ([w], X_1, \ldots, X_m, \{x\}) \in [\![M]\!]_\Pi\} .$$

Then, the modality M is PSPACE-effective.

Proof:
Since the automaton $\mathcal{B}_{M,\Pi}$ can be constructed by a PSPACE algorithm, its number of states is in $2^{\mathtt{poly}(|\Pi|)}$. We deduce from Lemma 7.3 that the automaton $\mathcal{C}_{M,\Pi}$ as defined in Definition 4.1 can be constructed by an algorithm working in space $O(\mathrm{GenVar}(\mathcal{B}_{M,\Pi}) \log(2^{\mathtt{poly}(|\Pi|)})) = \mathtt{poly}(|\Pi|)$. □

In some cases, e.g. for the modality $\mathcal{O}_p$, we were not able to obtain a Büchi automaton $\mathcal{B}_{M,\Pi}$ with *general* variance polynomial in $|\Pi|$. In these cases, our proof of PSPACE-effectiveness is based on the special variance.

Lemma 7.4. Let $\mathcal{B}_1$ and $\mathcal{B}_2$ be Büchi-automata over the alphabet Σ_1 such that $(w, \{x\}) \in \mathcal{L}(\mathcal{B}_2)$ iff $(w, \{x\}) \notin \mathcal{L}(\mathcal{B}_1)$ for all $(w, \{x\}) \in \Sigma_1^\infty$. If $\mathcal{B}_1$ and $\mathcal{B}_2$ have at most n states and special variance at most m then we can construct in space $O(m \log n)$ a Büchi-automaton $\mathcal{C}$ over the alphabet Σ_1 such that

$$\mathcal{L}(\mathcal{C}) = \{(w, X) \in \Sigma_1^\infty \mid \forall x : x \in X \leftrightarrow (w, \{x\}) \in \mathcal{L}(\mathcal{B}_1)\}.$$

Proof:
As in the proof of Lemma 7.3 we can construct two Büchi-automata $\mathcal{B}'_1$ and $\mathcal{B}'_2$ over Σ_2 with at most $2n+1$ states and special variance at most $m+1$ such that

$$\begin{aligned}\mathcal{L}(\mathcal{B}'_1) &= \{(w,X,\{x\}) \in \Sigma_2^\infty \mid x \in X \to (w,\{x\}) \in \mathcal{L}(\mathcal{B}_1)\}\\ \mathcal{L}(\mathcal{B}'_2) &= \{(w,X,\{x\}) \in \Sigma_2^\infty \mid x \notin X \to (w,\{x\}) \in \mathcal{L}(\mathcal{B}_2)\}\\ &= \{(w,X,\{x\}) \in \Sigma_2^\infty \mid x \notin X \to (w,\{x\}) \notin \mathcal{L}(\mathcal{B}_1)\}.\end{aligned}$$

where the last equality holds by the hypothesis on $\mathcal{B}_1$ and $\mathcal{B}_2$.

By Proposition 7.3, we can construct in space $O(m \log n)$ two automata $\mathcal{C}_1$ and $\mathcal{C}_2$ such that $\mathcal{L}(\mathcal{C}_1) = \mathcal{L}_\forall(\mathcal{B}'_1)$ and $\mathcal{L}(\mathcal{C}_2) = \mathcal{L}_\forall(\mathcal{B}'_2)$. We conclude as in the proof of Lemma 7.3 since the desired language is $\mathcal{L}(\mathcal{C}_1) \cap \mathcal{L}(\mathcal{C}_2)$. □

As a corollary, we deduce another sufficient condition based on the *special* variance to ensure PSPACE-effectiveness of a modality.

Proposition 7.5. Let M be a modality of arity m. Assume that there exist PSPACE algorithms which, given a finite set of processes Π, compute Büchi-automata $\mathcal{B}_{M,\Pi}$ and $\overline{\mathcal{B}}_{M,\Pi}$ with special variances in $\mathtt{poly}(|\Pi|)$ accepting the languages

$$\begin{aligned}\mathcal{L}(\mathcal{B}_{M,\Pi}) &= \{(w,X_1,\dots,X_m,\{x\}) \in \Sigma_{m+1}^\infty \mid ([w],X_1,\dots,X_m,\{x\}) \in [\![M]\!]_\Pi\}\\ \mathcal{L}(\overline{\mathcal{B}}_{M,\Pi}) &= \{(w,X_1,\dots,X_m,\{x\}) \in \Sigma_{m+1}^\infty \mid ([w],X_1,\dots,X_m,\{x\}) \notin [\![M]\!]_\Pi\}.\end{aligned}$$

Then, the modality M is PSPACE-effective.

Proof:
Since $\mathcal{B}_{M,\Pi}$ and $\overline{\mathcal{B}}_{M,\Pi}$ can both be constructed by PSPACE algorithms, their number of states are in $2^{\mathtt{poly}(|\Pi|)}$. We deduce from Lemma 7.4 that the automaton $\mathcal{C}_{M,\Pi}$ as defined in Definition 4.1 can be constructed by an algorithm working in space $\mathtt{poly}(|\Pi|)$. □

8. Examples of PSPACE-effective modalities

The aim of this section is to show that all modalities described in Section 3 are PSPACE-effective. Throughout this section, let Π denote some finite set of processes and let Σ be the set of nonempty subsets of Π.

8.1. Derived modalities

As a preliminary, we indicate how to construct more involved PSPACE-effective modalities from simpler ones. This will be used repeatedly in the following sections. For instance, the modality X_p is derived from the strict until SU and the Boolean connectives: $\mathsf{X}_p\,\varphi = (\neg p)\ \mathsf{SU}\ (p \wedge \varphi)$.

Let $\mathrm{TL}(B)$ be some local temporal logic. The set of *terms* of $\mathrm{TL}(B)$ is defined by the grammar

$$\tau ::= M(\underbrace{\tau,\dots,\tau}_{\mathrm{arity}(M)}) \mid p \mid X$$

where M ranges over B, p over the infinite alphabet $\mathcal{P}$, and X over the set variables $\{X_1, X_2, \dots\}$. For instance, $(\neg p)\ \mathsf{SU}\ (p \wedge X_1)$ is a term of $\mathrm{TL}(\neg, \wedge, \mathsf{SU})$.

Recall that the semantics of a formula is a set of positions in a trace. Similarly, the semantics of a term τ with free variables $\mathrm{Free}(\tau) \subseteq \{X_1, \dots, X_k\}$ is a set of positions in a k-extended trace. Let $t = (V, \preceq, \lambda)$ be a trace over some set of processes Π and $V_1, \dots, V_k \subseteq V$ be sets of positions. For $p \in \mathcal{P}$, the semantics of the term p is $p^{(t,V_1,\dots,V_k)} = \{v \in V \mid p \in \lambda(v)\}$. For $1 \le i \le k$, we set $X_i^{(t,V_1,\dots,V_k)} = V_i$. The induction then proceeds as in the case of formulas: if $\tau = M(\tau_1, \dots, \tau_m)$ where $M \in B$ is of arity $m \ge 0$, then

$$\tau^{(t,V_1,\dots,V_k)} = \{v \in V \mid (t, \tau_1^{(t,V_1,\dots,V_k)}, \dots, \tau_m^{(t,V_1,\dots,V_k)}, \{v\}) \in [\![M]\!]_\Pi\}.$$

Definition 8.1. Let $\mathrm{TL}(B)$ be some local temporal logic and let M be some m-ary modality. Then M is a *derived modality* if there exists a term τ of $\mathrm{TL}(B)$ with m free variables such that for any finite set of processes Π, we have

$$[\![M]\!]_\Pi = \{(t, V_1, \dots, V_m, \{v\}) \in \mathbb{R}_{m+1}(\Pi) \mid v \in \tau^{(t,V_1,\dots,V_m)}\}\ .$$

If M is a derived modality, then we also say that it can be *expressed with the modalities from* B.

Proposition 8.1. Let $\mathrm{TL}(B)$ be some PSPACE-effective temporal logic and let M be some derived m-ary modality. Then M is PSPACE-effective.

Proof:
We use the notations from Section 4, adapted naturally from formulas to terms. Let τ be the term that defines the modality M. Then, the automaton $\mathcal{A}_\tau$ from Section 4 can be constructed from Π in PSPACE. Note that its alphabet is $\overline{\Sigma}_m = \Sigma_m \times \{0,1\}^{\mathrm{Sub}(\tau)}$. Then, by Lemma 4.1, a word $(w, V_1, \dots, V_m, (V_\sigma)_{\sigma \le \tau})$ is accepted by $\mathcal{A}_\tau$ iff, for any subterm σ of τ, we have $V_\sigma = \sigma^{([w],V_1,\dots,V_m)}$. Hence the projection of the automaton $\mathcal{A}_\tau$ to the alphabet $\Sigma_m \times \{0,1\}$ where we project away all components associated with proper subterms of τ can serve as automaton $\mathcal{C}_{M,\Pi}$ from Definition 4.1. □

8.2. Universal modalities

This section is concerned with the strict universal until SU and its past version, the strict universal since SS and with the modalities that can be derived from them. To this aim, we will construct automata $\mathcal{B}_{\mathsf{SU},\Pi}$ and $\mathcal{B}_{\mathsf{SS},\Pi}$ whose general variances are polynomial in the size of Π. Both these automata are based on the following automaton $\mathcal{B}$.

Construction. The alphabet of the automaton $\mathcal{B}$ is Σ_3 and $\mathcal{B}$ will accept a word (w, X, Y, Z) iff there are $i \in X$ and $k \in Z$ with $i \prec k$ and such that $j \in Y$ for all j with $i \prec j \prec k$. Note here, that we have two orders: the natural linear order $\le$ on the positions of the word w as well as the partial order $\preceq$ of the trace $[w]$.

The set of states of the automaton $\mathcal{B}$ is $Q = \{\mathrm{init}, \mathrm{OK}\} \uplus (2^\Pi \times 2^\Pi)$, init is the unique initial state and OK is the only accepting state both for finite runs and for infinite runs. We first describe intuitively the expected behaviour of $\mathcal{B}$. Let $w = a_1 a_2 \cdots \in \Sigma^\infty$. Now, let $(w, X, Y, Z) =$

$(a_1,x_1,y_1,z_1)(a_2,x_2,y_2,z_2)\cdots \in \Sigma_3^\infty$. If there is a run

$$\mathrm{init} = q_0 \xrightarrow{(a_1,x_1,y_1,z_1)} q_1 \xrightarrow{(a_2,x_2,y_2,z_2)} q_2 \dots$$

of $\mathcal{B}$ then either $q_n = \mathrm{init}$ for all $n \geq 0$ or with $i = \min\{n \geq 0 \mid q_n \neq \mathrm{init}\}$ we have $i \in X$ and for all $n \geq i$, if $q_n \neq \mathrm{OK}$ then $q_n = (A_n, B_n)$ with

$$A_n = \bigcup\{a_j \mid i \preceq j \leq n\} \tag{3}$$

$$B_n = \bigcup\{a_j \mid \exists j' \notin Y : i \prec j' \preceq j \leq n\}\,. \tag{4}$$

Moreover, if $q_n = \mathrm{OK}$ for some n then with $k = \min\{n \geq 0 \mid q_n = \mathrm{OK}\}$ we have $i \prec k$ and $k \in Z$ and $j \in Y$ for all $i \prec j \prec k$.

To this aim, a triple $(p,(a,x,y,z),q)$ is a transition iff one of the following conditions holds

$$\begin{aligned}
& p = \mathrm{init} \wedge q = \mathrm{init} \\
\text{or}\quad & p = \mathrm{init} \wedge x = 1 \wedge q = (a,\emptyset) \\
\text{or}\quad & p = (A,B) \wedge a \cap A = \emptyset \wedge q = (A,B) \\
\text{or}\quad & p = (A,B) \wedge a \cap A \neq \emptyset \wedge a \cap B = \emptyset \wedge z = 1 \wedge q = \mathrm{OK} \\
\text{or}\quad & p = (A,B) \wedge a \cap A \neq \emptyset \wedge a \cap B = \emptyset \wedge z = 0 \wedge y = 0 \wedge q = (A \cup a, B \cup a) \\
\text{or}\quad & p = (A,B) \wedge a \cap A \neq \emptyset \wedge a \cap B = \emptyset \wedge z = 0 \wedge y = 1 \wedge q = (A \cup a, B) \\
\text{or}\quad & p = (A,B) \wedge a \cap A \neq \emptyset \wedge a \cap B \neq \emptyset \wedge q = (A \cup a, B \cup a) \\
\text{or}\quad & p = \mathrm{OK} \wedge q = \mathrm{OK}\,.
\end{aligned}$$

Note that the non-determinism in $\mathcal{B}$ reduces to the choice of whether we leave the state init or not when we are in a position from X (i.e., when $x = 1$).

Lemma 8.1. The automaton $\mathcal{B}$ accepts a word $(w,X,Y,Z) \in \Sigma_3^\infty$ iff there exist $i \in X$ and $k \in Z$ with $i \prec k$ and such that $j \in Y$ for all $i \prec j \prec k$.

Proof:
We first show that $\mathcal{B}$ satisfies the intuition described above. So we consider a run of $\mathcal{B}$ on $(w,X,Y,Z) = (a_1,x_1,y_1,z_1)(a_2,x_2,y_2,z_2)\cdots \in \Sigma_3^\infty$:

$$\mathrm{init} = q_0 \xrightarrow{(a_1,x_1,y_1,z_1)} q_1 \xrightarrow{(a_2,x_2,y_2,z_2)} q_2 \dots$$

and we assume that $q_n \neq \mathrm{init}$ for some $n \geq 0$. Let $i = \min\{n \geq 0 \mid q_n \neq \mathrm{init}\}$. From the second line of the definition of the transition relation we deduce that $i \in X$ and $q_i = (a_i, \emptyset)$. Hence (3,4) holds for $n = i$. Now, let $n > i$ be such that $q_n \neq \mathrm{OK}$. Then we must have $q_{n-1} \neq \mathrm{OK}$ and by induction we may assume that (3,4) holds for $n-1$. We have $A_{n-1} \cap a_n \neq \emptyset$ iff $a_j \cap a_n \neq \emptyset$ for some $i \preceq j < n$ iff $i \prec n$. By definition of the transition relation, we have $A_n = A_{n-1}$ if $A_{n-1} \cap a_n = \emptyset$ and $A_n = A_{n-1} \cup a_n$ otherwise. We deduce that (3) holds for n. Now, if $a_n \cap B_{n-1} \neq \emptyset$ then we find $j' \notin Y$ and $j < n$ such that $j' \preceq j$ and $a_n \cap a_j \neq \emptyset$. We deduce that $j' \prec n$ and $B_n = B_{n-1} \cup a_n$ satisfies (4). Similarly, if $y_n = 0$ and $a_n \cap A_{n-1} \neq \emptyset$ then $i \prec j' = n \notin Y$ and $B_n = B_{n-1} \cup a_n$ satisfies (4). Finally, if

$a_n \cap B_{n-1} = \emptyset$ then there is no $j' \notin Y$ with $i \prec j' \prec n$ and if in addition $y_n = 1$ then there is no $j' \notin Y$ with $i \prec j' \preceq n$. We deduce that in this case (4) holds with $B_n = B_{n-1}$. Moreover, assume that $q_n = \mathrm{OK}$ for some n and let $k = \min\{n \geq 0 \mid q_n = \mathrm{OK}\}$. Since $q_i = (a_i, \emptyset)$ we have $k > i$ and (3,4) holds for $n = k - 1$. By definition of the transition relation we have $z_k = 1$ and $a_k \cap A_{k-1} \neq \emptyset$ and $a_k \cap B_{k-1} = \emptyset$. We deduce that $k \in Z$ and $i \prec k$ and $j \in Y$ for all $i \prec j \prec k$.

Now, assume that (w, X, Y, Z) is accepted by $\mathcal{B}$ and consider an accepting run of $\mathcal{B}$ using the same notations as above. Since the run is accepting, it starts in state init and eventually loops on state OK. Let i and k be minimal with $q_i \neq \mathrm{init}$ and $q_k = \mathrm{OK}$, resp. We have seen above that $i \in X$, $i \prec k$, $k \in Z$ and $j \in Y$ for all $i \prec j \prec k$.

Conversely, assume that there are $i \in X$, $k \in Z$ with $i \prec k$ and $j \in Y$ for all $i \prec j \prec k$. Consider the unique run

$$\mathrm{init} = q_0 \xrightarrow{(a_1,x_1,y_1,z_1)} q_1 \xrightarrow{(a_2,x_2,y_2,z_2)} q_2 \dots$$

of $\mathcal{B}$ with $q_n = \mathrm{init}$ for all $n < i$ and $q_i = (a_i, \emptyset)$, which is indeed possible since $i \in X$. If $q_{k-1} = \mathrm{OK}$ then the run is accepting. So assume that $q_{k-1} \neq \mathrm{OK}$. Then, from the property of $\mathcal{B}$ we have $q_{k-1} = (A_{k-1}, B_{k-1})$ and (3,4) holds for $n = k - 1$. Now, from $i \prec k$ we deduce that $a_k \cap A_{k-1} \neq \emptyset$. Using $j \in Y$ for all $i \prec j \prec k$ we deduce that $a_k \cap B_{k-1} = \emptyset$. Since $k \in Z$ the definition of the transition function implies $q_k = \mathrm{OK}$. Therefore, the run is accepting. □

From the following lemma we will deduce that the general variance of the two automata $\mathcal{B}_{\mathsf{SU},\Pi}$ and $\mathcal{B}_{\mathsf{SS},\Pi}$ derived from $\mathcal{B}$ is polynomial in $|\Pi|$.

Lemma 8.2. Let $w = a_1 a_2 \dots a_n$ and $Y \subseteq \{1, \dots, n\}$. Then the set

$$\bigcup\{\mathrm{init} \cdot (w, X, Y, Z) \mid X, Z \subseteq \{1, \dots, n\}\}$$

contains at most $2 + |\Pi|^2(|\Pi| + 1)$ many elements.

Proof:
Let $X, Z \subseteq \{1, 2, \dots, n\}$ and consider a run

$$\mathrm{init} = q_0 \xrightarrow{(a_1,x_1,y_1,z_1)} q_1 \quad \cdots \quad q_{n-1} \xrightarrow{(a_n,x_n,y_n,z_n)} q_n \,.$$

Then, either $q_n \in \{\mathrm{init}, \mathrm{OK}\}$ or we have $q_n = (A(i), B(i))$ with i minimal such that $q_i \neq \mathrm{init}$ and

$$A(i) = \bigcup\{a_j \mid i \preceq j \leq n\} \quad \text{and} \quad B(i) = \bigcup\{a_j \mid \exists j' \notin Y : i \prec j' \preceq j \leq n\} \,.$$

Therefore, the set $\bigcup\{\mathrm{init} \cdot (w, X, Y, Z) \mid X, Z \subseteq \{1, \dots, n\}\}$ is contained in $H = \{\mathrm{init}, \mathrm{OK}\} \cup \{(A(i), B(i)) \mid 1 \leq i \leq n\}$. Towards a contradiction, suppose the set in question and therefore this set H contains properly more than $2 + |\Pi|^2(|\Pi| + 1)$ states. Then there exist $0 < i_0 < i_1 < \cdots < i_{|\Pi|^2(|\Pi|+1)} \leq n$ such that the tuples $(A(i_j), B(i_j))$ are pairwise distinct. Since the positions on process p are totally ordered for the causal ordering $\prec$, there are at least $1 + |\Pi|(|\Pi| + 1)$ positions totally ordered for $\prec$. Therefore, after renaming if necessary, we can assume that $i_0 \prec i_1 \prec \cdots \prec i_{|\Pi|(|\Pi|+1)} \leq n$. We easily see that $i \preceq i'$ implies $A(i) \supseteq A(i')$. Therefore, we obtain

$$A(i_0) \supseteq A(i_1) \supseteq \cdots \supseteq A(i_{|\Pi|(|\Pi|+1)}) \,.$$

Since all these are nonempty subsets of Π, among the remaining positions, there are at least $|\Pi| + 2$ positions with equal sets. Again, after renaming if necessary, we can assume that $i_0 \prec i_1 \prec \cdots \prec i_{|\Pi|+1} \leq n$ and

$$A(i_0) = A(i_1) = \cdots = A(i_{|\Pi|+1}) .$$

Finally, $i \preceq i'$ also implies $B(i) \supseteq B(i')$. Therefore,

$$B(i_0) \supseteq B(i_1) \supseteq \cdots \supseteq B(i_{|\Pi|+1}) .$$

We deduce that among these subsets of Π, at least two are equal, which is a contradiction. □

We show now that the universal modalities are PSPACE-effective. The strict universal until SU was already defined in Section 3. Here we deal simultaneously with its past version, the strict universal since SS whose semantics $[\![\mathsf{SS}]\!]_\Pi$ is defined by

$$\{(V, \preceq, \lambda, X, Y, \{z\}) \in \mathbb{R}_3(\Pi) \mid \exists y \in Y : y \prec z \wedge \forall x : y \prec x \prec z \to x \in X\} .$$

Proposition 8.2. The modalities SS and SU are PSPACE-effective.

Proof:
We start with the strict universal since. Let $(w, X, Y, \{z\}) \in \Sigma_3^\infty$. Then $([w], X, Y, \{z\}) \in [\![\mathsf{SS}]\!]_\Pi$ iff the word $(w, Y, X, \{z\})$ is accepted by $\mathcal{B}$. The automaton $\mathcal{B}_{\mathsf{SS},\Pi}$ is thus the automaton $\mathcal{B}$ where the two lines for X and Y have been exchanged and which checks in addition that the set Z is a singleton. The automaton $\mathcal{B}$ can be constructed in PSPACE, hence also the automaton $\mathcal{B}_{\mathsf{SS},\Pi}$. The general variance of $\mathcal{B}_{\mathsf{SS},\Pi}$ is polynomial in Π by Lemma 8.2. Hence the result follows from Proposition 7.4.

We turn now to the strict universal until. With the same notations, we have $([w], X, Y, \{z\}) \in [\![\mathsf{SU}]\!]_\Pi$ iff the word $(w, \{z\}, X, Y)$ is accepted by $\mathcal{B}$. Hence, we can conclude as above. □

We have already seen that the Boolean connectives are PSPACE-effective, hence the temporal logic $\mathrm{TL}(\vee, \neg, \mathsf{SU})$ is PSPACE-effective. Also, since the modalities EX and U can be expressed with SU we deduce that the logic $\mathrm{TL}(\vee, \neg, \mathsf{EX}, \mathsf{U})$ is also PSPACE-effective. Similarly, the pure future process based modalities X_p and U_p can be expressed with SU, hence the process based temporal logic $\mathrm{TL}(\vee, \neg, \mathsf{X}_p, \mathsf{U}_p)$ is PSPACE-effective.

The past versions EY, S, Y_p and S_p of EX, U, X_p and U_p can be expressed using SS. Hence they are also PSPACE-effective. Therefore, we can enhance the PSPACE-complete logics mentioned above by past versions of their modalities. The uniform satisfiability problem of the resulting logics is still in PSPACE.

8.3. Modalities used in TrPTL

We show here that the modalities $\mathcal{O}_p$ and $\mathcal{U}_p$ are also PSPACE-effective. Recall that these modalities are neither pure future nor pure past. We will define non-deterministic automata with small special variances in order to use Proposition 7.5.

Proposition 8.3. The modality $\mathcal{O}_p$ is PSPACE-effective.

Proof:
We first define a non-deterministic automaton $\mathcal{A}$ with 2^Π as set of states, where all states except $\emptyset$ are initial and $\emptyset$ is the only accepting state. Even though $\mathcal{A}$ is non-deterministic, it will have a unique accepting run on any word $(w, \{k\}) \in \Sigma_1^\infty$. If we write $(w, \{k\}) = (a_1, y_1)(a_2, y_2)\dots$ then the accepting run will be the sequence $(A_n)_{0\le n\le |w|}$ such that

$$A_n = \bigcup \{a_j \mid n < j \preceq k\} \,. \tag{5}$$

We have a transition $A \xrightarrow{(a,y)} A'$ iff the following holds:

$$\begin{aligned} & y = 1 \wedge A = a \wedge A' = \emptyset \\ \text{or} \quad & y = 0 \wedge a \cap A' = \emptyset \wedge A = A' \\ \text{or} \quad & y = 0 \wedge a \cap A' \neq \emptyset \wedge A' \cup a = A \,. \end{aligned}$$

We first show that the sequence $(A_n)_{n\ge 0}$ defined in (5) forms a successful run on $(w, \{k\})$. If $n = k$ then we have $y_n = 1$ and $A_n = \emptyset$ and $A_{n-1} = a_n$ hence $A_{n-1} \xrightarrow{(a_n,y_n)} A_n$ is a transition of $\mathcal{A}$. If $n > k$ then $A_{n-1} = A_n = \emptyset$ and $y_n = 0$ hence again $A_{n-1} \xrightarrow{(a_n,y_n)} A_n$ is a transition of $\mathcal{A}$. If $0 < n < k$ then $y_n = 0$ and either $a_n \cap A_n = \emptyset$ in which case $n \not\preceq k$ and $A_{n-1} = A_n$, or $a_n \cap A_n \neq \emptyset$ in which case $n \preceq k$ and $A_{n-1} = A_n \cup a_n$. In both cases we have $A_{n-1} \xrightarrow{(a_n,y_n)} A_n$.

Conversely, let $(A_n)_{n\ge 0}$ be a successful run of $\mathcal{A}$ on (w, Y). Let $k = \min\{n \mid A_n = \emptyset\}$. We have $y_k = 1$ and $A_{k-1} = a_k$ hence (5) holds for $k-1$. From the definition of the transition function, it is easy to see that $A_n = \emptyset$ and $y_n = 0$ for all $n > k$ hence (5) holds also for $n \ge k$. Now, assume that (5) holds for some $0 < n < k$. Since $A_{n-1} \xrightarrow{(a_n,y_n)} A_n$ is a transition, we have $y_n = 0$. Since (5) holds for n we have $n \preceq k$ iff $a_n \cap A_n \neq \emptyset$. Hence, $A_{n-1} = A_n \cup a_n$ if $n \preceq k$ and $A_{n-1} = A_n$ otherwise. We deduce that A_{n-1} satisfies (5).

Now, we define the automaton $\mathcal{B} = \mathcal{B}_{\mathcal{O}_p,\Pi}$ over the alphabet Σ_2 whose first component will be $\mathcal{A}$. Its set of states is $2^\Pi \times \{0, 1, 2\}$ and the initial states are $(2^\Pi \setminus \{\emptyset\}) \times \{0\}$. The only accepting state is $(\emptyset, 1)$. We have a transition $(A, q) \xrightarrow{(a,x,y)} (A', q')$ if $A \xrightarrow{(a,y)} A'$ is a transition of $\mathcal{A}$ and

$$q' = \begin{cases} 0 & \text{if } q = 0 \wedge (p \notin a \vee a \cap A' \neq \emptyset) \\ 1 & \text{if } q = 0 \wedge p \in a \wedge a \cap A' = \emptyset \wedge x = 1 \\ 2 & \text{if } q = 0 \wedge p \in a \wedge a \cap A' = \emptyset \wedge x = 0 \\ q & \text{if } q \neq 0. \end{cases}$$

We have seen above that there is only one successful run for the first component. Moreover, the second component of the automaton $\mathcal{B}$ is deterministic once the first component of the run is fixed. Let $(w, X, \{k\}) = (a_1, x_1, y_1)(a_2, x_2, y_2)\cdots \in \Sigma_2^\infty$ and consider the unique run $(A_n, q_n)_{n\ge 0}$ of $\mathcal{B}$ such that the first component is successful. Let $i = \min\{j \mid p \in a_j \wedge j \not\preceq k\}$ with the convention $i = \infty$ if this set is empty. Then, we can check that for all $n \ge 0$,

$$q_n = \begin{cases} 0 & \text{if } n < i \\ 1 & \text{if } i \le n \wedge i \in X \\ 2 & \text{if } i \le n \wedge i \notin X. \end{cases}$$

We deduce that $\mathcal{L}(\mathcal{B}) = \{(w, X, \{k\}) \mid ([w], X, \{k\}) \in [\![\mathcal{O}_p]\!]_\Pi\}$. Moreover, if we change the accepting states to $\{\emptyset\} \times \{0, 2\}$ then we obtain the complementary automaton $\overline{\mathcal{B}}_{\mathcal{O}_p,\Pi}$.

Finally, we show that $\mathrm{SpeVar}(\mathcal{B}) \leq 2|\Pi|(|\Pi| + 1)$. Fix a word (w, X) and $n \in \mathbb{N}$ and assume towards a contradiction that $|\mathrm{states}(\mathcal{B}, (w, X), n)| > 2|\Pi|(|\Pi| + 1)$. For each $k > 0$, let $(A_n(k), q_n(k))$ be the state reached on the successful run of $\mathcal{B}$ on $(w, X, \{k\})$. Note that in a successful run of $\mathcal{B}$, the value $q = 2$ cannot occur. Then we find $k_0 < k_1 < \cdots < k_{|\Pi|(|\Pi|+1)}$ such that the sets $A_n(k_i)$ are pairwise distinct and the values $q_n(k_i)$ are all equal. Since the positions on a process q are totally ordered for the causal ordering $\prec$, there are at least $|\Pi| + 2$ among these positions totally ordered for $\prec$. Therefore, after renaming if necessary, we can assume that $k_0 \prec k_1 \prec \cdots \prec k_{|\Pi|+1}$. We deduce that $A_n(k_0) \subseteq A_n(k_1) \subseteq \cdots \subseteq A_n(k_{|\Pi|+1})$ which contradicts the fact that these sets are pairwise distinct. The same arguments yield the analogous result for the automaton $\overline{\mathcal{B}}_{\mathcal{O}_p,\Pi}$.

Using Proposition 7.5 we deduce that $\mathcal{O}_p$ is PSPACE-effective. □

Next, we turn to the modality $\mathcal{U}_p$. Recall that $\varphi \,\mathcal{U}_p\, \psi$ means that we have φ until ψ on the sequence of vertices located on process p and starting from the last vertex of process p which is in the past of the current vertex if it exists and starting from the first vertex of process p which is not in the past of the current vertex otherwise. To deal with $\mathcal{U}_p$ we introduce another unary modality $\mathcal{O}'_p$. Intuitively, $\mathcal{O}'_p\varphi$ means that φ holds at the last vertex on process p which is in the past of the current vertex (and that this vertex exists). Formally, its semantics is defined by

$$[\![\mathcal{O}'_p]\!]_\Pi = \{(V, \preceq, \lambda, X, \{y\}) \in \mathbb{R}_2(\Pi) \mid \exists x \in X : \\ p \in \lambda(x) \wedge x \preceq y \wedge \forall z : (z \preceq x \wedge p \in \lambda(z)) \to z \preceq y\} .$$

Then, we have $\varphi \,\mathcal{U}_p\, \psi = \mathcal{O}'_p(\varphi \,\mathsf{U}_p\, \psi) \vee (\neg\mathcal{O}'_p\top \wedge \mathcal{O}_p(\varphi \,\mathsf{U}_p\, \psi))$. Recall from Section 8.2 that U_p is PSPACE-effective since it can be expressed with SU. Hence, it remains to show that $\mathcal{O}'_p$ is PSPACE-effective. The proof is almost the same as the one of Proposition 8.3 for the modality $\mathcal{O}_p$. The only difference is in the definition of the transition relation for the second component. We replace the definition by:

$$q' = \begin{cases} 0 & \text{if } q = 0 \wedge (p \notin a \setminus A' \vee a \cap A' = \emptyset) \\ 1 & \text{if } q = 0 \wedge p \in a \setminus A' \wedge a \cap A' \neq \emptyset \wedge x = 1 \\ 2 & \text{if } q = 0 \wedge p \in a \setminus A' \wedge a \cap A' \neq \emptyset \wedge x = 0 \\ q & \text{if } q \neq 0. \end{cases}$$

8.4. The modality Eco

We can show that the modality Eco is PSPACE-effective using an idea similar to the one used for $\mathcal{O}_p$. Indeed, let $(w, X, \{y\}) \in \Sigma_2^\infty$ and let $z > 0$ be any position. Thanks to the non-deterministic automaton $\mathcal{A}$ from the proof of Proposition 8.3 we can check whether $z \preceq y$. It is also easy to construct a deterministic automaton $\mathcal{A}'$ which allows to check whether $y \preceq z$. It suffices to compute, after reading the prefix of length n of $(w, X, \{y\})$ the set $A'_n = \bigcup\{a_j \mid y \preceq j \leq n\}$. Using these two automata $\mathcal{A}$ and $\mathcal{A}'$ it is easy to check whether $([w], X, \{y\}) \in [\![\mathsf{Eco}]\!]_\Pi$. Thus, we get the automata $\mathcal{B}_{\mathsf{Eco},\Pi}$ and $\overline{\mathcal{B}}_{\mathsf{Eco},\Pi}$ and we can show as in the previous proofs that their special variance is in $\mathtt{poly}(|\Pi|)$.

8.5. Path modalities

In this section, we show that the remaining modalities from the temporal logic for causality TLC are PSPACE-effective. The proof is based on Proposition 7.4, in particular on the notion of general variance.

Since the modalities EU, ES and EG claim the existence of a path for the causal successor relation $\prec$, we need to know what are the positions that are covered by a new letter. Let $w = a_1 a_2 \cdots \in \Sigma^\infty$ and let i, n be positions in w. Then, $i \prec n$ iff for some process $p \in a_i \cap a_n$ we have $a_j \cap a_n = \emptyset$ for all $i \prec j < n$.

This motivates the definition of the following deterministic automaton $\mathcal{A}$. The set of states is $Q_1 = (2^\Pi \times 2^\Pi)^\Pi$ and the initial state is $\mathrm{init}_1 = (\emptyset, \emptyset)_{p\in\Pi}$. We first specify the expected behavior of $\mathcal{A}$. For each word $w = a_1 \dots a_n \in \Sigma^*$, there is a unique run $\mathrm{init}_1 \xrightarrow{w} (A^p_n, B^p_n)_{p\in\Pi}$ where for each process p, if $\{j \le n \mid p \in a_j\} = \emptyset$ then $(A^p_n, B^p_n) = (\emptyset, \emptyset)$ and otherwise, with $i = \max\{j \le n \mid p \in a_j\}$, we have

$$A^p_n = \bigcup\{a_j \mid i \preceq j \le n\} \qquad \text{and} \qquad B^p_n = \bigcup\{a_j \mid i \prec j \le n\}\ . \tag{6}$$

To achieve this goal, we define transitions $(A^p, B^p)_{p\in\Pi} \xrightarrow{a} (A'^p, B'^p)_{p\in\Pi}$ if for all $p \in \Pi$ we have

$$(A'^p, B'^p) = \begin{cases} (a, \emptyset) & \text{if } p \in a \\ (A^p, B^p) & \text{if } a \cap A^p = \emptyset \\ (A^p \cup a, B^p \cup a) & \text{otherwise.} \end{cases}$$

Note that the number of states of $\mathcal{A}$ is in $2^{\mathtt{poly}(|\Pi|)}$ and that we can compute the transition function of $\mathcal{A}$ in space $\mathtt{poly}(|\Pi|)$.

We show by induction that the specification of $\mathcal{A}$ is satisfied. Assume that $p \in a_n$. Then, by definition of the transition function, we have $A^p_n = a_n$ and $B^p_n = \emptyset$. Since in this case $n = \max\{i \le n \mid p \in a_i\}$ we deduce that (6) holds. Assume now that $p \notin a_n$. If $p \notin \bigcup\{a_j \mid j \le n-1\}$ then also $p \notin \bigcup\{a_j \mid j \le n\}$ and we get $(A^p_n, B^p_n) = (A^p_{n-1}, B^p_{n-1}) = (\emptyset, \emptyset)$ as desired. Otherwise, $i = \max\{j \le n-1 \mid p \in A_j\} = \max\{j \le n \mid p \in A_j\}$. If $a \cap A^p_{n-1} = \emptyset$ then using the inductive hypothesis, we deduce that $i \not\preceq n$. Therefore, (6) holds with $(A^p_n, B^p_n) = (A^p_{n-1}, B^p_{n-1})$. On the other hand, if $a \cap A^p_{n-1} \neq \emptyset$ then $i \prec n$ and we also obtain (6) with $(A^p_n, B^p_n) = (A^p_{n-1} \cup a_n, B^p_{n-1} \cup a_n)$.

As explained above, the automaton $\mathcal{A}$ is important since it allows us to know which positions are covered by a new letter, i.e., when a new letter a_n arrives, which are the positions $i < n$ such that $i \prec n$. This is the case iff there exists $p \in a_i \cap a_n$ such that $p \notin a_j$ for all $i < j < n$ and $a_n \cap B^p_{n-1} = \emptyset$. Note that we only use the sets B^p to check this property, while the sets A^p are used to define the transitions of the automaton $\mathcal{A}$.

Lemma 8.3. There is a PSPACE algorithm which, given a finite set of processes Π, computes a Büchi-automaton $\mathcal{B}$ that accepts a word $(w, X, Y, Z) \in \Sigma_3^\infty$ iff there exists a path $i_0 \prec \cdots \prec i_\ell$ in $[w]$ such that $\ell > 0$, $i_0 \in X$, $i_\ell \in Z$ and $i_1, i_2, \dots, i_{\ell-1} \in Y$.

Moreover, if init is the initial state of $\mathcal{B}$ then for each $w \in \Sigma^*$, we have

$$\left|\bigcup\{\mathrm{init} \cdot (w, X, Y, Z) \mid X, Y, Z \subseteq \{1, \dots, |w|\}\}\right| \le |\Pi| + 2.$$

Proof:
The automaton $\mathcal{B}$ has two components. The first one is $\mathcal{A}$ and the set of states of the second component

is $Q_2 = \{\mathrm{init}_2, \mathrm{OK}\} \uplus \Pi$. The initial state of $\mathcal{B}$ is $\mathrm{init} = (\mathrm{init}_1, \mathrm{init}_2)$ and the accepting states are $F = Q_1 \times \{\mathrm{OK}\}$. There is a transition $((A^p, B^p)_{p\in\Pi}, q) \xrightarrow{(a,x,y,z)} ((A'^p, B'^p)_{p\in\Pi}, q')$ if in $\mathcal{A}$ we have the transition $(A^p, B^p)_{p\in\Pi} \xrightarrow{a} (A'^p, B'^p)_{p\in\Pi}$ and one of the following holds

$$\begin{aligned}
& q = \mathrm{init}_2 \wedge q' = \mathrm{init}_2 \\
\text{or}\quad & q = \mathrm{init}_2 \wedge x = 1 \wedge q' \in a \\
\text{or}\quad & q \notin a \wedge q' = q \\
\text{or}\quad & q \in a \wedge a \cap B^q = \emptyset \wedge y = 1 \wedge q' \in a \\
\text{or}\quad & q \in a \wedge a \cap B^q = \emptyset \wedge z = 1 \wedge q' = \mathrm{OK} \\
\text{or}\quad & q = \mathrm{OK} \wedge q' = \mathrm{OK}\,.
\end{aligned}$$

Let $(w, X, Y, Z) = (a_1, x_1, y_1, z_1)(a_2, x_2, y_2, z_2)\cdots \in \Sigma_3^\infty$. Assume that we have a path $i_0 \prec \cdots \prec i_\ell$ in $[w]$ such that $\ell > 0$, $i_0 \in X$, $i_1, i_2, \ldots, i_{\ell-1} \in Y$, and $i_\ell \in Z$. Then $x_{i_0} = 1$, $y_{i_j} = 1$ for $0 < j < \ell$, and $z_{i_\ell} = 1$. Let $(A_n^p, B_n^p)_{p\in\Pi}$ be the state reached by $\mathcal{A}$ after reading the prefix of w of length n. For $0 \le j < \ell$, we have $i_j \prec i_{j+1}$. Hence we find $q_{i_j} \in a_{i_j} \cap a_{i_{j+1}}$ such that $q_{i_j} \notin a_n$ for all $i_j < n < i_{j+1}$ and $a_{i_{j+1}} \cap B^{q_{i_j}}_{i_{j+1}-1} = \emptyset$. Now, let $q_n = \mathrm{init}_2$ for $n < i_0$, $q_n = q_{i_j}$ for $i_j \le n < i_{j+1}$ with $0 \le j < \ell$, and $q_n = \mathrm{OK}$ for $n \ge i_\ell$. We can easily check that the sequence $((A_0^p, B_0^p)_{p\in\Pi}, q_0), ((A_1^p, B_1^p)_{p\in\Pi}, q_1), \ldots$ defines an accepting run of $\mathcal{B}$ on the word (w, X, Y, Z).

Conversely, assume that $(w, X, Y, Z) \in \mathcal{L}(\mathcal{B})$. Let $((A_0^p, B_0^p)_{p\in\Pi}, q_0), ((A_1^p, B_1^p)_{p\in\Pi}, q_1), \ldots$ be an accepting run of $\mathcal{B}$ on $(w, X, Y, Z) = (a_1, x_1, y_1, z_1)(a_2, x_2, y_2, z_2)\cdots \in \Sigma_3^\infty$. Since the run is accepting, $q_0 = \mathrm{init}_2$ and $q_n = \mathrm{OK}$ for all but finitely many n.

We construct inductively a sequence $i_0 \prec \cdots \prec i_\ell$ such that $i_0 \in X$ and $i_\ell \in Y \cup Z$ if $\ell > 0$ and $i_j \in Y$ for $0 < i < \ell$ and $q_{i_j} \in a_{i_j} \cap a_{i_{j+1}}$ for $0 \le j < \ell$. To start the induction, we let i_0 be minimal with $q_{i_0} \neq \mathrm{init}_2$. Then, from the transition relation, we deduce $i_0 \in X$ and $q_{i_0} \in a_{i_0}$. Now, assume we have already constructed a sequence $i = i_0 \prec \cdots \prec i_\ell$ with the above property. If $i_\ell \in Z$, then the construction stops. Otherwise, we claim that $q_{i_\ell} \in a_{i_\ell}$. This is clearly the case if $\ell = 0$. So assume that $\ell > 0$. For $0 \le j < \ell$, we have $q_{i_j} \in a_{i_j} \cap a_{i_{j+1}}$ and $i_j \prec i_{j+1}$. Hence, for $i_j < n < i_{j+1}$, we have $q_{i_j} \notin a_n$ and $q_n = q_{i_j}$. In particular, $q_{i_\ell - 1} = q_{i_{\ell-1}} \in a_{i_\ell}$ and since $i_\ell \notin Z$ we have $q_{i_\ell} \neq \mathrm{OK}$ and $q_{i_\ell} \in a_{i_\ell}$ by definition of the transitions, which concludes the proof of our claim. Now, let $i_{\ell+1} = \min\{n > i_\ell \mid q_{i_\ell} \in a_n\}$ (this is well-defined since otherwise the run would stay forever in state $q_{i_\ell} \neq \mathrm{OK}$ and would not be successful). Thus, $q_{i_\ell} \in a_{i_\ell} \cap a_{i_{\ell+1}}$ and $q_{i_\ell} \notin a_n$ for $i_\ell < n < i_{\ell+1}$. The definition of the transitions also implies that $i_{\ell+1} \cap B^{q_{i_\ell}}_{i_{\ell+1}-1} = \emptyset$ and $i_{\ell+1} \in Y \cup Z$. We deduce that $i_\ell \prec i_{\ell+1}$ and we have extended the sequence. Finally, the run being successful, we eventually reach a state $q_n = \mathrm{OK}$ and the sequence cannot be extended forever. Therefore, we eventually get $i_\ell \in Z$ which implies the existence of a path as required.

The last property of $\mathcal{B}$ is trivial to check. Indeed, the first component of $\mathcal{B}$, i.e., the deterministic automaton $\mathcal{A}$, only depends on the word $w \in \Sigma^\infty$ and not on the sets X, Y, Z. The second component can take at most $|\Pi| + 2$ values.

Finally, given Π, the automaton $\mathcal{B}$ can be constructed in PSPACE. □

Proposition 8.4. The modalities EU and ES are PSPACE-effective.

Proof:
Let $(w, X, Y, \{z\}) \in \Sigma_3^\infty$ and let $\mathcal{B}$ be the automaton from Lemma 8.3. Then

- $([w], X, Y, \{z\}) \in [\![\mathsf{EU}]\!]_\Pi$ iff $z \in Y$ or the word $(w, \{z\}, X, Y) \in \mathcal{L}(\mathcal{B})$.
- $([w], X, Y, \{z\}) \in [\![\mathsf{ES}]\!]_\Pi$ iff $z \in Y$ or the word $(w, Y, X, \{z\}) \in \mathcal{L}(\mathcal{B})$.

The necessary changes to $\mathcal{B}$ can be done in polynomial space and the general variances of the resulting automata are in $\mathtt{poly}(|\Pi|)$. Hence the result follows from Proposition 7.4. □

Proposition 8.5. The modality EG is PSPACE-effective.

Proof:
First, note that $\mathsf{EG}\,\varphi = \mathsf{EG}(\varphi \wedge \mathsf{EX}\,\top) \vee (\varphi\,\mathsf{EU}(\varphi \wedge \neg\,\mathsf{EX}\,\top))$. The first conjunct claims the existence of an infinite $\prec$-path satisfying φ while the second conjunct claims the existence of a finite and *maximal* $\prec$-path satisfying φ. We have already seen that EU and EX are PSPACE-effective hence it remains to deal with $\mathsf{EG}(\varphi \wedge \mathsf{EX}\,\top)$.

The construction parallels that from the proof of Lemma 8.3. The main difference is that the acceptance conditions is now some flag-construction checking that the path is indeed infinite. The new automaton $\mathcal{B}$ has two components. The first one is $\mathcal{A}$ and the set of states of the second component is $Q_2 = (\{\mathrm{init}_2\} \uplus \Pi) \times \{0, 1\}$. The initial state of $\mathcal{B}$ is $\mathrm{init} = (\mathrm{init}_1, \mathrm{init}_2, 0)$ and the accepting states are $F = Q_1 \times \Pi \times \{1\}$.

There is in $\mathcal{B}$ a transition $((A^p, B^p)_{p\in\Pi}, q, \varepsilon) \xrightarrow{(a,x,y)} ((A'^p, B'^p)_{p\in\Pi}, q', \varepsilon')$ if in $\mathcal{A}$ we have the transition $(A^p, B^p)_{p\in\Pi} \xrightarrow{a} (A'^p, B'^p)_{p\in\Pi}$ and one of the following hold

$$q = \mathrm{init}_2 \wedge y = 0 \wedge q' = \mathrm{init}_2 \wedge \varepsilon' = 0 \tag{7}$$
$$\text{or} \quad q = \mathrm{init}_2 \wedge x = 1 \wedge y = 1 \wedge q' \in a \wedge \varepsilon' = 1 \tag{8}$$
$$\text{or} \quad q \notin a \wedge y = 0 \wedge q' = q \wedge \varepsilon' = 0 \tag{9}$$
$$\text{or} \quad q \in a \wedge a \cap B^q = \emptyset \wedge x = 1 \wedge y = 0 \wedge q' \in a \wedge \varepsilon' = 1\,. \tag{10}$$

Let $(w, X, Y) \in \mathcal{L}(\mathcal{B})$. Then, $Y = \{i_0\}$ is a singleton. Let $i_1, i_2, \ldots$ be the positions where a transition of the form (10) is taken. As in the proof of Lemma 8.3 we can show that $i_0 \prec i_1 \prec i_2 \ldots$ and that $i_j \in X$ for all these positions. Now, the run being successful, infinitely many transitions of type (10) are taken and the path is infinite.

Conversely, let $(w, X, \{i_0\}) \in \Sigma_2^\omega$ be such that there exists an infinite path $i_0 \prec i_1 \prec i_2 \ldots$ with $i_j \in X$ for all $j \geq 0$. As in the proof of Lemma 8.3 we can build an accepting path in $\mathcal{B}$ for $(w, X, \{i_0\})$ where transition (8) is taken at position i_0 and transitions (10) are taken at the positions $i_1, i_2, \ldots$

Finally, the general variance of $\mathcal{B}$ is at most $2(|\Pi| + 1)$. Hence, we deduce from Proposition 7.4 that the modality $\mathsf{EG}(\varphi \wedge \mathsf{EX}\,\top)$ is PSPACE-effective. □

References

[1] R. Alur, R. Peled, and W. Penczek. Model checking of causality properties. In *LICS 95*, pages 90–100. IEEE Computer Society Press, 1995.

[2] V. Diekert and P. Gastin. LTL is expressively complete for Mazurkiewicz traces. *Journal of Computer and System Sciences*, 64:396–418, 2002. A preliminary version appeared at ICALP'00, Lecture Notes in Comp. Science vol. 1853, pages 211-222, Springer.

[3] V. Diekert and P. Gastin. Local temporal logic is expressively complete for cograph dependence alphabets. *Information and Computation*, 195:30–52, 2004. A preliminary version appeared at LPAR'01, Lecture Notes in Artificial Intelligence vol. 2250, pages 55–69, Springer.

[4] V. Diekert and P. Gastin. Pure future local temporal logics are expressively complete for Mazurkiewicz traces. *Information and Computation*, 204:1597–1619, 2006. A preliminary version appeared at LATIN'04, Lecture Notes in Comp. Science vol. 2976, pages 232–241, Springer.

[5] V. Diekert and G. Rozenberg. *The Book of Traces*. World Scientific Publ. Co., 1995.

[6] P. Gastin and D. Kuske. Satisfiability and model checking for MSO-definable temporal logics are in PSPACE. In R. Amadio and D. Lugiez, editors, *CONCUR'03*, Lecture Notes in Comp. Science vol. 2761, pages 222–236. Springer, 2003.

[7] P. Gastin and D. Kuske. Uniform satisfiability problem for local temporal logics over Mazurkiewicz traces. In L. de Alfaro, editor, *CONCUR'05*, Lecture Notes in Comp. Science vol. 3653, pages 533–547. Springer, 2005.

[8] P. Gastin and M. Mukund. An elementary expressively complete temporal logic for Mazurkiewicz traces. In *ICALP'02*, Lecture Notes in Comp. Science vol. 2380, pages 938–949. Springer, 2002.

[9] S. Katz and D. Peled. Interleaving set temporal logic. *Theoretical Comp. Science*, 75:21–43, 1991.

[10] O. Kupferman and M. Vardi. Weak alternating automata are not that weak. *ACM Transaction on Computational Logic*, 2(3):408–429, 2001.

[11] A. Mazurkiewicz. Concurrent program schemes and their interpretation. Technical report, DAIMI Report PB-78, Aarhus University, 1977.

[12] A. Mazurkiewicz. Trace theory. In W. Brauer and others, editor, *Petri Nets, Applications and Relationship to other Models of Concurrency*, Lecture Notes in Comp. Science vol. 255, pages 279–324. Springer, 1987.

[13] S. Miyano and T. Hayashi. Alternating finite automata on ω-words. *Theoretical Comp. Science*, 32:321–330, 1984.

[14] W. Penczek. A concurrent branching time temporal logic. In E. Börger, H. Kleine Büning, and M.M. Richter, editors, *3rd Workshop on Computer Science Logic*, Lecture Notes in Comp. Science vol. 440, pages 337–354. Springer, 1990.

[15] W. Penczek. On undecidability of temporal logics on trace systems. *Information Processing Letters*, 43:147–153, 1992.

[16] D. Perrin and J.-E. Pin. *Infinite Words*. Pure and Applied Mathematics vol. 141. Elsevier, 2004.

[17] S. Safra. On the complexity of omega-automata. In *FOCS'88*, pages 319–327. IEEE Computer Society Press, 1988.

[18] P.S. Thiagarajan. A trace based extension of linear time temporal logic. In *LICS'94*, pages 438–447. IEEE Computer Society Press, 1994.

[19] P.S. Thiagarajan and I. Walukiewicz. An expressively complete linear time temporal logic for Mazurkiewicz traces. *Information and Computation*, 179:230–249, 2002. A preliminary version appeared at LICS'97, pages 183–194, IEEE Computer Society Press.

[20] I. Walukiewicz. Difficult configurations – on the complexity of LTrL. In *ICALP'98*, Lecture Notes in Comp. Science vol. 1443, pages 140–151. Springer, 1998.