





# Temporal Hyperproperties for Population Protocols

Nicolas Waldburger<sup>1</sup>, Chana Weil-Kennedy<sup>2</sup>,  
Pierre Ganty<sup>2</sup>, and César Sánchez<sup>2</sup>

<sup>1</sup> Université de Rennes, IRISA, INRIA, France  
`nicolas.waldburger@irisa.fr`

<sup>2</sup> IMDEA Software Institute, Pozuelo de Alarcón, Spain  
`{chana.weilkennedy,pierre.ganty,cesar.sanchez}@imdea.org`

**Abstract.** Hyperproperties are properties over sets of traces (or runs) of a system, as opposed to properties of just one trace. They were introduced in 2010 and have been much studied since, in particular via an extension of the temporal logic LTL called HyperLTL. Most verification efforts for HyperLTL are restricted to finite-state systems, usually defined as Kripke structures. In this paper we study hyperproperties for an important class of infinite-state systems. We consider population protocols, a popular distributed computing model in which arbitrarily many identical finite-state agents interact in pairs. Population protocols are a good candidate for studying hyperproperties because the main decidable verification problem, well-specification, is a hyperproperty. We first show that even for simple (monadic) formulas, HyperLTL verification for population protocols is undecidable. We then turn our attention to immediate observation population protocols, a simpler and well-studied subclass of population protocols. We show that verification of monadic HyperLTL formulas without the next operator is decidable in 2-EXPSPACE, but that all extensions make the problem undecidable.

## 1 Introduction

Hyperproperties are properties that allow to relate multiple traces (also called runs) of a system simultaneously [12]. They generalize regular run properties to properties of sets of runs, and formalize a wide range of important properties such as information-flow security policies like noninterference [30,36] and observational determinism [48], consistency models in concurrent computing [10], and robustness models in cyber-physical systems [47,9].

HyperLTL [11] was introduced as an extension of LTL (linear temporal logic) with quantification over runs which can then be related across time. HyperLTL enjoys a decidable model-checking problem for finite-state systems, expressed as Kripke structures. Other logics for hyperproperties were later introduced, like HyperCTL\* [27], HyperQPTL [40,13], and HyperPDL- $\Delta$  [31] which extend CTL\*, QPTL [44], and PDL [28] respectively. These logics also enjoy decidable model-checking problems for finite-state systems.

Most algorithmic verification results for verifying hyperproperties of temporal logics are restricted to finite-state systems. In the case of software verification, which is inherently infinite-state, the analysis of hyperproperties [7,26,43,45,46] has been limited to the class of  $k$ -safety properties — which only allow to establish the absence of a bad interaction between any  $k$  runs — and do not extend to a temporal logic for hyperproperties. A notable exception is [7], but the logic used (OHyperLTL) is a simple asynchronous logic for hyperproperties and it requires restrictions on the underlying theories of the data used in the program.

In this paper we focus on the verification of HyperLTL for an important class of infinite-state systems. We consider population protocols (PP) [2], an extensively studied (see e.g. [1,18,19]) model of distributed computation in which anonymous finite-state agents interact pairwise to change their states, following a common protocol. In a *well-specified* PP, the agents compute a predicate: the input is the initial configuration of the agents' states, and the agents interact in pairs to eventually reach a consensus opinion corresponding to the evaluation of the predicate (for *any* number of agents). Interactions are selected at random, which is modelled by considering only *fair* runs. LTL verification has been investigated for PPs in [20]. The authors consider LTL over actions, where formulas are evaluated over fair runs. They show that it is decidable, given a PP and an LTL formula, to check if all fair runs from initial configurations of the protocol verify the formula. Another related work on LTL verification for infinite-state systems is [29], where the authors consider stuttering-invariant LTL verification over shared-memory pushdown systems.

We consider PPs because, though they are infinite-state, they enjoy several decidable problems. In particular, the central verification problem checking whether a protocol is well-specified is decidable [21] and has a hyperproperty “flavor”. A PP is well-specified if for every initial configuration  $\gamma_0$ , every fair run starting in  $\gamma_0$  stabilizes to the same opinion. A run stabilizes to an opinion  $b \in \{0,1\}$  if from some position onwards it visits no configuration with an agent whose opinion differs from  $b$ . With  $\mathcal{I}$  the set of initial configurations and  $\text{FRuns}(\gamma)$  the set of fair runs starting in  $\gamma$ , well-specification can be expressed as:

$$\forall \gamma_0 \in \mathcal{I}, \text{FRuns}(\gamma_0) \models \forall \rho_1. \forall \rho_2. \bigvee_{b \in \{0,1\}} (\text{FG}(\rho_1 \text{ sees } b) \wedge \text{FG}(\rho_2 \text{ sees } b))$$

where “ $\rho$  sees  $b$ ” means that the run takes a transition that puts agents into states with opinion  $b$ . Then  $\text{FG}(\rho_i \text{ sees } b)$  ensures that  $\rho_i$  converges to  $b$ . The formula above is a proper relational hyperproperty and cannot be expressed in LTL because it requires to quantify over two traces  $\rho_1$  and  $\rho_2$  or quantify within the scope of an outer conjunction.

We show that for the general PP model, HyperLTL verification is already undecidable for simple (*monadic*) formulas which can be decomposed into formulas referring to only one run each (Section 3). We turn our attention to *immediate observation population protocols* (IOPP), a subclass of PP [3]. We show that HyperLTL verification over IOPP is a problem decidable in 2-EXPSpace when the formula is monadic and does not use the temporal operator  $\mathbf{X}$  (the formula

is then *stuttering-invariant*). This result delineates the decidability frontier for verification in PP: non-monadic or non-stuttering-invariant HyperLTL verification over IOPP is undecidable (Section 4). The decidability result for HyperLTL verification of IOPP is the most technical result of the paper. In particular, the technical results of Section 5 reason on the flow of agents in runs of an IOPP in conjunction with reading the transitions in a Rabin automaton. Note that if one fixed the initial configuration (focusing on non-global model checking) the verification of monadic HyperLTL $\setminus X$  can be performed by a Boolean combination of LTL verification problems. However, monadic HyperLTL $\setminus X$  is strictly more expressive than LTL even for the non-global case (including properties like termination-sensitive non-interference [41]).

## 2 Preliminaries

A *finite multiset* over a finite set  $S$  is a mapping  $\mu: S \rightarrow \mathbb{N}$  such that for each  $s \in S$ ,  $\mu(s)$  denotes the number of occurrences of element  $s$  in  $\mu$ . Given a set  $S$ ,  $\mathcal{M}(S)$  denotes the set of finite multisets over  $S$ . Given  $s \in S$ , we denote by  $\vec{s}$  the multiset  $\mu$  such that  $\mu(s) = 1$  and  $\mu(s') = 0$  for all  $s' \neq s$ . Given  $\mu, \mu' \in \mathcal{M}(S)$ , the multiset  $\mu + \mu'$  is defined by  $(\mu + \mu')(s) = \mu(s) + \mu'(s)$  for all  $s \in S$ . We let  $\mu \leq \mu'$  when  $\mu(s) \leq \mu'(s)$  for all  $s \in S$ . When  $\mu' \leq \mu$ , we let  $\mu - \mu'$  be the multiset such that  $(\mu - \mu')(s) = \mu(s) - \mu'(s)$  for all  $s \in S$ . We call  $|\mu| = \sum_{s \in S} \mu(s)$  the *size* of  $\mu$ . A set  $\mathcal{S} \subseteq \mathcal{M}(S)$  is *Presburger* if it can be written as a formula in Presburger arithmetic, *i.e.*, in  $FO(\mathbb{N}, +)$ .

A *strongly connected component* (SCC) in a graph is a non-empty maximal set of mutually reachable vertices. A SCC is *bottom* if no path leaves it.

### 2.1 Population Protocols

A *population protocol* (PP) is a tuple  $\mathcal{P} = (Q, \Delta, I)$  where  $Q$  is a finite set of *states*,  $\Delta \subseteq Q^2 \times Q^2$  is a set of *transitions* and  $I \subseteq Q$  is the set of *initial states*. A transition  $t = ((q_1, q_2), (q_3, q_4)) \in \Delta$  is denoted  $(q_1, q_2) \xrightarrow{t} (q_3, q_4)$ . We let  $|\mathcal{P}| := |Q| + |\Delta|$  denote the *size* of  $\mathcal{P}$ . A *configuration* of  $\mathcal{P}$  is a multiset over  $Q$ . We denote by  $\Gamma := \{\mu \in \mathcal{M}(Q) \mid |\mu| \geq 2\}$  the set of configurations; configurations must have at least 2 agents. We note  $\mathcal{I} := \{\gamma \in \Gamma \mid \forall q \notin I, \gamma(q) = 0\}$  the set of *initial configurations*. Given  $\gamma, \gamma' \in \Gamma$  and  $(q_1, q_2) \xrightarrow{t} (q_3, q_4) \in \Delta$ , there is a *step*  $\gamma \xrightarrow{t} \gamma'$  if  $\gamma \geq \vec{q_1} + \vec{q_2}$  and  $\gamma' = \gamma - \vec{q_1} - \vec{q_2} + \vec{q_3} + \vec{q_4}$ . A transition  $(q_1, q_2) \rightarrow (q_3, q_4)$  is *activated* at  $\gamma$  if  $\gamma \geq \vec{q_1} + \vec{q_2}$ , *i.e.*, if there is an agent in  $q_1$  and an agent in  $q_2$  (or two agents in  $q_1$  if  $q_1 = q_2$ ). Henceforth, we assume that for every  $q_1, q_2 \in Q$ , there exist  $q_3, q_4 \in Q$  such that  $(q_1, q_2) \rightarrow (q_3, q_4) \in \Delta$ , so that there is always an activated transition. This can be done by adding self-loops  $(q_1, q_2) \rightarrow (q_1, q_2)$ .

A *finite run* is a sequence  $\gamma_0, t_0, \gamma_1, \dots, t_{k-1}, \gamma_k$  where  $\gamma_i \xrightarrow{t_i} \gamma_{i+1}$  for all  $i \leq k-1$ ; we say  $t_i$  is *fired* at  $\gamma_i$ . We write  $\gamma \xrightarrow{*} \gamma'$  if there exists a finite run from  $\gamma$  to  $\gamma'$ , and we say  $\gamma'$  is *reachable* from  $\gamma$ . Given  $\mathcal{S} \subseteq \Gamma$ , let  $\text{post}^*(\mathcal{S})$  be the

set of configurations reachable from  $\mathcal{S}$ , *i.e.*,  $\text{post}^*(\mathcal{S}) := \{\gamma \mid \exists \gamma' \in \mathcal{S}. \gamma' \xrightarrow{*} \gamma\}$ . Similarly, let  $\text{pre}^*(\mathcal{S}) := \{\gamma \mid \exists \gamma' \in \mathcal{S}. \gamma \xrightarrow{*} \gamma'\}$ .

An *infinite run* is an infinite sequence  $\rho = \gamma_0, t_0, \gamma_1, t_1, \dots$  with  $\gamma_i \xrightarrow{t_i} \gamma_{i+1}$  for all  $i \in \mathbb{N}$ . A configuration  $\gamma$  is *visited* in  $\rho$  when there is  $i$  such that  $\gamma_i = \gamma$ ; it is *visited infinitely often* when there are infinitely many such  $i$ . Similarly,  $t \in \Delta$  is *fired infinitely often* in  $\rho$  where there are infinitely many  $i$  such that  $t_i = t$ . A finite run  $\gamma'_0, t'_0, \gamma'_1, \dots, t'_{k-1}, \gamma'_k$  *appears infinitely often* in  $\rho$  when there are infinitely many  $i$  such that  $\gamma_{i+j} = \gamma'_j$  for all  $j \in [0, k]$  and  $t_{i+j} = t'_j$  for all  $j \in [0, k-1]$ . Also,  $\rho$  is *strongly fair* when, for every finite run  $\rho'$ , by letting  $\gamma'_0$  the first configuration in  $\rho'$ , if  $\gamma'_0$  is visited infinitely often in  $\rho$  then  $\rho'$  appears infinitely often in  $\rho$ . Given a configuration  $\gamma_0$ , the set of strongly fair runs from  $\gamma_0$  is denoted  $\text{FRuns}(\gamma_0)$ . Note that this notion of fairness differs from the one usually used for PPs. We will discuss this choice in Section 2.4.

## 2.2 LTL and HyperLTL

Linear temporal logic [39] (LTL) extends propositional logic with modalities to relate different positions in a run, allowing to define temporal properties of systems. HyperLTL [11] is an extension of LTL for hyperproperties, with explicit quantification over runs. We here define LTL and HyperLTL for population protocols. Let  $\mathcal{P} = (Q, \Delta, I)$  be a PP. Our atomic propositions are the transitions of the run(s); we discuss this choice at the end of this section.

*LTL.* The syntax of LTL over  $\mathcal{P}$  is:

$$\varphi ::= t \mid \varphi \vee \varphi \mid \neg \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \quad \text{where } t \in \Delta.$$

The operators  $\mathbf{X}$  (next) and  $\mathbf{U}$  (until) are the temporal modalities. We use the usual additional operators:  $\text{true} = t \vee \neg t$ ,  $\text{false} = \neg \text{true}$ ,  $\varphi \wedge \varphi = \neg(\neg \varphi \vee \neg \varphi)$ ,  $\mathbf{F}\varphi = \text{true} \mathbf{U} \varphi$  and  $\mathbf{G}\varphi = \neg \mathbf{F} \neg \varphi$ . The *size*  $|\varphi|$  of an LTL formula  $\varphi$  is the number of (temporal and Boolean) operators of  $\varphi$ . The semantics of LTL is defined over runs in the usual way (*e.g.*, [4]) over  $\Delta^\omega$ . An infinite run  $\rho = \gamma_0, t_0, \gamma_1, t_1, \dots$  *satisfies* an LTL formula  $\varphi$ , denoted  $\rho \models \varphi$ , when  $w \models \varphi$  where  $w = t_0 t_1 t_2 \dots \in \Delta^\omega$ . A configuration  $\gamma$  satisfies an LTL formula  $\varphi$ , denoted  $\gamma \models \varphi$ , when  $\rho \models \varphi$  for all  $\rho \in \text{FRuns}(\gamma)$ , *i.e.*, when *all* strongly fair runs starting from  $\gamma$  satisfy  $\varphi$ .

*HyperLTL.* The syntax of HyperLTL over  $\mathcal{P}$  is:

$$\psi ::= \exists \rho. \psi \mid \forall \rho. \psi \mid \varphi \quad \varphi ::= t_\rho \mid \varphi \vee \varphi \mid \neg \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi$$

where  $t \in \Delta$  and  $\rho$  is a *run variable*. Note that  $\varphi$  is an LTL formula with, as atomic propositions, the transitions of the run variables. A HyperLTL formula  $\psi$  must additionally be well-formed: all appearing variables are quantified and no variable is quantified twice. The size  $|\psi|$  of an HyperLTL formula  $\psi$  is the number of (temporal and Boolean) operators and quantifiers of  $\psi$ . HyperLTL formulas are interpreted over strongly fair runs starting from a configuration as follows:

a configuration  $\gamma$  satisfies a HyperLTL formula  $\psi$ , denoted  $\gamma \models \psi$ , whenever  $\text{FRuns}(\gamma) \models \psi$ . See the appendix for a formal definition of the semantics. Notice that, given a configuration  $\gamma$  and an LTL formula  $\varphi$ ,  $\gamma \models \varphi$  if and only if  $\gamma \models \forall \rho. \varphi_\rho$  where  $\varphi_\rho$  is equal to  $\varphi$  where  $t$  is replaced by  $t_\rho$  for all  $t \in \Delta$ .

*Example 1.* Suppose that  $\Delta = \{s, t\}$ . Let  $\psi := \forall \rho_1. \exists \rho_2. \text{FG}((s_{\rho_1} \wedge t_{\rho_2}) \vee (t_{\rho_1} \wedge s_{\rho_2}))$ . Given  $\gamma \in \Gamma$ , we have that  $\gamma \models \psi$  if and only if, for every strongly fair run  $\rho_1 \in \text{FRuns}(\gamma)$  from  $\gamma$ , there is a strongly fair run  $\rho_2 \in \text{FRuns}(\gamma)$  from  $\gamma$  such that, always after some point,  $\rho_1$  fires  $s$  whenever  $\rho_2$  fires  $t$  and vice versa.

A HyperLTL formula  $\psi : Q_1 \rho_1 \dots Q_k \rho_k. \varphi$  is *monadic* if  $\varphi$  has a *decomposition* as a Boolean combination of temporal formulas  $\varphi_1, \dots, \varphi_n$ , each of which refer to exactly one run variable. We assume that a monadic formula is always given by its decomposition, *i.e.*, by giving  $\varphi_1$  to  $\varphi_n$  and the Boolean combination.

*Verification Problems.* Given a PP  $\mathcal{P} = (Q, \Delta, I)$  and an LTL formula  $\varphi$  (resp. a HyperLTL formula  $\psi$ ), we denote  $\mathcal{P} \models^\forall \varphi$  when  $\gamma_0 \models \varphi$  (resp.  $\gamma_0 \models \psi$ ) for all  $\gamma_0 \in \mathcal{I}$ . Dually, we let  $\mathcal{P} \models^\exists \varphi$  (resp.  $\mathcal{P} \models^\exists \psi$ ) when there is  $\gamma_0 \in \mathcal{I}$  such that  $\gamma_0 \models \varphi$  (resp.  $\gamma_0 \models \psi$ ).

The *LTL verification problem for population protocols* consists on determining, given  $\mathcal{P}$  and an LTL formula  $\varphi$ , whether  $\mathcal{P} \models^\forall \varphi$ , *i.e.*, whether all strongly fair runs from all initial configurations satisfy  $\varphi$ . We also consider a variant problem, the *existential LTL verification problem*, that asks whether  $\mathcal{P} \models^\exists \varphi$ , *i.e.*, whether there is an initial configuration from which all strongly fair runs satisfy  $\varphi$ . Given a HyperLTL formula  $\psi$ , the *HyperLTL verification problem for population protocols* consists on determining whether  $\mathcal{P} \models^\forall \psi$ ; again, the existential variant consists in asking whether  $\mathcal{P} \models^\exists \psi$ .

*Example 2.* A PP  $\mathcal{P} = (Q, \Delta, I)$  equipped with an *opinion* function  $O : Q \rightarrow \{0, 1\}$  is *well-specified* if for every  $\gamma_0 \in \mathcal{I}$ , every run in  $\text{FRuns}(\gamma_0)$  eventually visits only configurations where either all agents are in states  $O^{-1}(0)$  or all agents are in states  $O^{-1}(1)$ . Let  $\Delta_b$  be the set of transitions  $(q_1, q_2) \rightarrow (q_3, q_4) \in \Delta$  such that  $O(q_3) = O(q_4) = b$ . Well-specification of  $\mathcal{P} = (Q, \Delta, I)$  with opinion function  $O$  corresponds to the HyperLTL verification problem over a monadic formula:

$$\mathcal{P} \models^\forall \forall \rho_1, \rho_2. \bigvee_{b \in \{0,1\}} \text{FG}(\bigvee_{t \in \Delta_b} t_{\rho_1}) \wedge \text{FG}(\bigvee_{t \in \Delta_b} t_{\rho_2}) .$$

*LTL over transitions and LTL over states.* Our LTL formulas are *over transitions*, *i.e.*, their atomic propositions are the transitions of the run. In [20, Theorems 9 and 10], the LTL verification problem defined above is proven to be decidable, although as hard as reachability for Petri nets and therefore Ackermann-complete [35,14]. The authors of [20] also show that *LTL over states*, where the atomic predicates indicate whether or not a state is currently visited by an agent, is undecidable. A slight difference between their model and ours is that their initial configurations are given by a Presburger set; however, their undecidability proof, which relies on 2-counter machines, can easily be translated to our setting. In the rest of the paper we consider only (Hyper)LTL over transitions.

**Proposition 3.** *The LTL over states verification problem for PP is undecidable.*

### 2.3 Rabin Automata and LTL

Let  $\Sigma$  be a finite set. The set of finite words (resp. infinite words) over  $\Sigma$  is denoted  $\Sigma^*$  (resp.  $\Sigma^\omega$ ). A *deterministic Rabin automaton* over  $\Sigma$  is a tuple  $\mathcal{A} = (\mathcal{L}, T, \ell_0, \mathcal{W})$ , where  $\mathcal{L}$  is a finite set of states,  $\ell_0 \in \mathcal{L}$  is the initial state,  $T : \mathcal{L} \times \Sigma \rightarrow \mathcal{L}$  is the transition function and  $\mathcal{W} \subseteq 2^{\mathcal{L}} \times 2^{\mathcal{L}}$  is a finite set of *Rabin pairs*. An infinite word  $w \in \Sigma^\omega$  is *accepted* if there exists  $(F, G) \in \mathcal{W}$  such that the run of  $\mathcal{A}$  reading  $w$  visits  $F$  finitely often and  $G$  infinitely often.

**Theorem 4 ([23]).** *Given  $\Sigma$  a finite set and  $\varphi$  an LTL formula over  $\Sigma$ , one can compute, in time doubly-exponential in  $|\varphi|$ , a deterministic Rabin automaton  $\mathcal{A}_\varphi$  over  $\Sigma$ , of doubly-exponential size, that recognizes (the language of)  $\varphi$ .*

### 2.4 Why Strong Fairness?

Usually, fairness in population protocols is either of the form “all configurations reachable infinitely often are reached infinitely often” [3,21], or “all steps possible infinitely often are taken infinitely often” [20]. Our notion of fairness, dubbed strong fairness, is more restrictive. A sanity check is that a (reasonable) stochastic scheduler yields a strongly fair run with probability 1. This alone does not justify using a new notion of fairness different from the literature and in particular from the prior work on LTL verification [20]. The authors motivate their choice of fairness by claiming that there is a fair run satisfying an LTL formula  $\varphi$  if and only if, under a stochastic scheduler,  $\varphi$  is satisfied with non-zero probability [20, Proposition 7]. However, we show that this claim is incorrect.

The intuition is that a (not strongly) fair run may exhibit infinite regular patterns. Consider three configurations  $\gamma_1, \gamma_2, \gamma_3$  and three transitions  $a, b, c$  such that  $\gamma_1 \xrightarrow{a} \gamma_2$ ,  $\gamma_2 \xrightarrow{b} \gamma_1$ ,  $\gamma_1 \xrightarrow{c} \gamma_3$  and  $\gamma_3 \xrightarrow{d} \gamma_1$ , and these are the only steps possible from each of the configurations. Consider  $\varphi = \neg F(a \wedge (Xb) \wedge (X^2a) \wedge (X^3b))$ , which expresses that the sequence of transitions  $abab$  does not appear. Under a stochastic scheduler,  $\varphi$  is satisfied with probability 0 from  $\gamma_1$ . However, the run which repeats sequence  $abcd$  satisfies  $\varphi$ , and it is fair. This proves that [20, Proposition 7] does not hold; we explain the mistake in detail in the appendix.

The run from this counterexample is not strongly fair. We show that strong fairness does in fact allow the desired equivalence with stochastic schedulers. As in [20], fix a stochastic scheduler, assumed to be memoryless and guaranteeing non-zero probability for every activated transition;  $\Pr[\gamma \models \varphi]$  denotes the probability that a run from  $\gamma$  satisfies  $\varphi$ .

**Proposition 5.** *Given an LTL formula  $\varphi$  and  $\gamma_0 \in \Gamma$ ,  $\Pr[\gamma_0 \models \varphi] = 1$  if and only if, for all  $\rho \in \text{FRuns}(\gamma_0)$ ,  $\rho \models \varphi$ .*

*Proof (sketch).* Let  $\mathcal{A}_\varphi = (\mathcal{L}, T, \ell_0, \mathcal{W})$  be a Rabin automaton that recognizes  $\varphi$  (see Theorem 4). Like in the proof of [20, Proposition 7], we consider a Petri net obtained by combining  $\mathcal{P}$  and  $\mathcal{A}_\varphi$ . We reason on the bottom SCCs of the configuration graph of this Petri net. An SCC is *winning* where there is  $(F, G) \in \mathcal{W}$  such that  $S$  has some configuration with Rabin state in  $G$  but none with Rabin

state in  $F$ . By [20, Proposition 6], we have  $\Pr[\gamma_0 \models \varphi] = 1$  iff all bottom SCC  $S$  reachable from  $(\gamma_0, \ell_0)$  are winning. We show that strong fairness guarantees that a run (1) always reaches a bottom SCC and (2) visits all the configurations of this bottom SCC infinitely often. The proofs of the two statements are similar; we explain here the proof of (2). Configurations of the Petri net are of the form  $(\gamma, \ell)$  with  $\gamma$  a configuration of  $\mathcal{P}$  and  $\ell \in \mathcal{L}$ . Consider  $(\gamma, \ell)$  in a bottom SCC and  $t \in \Delta$  activated from  $\gamma$ . Since strong fairness is only related to  $\mathcal{P}$ , it does not guarantee that  $t$  is eventually fired from  $(\gamma, \ell)$ . We circumvent this difficulty by constructing a sequence of transitions  $\sigma$  that, when fired from any  $(\gamma, \ell')$  in the SCC, makes us go through  $(\gamma, \ell)$  and fire  $t$ . Strong fairness ensures  $\sigma$  is fired from some  $(\gamma, \ell')$ , which proves the result.  $\square$

This therefore justifies our choice to consider strong fairness for LTL verification. In particular, all results from [20] hold if strong fairness is considered instead of the usual fairness. An alternative to strong fairness for (non-Hyper)LTL verification would be to work directly with a stochastic scheduler. However, HyperLTL requires quantification over a subset of the set of runs; we make the choice to consider, for this subset, the set of strongly fair runs.

### 3 Undecidability of HyperLTL

One can show that verification of HyperLTL over transitions is undecidable for PP, using a proof with counter machines similar to the one for undecidability of LTL over states [20]. Intuitively, HyperLTL can be used to express whether a transition is activated at some point in the run, and hence encode zero-tests<sup>3</sup>. We show an even stronger undecidability result: verification of monadic HyperLTL formulas over two runs using only FG as temporal operator is undecidable.

**Theorem 6.** *Verification of monadic HyperLTL for PP is undecidable. If fact, it is already undecidable for formulas of the form:*

$$\forall \rho_1. \exists \rho_2. \neg(\text{GF } a_{\rho_1}) \vee (\text{GF } b_{\rho_2}) \quad \text{where } a, b \in \Delta .$$

This verification problem asks whether, for all  $\gamma_0 \in \mathcal{I}$ , for all  $\rho_1 \in \text{FRuns}(\gamma_0)$ , there is  $\rho_2 \in \text{FRuns}(\gamma_0)$  such that if  $\rho_1$  fires  $a$  infinitely often then  $\rho_2$  fires  $b$  infinitely often. We first observe that the  $\forall\text{-}\exists$  sequence of quantifiers is reminiscent of inclusion problems. Since the population protocol model is close to Petri nets, it is natural to look for undecidable inclusion-like problems for that model. Indeed, undecidability was shown multiple times [5,32] for the problem asking whether the set of reachable markings of a Petri net is included in the set of reachable marking of another Petri net with equally many places. We call this problem the *reachability set inclusion problem*. Our attempts at reducing the reachability set inclusion problem to the above problem faced a major obstacle: Petri nets allow the creation/destruction of tokens while in PPs the number of agents remains the same. We sidestepped this obstacle by looking at a particular

<sup>3</sup> See the proof of Theorem 11 for an illustration of this.

proof of undecidability for the reachability set inclusion problem which leverages Hilbert’s Tenth Problem (shown to be undecidable by Matijasevic in the seventies). We thus obtain a reduction from Hilbert’s Tenth Problem to the above problem for PPs. Our reduction uses PPs to “compute” the value of polynomials while keeping the number of agents constant during the computation.

The detailed proof is given in the appendix and we only provide here the statement of the variant of Hilbert’s Tenth Problem used in the reduction.

**Proposition 7 ([32]).** *The following problem is undecidable:*

**Input:** two polynomials  $P_1(x_1, \dots, x_r), P_2(x_1, \dots, x_r)$  with natural coefficients

**Question:** Does it hold that, for all  $x_1, \dots, x_r \in \mathbb{N}$ ,  $P_1(x_1, \dots, x_r) \leq P_2(x_1, \dots, x_r)$ ?

## 4 Verification of HyperLTL for IOPP

Section 3 showed that verification of HyperLTL in PPs is undecidable, even when the formulas are monadic and have a simple shape. We thus turn to a subclass of PPs called *immediate observation population protocols* (IOPP) [3] that has been studied extensively (see e.g. [24,33,8,6]).

### 4.1 Immediate Observation PP and Preliminary Results

**Definition 8.** *An immediate observation population protocol (IOPP) is a population protocol where all transitions are of the form  $(q_1, q_2) \rightarrow (q_3, q_2)$ .*

We denote a transition  $(q_1, q_2) \rightarrow (q_3, q_2)$  as  $q_1 \xrightarrow{q_2} q_3$ . Intuitively, when two agents interact, one remains in its state, as if it was observed by the other agent.

The IOPP model tends to be simpler to verify than standard PP [24], notably because it enjoys a convenient monotonicity property: whenever an agent observes an agent in  $q_3$  and goes from  $q_1$  to  $q_2$ , another agent in  $q_1$  may do the same “for free”. This property is however broken by the  $X$  operator of LTL. In fact, under LTL, IOPP has similar power to regular PP. Indeed, consider a PP transition  $t : (q_1, q_2) \rightarrow (q_3, q_4)$ . One may split this transition into immediate observation transitions  $t_1 : q_1 \xrightarrow{q_2} q_3$  and  $t_2 : q_2 \xrightarrow{q_3} q_4$ . Using an LTL formula with the  $X$  operator, one can enforce that, whenever  $t_1$  is fired,  $t_2$  must be fired directly after.

We start by establishing that verification of LTL for IOPP is as hard as its counterpart for PP:

**Proposition 9.** *Verification of LTL for IOPP is Ackermann-complete.*

*Remark 10.* The fragment of LTL with no  $X$  operator is equivalent to stutter-invariant LTL [38,25]. Let  $\varphi$  be an LTL\( $X$ ) formula  $\varphi$ , let  $t_1, t_2, \dots \in \Delta$  and  $k_1, k_2, \dots \geq 1$ . This means that we have  $t_1^{k_1} t_2^{k_2} \dots \models \varphi$  if and only if  $t_1 t_2 \dots \models \varphi$ .

Below, we consider the fragment LTL\( $X$ ) as done in prior work [29] in which the systems under study feature monotonicity due to non-atomic writes: stuttering-invariance is a natural choice for systems with monotonicity properties. We show that, even then, verification of HyperLTL\( $X$ ) formulas for IOPP is undecidable.



**Theorem 11.** *Verification of  $\text{HyperLTL} \setminus \mathbf{X}$  is undecidable for IOPP.*

*Proof (sketch).* We proceed by reducing from the halting problem for 2-counter machines with zero-tests, an undecidable problem [37]. A *2-counter machine* consists in two counters  $c_1, c_2$  and a list of instructions  $l_1, \dots, l_n, \text{halt}$ . An instruction  $l_i$  can increment a counter, decrement a counter, or test whether a counter's value is zero. Given a 2-counter machine  $\mathcal{M}$ , we build an IOPP  $\mathcal{P}$ , with the goal of simulating executions of  $\mathcal{M}$  faithfully using runs of  $\mathcal{P}$ . We introduce gadgets to simulate the instructions; to ensure that the gadgets are used correctly we add *bad* transitions which are activated when the simulation “cheats”. A bad transition is activated in a run  $\rho$  if there exists another run which takes all the same transitions as  $\rho$  until it takes a bad transition  $b$ :  $\psi_{\mathcal{B}}(\rho) = \exists \rho'. (\bigvee_{t \in \Delta} t_\rho \wedge t_{\rho'}) \mathcal{U} (\bigvee_{b \in \mathcal{B}} b_{\rho'})$ , where  $\mathcal{B}$  is the set of bad transitions. We define the  $\text{HyperLTL} \setminus \mathbf{X}$  formula  $\psi = \forall \rho. \neg(\mathbf{F} \text{halt}_\rho) \vee \psi_{\mathcal{B}}(\rho)$ . Then  $\mathcal{P} \models^\forall \psi$  if and only if  $\mathcal{M}$  does not halt, and we are done.  $\square$

However, we will show that the monadic  $\text{HyperLTL} \setminus \mathbf{X}$  case is decidable and in 2-EXPSpace.

## 4.2 Product Systems

Our approach consists, as in the proof of Proposition 5, to define *product systems* that combine the IOPP with a Rabin automaton recognizing an LTL formula. We will then characterize the set of configurations of the IOPP that satisfy the formula using reachability sets of the product system.

**Definition 12.** *A product system is a pair  $\mathcal{PS} = (\mathcal{P}, \mathcal{A})$  where:*

- $\mathcal{P} = (Q, \Delta, I)$  is an IOPP,
- $\mathcal{A} = (\mathcal{L}, T, \ell_0, \mathcal{W})$  is a deterministic Rabin automaton over  $\Delta$ .

We refer to the part with the Rabin automaton as the *control part*. There are two distinct notions of size for a product system: the *protocol size*  $|\mathcal{PS}|_{\text{prot}} := |Q|$  and the *control size*  $|\mathcal{PS}|_{\text{cont}} := |\mathcal{L}|$ . The reason for this distinction is that the control size is typically exponential in the size of the LTL formulas, so that keeping track of the two sizes separately will later improve our complexity analysis.

*Semantics of Product Systems.* A configuration of  $\mathcal{PS}$  is an element of  $\mathcal{C} := \mathcal{M}(Q) \times \mathcal{L}$ . Moreover, we let  $\mathcal{C}_0 := \{(\gamma, \ell_0) \mid \gamma \in \mathcal{I}\}$  be the set of initial configurations of the product system. In product systems, unlike in the proof of Proposition 5, the semantics in the PP is modified to match the monotonicity properties of the system. More precisely, we rely on *accelerated semantics* for the IOPP: in  $\mathcal{P}$ , there is an *accelerated step* from  $\gamma$  to  $\gamma'$  with transition  $t \in \Delta$  when there is  $k \geq 1$  such that  $\gamma \xrightarrow{t^k} \gamma'$ . Given two configurations  $c = (\gamma, \ell), c' = (\gamma', \ell') \in \mathcal{C}$  and transition  $t \in \Delta$ , we let  $c \xrightarrow{t} c'$  when there is  $k \geq 1$  such that  $\gamma \xrightarrow{t^k} \gamma'$  in  $\mathcal{P}$  and  $\Delta(\ell, t) = \ell'$ . A step in the product system corresponds to an accelerated step in  $\mathcal{P}$  whose transition is read by  $\mathcal{A}$ . Note that there is no communication from the control part to the IOPP. In product systems, runs and operators  $\text{pre}^*(\cdot)$ ,  $\text{post}^*(\cdot)$  are defined as expected.

### 4.3 Satisfiability as a Reachability Problem

We fix  $\mathcal{P}$  an IOPP,  $\varphi$  an LTL $\setminus X$  formula,  $\mathcal{A} = (\mathcal{L}, T, \ell_0, \mathcal{W})$  a deterministic Rabin automaton recognizing  $\varphi$  obtained using Theorem 4, and we let  $\mathcal{PS} = (\mathcal{P}, \mathcal{A})$ .

Recall that, in  $\mathcal{P}$ , there is an *accelerated step* from  $\gamma$  to  $\gamma'$  using  $t$  when there are  $k \geq 1$  and  $t \in \Delta$  such that  $\gamma \xrightarrow{t^k} \gamma'$ . A (finite) *accelerated run* is a sequence  $\gamma_0, t_1, \gamma_1, \dots, t_m$  such that, for all  $i \in [1, m]$ , there is an accelerated step from  $\gamma_{i-1}$  to  $\gamma_i$  using  $t_i$ . We similarly define infinite accelerated runs. We extend the notion of strong fairness: an infinite accelerated run  $\alpha$  is *strongly fair* when, for every finite accelerated run  $\alpha'$ , if the first configuration of  $\alpha'$  is visited infinitely often in  $\alpha$  then  $\alpha'$  appears infinitely often in  $\alpha$ . A run  $\rho$  of  $\mathcal{PS}$  can be projected onto  $\mathcal{P}$  to obtain an accelerated run of  $\mathcal{P}$ , denoted  $\text{pr}(\rho)$ ;  $\rho$  is called *protocol-fair* when the accelerated run  $\text{pr}(\rho)$  is strongly fair. Given an accelerated run  $\alpha = \gamma_0, t_1, \gamma_1, t_2, \dots$ , we let  $\alpha \models \varphi$  when  $t_1 t_2 \dots \models \varphi$ . An accelerated infinite run  $\alpha = \gamma_0, t_1, \gamma_1, t_2, \dots$  is an *acceleration* of an infinite run  $\rho$  when there are  $k_1, k_2, \dots \geq 1$  such that  $\rho$  is of the form  $\gamma_0, t_1^{k_1}, \gamma_1, t_2^{k_2}, \gamma_2, \dots$ .

**Lemma 13.** *Given a strongly fair accelerated run  $\alpha$ , there is a strongly fair run  $\rho$  such that  $\alpha$  is an acceleration of  $\rho$ . Conversely, given a strongly fair run  $\rho$ , there is a strongly fair acceleration  $\alpha$  of  $\rho$ .*

*Proof (sketch).* Given a strongly fair accelerated run  $\alpha$ , we build  $\rho$  by choosing, for each accelerated step, the minimal number of repetitions of the transition. Any finite run  $\rho'$  available infinitely often in  $\rho$  can be seen as an accelerated finite run  $\alpha'$ ; it is easy to prove that  $\alpha'$  is available infinitely often in  $\alpha$ , and thus appears infinitely often in  $\alpha$ . By minimality of the number of repetitions,  $\rho'$  appears infinitely often in  $\rho$ , so that  $\rho$  is strongly fair. For the other implication, let  $\rho$  be a strongly fair run of  $\mathcal{P}$ ; we build an acceleration of  $\rho$  using randomization. To do so, we iteratively pick at random  $m \in \mathbb{N}$  and we group the next  $m$  transitions if possible; if not, we leave the next transition without accelerating it. Assuming that the probability distribution for  $m$  gives non-zero probability for all integers, the obtained run is strongly fair with probability 1.  $\square$

For  $L \subseteq \mathcal{L}$ , we write  $\mathcal{C}_L := \Gamma \times L \subseteq \mathcal{C}$ ; also, for  $\mathcal{S} \subseteq \mathcal{C}$ , we write  $\overline{\mathcal{S}} := \mathcal{C} \setminus \mathcal{S}$ . We define  $\llbracket \exists \rho. \varphi \rrbracket := \{\gamma \in \Gamma \mid \exists \rho \in \text{FRuns}(\gamma), \rho \models \varphi\}$  the set of configurations  $\gamma$  of  $\mathcal{P}$  such that there exists a strongly fair run from  $\gamma$  satisfying formula  $\varphi$ . Similarly, we define  $\llbracket \forall \rho. \varphi \rrbracket := \{\gamma \in \Gamma \mid \forall \rho \in \text{FRuns}(\gamma), \rho \models \varphi\} = \Gamma \setminus \llbracket \exists \rho. \neg \varphi \rrbracket$  the set of configurations  $\gamma$  of  $\mathcal{P}$  such that all strongly fair runs from  $\gamma$  satisfy formula  $\varphi$ ; in other words, the set of configurations of  $\mathcal{P}$  satisfying  $\varphi$ . We characterize these sets using the product system.

**Theorem 14.** *A configuration  $\gamma$  of  $\mathcal{P}$  is in  $\llbracket \exists \rho. \varphi \rrbracket$  if and only if  $(\gamma, \ell_0)$  is in*

$$\mathcal{S}_{\mathcal{W}} := \text{pre}^* \left( \bigcup_{(F, G) \in \mathcal{W}} \overline{\text{pre}^*(\mathcal{C}_F)} \cap \overline{\text{pre}^*(\text{pre}^*(\mathcal{C}_G))} \right)$$

*Proof.* Let  $\gamma \in \Gamma$ . By Lemma 13 and Remark 10,  $\gamma \in \llbracket \exists \rho. \varphi \rrbracket$  if and only if there is a strongly fair accelerated run  $\alpha$  from  $\gamma$  such that  $\alpha \models \varphi$ . Let  $G$  denote

the graph whose vertices are the configurations of the product system reachable from  $(\gamma, \ell_0)$  and where there is an edge from  $c$  to  $c'$  whenever  $c \xrightarrow{t} c'$  for some  $t \in \Delta$ . We claim that there is a strongly fair accelerated run  $\alpha$  from  $\gamma$  such that  $\alpha \models \varphi$  if and only if there is a bottom SCC  $S$  of  $G$  reachable from  $(\gamma, \ell_0)$  that is *winning*, i.e., such that there is  $(F, G) \in \mathcal{W}$  for which  $S \cap \mathcal{C}_G \neq \emptyset$  but  $S \cap \mathcal{C}_F = \emptyset$ .

The arguments are the same as in the proof of Proposition 5, but with accelerated semantics in  $\mathcal{P}$ . If we have such an SCC  $S$ , it is easy to build a protocol-fair run  $\rho$  of  $\mathcal{PS}$  that goes to  $S$  and visits all configurations in  $S$  infinitely often. We let  $\alpha := \text{pr}(\rho)$ ;  $\alpha$  is strongly fair and, because  $S$  is winning,  $\alpha \models \varphi$ . Suppose now that we have a strongly fair accelerated run  $\alpha$  such that  $\alpha \models \varphi$ . Let  $\rho$  be the run of  $\mathcal{PS}$  such that  $\text{pr}(\rho) = \alpha$ ;  $\rho$  is protocol-fair. Let  $S$  be the SCC visited infinitely often in  $\rho$ ;  $S$  is bottom and  $\rho$  visits infinitely often all configurations in  $S$ . Indeed, the same arguments as in the proof of Proposition 5 apply, except that we rely on strong fairness of the accelerated run, which makes no difference since strong fairness is defined the same for accelerated and non-accelerated runs.

It remains to prove that there is a winning bottom SCC  $S$  reachable from  $(\gamma, \ell_0)$  if and only if  $(\gamma, \ell_0) \in \mathcal{S}_{\mathcal{W}}$ . Suppose first that there is such an SCC  $S$ ; let  $c \in S$  and let  $(F, G) \in \mathcal{W}$  such that  $S \cap \mathcal{C}_G \neq \emptyset$  and  $S \cap \mathcal{C}_F = \emptyset$ . We have  $(\gamma, \ell_0) \in \text{pre}^*(c)$ . Since  $S$  is bottom and  $S \cap \mathcal{C}_F = \emptyset$ , we have  $\text{post}^*(c) \cap \mathcal{C}_F = \emptyset$  and so  $c \in \text{pre}^*(\mathcal{C}_F)$ . We also have  $S = \text{post}^*(S)$ , and because  $S \cap \mathcal{C}_G \neq \emptyset$ , we have  $\text{post}^*(S) \subseteq \text{pre}^*(\mathcal{C}_G)$ ; therefore  $S \cap \text{pre}^*(\text{pre}^*(\mathcal{C}_G)) = \emptyset$ . This proves that  $c \in \text{pre}^*(\mathcal{C}_F) \cap \text{pre}^*(\text{pre}^*(\mathcal{C}_G))$ ; therefore  $(\gamma, \ell_0) \in \mathcal{S}_{\mathcal{W}}$ . Suppose now that  $(\gamma, \ell_0) \in \mathcal{S}_{\mathcal{W}}$ . Let  $(F, G) \in \mathcal{W}$ ,  $c \in \text{post}^*((\gamma, \ell_0))$  such that  $c \in \text{pre}^*(\mathcal{C}_F) \cap \text{pre}^*(\text{pre}^*(\mathcal{C}_G))$ . Let  $S$  be an SCC reachable from  $c$ . We claim that  $S$  is winning. Because  $S \subseteq \text{post}^*(c)$ , we have  $S \cap \mathcal{C}_F = \emptyset$ . Also, if we had  $S \cap \mathcal{C}_G \neq \emptyset$  then any configuration  $c_S \in S$  would be in  $\text{pre}^*(\mathcal{C}_G)$ , so that  $c$  would be in  $\text{pre}^*(\text{pre}^*(\mathcal{C}_G))$ , a contradiction.  $\square$

#### 4.4 $K$ -blind Sets

Observe that  $\mathcal{P} \models^\forall \varphi$  is false if and only if there exists an initial configuration of  $\mathcal{P}$  in the set  $\llbracket \exists \rho. \neg \varphi \rrbracket$ . We show, using the characterization of Theorem 14, that set  $\llbracket \exists \rho. \neg \varphi \rrbracket$  is “nice” in the following sense: if it is non-empty, then it contains a configuration with a bounded number of agents for a doubly exponential bound  $B(\varphi, \mathcal{P})$ . Checking  $\mathcal{P} \models^\forall \varphi$  is then achieved by exploring the finite reachability graph of the product system for configurations with less than  $B(\varphi, \mathcal{P})$  agents. Moreover, we will show that the set of configurations satisfying a monadic HyperLTL $\setminus X$  formula can be decomposed into a Boolean combination of sets of the form  $\llbracket \exists \rho. \varphi \rrbracket$ , for  $\varphi$  an LTL $\setminus X$  formula. This will allow us to check  $\mathcal{P} \models^\forall \psi$  by repeated applications of the exploration procedure described above. We start by formalizing our “nice” sets.

Let  $K \in \mathbb{N}$ . A set  $S \subseteq \Gamma$  of configurations of  $\mathcal{P}$  is *K-blind* when, for all  $\gamma \in \Gamma$  and  $q \in Q$  such that  $\gamma(q) \geq K$ ,  $\gamma \in S$  if and only if  $\gamma + \vec{q} \in S$ . Similarly, a set  $\mathcal{S} \subseteq \mathcal{C}$  of configurations of  $\mathcal{PS}$  is *K-blind* when, for all  $(\gamma, \ell) \in \mathcal{C}$  and  $q \in Q$  such that  $\gamma(q) \geq K$ ,  $(\gamma, \ell) \in \mathcal{S}$  if and only if  $(\gamma + \vec{q}, \ell) \in \mathcal{S}$ .

*Example 15.* The set  $\mathcal{I}$  is 1-blind, because  $\gamma \in \mathcal{I}$  if and only if  $\gamma(q)$  is non-zero when  $q \in I$  and zero otherwise. For the same reason, the set  $\mathcal{C}_0 \subseteq \mathcal{C}$  defined above is 1-blind. Also, for all  $L \subseteq \mathcal{L}$ , the set  $\mathcal{C}_L$  is 0-blind.

**Lemma 16.** *Let  $\mathcal{S}_1$  a  $K_1$ -blind set and  $\mathcal{S}_2$  a  $K_2$ -blind set of  $\mathcal{PS}$ . Then  $\mathcal{S}_1 \star \mathcal{S}_2$  is a  $\max(K_1, K_2)$ -blind set for  $\star \in \{\cup, \cap\}$ . Additionally,  $\overline{\mathcal{S}_1}$  is a  $K_1$ -blind set.*

The previous result states that  $K$ -blind sets are closed under Boolean operations. Next, we find that  $K$ -blind sets are closed under reachability if we enlarge  $K$ .

**Theorem 17.** *Let  $\mathcal{S}$  be a  $K'$ -blind set of  $\mathcal{PS}$ . Then  $\text{post}^*(\mathcal{S})$  and  $\text{pre}^*(\mathcal{S})$  are  $K$ -blind sets for  $K := |Q|^2 \max(K', 2B)$  where  $B = |\mathcal{L}|^{3^{|Q|^2+2} \cdot 2(\log(|Q|^2+2)+1)|Q|^2}$ .*

This theorem crucially relies on the immediate observation assumption, its proof is technical and presented in Section 5. Note that  $K$  is doubly-exponential in  $|Q|$  but polynomial in  $|\mathcal{L}|$  and in  $K'$ , so that this bound is doubly-exponential in  $|\varphi|$  if we let  $\mathcal{A} = \mathcal{A}_\varphi$  using Theorem 4. Let us apply this result to  $\llbracket \exists \rho. \varphi \rrbracket$ :

**Lemma 18.** *Set  $\llbracket \exists \rho. \varphi \rrbracket$  is  $K$ -blind with  $K$  doubly-exponential in  $|\mathcal{P}|$  and  $|\varphi|$ .*

*Proof.* By Theorem 14 we find that  $\llbracket \exists \rho. \varphi \rrbracket \times \{\ell_0\} = \mathcal{S}_\mathcal{W}$ . The sets  $\mathcal{C}_F$  and  $\mathcal{C}_G$  are 0-blind for each pair  $(F, G) \in \mathcal{W}$ . The result follows by iterative applications of Theorem 17 and Lemma 16.  $\square$

#### 4.5 LTL and HyperLTL Verification

We now apply the results from the previous sections to the verification of  $\text{LTL} \setminus \text{X}$  and verification of monadic  $\text{HyperLTL} \setminus \text{X}$  for IOPP; we prove that both problems are decidable and in 2-EXPSPACE. For  $\text{LTL} \setminus \text{X}$ , Lemma 18 shows that we only need to check emptiness of a  $K$ -blind set for  $K$  bounded doubly-exponentially.

**Theorem 19.** *Verification of  $\text{LTL} \setminus \text{X}$  for IOPP is in 2-EXPSPACE, and the same is true for its existential variant.*

*Proof.* By Savitch's Theorem, we can present a non-deterministic procedure. Let  $\varphi$  be an  $\text{LTL} \setminus \text{X}$  formula, and  $\mathcal{P}$  an IOPP. We construct  $\mathcal{A}_\varphi$  using Theorem 4; for this, we pay a doubly-exponential cost in  $|\varphi|$ , which is the most costly part of the procedure. We work in the product system  $\mathcal{PS} := (\mathcal{P}, \mathcal{A}_\varphi)$ .

Observe that  $\mathcal{P} \models^\forall \varphi$  if and only if  $\mathcal{I} \cap \llbracket \exists \rho. \neg \varphi \rrbracket = \emptyset$ , so that it suffices to consider the existential variant. We therefore want to decide whether  $\llbracket \exists \rho. \varphi \rrbracket \cap \mathcal{I} \neq \emptyset$ . The set  $\mathcal{I}$  is 1-blind; by Lemma 18 and Lemma 16,  $\mathcal{I} \cap \llbracket \exists \rho. \varphi \rrbracket$  is  $K$ -blind for  $K$  doubly-exponential in the size of  $\mathcal{P}$  and in the size of  $\varphi$ .

Hence,  $\mathcal{I} \cap \llbracket \exists \rho. \varphi \rrbracket \neq \emptyset$  if and only if it contains  $\gamma_0$  such that  $\gamma_0(q) \leq K$  for all  $q \in Q$ . We guess such a configuration  $\gamma_0$ . We can write  $\gamma_0$  in binary, and thus in exponential space. Checking if  $\gamma_0 \in \mathcal{I}$  is immediate. By Theorem 14, we can check if  $\gamma_0 \in \llbracket \exists \rho. \varphi \rrbracket$  by checking whether, in the product system  $\mathcal{PS} = (\mathcal{P}, \mathcal{A}_\varphi)$ ,  $(\gamma_0, \ell_0) \in \mathcal{S}_\mathcal{W} = \bigcup_{(F,G) \in \mathcal{W}} \text{pre}^* \left( \overline{\text{pre}^*(\mathcal{C}_F)} \cap \overline{\text{pre}^*(\text{pre}^*(\mathcal{C}_G))} \right)$ .

We guess a Rabin pair  $(F, G) \in \mathcal{W}$ . We only need to consider configurations in  $\mathcal{C}_{\gamma_0} := \{(\gamma, \ell) \in \mathcal{C} \mid |\gamma| = |\gamma_0|\}$ . Given a set  $\mathcal{S} \subseteq \mathcal{C}$  whose membership can be checked in 2-EXPSpace for configurations in  $\mathcal{C}_{\gamma_0}$ , checking whether a configuration  $c \in \mathcal{C}_{\gamma_0}$  is in  $\text{pre}^*(\mathcal{S})$  can also be done in 2-EXPSpace: guess a run starting at  $c$ , step by step. After each step, check if the current configuration  $c'$  is in  $\mathcal{S}$ . We only remember the previous configuration and the current one; checking the step can be done in 2-EXPSpace because we have constructed  $\mathcal{A}_\varphi$  and because, in the protocol, a step corresponds to simple arithmetic operations. For each  $H \in \{F, G\}$ , checking whether a configuration  $c \in \mathcal{C}_{\gamma_0}$  is in  $\mathcal{C}_H$  is easy. Therefore, checking whether  $c \in \mathcal{C}_{\gamma_0}$  is in  $\text{pre}^*(\mathcal{C}_H)$  can be done in 2-EXPSpace. By iterating this technique and treating Boolean operations in a natural manner, we check whether  $(\gamma_0, \ell_0) \in \text{pre}^*(\overline{\text{pre}^*(\mathcal{C}_F)} \cap \overline{\text{pre}^*(\mathcal{C}_G)})$ .  $\square$

Let  $\psi$  be a HyperLTL formula over  $\Delta$ , we write  $\llbracket \psi \rrbracket := \{\gamma \in \Gamma \mid \gamma \models \psi\}$ . We show that  $\llbracket \psi \rrbracket$  can be written as a Boolean combination of sets of the form  $\llbracket \exists \rho. \varphi \rrbracket$  with  $\varphi$  an LTL formula. Set  $\llbracket \psi \rrbracket$  is then  $K$ -blind, for  $K$  doubly-exponential, as a Boolean combination of  $K$ -blind sets.

**Lemma 20.** *Let  $\psi = Q_1 \rho_1 \dots Q_k \rho_k. \varphi$  be a monadic HyperLTL $\setminus X$  formula. Set  $\llbracket \psi \rrbracket$  is  $K$ -blind for  $K$  doubly-exponential in  $|\mathcal{P}|$  and  $|\varphi|$ .*

*Proof.* We show  $K$ -blindness where  $K$  is the bound obtained when applying Lemma 18 on  $\mathcal{P}$  and on a formula of size linear in  $|\varphi|$ . Hence, the bound does not depend on the number of quantifiers of  $\psi$ . We proceed by induction on the number of quantifiers  $k \geq 1$ . The base case  $k = 1$  is proved by Lemma 18. Let  $k \geq 2$ ; suppose that the result holds for any monadic HyperLTL formula with  $k - 1$  quantifiers. Let  $\psi = Q_1 \rho_1. Q_2 \rho_2 \dots Q_k \rho_k. \varphi$  with  $\varphi$  described as a Boolean combination of  $\varphi_1$  to  $\varphi_n$ , each referring to a single run variable. Note that  $\llbracket \psi \rrbracket = \Gamma \setminus \llbracket \text{neg}(\psi) \rrbracket$ , where  $\text{neg}(\psi)$  is the formula obtained from  $\psi$  by transforming  $\forall$  quantifiers into  $\exists$  and vice versa, and by replacing the inner formula  $\varphi$  by  $\neg\varphi$ . Therefore, we may assume that  $Q_1 = \exists$ .

Suppose w.l.o.g. that  $\varphi_1$  to  $\varphi_m$  are the formulas that refer to  $\rho_1$ . For every valuation  $\nu : [1, m] \rightarrow \{\text{true}, \text{false}\}$ , let  $\text{Ev}_\nu := \bigwedge_{i=1}^m \varphi_i(\rho) \Leftrightarrow \nu(i)$ ; note that  $\text{Ev}_\nu(\rho)$  only has run variable  $\rho$ . Let  $\varphi[\nu]$  denote the formula  $\varphi$  simplified assuming that, for all  $i \in [1, m]$ ,  $\varphi_i$  has truth value  $\nu(i)$ . Note that  $\rho_1$  does not appear in  $\varphi[\nu]$ . Let  $\psi_\nu := Q_2 \rho_2 \dots Q_k \rho_k. \varphi[\nu]$ . Let  $\gamma \in \Gamma$ ;  $\gamma \in \llbracket \exists \rho_1. \text{Ev}_\nu \rrbracket$  is equivalent to the existence of  $\rho_1 \in \text{FRuns}(\gamma)$  such that, for all  $i \in [1, m]$ ,  $\rho_1 \models \varphi_i$  iff  $\nu(i)$  is true. In words,  $\gamma \in \llbracket \exists \rho_1. \text{Ev}_\nu \rrbracket$  whenever there is  $\rho_1 \in \text{FRuns}(\gamma)$  that yields valuation  $\nu$ . Also,  $\psi_\nu$  corresponds to  $\psi$  simplified under the assumption that run variable  $\rho_1$  yields valuation  $\nu$ ; run variable  $\rho_1$  does not appear in  $\psi_\nu$  and  $\psi_\nu$  does not need quantifier  $Q_1$ . We deduce that  $\llbracket \psi \rrbracket = \bigcup_{\nu: [1, m] \rightarrow \{\text{true}, \text{false}\}} \llbracket \exists \rho_1. \text{Ev}_\nu \rrbracket \cap \llbracket \psi_\nu \rrbracket$ .

For every  $\nu$ ,  $\psi_\nu$  only has  $k - 1$  quantifiers; by induction hypothesis,  $\llbracket \psi_\nu \rrbracket$  is  $K$ -blind. This also holds for  $\llbracket \exists \rho_1. \text{Ev}_\nu \rrbracket$  because  $\text{Ev}_\nu$  has size at most linear in  $|\varphi|$ . Thanks to Lemma 16, we obtain that  $\llbracket \psi \rrbracket$  is  $K$ -blind.  $\square$

We can now extend Theorem 19 to monadic HyperLTL $\setminus X$ .

**Theorem 21.** *Verification of monadic HyperLTL\X for immediate observation population protocols is in 2-EXPSpace.*

*Proof.* Again, we present a non-deterministic procedure. Let  $\psi$  be a HyperLTL\X formula; as in the proof of Theorem 19, we may consider the existential case only, where one asks whether  $\llbracket \psi \rrbracket \cap \mathcal{I} \neq \emptyset$ . By Lemma 20 and Lemma 16,  $\llbracket \psi \rrbracket \cap \mathcal{I}$  is  $K$ -blind for some doubly-exponential  $K$ , so that  $\llbracket \psi \rrbracket \cap \mathcal{I} \neq \emptyset$  if and only if there is  $\gamma \in \llbracket \psi \rrbracket \cap \mathcal{I} \cap \Gamma_{\leq K}$  where  $\Gamma_{\leq K} := \{\gamma \mid \forall q, \gamma(q) \leq K\}$ . We guess such a  $\gamma \in \Gamma_{\leq K}$ . We can write  $\gamma$  in binary, and thus in exponential space. It is easy to check that  $\gamma \in \mathcal{I}$ . Let  $\psi = Q_1 \rho_1 \dots Q_k \rho_k \cdot \varphi$  with  $\varphi$  described as a Boolean combination of  $\varphi_1$  to  $\varphi_n$ , each referring to a single run variable. For each  $j \in [1, k]$ , let  $\ell_j$  be the number of  $\varphi_i$  that refer to run variable  $\rho_j$ . From the proof of Lemma 20, we can compute a *simple expression* for  $\llbracket \psi \rrbracket$  in the form of a Boolean combination of *elementary sets* of the form  $\llbracket \exists \rho. \varphi' \rrbracket$ . Moreover, with a straightforward induction, this simple expression is composed of at most  $O(2^{\ell_1 + \dots + \ell_k})$  elementary sets, because the union over the possible valuations has  $2^{\ell_j}$  disjuncts during induction step  $j$ ; also, each elementary set formula has size linear in  $|\varphi|$ . We compute, in exponential time, this simple expression. We check if  $\gamma \in \llbracket \psi \rrbracket$  by evaluating membership of  $\gamma$  in each elementary set with Theorem 19 using doubly-exponential space, and then evaluating the simple expression.  $\square$

## 5 A Structural Bound in Product Systems

This section is devoted to proving Theorem 17. We rely on the theory of well-quasi-orders (see, *e.g.*, [16]). A *quasi-order* is a set equipped with a transitive and symmetric relation. In a quasi-order  $(E, \preceq)$ , a set  $S \subseteq E$  is *upward-closed* (resp. *downward-closed*) when, for all  $s \in S$ , for all  $t \in E$ , if  $s \preceq t$  then  $t \in S$  (resp. if  $t \preceq s$  then  $s \in S$ ); also,  $\uparrow S := \{t \in E \mid \exists s \in S, s \preceq t\}$  is its *upward-closure* and  $\downarrow S := \{t \in E \mid \exists s \in S, t \preceq s\}$  its *downward-closure*. A *well-quasi-order* is a quasi-order  $(E, \preceq)$  such that, for every infinite sequence  $(x_i)_{i \in \mathbb{N}}$  of elements of  $E$ , there is  $i < j$  such that  $x_i \preceq x_j$ . In a well-quasi-order  $(E, \preceq)$ , any upward-closed set  $S$  has a finite set of minimal elements  $\text{basis}(S)$ , and  $S = \uparrow \text{basis}(S)$ .

### 5.1 Transfer Flows

We fix a product system  $\mathcal{PS} = (\mathcal{P}, \mathcal{A})$  with  $\mathcal{P} = (Q, \Delta, I)$  and  $\mathcal{A} = (\mathcal{L}, T, \ell_0, \mathcal{W})$ . We prove Theorem 17 using *transfer flows*, an abstraction representing the possibilities offered by sequences of transitions. Let  $\mathbb{N}_\# := \mathbb{N} \cup \{\#\}$ ; we extend  $(\mathbb{N}, \leq)$  to  $(\mathbb{N}_\#, \leq)$  where  $\#$  is incomparable with integers: for all  $x \in \mathbb{N}_\#$ ,  $x \sim \#$  iff  $x = \#$  for  $\sim \in \{\leq, \geq\}$ . We extend addition by  $\# + x = x$  for all  $x \in \mathbb{N}_\#$ .

**Definition 22.** A transfer flow is a triplet  $\text{tf} = (f, \ell, \ell')$  where  $f : Q^2 \rightarrow \mathbb{N}_\#$  and  $\ell, \ell' \in \mathcal{L}$ . We denote by  $\mathcal{F}$  the set of all transfer flows.

Intuitively,  $(f, \ell, \ell')$  represents possible finite runs of  $\mathcal{PS}$ , with  $f$  the transfer of agents in  $\mathcal{P}$  and  $\ell, \ell'$  the start and end states in  $\mathcal{A}$ . Having  $f(q_1, q_2) = \#$

represents the impossibility to send agents from  $q_1$  to  $q_2$ , while  $f(q_1, q_2) = n$  represents the need to send at least  $n$  agents from  $q_1$  to  $q_2$ ; in this case, any number in  $[n, +\infty[$  can be sent. The values  $\ell, \ell'$  are called the *control part* of  $\mathbf{tf}$ , while the function  $f$  is called the *agent part* of  $\mathbf{tf}$ . Given a transfer flow  $\mathbf{tf} = (f, \ell, \ell') \in \mathcal{F}$ , we define its *weight* by  $\text{weight}(\mathbf{tf}) := \sum_{q, q'} f(q, q')$ .

We define a partial order  $\preceq$  on  $\mathcal{F}$  as follows. For  $\mathbf{tf}_1 = (f_1, \ell_1, \ell'_1)$  and  $\mathbf{tf}_2 = (f_2, \ell_2, \ell'_2)$ , we let  $\mathbf{tf}_1 \preceq \mathbf{tf}_2$  when  $\ell_1 = \ell'_1$ ,  $\ell_2 = \ell'_2$  and, for all  $q, q'$ ,  $f_1(q, q') \leq f_2(q, q')$ . In particular, this requires that, for all  $q, q'$ ,  $f_1(q, q') = \#$  if and only if  $f_2(q, q') = \#$ . It is easy to see that  $(\mathcal{F}, \preceq)$  is a well-quasi-order. We highlight the following rule of thumb: *smaller transfer flows are more powerful*. Indeed, when  $\mathbf{tf}_1 \preceq \mathbf{tf}_2$ , for  $q, q'$  such that  $f_1(q, q'), f_2(q, q') \neq \#$ ,  $f_1(q, q') \leq f_2(q, q')$ :  $\mathbf{tf}_1$  allows to send from  $q$  to  $q'$  any number of agents in  $[f_1(q, q'), +\infty[$  while  $\mathbf{tf}_2$  allows to send from  $q$  to  $q'$  any number of agents in  $[f_2(q, q'), +\infty[ \subseteq [f_1(q, q'), +\infty[$ .

**Definition 23.** Given  $c_1 = (\gamma_1, \ell_1), c_2 = (\gamma_2, \ell_2) \in \mathcal{C}$  and  $\mathbf{tf} = (f, \ell, \ell') \in \mathcal{F}$ , we let  $c_1 \xrightarrow{\mathbf{tf}} c_2$  when  $\ell_1 = \ell$ ,  $\ell_2 = \ell'$  and there is a step witness  $g : Q^2 \rightarrow \mathbb{N}_\#$  such that  $f(q, q') \leq g(q, q')$  for all  $q, q' \in Q$ ,  $\gamma_1(q) = \sum_{q'} g(q, q')$  for all  $q \in Q$  and  $\gamma_2(q) = \sum_{q'} g(q', q)$  for all  $q \in Q$ .

Note that if  $c_1 \xrightarrow{\mathbf{tf}} c_2$ , then  $c_1 \xrightarrow{\mathbf{tf}'} c_2$  for all  $\mathbf{tf}' \preceq \mathbf{tf}$ : again, smaller transfer flows are more powerful. Intuitively,  $g$  corresponds to a transfer of agents in  $\mathcal{PS}$  concretizing  $c_1 \xrightarrow{\mathbf{tf}} c_2$ . We now build transfer flows corresponding to transitions of  $\mathcal{PS}$ . For each  $t = (q_1, q_2) \rightarrow (q_1, q_3) \in \Delta$ , we define the set  $F[t] \subseteq \mathcal{F}$  that contains all transfer flows  $(f, \ell, \ell')$  such that  $T(\ell, t) = \ell'$  holds where  $T$  is the transition function of the Rabin automaton and:

- if  $q_1 \neq q_2$  or  $q_1 \neq q_3$  then  $f(q_1, q_1) \geq 1, f(q_2, q_3) \geq 1$ ;
- if  $q_1 = q_2 = q_3$  then  $f(q_1, q_1) \geq 2$ ;
- for all  $q \neq q_1$  such that  $(q, q) \neq (q_2, q_3)$ ,  $f(q, q) \geq 0$ ;
- for all  $q \neq q'$  such that  $(q, q') \neq (q_2, q_3)$ ,  $f(q, q') = \#$ .

That is, at least one agent is in  $q_1$ , some agents are sent from  $q_2$  to  $q_3$  and the control part is changed according to  $t$ . The set  $F[t]$  is upward-closed with respect to  $\preceq$ : the number of agents going from  $q_2$  to  $q_3$  can be arbitrarily large, which corresponds to an accelerated step of  $\mathcal{P}$  using transition  $t$ .

**Lemma 24.** For all  $c, c' \in \mathcal{C}$ ,  $t \in \Delta$ ,  $c \xrightarrow{t} c'$  iff there is  $\mathbf{tf} \in F[t]$  s.t.  $c \xrightarrow{\mathbf{tf}} c'$ .

We define the product set  $\mathbf{tf}_1 \otimes \mathbf{tf}_2 \subseteq \mathcal{F}$  of two transfer flows. This set is meant to encode the possibilities given by using  $\mathbf{tf}_1$  followed by  $\mathbf{tf}_2$ . Let  $\mathbf{tf}_1 = (f_1, \ell_1, \ell'_1), \mathbf{tf}_2 = (f_2, \ell_2, \ell'_2) \in \mathcal{F}$ . If  $\ell'_1 \neq \ell_2$ , then we set  $\mathbf{tf}_1 \otimes \mathbf{tf}_2 = \emptyset$ . Assume now  $\ell'_1 = \ell_2$ . The set  $\mathbf{tf}_1 \otimes \mathbf{tf}_2$  contains all transfer flows of the form  $(h, \ell_1, \ell'_2)$  for which there is a *product witness*  $H : Q^3 \rightarrow \mathbb{N}_\#$  such that:

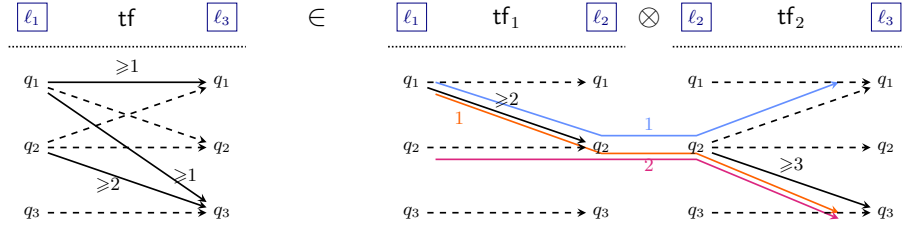
- (prod.i) for all  $(q_1, q_3)$ ,  $\sum_{q_2} H(q_1, q_2, q_3) = h(q_1, q_3)$ ;
- (prod.ii) for all  $(q_1, q_2)$ ,  $\sum_{q_3} H(q_1, q_2, q_3) \geq f_1(q_1, q_2)$ ;
- (prod.iii) for all  $(q_2, q_3)$ ,  $\sum_{q_1} H(q_1, q_2, q_3) \geq f_2(q_2, q_3)$ .

In particular, for all  $q_1, q_2$ ,  $f_1(q_1, q_2) = \#$  if and only if, for all  $q_3$ ,  $H(q_1, q_2, q_3) = \#$ . Similarly,  $f_2(q_2, q_3) = \#$  if and only if, for all  $q_1$ ,  $H(q_1, q_2, q_3) = \#$ . We extend  $\otimes$  to sets of transfer flows: for  $F, F' \subseteq \mathcal{F}$ ,  $F \otimes F' := \bigcup_{\text{tf} \in F, \text{tf}' \in F'} \text{tf} \otimes \text{tf}'$ .

**Lemma 25.** *Let  $\text{tf}_1, \text{tf}_2, \text{tf}_3 \in \mathcal{F}$ . We have the following properties:*

- (25.i) *the set  $\text{tf}_1 \otimes \text{tf}_2$  is upward-closed with respect to  $\preceq$ ;*
- (25.ii) *for all  $\text{tf}'_1 \preceq \text{tf}_1$  and  $\text{tf}'_2 \preceq \text{tf}_2$ ,  $\text{tf}_1 \otimes \text{tf}_2 \subseteq \text{tf}'_1 \otimes \text{tf}'_2$ ;*
- (25.iii)  *$\otimes$  is associative:  $(\text{tf}_1 \otimes \text{tf}_2) \otimes \text{tf}_3 = \text{tf}_1 \otimes (\text{tf}_2 \otimes \text{tf}_3)$ ;*
- (25.iv) *for every  $\text{tf} \in \text{basis}(\text{tf}_1 \otimes \text{tf}_2)$ ,  $\text{weight}(\text{tf}) \leq \text{weight}(\text{tf}_1) + \text{weight}(\text{tf}_2)$ .*

*Example 26.* Consider Fig. 1. Let  $\text{tf}_1 = (f_1, \ell_1, \ell_2)$  and  $\text{tf}_2 = (f_2, \ell_2, \ell_3)$ , with  $f_1(q_1, q_2) = 2$ ,  $f_2(q_2, q_3) = 3$ ,  $f_1(q, q) = f_2(q, q) = 0$  for all  $q$ ,  $f_2(q_2, q_1) = 0$  and all other values equal to  $\#$ . Let  $\text{tf} = (f, \ell_1, \ell_3)$ , with  $f(q_1, q_1) = 1$ ,  $f(q_1, q_3) = 1$ ,  $f(q_2, q_3) = 2$ ,  $f(q_2, q_2) = f(q_3, q_3) = f(q_1, q_2) = f(q_2, q_1) = 0$  and  $f(q, q') = \#$  for all other  $(q, q')$ . We have  $\text{tf} \in \text{tf}_1 \otimes \text{tf}_2$ . Indeed, we have a product witness  $H$  defined by  $H(q_1, q_2, q_1) = 1$ ,  $H(q_1, q_2, q_3) = 1$ ,  $H(q_2, q_2, q_3) = 2$ ,  $H(q_1, q_2, q_2) = H(q_2, q_2, q_1) = H(q_2, q_2, q_2) = H(q_3, q_3, q_3) = H(q_1, q_1, q_1) = 0$  and all other values equal to  $\#$ . In fact,  $\text{tf}$  is minimal for  $\preceq$  in  $\text{tf}_1 \otimes \text{tf}_2$ .



**Fig. 1.** Dashed arrows correspond to value 0, no arrow corresponds to  $\#$ . The product witness  $H$  is represented with colored arrows. We do not depict  $H$  when its value is 0.

Given a sequence  $t_1 \dots t_k$  of transitions, we let  $F[t_1 \dots t_k] := F[t_1] \otimes F[t_2] \otimes \dots \otimes F[t_k]$ . For the empty sequence  $\epsilon$ , we define  $F[\epsilon]$  as the set of  $(f, \ell, \ell')$  where  $\ell = \ell'$ ,  $f(q, q) \in \mathbb{N}$  for all  $q$  and  $f(q, q') = \#$  for all  $q \neq q'$ . For all upward-closed sets  $F \subseteq \mathcal{F}$ , we have  $F \otimes F[\epsilon] = F[\epsilon] \otimes F = F$ . Observe that, for every  $t_1 \dots t_k$ , all transfer flows  $(f, \ell, \ell') \in F[t_1 \dots t_k]$  are such that  $f(q, q) \in \mathbb{N}$  for all  $q$ .

**Lemma 27.** *For all  $k \geq 0$ , for all  $t_1, t_2, \dots, t_k \in \Delta$ , and for all  $c, c' \in \mathcal{C}$ ,  $c \xrightarrow{t_1 \dots t_k} c'$  if and only if there exists  $\text{tf} \in F[t_1 \dots t_k]$  such that  $c \xrightarrow{\text{tf}} c'$ .*

*Proof (sketch).* The proof is by induction on  $k$ . The difficult case is  $k = 2$ : there exists  $\text{tf} \in \text{tf}_1 \otimes \text{tf}_2$  s.t.  $c_1 \xrightarrow{\text{tf}} c_3$  iff there exists  $c_2 \in \mathcal{C}$  s.t.  $c_1 \xrightarrow{\text{tf}_1} c_2 \xrightarrow{\text{tf}_2} c_3$ . First, if there is  $\text{tf} \in \text{tf}_1 \otimes \text{tf}_2$  for which  $c_1 \xrightarrow{\text{tf}} c_3$  then we let  $H$  be a witness for  $\text{tf} \in \text{tf}_1 \otimes \text{tf}_2$ , and we build the multiset  $\mu_2$  of  $c_2$  by  $\mu_2 : q_2 \in Q \mapsto \sum_{q_1, q_3} H(q_1, q_2, q_3)$ .



Conversely, given  $c_2 \in \mathcal{C}$  such that  $c_1 \xrightarrow{\text{tf}_1} c_2 \xrightarrow{\text{tf}_2} c_3$ , let  $h_1$  be a step witness for  $c_1 \xrightarrow{\text{tf}_1} c_2$  and  $h_2$  for  $c_2 \xrightarrow{\text{tf}_2} c_3$ ; we build a product witness  $H : Q^3 \rightarrow \mathbb{N}_\#$  such that  $\sum_{q_3} H(q_1, q_2, q_3) = h_1(q_1, q_2)$  and  $\sum_{q_1} H(q_1, q_2, q_3) = h_2(q_2, q_3)$ , which is possible because  $\sum_{q_1} h_1(q_1, q_2) = \sum_{q_3} h_2(q_2, q_3) = \mu_2(q_2)$ .  $\square$

Given  $T \subseteq \Delta^*$ , we let  $F[T] := \bigcup_{w \in T} F[w]$ . For all  $k \geq 0$ , we denote by  $\Delta^{\leq k} \subseteq \Delta^*$  the set of sequences of length at most  $k$ . Let  $m = |Q|$  and  $M = |\mathcal{L}|$ . We prove Theorem 17 using the following theorem, which we prove in Section 5.3.

**Theorem 28 (Structural theorem).** *Let  $B := (M+1)^{3^{m^2+2} \cdot 2(\log(m^2+2)+1)m^2}$ . We have  $F[\Delta^{\leq B}] = F[\Delta^*]$  and elements of  $\text{basis}(F[\Delta^*])$  have norm at most  $2B$ .*

## 5.2 Proof of Theorem 17

Again, we write  $m = |Q|$  and  $M = |\mathcal{L}|$ . Let  $K' \geq 0$ ,  $K := m^2 \max(K', 2B)$  and  $\mathcal{S}$  a  $K'$ -blind set. We prove that  $\text{post}^*(\mathcal{S})$  is  $K$ -blind; the proof for  $\text{pre}^*(\mathcal{S})$  is similar. We start with the following observation.

**Lemma 29.** *A configuration  $c$  is in  $\text{post}^*(\mathcal{S})$  if and only if there are  $c_{\mathcal{S}} \in \mathcal{S}$  and  $\text{tf} \in F[\Delta^*]$  such that  $c_{\mathcal{S}} \xrightarrow{\text{tf}} c$  and  $\text{weight}(\text{tf}) \leq 2B$ .*

*Proof.* By Lemma 27, if we have such  $c_{\mathcal{S}}$  and  $\text{tf}$  then  $c \in \text{post}^*(\mathcal{S})$ . Conversely, if  $c = (\gamma, \ell) \in \text{post}^*(\mathcal{S})$ , there are  $c_{\mathcal{S}} = (\gamma_{\mathcal{S}}, \ell_{\mathcal{S}}) \in \mathcal{S}$  and  $w \in \Delta^*$  such that  $\gamma_{\mathcal{S}} \xrightarrow{w} \gamma$ . By Lemma 27, there is  $\text{tf} = (f, \ell_{\mathcal{S}}, \ell) \in F[w] \subseteq F[\Delta^*]$  such that  $c_{\mathcal{S}} \xrightarrow{\text{tf}} c$ ; by Theorem 28 and Lemma (25.iv), one may assume that  $\text{weight}(\text{tf}) \leq 2B$ .  $\square$

Let  $c = (\gamma, \ell) \in \mathcal{C}$  and  $q \in Q$  be such that  $\gamma(q) \geq K$ ; we show that  $(\gamma, \ell) \in \text{post}^*(\mathcal{S})$  if and only if  $(\gamma + \vec{q}, \ell) \in \text{post}^*(\mathcal{S})$ . First, suppose that  $c = (\gamma, \ell) \in \text{post}^*(\mathcal{S})$ . Let  $\text{tf}, c_{\mathcal{S}} = (\gamma_{\mathcal{S}}, \ell_{\mathcal{S}})$  obtained thanks to Lemma 29. Let  $g : Q^2 \rightarrow \mathbb{N}_\#$  be a step witness for  $c_{\mathcal{S}} \xrightarrow{\text{tf}} c$ . We have  $\sum_{r \in Q} g(r, q) = \gamma(q) \geq K$ . By the pigeonhole principle, there is  $r$  such that  $g(r, q) \geq \frac{K}{m^2} \geq K'$  therefore  $\gamma_{\mathcal{S}}(r) \geq K'$ . Let  $g'$  be such that  $g'(q_1, q_2) = g(q_1, q_2)$  for all  $(q_1, q_2) \neq (r, q)$  and  $g'(r, q) = g(r, q) + 1$ ;  $g'$  is a witness that  $(\gamma_{\mathcal{S}} + \vec{r}, \ell_{\mathcal{S}}) \xrightarrow{\text{tf}} (\gamma + \vec{q}, \ell)$ . Thanks to Lemma 27, this proves that  $(\gamma_{\mathcal{S}} + \vec{r}, \ell_{\mathcal{S}}) \xrightarrow{*} (\gamma + \vec{q}, \ell)$ . Because  $\mathcal{S}$  is  $K'$ -blind, we conclude that  $(\gamma + \vec{q}, \ell) \in \text{post}^*(\mathcal{S})$ . Conversely, suppose that  $(c + \vec{q}, \ell) \in \text{post}^*(\mathcal{S})$ . With the same reasoning as above, we obtain  $c_{\mathcal{S}} = (\gamma_{\mathcal{S}}, \ell_{\mathcal{S}}) \in \mathcal{S}$ ,  $\text{tf} = (f, \ell_{\mathcal{S}}, \ell)$ ,  $g, r$  such that  $g$  is a witness that  $c_{\mathcal{S}} \xrightarrow{\text{tf}} c$ . By the pigeonhole principle, there is  $r$  such that  $g(r, q) \geq K' + 1$  and  $g(r, q) \geq 2B + 1 > f(r, q)$ . Because  $\mathcal{S}$  is  $K'$ -blind and  $\gamma_{\mathcal{S}}(r) \geq g(r, q) \geq K' + 1$ , we have  $(\gamma_{\mathcal{S}} - \vec{r}, \ell_{\mathcal{S}}) \in \mathcal{S}$ . Let  $g'(q_1, q_2) = g(q_1, q_2)$  for all  $(q_1, q_2) \neq (r, q)$  and  $g'(r, q) = g(r, q) - 1$ . Because  $g' \geq f$ ,  $g'$  is a step witness that  $(\gamma_{\mathcal{S}} - \vec{r}, \ell_{\mathcal{S}}) \xrightarrow{\text{tf}} (\gamma, \ell)$ . Thanks to Lemma 27, this proves that  $(\gamma, \ell) \in \text{post}^*(\mathcal{S})$ .

### 5.3 Proving the Structural Theorem with Descending Chains

To prove Theorem 28, we use a result bounding the length of descending chains in  $\mathbb{N}^d$  from [34,42]. We recall the result and some definitions. Let  $d \geq 1$ . Given  $\vec{v}$  of  $\mathbb{N}^d$  and  $i \in [1, d]$ , we denote by  $\vec{v}(i)$  its  $i$ -th component. Let  $\leq_\times$  be the order over  $\mathbb{N}^d$  such that  $\vec{u} \leq_\times \vec{v}$  if and only if, for all  $i \in [1, d]$ ,  $\vec{u}(i) \leq \vec{v}(i)$ . The obtained  $(\mathbb{N}^d, \leq_\times)$  is a well-quasi-order (Dickson's lemma [17]). A *descending chain* is a sequence  $D_0 \supsetneq D_1 \supsetneq D_2 \dots$  of sets  $D_k \subseteq \mathbb{N}^d$  that are downward-closed for  $\leq_\times$ . Because  $(\mathbb{N}^d, \leq_\times)$  is a well-quasi-order, all descending chains have finite length, *i.e.*, are of the form  $D_0, \dots, D_\ell$  with  $\ell \in \mathbb{N}$ . To bound the length of descending chains [34,42] we need the sequence to be *controlled* and  $\omega$ -*monotone*.

We extend  $\mathbb{N}$  to  $\mathbb{N}_\omega := \mathbb{N} \cup \{\omega\}$  with  $n < \omega$  for all  $n \in \mathbb{N}$ . Given  $\vec{v} \in \mathbb{N}_\omega^d$ , its *norm*  $\|\vec{v}\|$  is the largest  $\vec{v}(i)$  that is not  $\omega$ . An *ideal*  $I$  is the downward-closure in  $\mathbb{N}_\omega^d$  of a vector  $\vec{v} \in \mathbb{N}_\omega^d$ , *i.e.*,  $I = \downarrow\{\vec{v}\} \cap \mathbb{N}_\omega^d$ ; its *norm*  $\|I\|$  is  $\|\vec{v}\|$ . A downward-closed set  $D \subseteq \mathbb{N}_\omega^d$  is canonically represented as a finite union of ideals; its *norm*  $\|D\|$  is the maximum of the norms of its ideals. Given  $N > 0$  and a descending chain  $(D_k)$ , we call  $(D_k)$   $N$ -*controlled* when, for all  $k$ ,  $\|D_k\| \leq (k+1)N$ . In a descending chain  $(D_k)$ , an ideal  $I$  is *proper* at step  $k$  if  $I$  is in the canonical representation of  $D_k$  but  $I \not\subseteq D_{k+1}$ . The sequence  $(D_k)$  is  $\omega$ -*monotone* if, when an ideal  $I_{k+1}$  represented by some vector  $\vec{v}_{k+1}$  is proper at step  $k+1$ , there is  $I_k$  that is proper at step  $k$  and that is represented by  $\vec{v}_k$  such that, for all  $i \in [1, d]$ , if  $\vec{v}_{k+1}(i) = \omega$  then  $\vec{v}_k(i) = \omega$ .

**Theorem 30 ([42]).** *Let  $d, n > 0$ . Every descending chain  $(D_k)$  of  $\mathbb{N}^d$  that is  $n$ -controlled and  $\omega$ -monotone has length at most  $n^{3^d(\log(d)+1)}$ .*

We now use this bound to prove Theorem 28. Recall that we write  $m = |Q|$  and  $M = |\mathcal{L}|$ . Let  $d := m^2 + 2$  and  $N := M^2 \cdot 2^{m^2} = |\mathcal{L}^2 \times 2^{Q^2}|$ . We fix two arbitrary bijective mappings  $\theta : \mathcal{L}^2 \times 2^{Q^2} \rightarrow [1, N]$  and  $\text{index} : Q^2 \rightarrow [1, m^2]$ . We map transfer flows to sets of elements of  $\mathbb{N}^d$  with  $\chi : \mathcal{F} \rightarrow 2^{\mathbb{N}^d}$ . Let  $\text{tf} = (f, \ell, \ell') \in \mathcal{F}$  and  $S := \{(q, q') \mid f(q, q') = \#\}$ . A vector  $\vec{v} \in \mathbb{N}^d$  is in  $\chi(\text{tf})$  when:

- for all  $(q, q')$  such that  $f(q, q') \neq \#$ ,  $\vec{v}(\text{index}(q, q')) = f(q, q')$ ;
- $\vec{v}(m^2 + 1) = \theta(\ell, \ell', S)$ ;
- $\vec{v}(m^2 + 2) = N + 1 - \theta(\ell, \ell', S)$ .

Note that there is no restriction to  $\vec{v}(i)$  when the corresponding pair  $(q, q') = \text{index}^{-1}(i)$  is such that  $f(q, q') = \#$ . Also, if  $\vec{v} \in \chi(\text{tf})$  and  $\vec{u} \in \chi(\text{tf}')$  are such that  $\vec{v} \leq_\times \vec{u}$ , then  $\vec{u}(m^2 + 1) = \vec{v}(m^2 + 1)$  and  $\vec{u}(m^2 + 2) = \vec{v}(m^2 + 2)$ , so that  $\text{tf}$  and  $\text{tf}'$  have the same states of  $\mathcal{L}$  and the same  $\#$  components. For  $\text{tf} \neq \text{tf}'$ , we have  $\chi(\text{tf}), \chi(\text{tf}') \neq \emptyset$  but  $\chi(\text{tf}) \cap \chi(\text{tf}') = \emptyset$ , a property that we call *strong injectivity* of  $\chi$ . The vectors of  $\mathbb{N}^d \cap \chi(\mathcal{F})$  are exactly those whose last two components are strictly positive and sum to  $N + 1$ . We build a decreasing chain  $(D_k)$  such that  $D_k \cap \chi(\mathcal{F}) = \chi(\mathcal{F} \setminus F[\Delta^{\leq k}])$ .

Let  $V_0$  denote the set of vectors  $\vec{v}$  such that either  $(\vec{v}(m^2 + 1), \vec{v}(m^2 + 2)) = (N + 1, 0)$  or  $(\vec{v}(m^2 + 1), \vec{v}(m^2 + 2)) = (0, N + 1)$ . For technical reasons (related to  $\omega$ -monotonicity), we will enforce that  $D_k \cap V_0 = \emptyset$  for every  $k$ . Note that  $V_0 \cap \chi(\mathcal{F}) = \emptyset$ : vectors in  $V_0$  have no relevance in terms of transfer flows. For all

$k \geq 0$ , let  $U_k := \uparrow\chi(F[\Delta^{\leq k}]) \cup V_0$ , and let  $D_k = \mathbb{N}^d \setminus U_k$ ;  $(D_k)$  is a decreasing chain because all  $D_k$  are downward-closed and  $F[\Delta^{\leq k}] \subseteq F[\Delta^{\leq k+1}]$  for all  $k$ .

**Lemma 31.** *For all  $k$ ,  $U_k \cap \chi(\mathcal{F}) = \chi(F[\Delta^{\leq k}])$  and  $D_k \cap \chi(\mathcal{F}) = \chi(\mathcal{F} \setminus F[\Delta^{\leq k}])$ .*

Note that if  $D_{k+1} = D_k$  then, by Lemma 31,  $\chi(F[\Delta^{\leq k+1}]) = \chi(F[\Delta^{\leq k}])$  and, by injectivity of  $\chi$ ,  $F[\Delta^{\leq k+1}] = F[\Delta^{\leq k}]$ . This means that if  $D_{k+1} = D_k$  then  $F[\Delta^{\leq k}]$  is stable under product by  $F[t]$  for all  $t$ , hence that  $F[\Delta^*] = F[\Delta^{\leq k}]$ . Let  $L$  be the smallest  $k \in \mathbb{N}$  such that  $D_k \neq D_{k-1}$ ; it exists because  $(\mathbb{N}^d, \leq_x)$  is a well-quasi-order. To prove Theorem 28, we want  $L \leq N^{3^d(\log(d)+1)}$ . To apply Theorem 30, we need to prove that  $(D_k)$  is  $(N+1)$ -controlled and  $\omega$ -monotone.

Transfer flows in  $\text{basis}(F[\Delta])$  have weight bounded by 2. Let  $\text{tf} \in \text{basis}(F[\Delta^{\leq k}])$ , there are  $\ell \leq k$  and  $\text{t}_1, \dots, \text{t}_\ell \in \text{basis}(F[\Delta])$  such that  $\text{tf} \in \text{t}_1 \otimes \dots \otimes \text{t}_\ell$ . A straightforward induction using (25.ii) proves that  $\text{weight}(\text{tf}) \leq 2\ell \leq 2k$ . This proves that minimal elements of  $F[\Delta^{\leq k}]$  have weight bounded by  $2k$ . In turn, this bounds the norm of minimal elements of  $U_k$  by  $\max(N+1, 2k)$ . Because  $D_k = \mathbb{N}^d \setminus U_k$ , this last bound applies to the norm of  $D_k$ .

**Lemma 32.**  *$(D_k)$  is  $(N+1)$ -controlled and  $\omega$ -monotone.*

We apply Theorem 30 on  $(D_k)$  to prove that  $(D_k)$  and  $(U_k)$  stabilize at index at most  $(N+1)^{3^d(\log(d)+1)} \leq (M+1)^{3^{m^2+2} \cdot 2(\log(m^2+2)+1)m^2} = B$ , so that  $F[\Delta^*] = F[\Delta^{\leq B}]$ . By above, transfer flows in  $\text{basis}(F[\Delta^{\leq k}])$  have weight bounded by  $k$ , therefore transfer flows of  $\text{basis}(F[\Delta^*]) = \text{basis}(F[\Delta^{\leq B}])$  have weight at most  $2B$ . This concludes the proof of Theorem 28.

## 6 Conclusion

When compared to the NEXPTIME result for LTL\X verification of shared-memory systems with pushdown machines [29], our 2-EXPSpace LTL result may seem weak. However, their techniques are quite specific, while ours are generic, enabling us to go from LTL to monadic HyperLTL with little extra work. Additionally, we believe transfer flows,  $K$ -blind sets and the results thereof apply to other problems and systems, such as reconfigurable broadcast networks [15] or asynchronous shared-memory systems [22], which enjoy a similar monotonicity property to IOPP.

Most problems considered in this paper are undecidable; this was to be expected for infinite-state systems. However, our decidability result (Theorem 21) sheds light on a decidable fragment, suggesting that further research on verification of hyperproperties for infinite-state systems should be pursued.

**Acknowledgments.** We thank the reviewers for their helpful comments and suggestions to improve readability. This publication is part of the grant PID2022-138072OB-I00, funded by MCIN, FEDER, UE. This work has been partially supported by the PRODIGY Project (TED2021-132464B-I00) funded by MCIN and the European Union NextGeneration and by the ESF, as well as by a research grant from Nomadic Labs and the Tezos Foundation.

## References

1. Alistarh, D., Gelashvili, R.: Recent Algorithmic Advances in Population Protocols. *SIGACT News* **49**(3), 63–73 (2018). <https://doi.org/10.1145/3289137.3289150>
2. Angluin, D., Aspnes, J., Diamadi, Z., Fischer, M.J., Peralta, R.: Computation in Networks of Passively Mobile Finite-state Sensors. *Distributed Comput.* **18**(4), 235–253 (2006). <https://doi.org/10.1007/s00446-005-0138-3>
3. Angluin, D., Aspnes, J., Eisenstat, D., Ruppert, E.: The Computational Power of Population Protocols. *Distributed Comput.* **20**(4), 279–304 (2007). <https://doi.org/10.1007/s00446-007-0040-2>
4. Baier, C., Katoen, J.: Principles of model checking. MIT Press (2008)
5. Baker, H.G.: Rabin’s proof of the undecidability of the reachability set inclusion problem of vector addition systems. Massachusetts Institute of Technology, Project MAC (1973)
6. van Bergerem, S., Guttenberg, R., Kiefer, S., Mascle, C., Waldburger, N., Weil-Kennedy, C.: Verification of Population Protocols with Unordered Data. In: 51st International Colloquium on Automata, Languages, and Programming, ICALP 2024. LIPIcs, vol. 297, pp. 156:1–156:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2024). <https://doi.org/10.4230/LIPICS.ICALP.2024.156>
7. Beutner, R., Finkbeiner, B.: Software Verification of Hyperproperties Beyond k-Safety. In: Proc. of the 34th Int’l Conf. on Computer Aided Verification (CAV’22), Part I. LNCS, vol. 13371, pp. 341–362. Springer (2022). [https://doi.org/10.1007/978-3-031-13185-1\\_17](https://doi.org/10.1007/978-3-031-13185-1_17)
8. Blondin, M., Ladouceur, F.: Population Protocols with Unordered Data. In: 50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10–14, 2023, Paderborn, Germany. LIPIcs, vol. 261, pp. 115:1–115:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2023). <https://doi.org/10.4230/LIPICS.ICALP.2023.115>
9. Bonakdarpour, B., Prabhakar, P., Sánchez, C.: Model Checking Timed Hyperproperties in Discrete-Time Systems. In: Proc. of the 12th NASA Formal Methods Symposium (NFM’20). LNCS, vol. 12229, pp. 311–328. Springer (2020)
10. Bonakdarpour, B., Sánchez, C., Schneider, G.: Monitoring Hyperproperties by Combining Static Analysis and Runtime Verification. In: Proc. of the 8th Int’l Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA’18), Part II. LNCS, vol. 11245, pp. 8–27. Springer (2018)
11. Clarkson, M.R., Finkbeiner, B., Koleini, M., Micinski, K.K., Rabe, M.N., Sánchez, C.: Temporal Logics for Hyperproperties. In: Proc. of the 3rd Conference on Principles of Security and Trust (POST 2014). LNCS, vol. 8414, pp. 265–284. Springer (2014). [https://doi.org/10.1007/978-3-642-54792-8\\_15](https://doi.org/10.1007/978-3-642-54792-8_15)
12. Clarkson, M.R., Schneider, F.B.: Hyperproperties. *Journal of Computer Security* **18**(6), 1157–1210 (2010). <https://doi.org/10.3233/JCS-2009-0393>
13. Coenen, N., Finkbeiner, B., Hahn, C., Hofmann, J.: The Hierarchy of Hyperlogics. In: Proc. 34th LICS. pp. 1–13. IEEE (2019). <https://doi.org/10.1109/LICS.2019.8785713>
14. Czerwinski, W., Orlikowski, L.: Reachability in Vector Addition Systems is Ackermann-complete. In: 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS). IEEE (2022). <https://doi.org/10.1109/focs52979.2021.00120>
15. Delzanno, G., Sangnier, A., Traverso, R., Zavattaro, G.: On the Complexity of Parameterized Reachability in Reconfigurable Broadcast Networks. In: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer

- Science, FSTTCS 2012. LIPIcs, vol. 18, pp. 289–300. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2012). <https://doi.org/10.4230/LIPICS.FSTTCS.2012.289>
16. Demri, S., Finkel, A., Goubault-Larrecq, J., Schmitz, S., Schnoebelen, P.: Well-Quasi-Orders for Algorithms. Lecture Notes, MPRI Course 2.9.1 – 2017/2018 (2017), <https://wikimpri.dptinfo.ens-cachan.fr/lib/exe/fetch.php?media=cours:upload:poly-2-9-1v02oct2017.pdf>
  17. Dickson, L.E.: Finiteness of the Odd Perfect and Primitive Abundant Numbers with  $n$  Distinct Prime Factors. *American Journal of Mathematics* **35**(4), 413–422 (1913), <http://www.jstor.org/stable/2370405>
  18. Elsässer, R., Radzik, T.: Recent Results in Population Protocols for Exact Majority and Leader Election. *Bulletin of the EATCS* **126** (2018)
  19. Esparza, J.: Population Protocols: Beyond Runtime Analysis. In: Reachability Problems - 15th International Conference, RP 2021, Liverpool, UK, October 25–27, 2021, Proceedings. Lecture Notes in Computer Science, vol. 13035, pp. 28–51. Springer (2021). [https://doi.org/10.1007/978-3-030-89716-1\\_3](https://doi.org/10.1007/978-3-030-89716-1_3)
  20. Esparza, J., Ganty, P., Leroux, J., Majumdar, R.: Model Checking Population Protocols. In: 36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016, December 13–15, 2016, Chennai, India. LIPIcs, vol. 65, pp. 27:1–27:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2016). <https://doi.org/10.4230/LIPICS.FSTTCS.2016.27>
  21. Esparza, J., Ganty, P., Leroux, J., Majumdar, R.: Verification of Population Protocols. *Acta Informatica* **54**(2), 191–215 (2017). <https://doi.org/10.1007/S00236-016-0272-3>
  22. Esparza, J., Ganty, P., Majumdar, R.: Parameterized Verification of Asynchronous Shared-Memory Systems. In: Computer Aided Verification - 25th International Conference, CAV 2013. Lecture Notes in Computer Science, vol. 8044, pp. 124–140. Springer (2013). [https://doi.org/10.1007/978-3-642-39799-8\\_8](https://doi.org/10.1007/978-3-642-39799-8_8)
  23. Esparza, J., Kretínský, J., Sickert, S.: One Theorem to Rule Them All: A Unified Translation of LTL into  $\omega$ -Automata. *CoRR* **abs/1805.00748** (2018), <http://arxiv.org/abs/1805.00748>
  24. Esparza, J., Raskin, M.A., Weil-Kennedy, C.: Parameterized Analysis of Immediate Observation Petri Nets. In: Application and Theory of Petri Nets and Concurrency - 40th International Conference, PETRI NETS 2019, Aachen, Germany, June 23–28, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11522, pp. 365–385. Springer (2019). [https://doi.org/10.1007/978-3-030-21571-2\\_20](https://doi.org/10.1007/978-3-030-21571-2_20)
  25. Etessami, K.: A note on a question of Peled and Wilke regarding stutter-invariant LTL. *Inf. Process. Lett.* **75**(6), 261–263 (2000). [https://doi.org/10.1016/S0020-0190\(00\)00113-7](https://doi.org/10.1016/S0020-0190(00)00113-7)
  26. Farzan, A., Vandikas, A.: Automated Hypersafety Verification. In: Proc. of CAV 2019. LNCS, vol. 11561, pp. 200–218 (2019). <https://doi.org/10.1007/978-3-030-25540-411>
  27. Finkbeiner, B., Rabe, M.N., Sánchez, C.: A Temporal Logic for Hyperproperties. *CoRR* **abs/1306.6657** (2013), <http://arxiv.org/abs/1306.6657>
  28. Fischer, M.J., Ladner, R.E.: Propositional Dynamic Logic of Regular Programs. *J. Comput. Syst. Sci.* **18**(2), 194–211 (1979). [https://doi.org/10.1016/0022-0000\(79\)90046-1](https://doi.org/10.1016/0022-0000(79)90046-1)
  29. Fortin, M., Muscholl, A., Walukiewicz, I.: Model-Checking Linear-Time Properties of Parametrized Asynchronous Shared-Memory Pushdown Systems. In: Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg,

- Germany, July 24-28, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10427, pp. 155–175. Springer (2017). [https://doi.org/10.1007/978-3-319-63390-9\\_9](https://doi.org/10.1007/978-3-319-63390-9_9)
30. Goguen, J.A., Meseguer, J.: Security Policies and Security Models. In: IEEE Symposium on Security and Privacy. pp. 11–20. IEEE Computer Society (1982). <https://doi.org/10.1109/SP.1982.10014>
  31. Gutsfeld, J.O., Müller-Olm, M., Ohrem, C.: Propositional Dynamic Logic for Hyperproperties. In: Proc. 31st CONCUR. pp. 50:1–50:22. LIPIcs 171, Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). <https://doi.org/10.4230/LIPIcs.CONCUR.2020.50>
  32. Hack, M.: The Equality Problem for Vector Addition Systems is Undecidable. *Theor. Comput. Sci.* **2**(1), 77–95 (1976). [https://doi.org/10.1016/0304-3975\(76\)90008-6](https://doi.org/10.1016/0304-3975(76)90008-6)
  33. Jancar, P., Valusek, J.: Structural Liveness of Immediate Observation Petri Nets. *Fundam. Informaticae* **188**(3), 179–215 (2022). <https://doi.org/10.3233/FI-222146>
  34. Lazic, R., Schmitz, S.: The ideal view on Rackoff’s coverability technique. *Inf. Comput.* **277**, 104582 (2021). <https://doi.org/10.1016/J.IC.2020.104582>
  35. Leroux, J.: The Reachability Problem for Petri Nets is Not Primitive Recursive. In: 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS). IEEE (2022). <https://doi.org/10.1109/focs52979.2021.00121>
  36. McLean, J.D.: A General Theory of Composition for a Class of “Possibilistic” Properties. *IEEE Trans. Software Eng.* **22**(1), 53–67 (1996). <https://doi.org/10.1109/32.481534>
  37. Minsky, M.L.: *Computation: Finite and Infinite Machines*. Prentice-Hall, Inc. (1967)
  38. Peled, D.A., Wilke, T.: Stutter-Invariant Temporal Properties are Expressible Without the Next-Time Operator. *Inf. Process. Lett.* **63**(5), 243–246 (1997). [https://doi.org/10.1016/S0020-0190\(97\)00133-6](https://doi.org/10.1016/S0020-0190(97)00133-6)
  39. Pnueli, A.: The Temporal Logic of Programs. In: Proc. of the 18th IEEE Symp. on Foundations of Computer Science (FOCS’77). pp. 46–67. IEEE CS Press (1977)
  40. Rabe, M.N.: A temporal logic approach to information-flow control. Ph.D. thesis, Saarland University (2016)
  41. Sabelfeld, A., Sands, D.: A per model of secure information flow in sequential programs. *Higher Order Symbolic Computation* **14**(1), 59–91 (2001)
  42. Schmitz, S., Schütze, L.: On the Length of Strongly Monotone Descending Chains over  $\mathbb{N}^d$ . In: 51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia. LIPIcs, vol. 297, pp. 153:1–153:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2024). <https://doi.org/10.4230/LIPIcs.ICALP.2024.153>
  43. Shemer, R., Gurfinkel, A., Shoham, S., Vizel, Y.: Property Directed Self Composition. In: Proc. of CAV’19. LNCS, vol. 11560, pp. 161–179. Springer (2019). [https://doi.org/10.1007/978-3-030-25540-4\\_9](https://doi.org/10.1007/978-3-030-25540-4_9)
  44. Sistla, A.P., Vardi, M.Y., Wolper, P.: The Complement Problem for Büchi Automata with Applications to Temporal Logic. *Theoretical Computer Science* **49**, 217–237 (1987). [https://doi.org/10.1016/0304-3975\(87\)90008-9](https://doi.org/10.1016/0304-3975(87)90008-9)
  45. Sousa, M., Dillig, I.: Cartesian Hoare logic for verifying k-safety properties. In: Proc. of ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI’16). ACM (2016). <https://doi.org/10.1145/2908080.2908092>

- 46. Unno, H., Terauchi, T., Koskinen, E.: Constraint-based Relational Verification. In: Proc. of CAV 2021. LNCS, vol. 12759, pp. 742—766. Springer (2021). <https://doi.org/10.1007/978-3-030-81685-835>
- 47. Wang, Y., Zarei, M., Bonakdarpour, B., Pajic, M.: Statistical Verification of Hyperproperties for Cyber-Physical Systems. *ACM Transactions on Embedded Computing systems* **18**(5s), 92:1–92:23 (2019)
- 48. Zdancewic, S., Myers, A.C.: Observational Determinism for Concurrent Program Security. In: Proc. 16th IEEE CSFW-16. pp. 29–43. IEEE Computer Society (2003). <https://doi.org/10.1109/CSFW.2003.1212703>