A Probabilistic Higher-Order Fixpoint Logic

Yo Mitani
The University of Tokyo, Japan
mitaniyo@kb.is.s.u-tokyo.ac.jp

Naoki Kobayashi [ORCID]
The University of Tokyo, Japan
koba@kb.is.s.u-tokyo.ac.jp

Takeshi Tsukada
The University of Tokyo, Japan
tsukada@kb.is.s.u-toyko.ac.jp

─── **Abstract** ───

We introduce PHFL, a probabilistic extension of higher-order fixpoint logic, which can also be regarded as a higher-order extension of probabilistic temporal logics such as PCTL and the μ^p-calculus. We show that PHFL is strictly more expressive than the μ^p-calculus, and that the PHFL model-checking problem for finite Markov chains is undecidable even for the μ-only, order-1 fragment of PHFL. Furthermore the full PHFL is far more expressive: we give a translation from Lubarsky's μ-arithmetic to PHFL, which implies that PHFL model checking is Π_1^1-hard and Σ_1^1-hard. As a positive result, we characterize a decidable fragment of the PHFL model-checking problems using a novel type system.

1 Introduction

Temporal logics such as CTL and CTL* have been playing important roles, for example, in system verification. Among the most expressive temporal logics is the *higher-order fixpoint logic* (*HFL* for short) proposed by Viswanathan and Viswanathan [22], which is a higher-order extension of the *modal μ-calculus* [13]. HFL is known to be strictly more expressive than the modal μ-calculus but the model-checking problem against finite models is still decidable.

In view of the increasing importance of probabilistic systems, temporal logics for probabilistic systems (such as PCTL [7]) and their model-checking problems have been studied and applied to verification and analysis of probabilistic systems and randomized distributed algorithms [14]. Recently Castro et al. [2] have proposed a probabilistic extension of the modal μ-calculus, called the *μ^p-calculus*. They showed that the μ^p-calculus is strictly more expressive than PCTL and that the model-checking problem for the μ^p-calculus belongs to NP ∩ co-NP.

In the present paper, we introduce *PHFL*, a probabilistic higher-order fixpoint logic, which can be regarded as a probabilistic extension of HFL and as a higher-order extension of the μ^p-calculus. PHFL strictly subsumes the μ^p-calculus [2], which coincides with order-0 PHFL.

We prove that PHFL model checking for finite Markov chains is undecidable even for the order-1 fragment of PHFL without fixpoint alternations, by giving a reduction of the *value problem* of probabilistic automata [21, 20]. In the presence of fixpoint alternations (i.e.,

with both least and greatest fixpoint operators), PHFL model checking is even harder: the order-1 PHFL model-checking problem is Π_1^1-hard and Σ_1^1-hard. The proof is by a reduction from the validity checking problem for μ-arithmetic [16] to PHFL model checking. This may be surprising, because both order-0 PHFL model checking (i.e. μ^p-calculus model checking) for finite Markov chains [2] and HFL model checking for finite state systems [22] are decidable. The combination of probabilities and higher-order predicates suddenly makes the model-checking problem highly undecidable.

As a positive result, we identify a decidable subclass of PHFL model-checking problems. To characterize the subclass, we introduce a type system for PHFL formulas, which is parameterized by Markov chains M. We show that the model-checking problem $M \models \varphi$ is decidable provided that φ is typable by the type system for M, by giving a decision procedure using the decidability of existential theories of reals. The decidable subclass is reasonably expressive: the problem of computing termination probabilities of *recursive Markov chains* [3] can be reduced to the subclass.

The rest of this paper is organized as follows. Section 2 introduces PHFL and shows that it is strictly more expressive than the μ^p-calculus. Section 3 proves undecidability of the model-checking problem for μ-only and order-1 PHFL. Section 4 proves that the PHFL model-checking problem is both Π_1^1-hard and Σ_1^1-hard. Section 5 introduces a decidable subclass of PHFL model-checking problems, and shows that the subclass is reasonably large. Section 6 discusses related work, and Section 7 concludes the paper. Proofs omitted in the paper are found in a longer version of this paper [18].

2   PHFL: Probabilistic Higher-order Fixpoint Logic

This section introduces PHFL, a probabilistic extension of HFL [22]. It is a logic used for describing properties of Markov chains. We define its syntax and semantics and show that it is more expressive than the μ^p-calculus [2].

2.1   Markov Chains

We first recall the standard notion of Markov chains. Our definitions follow those in [2].

▶ **Definition 1.** *A* Markov chain *over a set AP of atomic propositions is a tuple $(S, P, \rho_{AP}, s_{in})$ where*

- S *is a finite set of states,*
- $P : S \times S \to [0, 1]$ *satisfying* $\forall s. \sum_{s' \in S} P(s, s') = 1$ *describes transition probabilities,*
- $\rho_{AP} : AP \to 2^S$ *is a labeling function, and*
- $s_{in} \in S$ *is an initial state.*

For a Markov chain $M = (S, P, \rho_{AP}, s_{in})$, its embedded Kripke structure *is $K = (S, R, \rho_{AP}, s_{in})$ where $R \subseteq S \times S$ is a relation such that $R = \{(s, s') | P(s, s') > 0\}$.*

Intuitively, $P(s, s')$ denotes the probability that the state s transits to the state s', and $\rho_{AP}(p)$ gives the set of states where p is true. Throughout the paper, we assume that the set AP of atomic propositions is closed under negations, in the sense that for any $p \in AP$, there exists $\overline{p} \in AP$ such that $\rho_{AP}(\overline{p}) = S \setminus \rho_{AP}(p)$.

Given a Markov chain M, we often write $S_M, P_M, \rho_{AP,M}, s_{in,M}$ for its components; we omit the subscript M when it is clear from the context.

2.2   Syntax of PHFL Formulas

As in HFL [22, 11], we need the notion of types to define the syntax of PHFL formulas.

The set of types, ranged over by τ, is given by:

$$\tau ::= Prop_{\{0,1\}} \mid Prop_{[0,1]} \mid \tau_1 \rightarrow \tau_2.$$

The type $Prop_{\{0,1\}}$ is for *qualitative* propositions, which take truth values (0 for false, and 1 for true). In contrast, $Prop_{[0,1]}$ is the type of *quantitative* propositions, whose values range over $[0,1]$. Intuitively, the value of a quantitative proposition represents the *probability* that the proposition holds. The type $\tau_1 \rightarrow \tau_2$ is for functions from τ_1 to τ_2. For example, $(Prop_{\{0,1\}} \rightarrow Prop_{\{0,1\}}) \rightarrow Prop_{[0,1]}$ represents the type of (higher-order) quantitative predicates on a qualitative predicate.

We assume a countably infinite set *Var* of variables, ranged over by $X_1, X_2, \ldots$. The set of PHFL (pre-)formulas, ranged over by ϕ, is given by:

$$\phi ::= p \mid X \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid [\phi]_J \mid \{\phi\} \mid \Box\phi \mid \Diamond\phi \mid \bigcirc\phi \mid \mu X.\phi \mid \nu X.\phi \mid \lambda X.\phi \mid \phi_1\,\phi_2.$$

Here, p ranges over the set AP of atomic propositions (of the underlying Markov chains; we thus assume that AP is closed under negations). The subscript J of $[\phi]_J$ is either "$> r$" or "$\geq r$" for some rational number $r \in [0,1]$. We often identify J with an interval: for example, "$> r$" is regarded as $(r,1] = \{ x \mid r < x \leq 1 \}$. Given a quantitative proposition ϕ, the formula $[\phi]_{>r}$ (resp. $[\phi]_{\geq r}$) is a qualitative formula, which is true just if the probability that ϕ holds is greater than r (resp. no less than r). The formulas $\Box\phi$, $\Diamond\phi$, and $\bigcirc\phi$ respectively mean the minimum, maximum, and average probabilities that ϕ holds after a one-step transition. The formulas $\mu X.\phi$ and $\nu X.\phi$ respectively denote the least and greatest fixpoints of $\lambda X.\phi$. Note that ϕ may denote higher-order predicates (unlike in the modal μ-calculus and its probabilistic variants [2, 17, 19], where fixpoints are restricted to propositions). We have also λ-abstractions and applications, to manipulate higher-order predicates. The prefixes μX, νX and λX bind the variable X. As usual, we identify formulas up to the renaming of bound variables and implicitly allow α-conversions.

In order to exclude out ill-formed formulas like $(p_1 \vee p_2)(\phi)$, we restrict the shape of formulas through a type system. A *type environment* is a map from a finite set of variables to the set of types. A *type judgment* is of the form $\Gamma \vdash \phi : \tau$. The typing rules are shown in Figure 1. In the figure, P is a meta-variable ranging over the set $\{Prop_{\{0,1\}}, Prop_{[0,1]}\}$ of proposition types. For example, the rule for $\phi_1 \wedge \phi_2$ means that $\Gamma \vdash \phi_i : Prop_{\{0,1\}}$ for each $i \in \{1,2\}$ implies $\Gamma \vdash \phi_1 \wedge \phi_2 : Prop_{\{0,1\}}$ and that $\Gamma \vdash \phi_i : Prop_{[0,1]}$ for each $i \in \{1,2\}$ implies $\Gamma \vdash \phi_1 \wedge \phi_2 : Prop_{[0,1]}$. A formula ϕ is *well-typed* if $\Gamma \vdash \phi : \tau$ is derivable for some Γ and τ. Henceforth, we consider only well-typed formulas.

▶ **Example 2.** For a proposition $p \in AP$, the formula $\phi = (\mu F.\lambda X.X \vee F(\bigcirc X))\,\{p\}$ is a well-typed formula of type $Prop_{[0,1]}$. By unfolding the fixpoint formula, we obtain:

$$\begin{aligned}
\phi &\equiv (\lambda X.X \vee (\mu F.\lambda X.X \vee F(\bigcirc X))(\bigcirc X))\{p\} \\
&\equiv \{p\} \vee (\mu F.\lambda X.X \vee F(\bigcirc X))(\bigcirc\{p\}) \\
&\equiv \{p\} \vee \bigcirc\{p\} \vee (\mu F.\lambda X.X \vee F(\bigcirc X))(\bigcirc\bigcirc\{p\}) \\
&\equiv \{p\} \vee \bigcirc\{p\} \vee \bigcirc\bigcirc\{p\} \vee \cdots
\end{aligned}$$

Thus, intuitively, the formula represents the function that maps each state s to the value $\sup_{k \geq 0} q_k$ where q_k is the probability that a k-step transition sequence starting from the state s ends in a state satisfying p.   ◀

$$\frac{}{\Gamma \vdash p : Prop_{\{0,1\}}} \qquad \frac{}{\Gamma, X : \tau \vdash X : \tau} \qquad \frac{\Gamma \vdash \phi : Prop_{[0,1]}}{\Gamma \vdash [\phi]_J : Prop_{\{0,1\}}} \qquad \frac{\Gamma \vdash \phi : Prop_{\{0,1\}}}{\Gamma \vdash \{\phi\} : Prop_{[0,1]}}$$

$$\frac{\Gamma \vdash \phi_1, \phi_2 : \mathsf{P}}{\Gamma \vdash \phi_1 \wedge \phi_2 : \mathsf{P}} \qquad \frac{\Gamma \vdash \phi_1, \phi_2 : \mathsf{P}}{\Gamma \vdash \phi_1 \vee \phi_2 : \mathsf{P}} \qquad \frac{\Gamma \vdash \phi : \mathsf{P}}{\Gamma \vdash \Box\phi : \mathsf{P}}$$

$$\frac{\Gamma \vdash \phi : \mathsf{P}}{\Gamma \vdash \Diamond\phi : \mathsf{P}} \qquad \frac{\Gamma \vdash \phi : Prop_{[0,1]}}{\Gamma \vdash \bigcirc\phi : Prop_{[0,1]}} \qquad \frac{\Gamma, X : \tau \vdash \phi : \tau}{\Gamma \vdash \mu X.\phi : \tau}$$

$$\frac{\Gamma, X : \tau \vdash \phi : \tau}{\Gamma \vdash \nu X.\phi : \tau} \qquad \frac{\Gamma, X : \tau_1 \vdash \phi : \tau_2}{\Gamma \vdash \lambda X.\phi : \tau_1 \to \tau_2} \qquad \frac{\Gamma \vdash \phi : \tau_1 \to \tau_2 \qquad \Gamma \vdash \psi : \tau_1}{\Gamma \vdash \phi\,\psi : \tau_2}$$

Figure 1 Type Derivation Rules for PHFL.

▶ Remark 3. Following [11], we have excluded out negations. By a transformation similar to that in [15] and our assumption that the set of atomic propositions is closed under negations, any closed formula of PHFL extended with negations can be transformed to an equivalent negation-free formula. ◀

We define the *order* of a type τ by:

$$order(Prop_{\{0,1\}}) = order(Prop_{[0,1]}) = 0 \qquad order(\tau_1 \to \tau_2) = \max(order(\tau_1)+1, order(\tau_2)).$$

The order of a formula ϕ such that $\Gamma \vdash \phi : \tau$ is the largest order of types used in the derivation of $\Gamma \vdash \phi : \tau$. The *order-k PHFL* is the fragment of PHFL consisting of formulas of order up to k. Order-0 PHFL coincides with the μ^p-calculus [2].

2.3 Semantics

We first give the semantics of types. We write $\leq_{\mathbb{R}}$ for the natural order over the set $\mathbb{R}$ of real numbers, and often omit the subscript when there is no danger of confusion. For a map f, we write $dom(f)$ for the domain of f.

▶ **Definition 4** (Semantics of Types). *For each τ, we define a partially ordered set $[\![\tau]\!] = (D_\tau, \leq_\tau)$ inductively by:*

$$D_{Prop_{\{0,1\}}} = S \to \{0,1\} \qquad f \leq_{Prop_{\{0,1\}}} g \overset{def}{\iff} \forall s \in S.f(s) \leq g(s)$$
$$D_{Prop_{[0,1]}} = S \to [0,1] \qquad f \leq_{Prop_{[0,1]}} g \overset{def}{\iff} \forall s \in S.f(s) \leq g(s)$$
$$D_{\tau_1 \to \tau_2} = \{f \in D_{\tau_1} \to D_{\tau_2} \mid \forall x,y \in D_{\tau_1}.x \leq_{\tau_1} y \implies f(x) \leq_{\tau_2} f(y)\}$$
$$f \leq_{\tau_1 \to \tau_2} g \overset{def}{\iff} \forall x \in D_{\tau_1}.f(x) \leq_{\tau_2} g(x).$$

For a type environment Γ, we write $[\![\Gamma]\!]$ for the set of maps f such that $dom(f) = dom(\Gamma)$ and $f(x) \in D_{\Gamma(x)}$ for every $x \in dom(\Gamma)$.

Note that $[\![\tau]\!]$ forms a complete lattice for each τ. We write $\bot_\tau$ for the least element of $[\![\tau]\!]$, and for a set $V \subseteq D_\tau$, we write $\bigvee_\tau V$ for the least upper bound of S with respect to $\leq_\tau$; we often omit the subscript τ if it is clear from the context. Note also that for every

functional type $\tau_1 \to \tau_2$, every element of $D_{\tau_1 \to \tau_2}$ is monotonic. Thus, for every type τ and every function $f \in D_{\tau \to \tau}$, the least and greatest fixed points of f exist.

We now define the semantics of formulas. Since the meaning of a formula depends on its type environment, we actually define the semantics $[\![\Gamma \vdash \phi : \tau]\!]_M$ for each type judgment $\Gamma \vdash \phi : \tau$. Here, M is the underlying Markov chain, which is often omitted.

▶ **Definition 5** (Semantics of Type Judgement). *Let M be a Markov chain and assume $\Gamma \vdash \phi : \tau$ is derivable. Then its semantics $[\![\Gamma \vdash \phi : \tau]\!]_M \in [\![\Gamma]\!] \to [\![\tau]\!]$ is defined by induction on the (unique) derivation of $\Gamma \vdash \phi : \tau$ by:*

$$
\begin{aligned}
[\![\Gamma \vdash p : Prop_{\{0,1\}}]\!]_M(\rho) &= \lambda s \in S_M. if\ s \in \rho_{AP,M}(p)\ then\ 1\ else\ 0 \\
[\![\Gamma \vdash X : \tau]\!]_M(\rho) &= \rho(X) \\
[\![\Gamma \vdash \phi_1 \wedge \phi_2 : \mathtt{P}]\!]_M(\rho) &= \lambda s \in S_M. \min_{i \in \{1,2\}} [\![\Gamma \vdash \phi_i : \mathtt{P}]\!]_M(\rho)(s) \\
[\![\Gamma \vdash \phi_1 \vee \phi_2 : \mathtt{P}]\!]_M(\rho) &= \lambda s \in S_M. \max_{i \in \{1,2\}} [\![\Gamma \vdash \phi_i : \mathtt{P}]\!]_M(\rho)(s) \\
[\![\Gamma \vdash [\phi]_J : Prop_{\{0,1\}}]\!]_M(\rho) &= \lambda s \in S_M. if\ [\![\Gamma \vdash \phi : Prop_{[0,1]}]\!]_M(\rho)(s) \in J\ then\ 1\ else\ 0 \\
[\![\Gamma \vdash \{\phi\} : Prop_{[0,1]}]\!]_M(\rho) &= [\![\Gamma \vdash \phi : Prop_{\{0,1\}}]\!]_M(\rho) \\
[\![\Gamma \vdash \Box\phi : \mathtt{P}]\!]_M(\rho) &= \lambda s \in S_M. \min_{s':P_M(s,s')>0} [\![\Gamma \vdash \phi : \mathtt{P}]\!]_M(\rho)(s') \\
[\![\Gamma \vdash \Diamond\phi : \mathtt{P}]\!]_M(\rho) &= \lambda s \in S_M. \max_{s':P_M(s,s')>0} [\![\Gamma \vdash \phi : \mathtt{P}]\!]_M(\rho)(s') \\
[\![\Gamma \vdash \bigcirc\phi : Prop_{[0,1]}]\!]_M(\rho) &= \lambda s \in S_M. \sum_{s' \in S_M} P_M(s,s')[\![\Gamma \vdash \phi : Prop_{[0,1]}]\!]_M(\rho)(s') \\
[\![\Gamma \vdash \mu X.\phi : \tau]\!]_M(\rho) &= LFP(\lambda v \in D_\tau.[\![\Gamma, X : \tau \vdash \phi : \tau]\!]_M(\rho[X \mapsto v])) \\
[\![\Gamma \vdash \nu X.\phi : \tau]\!]_M(\rho) &= GFP(\lambda v \in D_\tau.[\![\Gamma, X : \tau \vdash \phi : \tau]\!]_M(\rho[X \mapsto v])) \\
[\![\Gamma \vdash \lambda X.\phi : \tau_1 \to \tau_2]\!]_M(\rho) &= \lambda v \in D_{\tau_1}.[\![\Gamma, X : \tau_1 \vdash \phi : \tau_2]\!]_M(\rho[X \mapsto v]) \\
[\![\Gamma \vdash \phi_1\,\phi_2]\!]_M(\rho) &= [\![\Gamma \vdash \phi_1]\!]_M(\rho)\,([\![\Gamma \vdash \phi_2]\!]_M(\rho))
\end{aligned}
$$

Here $\mathtt{P} \in \{ Prop_{\{0,1\}}, Prop_{[0,1]} \}$.

In the definitions of the semantics of $\Box\phi$ and $\Diamond\phi$, the set $S' = \{s' \in S | P(s,s') > 0\}$ is non-empty and finite, because $\sum_{s' \in S} P(s,s') = 1$ and S is finite by the definition of Markov chains. Thus the max/min operations are well-defined. We also note that $[\![\Gamma \vdash \phi : \tau]\!]$ is a monotone function from $[\![\Gamma]\!]$ to $[\![\tau]\!]$ (here $[\![\Gamma]\!]$ is ordered by the component-wise ordering; note also Remark 6 below). This ensures the well-definedness of the semantics of abstractions.

▶ **Remark 6.** Recall that in a formula $[\phi]_J$, we allow the predicate J to be "$> r$" or "$\geq r$" (where $r \in [0,1]$), but neither "$< r$" nor "$\leq r$". Allowing "$< r$" would break the monotonicity of the semantics of a formula. For example, $[\![\emptyset \vdash \lambda X.[X]_{<1} : Prop_{[0,1]} \to Prop_{\{0,1\}}]\!] = \lambda v \in D_{Prop_{[0,1]}}.\lambda s \in S.(\text{if } v(s) < 1 \text{ then } 1 \text{ else } 0)$ is not monotonic. ◀

We often omit M, the type of the formula, and the type environment in the notation of semantics when there is no confusion and just write $[\![\phi]\!]$ or $[\![\Gamma \vdash \phi]\!]$ for $[\![\Gamma \vdash \phi : \tau]\!]_M$. For a Markov chain $M = (S, P, \rho_{AP}, s_{\mathrm{in}})$ and a closed PHFL formula ϕ of type $Prop_{\{0,1\}}$, we write $M \models \phi$ if $[\![\phi]\!](s_{\mathrm{in}}) = 1$.

▶ **Example 7.** Recall the PHFL formula $\phi = \psi\,\{p\}$ where $\psi = \mu F.\lambda X.X \vee F(\bigcirc X)$ in Example 2. We have

$$
[\![\psi]\!] = \mathrm{LFP}\Big(\lambda v \in D_{Prop_{[0,1]} \to Prop_{[0,1]}}.\lambda x \in D_{Prop_{[0,1]}}.\lambda s \in S.
$$

$$\max\left(x\,s, v\left(\lambda s' \in S. \sum_{s''} P(s', s'') \cdot (x s'')\right) s\right)\right)$$

$$\geq \left(\lambda v.\lambda x.\lambda s. \max\left(x\,s, v\left(\lambda s' \in S. \sum_{s''} P(s', s'') \cdot (x s'')\right) s\right)\right)^{n+1} (\bot_{Prop_{[0,1]} \to Prop_{[0,1]}})$$

$$= \lambda x.\lambda s. \max_{0 \leq k \leq n} \sum_{s_0 s_1 \ldots s_k \in S^{k+1}, s_0 = s} \left(x(s_k) \cdot \prod_{0 \leq j \leq k-1} P(s_j, s_{j+1})\right)$$

for every $n \geq 0$. Thus, we have:

$$[\![\psi]\!] \geq \lambda x.\lambda s \in S. \sup_{k \geq 0} \sum_{s_0 s_1 \ldots s_k \in S^{k+1}, s_0 = s} \left(x(s_k) \cdot \prod_{0 \leq j \leq k-1} P(s_j, s_{j+1})\right).$$

Actually, the equality holds, because the righthand side is a fixpoint of

$$\lambda v \in D_{Prop_{[0,1]} \to Prop_{[0,1]}}.\lambda x \in D_{Prop_{[0,1]}}. \max(x, v(\lambda s \in S. \sum_{s'} P(s, s') \cdot (x s'))).$$

The semantics of ϕ is, therefore, given by

$$[\![\phi]\!] = \lambda s \in S. \sup_{k \geq 0} \sum_{s_0 s_1 \ldots s_k \in S^{k+1}, s_0 = s} \left(\rho_{AP}(p)(s_k) \cdot \prod_{0 \leq j \leq k-1} P(s_j, s_{j+1})\right). \qquad \blacktriangleleft$$

2.4 Expressive Power

PHFL obviously subsumes the μ^p-calculus [2], which coincides with order-0 PHFL. Hence PHFL also subsumes PCTL [7], since the μ^p-calculus subsumes PCTL [2].

PHFL is *strictly* more expressive than the μ^p-calculus.

▶ **Theorem 8.** *Order-1 PHFL is strictly more expressive than the μ^p-calculus, i.e., there exists an order-1 PHFL proposition ϕ such that ϕ is not equivalent to any μ^p-formula.*

Proof. Let $\mathcal{M}$ be the set of Markov chains $M = (S, P, \rho_{AP}, s_{in})$ that satisfy the following conditions.

- $S = \{s_0, s_1, \ldots, s_n\}$ for a positive integer n,
- $P(s_i, s_{i+1}) = 1$ $(0 \leq i \leq n-1)$, $P(s_n, s_n) = 1$ and $P(s_i, s_j) = 0$ otherwise.
- There are three atomic propositions a, b, c with $\rho_{AP}(a) \cup \rho_{AP}(b) = \{s_0, s_1, \ldots, s_{n-1}\}$, $\rho_{AP}(a) \cap \rho_{AP}(b) = \emptyset$ and $\rho_{AP}(c) = \{s_n\}$.
- The initial state is $s_{in} = s_0$

Let ϕ be the order-1 PHFL formula of type $Prop_{\{0,1\}}$:

$$(\mu F.\lambda X.a \wedge \Diamond(X \vee F(b \wedge \Diamond X)))(b \wedge \Diamond c).$$

Note that, for $M \in \mathcal{M}$, $M \models \phi$ holds just if n is even, and ρ_{AP} satisfies $\rho_{AP}(a) = \{s_0, s_1, \ldots, s_{\frac{n}{2}-1}\}$ and $\rho_{AP}(b) = \{s_{\frac{n}{2}}, s_{\frac{n}{2}+1}, \ldots, s_{n-1}\}$.

We show that there is no μ^p-formula equivalent to ϕ. Suppose that a μ^p-formula ϕ' *were* equivalent to ϕ, which would imply that $M \models \phi$ if and only if $M \models \phi'$ for any $M \in \mathcal{M}$. For $M \in \mathcal{M}$, let us write K_M for the embedded Kripke structure of M. Since all the transitions in $\mathcal{M}$ are deterministic, there exists a modal μ-calculus formula ϕ'' such that $M \models \phi'$ if and only if $K_M \models \phi''$ (note that ϕ'' is obtained by replacing $\bigcirc$ with $\Diamond$, and replacing $[\phi_1]_J$ with true if J is "≥ 0" and with ϕ_1 otherwise). That would imply that $K_M \models \phi''$ for $M \in \mathcal{M}$, just if n is even and ρ_{AP} satisfies $\rho_{AP}(a) = \{s_0, s_1, \ldots, s_{\frac{n}{2}-1}\}$ and $\rho_{AP}(b) = \{s_{\frac{n}{2}}, s_{\frac{n}{2}+1}, \ldots, s_{n-1}\}$. But then ϕ'' corresponds to the non-regular language $\{a^m b^m \mid m \geq 1\}$, which contradicts the fact that the modal μ-calculus can express only regular properties. $\qquad \blacktriangleleft$

3   Undecidability of PHFL Model Checking

In this section we prove the undecidability of the following problem.

▶ **Definition 9** (PHFL Model Checking). *The* PHFL *model-checking problem for finite Markov chains is the problem of deciding whether $M \models \phi$, given a (finite) Markov chain M and a closed* PHFL *formula ϕ of type $Prop_{\{0,1\}}$ as input.*

We prove that the problem is undecidable even for the order-1 fragment of PHFL without fixpoint alternations, by a reduction from the undecidability of the value-1 problem [6] for probabilistic automata [21]. In contrast to the undecidability of PHFL model checking, the corresponding model-checking problems are *decidable* for the full fragments of the μ^p-calculus [2] and (non-probabilistic) HFL [22], with fixpoint alternations. Thus, the combination of probabilities and higher-order predicates introduces a new difficulty.

In Section 3.1, we review the definition of probabilistic automata and the value-1 problem. Section 3.2 shows the reduction from the value-1 problem to the PHFL model-checking problem.

3.1   Probabilistic Automata

We review probabilistic automata [21] and the undecidability of the value-1 problem. Our definition follows [4].

▶ **Definition 10** (Probabilistic Automata). *A probabilistic automaton A is a tuple $(Q, \Sigma, q_I, \Delta, F)$ where*
- Q *is a finite set of states,*
- Σ *is a finite set of input symbols,*
- $q_I \in Q$ *is an initial state,*
- $\Delta : Q \times \Sigma \to D(Q)$, *where $D(Q) := \{ f : Q \to [0,1] \mid \sum_{q \in Q} f(q) = 1 \}$ is the set of probabilistic distributions over the set Q, represents transition probabilities, and*
- $F \subseteq Q$ *is a set of accepting states.*

For a word $w = w_1 \cdots w_n \in \Sigma^n$, the probability that w is accepted by $A = (Q, \Sigma, q_I, \Delta, F)$, written $A(w)$, is defined by:

$$A(w) := \sum_{\substack{q_0,\ldots,q_{n-1} \in Q, q_n \in F \\ s.t.\ q_0 = q_I}} \prod_{1 \le i \le n} \Delta(q_{i-1}, w_i)(q_i).$$

The value *of a probabilistic automaton A, denoted by val(A), is defined by*

$$\mathrm{val}(A) := \sup_{w \in \Sigma^*} A(w).$$

The problem of deciding whether val$(A) = 1$, *called the* value-1 problem, *is known to be undecidable.*

▶ **Theorem 11** (Undecidability of The Value-1 Problem [6]). *Given a probabilistic automaton A, whether val$(A) = 1$ is undecidable.*

3.2   The Undecidability Result

Let $A = (Q, \Sigma, q_I, \Delta, F)$ be a probabilistic automaton, where $\Sigma = \{c_1, \ldots, c_{|\Sigma|}\}$ with $|\Sigma| > 0$. We shall construct a Markov chain M_A and a PHFL formula ϕ_A, so that $\text{val}(A) = 1$ if and only if $M_A \models \phi_A$. The undecidability of PHFL model checking then follows immediately from Theorem 11.

We first construct a Markov chain. The set AP of atomic propositions is $\{ p_c \mid c \in \Sigma \} \uplus \{ p_F \}$. The Markov chain $M_A = (S, P, \rho_{AP}, s_{\text{in}})$ is defined as follows.

- The set S of states is $Q \uplus (Q \times \Sigma)$.
- The transition probability P is given by:

$$P((q, c), q') = \Delta(q, c)(q') \qquad\qquad (c \in \Sigma \text{ and } q, q' \in Q)$$

$$P(q, (q, c)) = \frac{1}{|\Sigma|} \qquad\qquad (c \in \Sigma \text{ and } q \in Q)$$

$$P(s, s') = 0 \qquad\qquad\qquad (\text{otherwise})$$

  The first transition (from (q, c) to q') is used to simulate the transition of A from q to q' for the input symbol c. The second transition (from q to (q, c)) is used to choose the next input symbol to be supplied to the automaton; the probability is not important, and replacing $1/|\Sigma|$ with any non-zero probability does not affect the following arguments.

- ρ_{AP} is defined by:

$$\rho_{AP}(p_c) = \{ (q, c) \mid q \in Q \} \qquad\qquad \rho_{AP}(p_F) = \{ q \mid q \in F \}.$$

- The initial state is $s_{\text{in}} = q_I$.

Intuitively, the Markov chain M_A simulates the behavior of A. The atomic proposition p_c means that A is currently reading the symbol c, and p_F means that A is in a final state.

Based on this intuition, we now construct the PHFL formula ϕ_A. For each $c \in \Sigma$, we define a formula f_c of type $Prop_{[0,1]} \to Prop_{[0,1]}$ by:

$$f_c := \lambda X. \Diamond(\{p_c\} \wedge \bigcirc X).$$

Intuitively $f_c(\phi)$ denotes the probability that the automaton transits to a state satisfying ϕ given c as the next input. Given a word $w = w_1 w_2 \ldots w_n \in \Sigma^*$, we define the formula g_w by

$$g_w := f_{w_1}(f_{w_2}(\ldots (f_{w_n}\{p_F\}) \ldots)).$$

We write A_q for the automaton obtained from A by replacing the initial state with q.

▶ **Lemma 12.** $A_q(w) = [\![g_w]\!]_{M_A}(q)$ *for every* $q \in Q$.

Proof. By induction on the length of w. ◀

Using Lemma 12, we obtain $\text{val}(A) = \sup_{n \in \omega} [\![\bigvee_{w \in \Sigma^{\leq n}} g_w]\!]_{M_A}(q_I)$, where $\Sigma^{\leq n}$ is the set of words of length up to n. This can be expressed by using the least fixpoint operator.

▶ **Theorem 13.** *Let θ_A be the formula of type $Prop_{[0,1]} \to Prop_{[0,1]}$ defined by:*

$$\theta_A := \mu F. \big(\lambda X. X \vee \bigvee_{c \in \Sigma} F(f_c\, X)\big).$$

and let $\phi_A := [\theta_A\,\{p_F\}]_{\geq 1}$. Then $\text{val}(A) = [\![\theta_A\,\{p_F\}]\!]_{M_A}(q_I)$. Therefore $M_A \models \phi_A$ if and only if $\text{val}(A) = 1$.

Proof. Let

$$\xi := \lambda F.\lambda X.X \vee \bigvee_{c \in \Sigma} F (f_c X).$$

Then, it is easy to verify:

$$[\![\theta_A]\!]_M \quad = \quad [\![\mu F.\xi F]\!]_M \quad = \quad \bigvee_{Prop_{[0,1]} \to Prop_{[0,1]}} \{[\![\xi^n(\bot)]\!] \mid n \in \omega\}$$

where $\bot := \lambda Z.\mu U.U$ is the formula of type $Prop_{[0,1]} \to Prop_{[0,1]}$, and $\xi^n(x)$ denotes n-times applications of ξ to x.

We have also: $[\![\xi^n(\bot) \{p_F\}]\!]_M = [\![\bigvee_{w \in \Sigma^{\leq n}} g_w]\!]_M$. Therefore, we obtain:

$$\mathrm{val}(A) \quad = \quad \sup_n ([\![\bigvee_{w \in \Sigma^{\leq n}} g_w]\!]_{M_A}(q_I)) \quad = \quad \sup_n ([\![\xi^n(\bot)\{p_F\}]\!](q_I)) \quad = \quad [\![\theta_A \{p_F\}]\!]_{M_A}(q_I),$$

which implies the required result.   ◀

The following is an immediate corollary of Theorems 11 and 13.

▶ **Corollary 14** (Undecidability of PHFL Model-Checking Problem). *There is no algorithm that, given a Markov chain M and a closed order-1 formula ϕ of type $Prop_{\{0,1\}}$, decides whether $M \models \phi$.*

We close this section with some remarks.[1]

▶ **Remark 15.** Note that the value $\mathrm{val}(A)$ of a probabilistic automaton cannot even be approximately computable [4]: there is no algorithm that outputs "Yes" if $\mathrm{val}(A) = 1$ and "No" if $\mathrm{val}(A) \leq \frac{1}{2}$. Thus, the proof of Theorem 13 (in particular, the result $\mathrm{val}(A) = [\![\theta_A \{p_F\}]\!]_{M_A}(q_I)$) also implies that for a qualitative formula of PHFL ψ, $[\![\psi]\!]$ is not approximately computable in general.

▶ **Remark 16.** It would be interesting to study a converse encoding, i.e., to find an encoding of some fragment of the PHFL model checking problem into the value-1 problem. Such an encoding may help us find a decidable class of the PHFL model checking problem, based on decidable subclasses for the value-1 problem, such as the one studied in [5].

4   Hardness of the PHFL Model-Checking Problem

In the previous section, we have seen that PHFL model checking is undecidable even for the fragment of PHFL without fixpoint alternations. In this section, we give a lower bound of the hardness of the PHFL model-checking problem in the presence of fixpoint alternations. The following theorem states the main result of this section.

▶ **Theorem 17.** *The order-1 PHFL model-checking problem is Π_1^1-hard and Σ_1^1-hard.*

Note that Π_1^1 and Σ_1^1, defined in terms of the second-order arithmetic, contain very hard problems. For example, the problem of deciding whether a given first-order Peano arithmetic formula is true is in those classes.

We prove this theorem by reducing the validity checking problem of the μ-arithmetic [16] to the PHFL model-checking problem. It is even possible to reduce the validity checking problem of a higher-order extension of the μ-arithmetic to the PHFL model-checking problem. The key in the proof is a representation of natural numbers as quantitative propositions such that all the operations on natural numbers in the μ-arithmetic are expressible in PHFL.

This section is structured as follows. Section 4.1 reviews the basic notions of the μ-arithmetic. Section 4.2 describes the reduction and proves the theorem above.

---

[1] We would like to thank an anonymous reviewer for pointing them out.

$$\frac{}{\Gamma, X : A \vdash_\mu X : A} \qquad \frac{}{\Gamma \vdash_\mu Z : N} \qquad \frac{\Gamma \vdash_\mu s : N}{\Gamma \vdash_\mu S\,s : N} \qquad \frac{\Gamma \vdash_\mu s, t : N}{\Gamma \vdash_\mu s \leq t : \Omega}$$

$$\frac{\Gamma \vdash_\mu \phi, \psi : \Omega}{\Gamma \vdash_\mu \phi \wedge \psi : \Omega} \qquad \frac{\Gamma \vdash_\mu \phi, \psi : \Omega}{\Gamma \vdash_\mu \phi \vee \psi : \Omega} \qquad \frac{\Gamma, X : A \vdash_\mu \phi : T}{\Gamma \vdash_\mu \lambda X.\phi : A \to T}$$

$$\frac{\Gamma \vdash_\mu \phi : A \to T \qquad \Gamma \vdash_\mu \psi : A}{\Gamma \vdash_\mu \phi\,\psi : T} \qquad \frac{\Gamma, X : T \vdash_\mu \phi : T}{\Gamma \vdash_\mu \mu X.\phi : T} \qquad \frac{\Gamma, X : T \vdash_\mu \phi : T}{\Gamma \vdash_\mu \nu x.\phi : T}$$

Figure 2 Typing Rules for the Higher-order Fixpoint Arithmetic.

4.1 Higher-Order Fixpoint Arithmetic

The μ-arithmetic [16] is a first-order arithmetic with fixpoint operators. This section briefly reviews its higher-order extension, studied by Kobayashi et al. [12].

As in PHFL, we first define the types of μ-arithmetic formulas. The set of *types*, ranged over by A, is given by:

$$A ::= N \,|\, T \qquad\qquad\qquad T ::= \Omega \,|\, A \to T.$$

The type N is for natural numbers, Ω for (qualitative) propositions, and $A \to T$ for functions. We do not allow functions to return values of type N. We define the order of types of the μ-arithmetic similarly to the PHFL types, by: $order(N) = order(\Omega) = 0$ and $order(A \to T) = \max(order(A) + 1, order(T))$.

Assume a countably infinite set *Var* of variables ranged over by X. The set of formulas is given by the following grammar.

$$s ::= X \,|\, Z \,|\, Ss \qquad \phi ::= X \,|\, s_1 \leq s_2 \,|\, \phi_1 \wedge \phi_2 \,|\, \phi_1 \vee \phi_2 \,|\, \lambda X.\phi \,|\, \phi_1\,\phi_2 \,|\, \mu X.\phi \,|\, \nu X.\phi.$$

Here, Z and S respectively denote the constant 0 and the successor function on natural numbers.

The typing rules are shown in Fig. 2. We shall consider only well-typed formulas. We define the *order* of a formula as the largest order of the types of its subformulas.

▶ **Definition 18** (Semantics of Types). *The semantics of a type A is a partially ordered set* $[\![A]\!]_\mu = (D_A, \sqsubseteq_A)$ *defined inductively on the structure of A as follows.*
1. *The semantics of types N and Ω are defined as follows.*

$$D_N = \mathbb{N} \qquad\qquad\qquad n \sqsubseteq_N m \overset{def}{\Longleftrightarrow} n = m$$

$$D_\Omega = \{0, 1\} \qquad\qquad\qquad p \sqsubseteq_\Omega q \overset{def}{\Longleftrightarrow} p \leq q$$

2. *The semantics of the type $A \to T$ is defined as follows.*

$$D_{A \to T} = \{\, f : D_A \to D_T \mid \forall u, v \in D_A.u \sqsubseteq_A v \implies f(u) \sqsubseteq_T f(v) \,\}$$

$$f \sqsubseteq_{A \to T} g \overset{def}{\Longleftrightarrow} \forall v \in D_A.f(v) \sqsubseteq_T g(v)$$

The semantics $[\![T]\!]_\mu$ of a type T forms a complete lattice; we write $\bigvee_T$ for the least upper bound operation, and $\bot_T$ for the least element.

The interpretation $[\![\Gamma]\!]_\mu$ of a type environment Γ is the set of functions θ such that $dom(\theta) = dom(\Gamma)$ and that $\theta(X) \in [\![\Gamma(X)]\!]_\mu$ for every $X \in dom(\Gamma)$. It is ordered by the point-wise ordering.

▶ **Definition 19** (Semantics of Formulas). *The semantics of a formula ϕ with judgment $\Gamma \vdash_\mu \phi : A$ is a monotone map from $[\![\Gamma]\!]_\mu$ to $[\![A]\!]_\mu$, defined as follows.*

$$[\![\Gamma \vdash_\mu X : A]\!]_\mu(\theta) := \theta(X)$$

$$[\![\Gamma \vdash_\mu Z : N]\!]_\mu(\theta) := 0$$

$$[\![\Gamma \vdash_\mu Ss : N]\!]_\mu(\theta) := [\![\Gamma \vdash_\mu s : N]\!]_\mu(\theta) + 1$$

$$[\![\Gamma \vdash_\mu s \le t : \Omega]\!]_\mu(\theta) := \begin{cases} 1 & (\text{if } [\![\Gamma \vdash_\mu s : N]\!]_\mu(\theta) \le [\![\Gamma \vdash_\mu t : N]\!]_\mu(\theta)) \\ 0 & (\text{if } [\![\Gamma \vdash_\mu s : N]\!]_\mu(\theta) > [\![\Gamma \vdash_\mu t : N]\!]_\mu(\theta)) \end{cases}$$

$$[\![\Gamma \vdash_\mu \phi \wedge \psi : \Omega]\!]_\mu(\theta) := [\![\Gamma \vdash_\mu \phi : \Omega]\!]_\mu(\theta) \wedge [\![\Gamma \vdash_\mu \phi : \Omega]\!]_\mu(\theta)$$

$$[\![\Gamma \vdash_\mu \phi \vee \psi : \Omega]\!]_\mu(\theta) := [\![\Gamma \vdash_\mu \phi : \Omega]\!]_\mu(\theta) \vee [\![\Gamma \vdash_\mu \phi : \Omega]\!]_\mu(\theta)$$

$$[\![\Gamma \vdash_\mu \lambda X.\phi : A \to T]\!]_\mu(\theta) := \lambda v \in [\![A]\!]_\mu.[\![\Gamma, X : A \vdash_\mu \phi : T]\!]_\mu(\theta[X \mapsto v])$$

$$[\![\Gamma \vdash_\mu \phi\,\psi : T]\!]_\mu(\theta) := [\![\Gamma \vdash_\mu \phi : A \to T]\!]_\mu(\theta)\,([\![\Gamma \vdash_\mu \psi : A]\!]_\mu(\theta))$$

$$[\![\Gamma \vdash_\mu \mu X.\phi : T]\!]_\mu(\theta) := LFP(\lambda v \in D_T.[\![\Gamma, v : T \vdash_\mu \phi : T]\!](\theta[X \mapsto v]))$$

$$[\![\Gamma \vdash_\mu \nu X.\phi : T]\!]_\mu(\theta) := GFP(\lambda v \in D_T.[\![\Gamma, v : T \vdash_\mu \phi : T]\!](\theta[X \mapsto v]))$$

As in the case of PHFL, we write $[\![\phi]\!]_\mu(\theta)$ for $[\![\Gamma \vdash_\mu \phi : A]\!]_\mu(\theta)$ and just $[\![\phi]\!]_\mu$ for $[\![\phi]\!]_\mu(\emptyset)$ when there is no confusion.

▶ **Example 20.** Let $\phi = \mu F.\lambda X.(X = 100 \vee F(S(S\,X)))$ where 100 is an abbreviation of the term $\underbrace{S(S(\dots S\,Z)\dots)}_{100}$. The semantics $[\![\phi]\!]_\mu$ is a function $f : \mathbb{N} \to \{0,1\}$ where $f(n) = 1$ if and only if n is an even number no greater than 100.

The *validity checking problem* of the higher-order fixpoint arithmetic is the problem of, given a closed formula ϕ of type Ω, deciding whether $[\![\phi]\!]_\mu = 1$. The following result is probably folklore, which follows from the well-known fact that the *fair termination problem for programs* is Π^1_1-complete (see, e.g., Harel [8]), and the fact that the fair termination of a program can be reduced to the validity of a first-order fixpoint arithmetic formula (see, e.g., [12] for the reduction).

▶ **Theorem 21.** *The validity checking problem of the first-order fixpoint arithmetic is Π^1_1-hard and Σ^1_1-hard.*

▶ Remark 22. As for an upper bound, Lubarsky [16] has shown that predicates on natural numbers definable by μ-arithmetic formulas belong to Δ^1_2. One can prove that the validity problem for the μ-arithmetic is Δ^1_2 as well.

## 4.2 Hardness of PHFL Model Checking

We give a reduction of the validity checking problem of the higher-order fixpoint arithmetic to the PHFL model-checking problem. The main theorem of this section (Theorem 17) is an immediate consequence of this reduction and Theorem 21.

Given a formula ϕ of the higher-order fixpoint arithmetic, we need to effectively construct a pair (ψ, M) of a formula of PHFL and a Markov chain such that ϕ is true if and only if $M \models \psi$. The Markov chain M is independent of the formula ϕ. We first define the Markov chain and then explain the intuition of the translation of formulas.

The Markov chain $M = (S, P, \rho_{AP}, s_{\text{in}})$ is shown in Figure 3. It is defined as follows.

- The set of states is $S = \{s_0, s'_0, s_1, s'_1\}$.
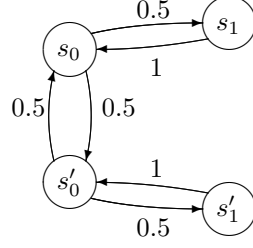
Figure 3 The Markov Chain for Reduction from Higher-order Fixpoint Arithmetic to PHFL.

- The transition probability satisfies $P(s_0, s_1) = P(s_0, s'_0) = P(s'_0, s_0) = P(s'_0, s'_1) = \frac{1}{2}$, $P(s_1, s_0) = P(s'_1, s'_0) = 1$ and $P(s_i, s_j) = 0$ for all other pairs of states.
- There are four atomic propositions $p_0, p'_0, p_1,$ and p'_1, representing each state (e.g. $\rho_{AP}(p_0) = \{s_0\}$).
- The initial state s_{in} is s_0.

For notational convenience, we write $v \in \llbracket Prop_{[0,1]} \rrbracket_M$ as a tuple $(v(s_0), v(s'_0), v(s_1), v(s'_1))$.

As mentioned at the beginning of this section, the key of the reduction is the representation of natural numbers, as well as operations on natural numbers. We represent a natural number n by a quantitative propositional formula ψ such that $\llbracket \psi \rrbracket_M = (\frac{1}{2^n}, 1 - \frac{1}{2^n}, \_, \_)$. Here, $\_$ denotes a "don't care" value. We implement primitives on natural numbers Z, S and $\leq$, as follows.

The constant Z can be represented by $\{p_0\}$: then $\llbracket \{p_0\} \rrbracket_M = (1, 0, 0, 0) = (1/2^0, 1 - (1/2^0), 0, 0)$ as expected.

Assuming that ψ represents n (i.e. $\llbracket \psi \rrbracket_M = (1/2^n, 1 - (1/2^n), \_, \_)$), the successor $n+1$ can be given by

$$\psi' \quad := \quad \bigcirc((\bigcirc\psi \wedge (p_1 \vee p'_1)) \vee p_0).$$

Indeed, we have:

$$\llbracket \bigcirc\psi \rrbracket_M = (\_, \_, \frac{1}{2^n}, 1 - \frac{1}{2^n})$$

$$\llbracket \bigcirc\psi \wedge (p_1 \vee p'_1) \rrbracket_M = (0, 0, \frac{1}{2^n}, 1 - \frac{1}{2^n})$$

$$\llbracket (\bigcirc\psi \wedge (p_1 \vee p'_1)) \vee p_0 \rrbracket_M = (1, 0, \frac{1}{2^n}, 1 - \frac{1}{2^n})$$

$$\llbracket \bigcirc((\bigcirc\psi \wedge (p_1 \vee p'_1)) \vee p_0) \rrbracket_M = (\frac{1}{2} \times \frac{1}{2^n}, \frac{1}{2} + \frac{1}{2} \times (1 - \frac{1}{2^n}), \_, \_)$$

$$= (\frac{1}{2^{n+1}}, 1 - \frac{1}{2^{n+1}}, \_, \_).$$

It remains to encode $\leq$. We use the fact that, for any natural numbers n and m,

$$n \leq m \quad \Leftrightarrow \quad \frac{1}{2^n} \geq \frac{1}{2^m} \quad \Leftrightarrow \quad \frac{1}{2^n} + (1 - \frac{1}{2^m}) \geq 1.$$

The s'_0-component of the representation of a natural number plays an important role below. Assume that ψ and χ represent n and m respectively. Then we have

$$\llbracket \bigcirc\psi \wedge p_1 \rrbracket_M = (0, 0, \frac{1}{2^n}, 0) \qquad \llbracket \chi \wedge p'_0 \rrbracket_M = (0, 1 - \frac{1}{2^m}, 0, 0)$$

and thus

$$[\![(\bigcirc \psi \wedge p_1) \vee (\chi \wedge p_0')]\!]_M = (0, 1 - \frac{1}{2^m}, \frac{1}{2^n}, 0).$$

Therefore

$$[\![\bigcirc ((\bigcirc \psi \wedge p_1) \vee (\chi \wedge p_0'))]\!]_M = (\frac{1}{2} \times (\frac{1}{2^n} + (1 - \frac{1}{2^m})), \_, \_, \_).$$

Therefore, $n \leq m$ if and only if the s_0-component of the above formula is $\geq \frac{1}{2}$.

Let us formalize the above argument. We first give the translation of types:

$$tr(N) = Prop_{[0,1]} \qquad tr(\Omega) = Prop_{\{0,1\}} \qquad tr(A \to T) = tr(A) \to tr(T).$$

The translation can be naturally extended to type environments. Following the above discussion, the translation of formulas of type N is given by

$$tr(Z) = \{p_0\} \qquad \text{and} \qquad tr(S\,s) = \bigcirc ((\bigcirc tr(s) \wedge (p_1 \vee p_1')) \vee p_0).$$

The comparison operator can be translated as follows:

$$tr(s \leq t) = [(\bigcirc ((\bigcirc tr(s) \wedge p_1) \vee (tr(t) \wedge p_0')))]_{\geq \frac{1}{2}}.$$

The translation of other connectives is straightforward:

$$tr(\phi \wedge \psi) = tr(\phi) \wedge tr(\psi) \qquad tr(\phi \vee \psi) = tr(\phi) \vee tr(\psi) \qquad tr(\lambda X.\phi) = \lambda X.tr(\phi)$$
$$tr(X) = X \qquad tr(\phi\,\psi) = tr(\phi)\,tr(\psi) \qquad tr(\mu X.\phi) = \mu X.tr(\phi) \qquad tr(\nu X.\phi) = \nu X.tr(\phi).$$

The following lemma states that the translation preserves types.

▶ **Lemma 23.** *If* $\Gamma \vdash_\mu \phi : A$, *then* $tr(\Gamma) \vdash tr(\phi) : tr(A)$.

We prove the correctness of the translation. For each type A of the higher-order fixpoint arithmetic, we define a relation $(\sim_A) \subseteq [\![A]\!]_\mu \times [\![tr(A)]\!]_M$ by induction on A as follows:

$$n \sim_N (r_0, r_0', r_1, r_1') \quad \overset{\text{def}}{\Longleftrightarrow} \quad r_0 = \frac{1}{2^n} \text{ and } r_0' = 1 - \frac{1}{2^n}$$

$$b \sim_\Omega (r_0, r_0', r_1, r_1') \quad \overset{\text{def}}{\Longleftrightarrow} \quad b = r_0$$

$$f \sim_{A \to T} g \quad \overset{\text{def}}{\Longleftrightarrow} \quad \forall x \in [\![A]\!]_\mu.\forall y \in [\![tr(A)]\!]_M.\, x \sim_A y \implies f\,x \sim_T g\,y.$$

This relation can be naturally extended to the interpretations of type environments: given a type environment Γ of the μ-arithmetic, the relation $(\sim_\Gamma) \subseteq [\![\Gamma]\!]_\mu \times [\![tr(\Gamma)]\!]_M$ is defined by

$$\theta \sim_\Gamma \rho \quad \overset{\text{def}}{\Longleftrightarrow} \quad \forall X \in dom(\Gamma).\, \theta(X) \sim_{\Gamma(X)} \rho(X).$$

▶ **Theorem 24.** *Let* $\Gamma \vdash_\mu \phi : A$ *be a formula of the higher-order fixpoint arithmetic. Assume* $\theta \in [\![\Gamma]\!]_\mu$ *and* $\rho \in [\![tr(\Gamma)]\!]$. *If* $\theta \sim_\Gamma \rho$, *then* $[\![\Gamma \vdash_\mu \phi : A]\!]_\mu(\theta) \sim_A [\![tr(\Gamma) \vdash tr(\phi) : tr(A)]\!]_M(\rho)$.

Proof. See Appendix A. ◀

▶ **Corollary 25.** *The validity problem of the order-k fixpoint arithmetic (where $k > 0$) is reducible to the order-k PHFL model-checking problem.*

Proof. Assume $\emptyset \vdash_\mu \phi : \Omega$. By Theorem 24, $[\![\phi]\!]_\mu \sim_\Omega [\![tr(\phi)]\!]_M$. Therefore, $[\![\phi]\!]_\mu = 1$ if and only if $[\![tr(\phi)]\!]_M(s_0) = 1$, i.e. $M \models tr(\phi)$. The mapping $\phi \mapsto (tr(\phi), M)$ is obviously effective, and preserves the order. ◀

Theorem 17 is an immediate consequence of Theorem 21 and Corollary 25.

5   Decidable Subclass of Order-1 PHFL Model Checking

As we have seen in the last section, PHFL model checking is undecidable in general, even for order 1. In this section, we identify a decidable subclass of the order-1 PHFL model-checking problems (i.e., a set of pairs (ϕ, M) such that whether $M \models \phi$ is decidable). We identify the subclass by using a type system: we define a type system $\mathcal{T}_M$ for PHFL formulas, parameterized by M, such that if ϕ is a proposition well-typed in $\mathcal{T}_M$, then $M \models \phi$ is decidable.

We first explain the idea of the restriction imposed by the type system. By definition, the semantics of a (closed) order-1 PHFL formula ϕ of type $Prop_{[0,1]} \to Prop_{[0,1]}$ with respect to the Markov chain M is a map f_ϕ from the set of functions $S \to [0,1]$ to the same set, where S is the set of states of M. Thus, if $S = \{s_1, s_2, \ldots, s_n\}$ is fixed, f_ϕ can be regarded as a function from $[0,1]^n$ to $[0,1]^n$. Now, if the function f_ϕ were affine, i.e., if there are functions $f_1, f_2, \ldots, f_n$ such that $f_\phi(r_1, r_2, \ldots, r_k) = (f_1(r_1, r_2, \ldots, r_k), \ldots, f_n(r_1, r_2, \ldots, r_k))$, where $f_i(r_1, r_2, \ldots, r_k) = c_{i,0} + c_{i,1}r_1 + \cdots + c_{i,k}r_k$ for some real numbers $c_{i,j}$, then the function f_ϕ would be representable by a finite number of reals $c_{i,j}$. The semantics of an (alternation-free) fixpoint formula would then be given as a solution of a fixpoint equation on the coefficients, which is solvable by appealing to the existential theories of reals.

Based on the observation above, we use a type system to restrict the formulas so that the semantics of every order-1 formula is affine. The conjunction $\phi_1 \wedge \phi_2$ is one of the problematic logical connectives that may make the semantics of an order-1 formula non-affine: recall that the min operator was used to define the semantics of conjunction. We require that for every subformula of the form $\phi_1 \wedge \phi_2$ and for each state $s \in S$, one of the values $[\![\phi_1]\!](s)$ and $[\![\phi_2]\!](s)$ is the constant 0 or 1. We can then remove the min operator, since we have $\min(0, x) = 0$ and $\min(1, x) = x$ for every $x \in [0,1]$.

The discussion above motivates us to refine the type $Prop_{[0,1]}$ of quantitative propositions to $Prop^{T,U}$ where $T, U \subseteq S$ and $T \cap U = \emptyset$. Intuitively, the type $Prop^{T,U}$ is a type for values $v \in Prop_{[0,1]}$ such that $v(s) = 0$ for all $s \in T$ and $v(s) = 1$ for all $s \in U$; there is no guarantee on the value of $v(s)$ for $s \in S \setminus (T \cup U)$. The syntax of refined types is given by:

$$\sigma ::= \kappa \mid Prop_{\{0,1\}} \qquad\qquad \kappa ::= Prop^{T,U} \mid Prop^{T,U} \to \kappa$$

where T and U range over the subsets of S satisfying $T \cap U = \emptyset$. Note that each type $\kappa \neq Prop_{\{0,1\}}$ can be expressed as $Prop^{T_1,U_1} \to Prop^{T_2,U_2} \to \cdots \to Prop^{T_k,U_k} \to Prop^{T,U}$ where $k \geq 0$. The formal definition of the semantics of types is given later.

We restrict PHFL formulas to those given by:

$$\psi ::= [\phi]_J \qquad\qquad \phi ::= \{p\} \mid x \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \bigcirc \phi \mid \mu x.\phi \mid \lambda x.\phi \mid \phi_1\,\phi_2$$

and further restrict them by using the typing rules in Figure 4. In the figure, the type environment $\mathcal{K}$ maps each variable to a type in the set ranged over by κ. The operator $[\cdot]$ has been restricted to the top-level, and the operators $\Diamond, \Box$ and ν have been removed. Note that ψ is a qualitative formula and ϕ is a quantitative formula.

A key rule is for conjunctions. Note that $[\![\phi_1 \wedge \phi_2]\!](s) = 0$ if either $[\![\phi_1]\!](s) = 0$ or $[\![\phi_2]\!](s) = 0$ holds; hence $s \in T_1 \cup T_2$ implies $[\![\phi_1 \wedge \phi_2]\!](s) = 0$. Note also that $[\![\phi_1 \wedge \phi_2]\!](s) = 1$ if both $[\![\phi_1]\!](s) = 1$ *and* $[\![\phi_2]\!](s) = 1$ hold. Thus, $s \in U_1 \cap U_2$ implies $[\![\phi_1 \wedge \phi_2]\!](s) = 1$. This is why $\phi_1 \wedge \phi_2$ has type $Prop^{T_1 \cup T_2, U_1 \cap U_2}$. The extra condition $T_1 \cup U_1 \cup T_2 \cup U_2 = S$ requires that, for each state s, either $[\![\phi_1]\!](s)$ or $[\![\phi_2]\!](s)$ is the constant 0 or 1; recall the earlier discussion on a sufficient condition for the semantics of an order-1 formula to be affine. The rule for disjunctions is analogous.

The following lemma states that a formula that is well-typed in $\mathcal{T}_M$ is also well-typed in the original PHFL type system.

$$\frac{\mathcal{K} \vdash_M \phi : Prop^{T,U} \qquad T' \subseteq T \qquad U' \subseteq U}{\mathcal{K} \vdash_M \phi : Prop^{T',U'}}$$

$$\frac{}{\mathcal{K} \vdash_M \{p\} : Prop^{\overline{\rho_{AP}(p)}, \rho_{AP}(p)}}$$

$$\frac{\mathcal{K} \vdash_M \phi_1 : Prop^{T_1,U_1} \qquad \mathcal{K} \vdash_M \phi_2 : Prop^{T_2,U_2} \qquad T_1 \cup U_1 \cup T_2 \cup U_2 = S}{\mathcal{K} \vdash_M \phi_1 \wedge \phi_2 : Prop^{(T_1 \cup T_2),(U_1 \cap U_2)}}$$

$$\frac{\mathcal{K} \vdash_M \phi_1 : Prop^{T_1,U_1} \qquad \mathcal{K} \vdash_M \phi_2 : Prop^{T_2,U_2} \qquad T_1 \cup U_1 \cup T_2 \cup U_2 = S}{\mathcal{K} \vdash_M \phi_1 \vee \phi_2 : Prop^{(T_1 \cap T_2),(U_1 \cup U_2)}}$$

$$\frac{}{\mathcal{K}, X : \kappa \vdash_M X : \kappa} \qquad \frac{\mathcal{K} \vdash_M \phi : Prop^{T,U}}{\mathcal{K} \vdash_M [\phi]_J : Prop_{\{0,1\}}} \qquad \frac{\mathcal{K} \vdash_M \phi : Prop^{T,U}}{\mathcal{K} \vdash_M \bigcirc \phi : Prop^{\emptyset,\emptyset}}$$

$$\frac{\mathcal{K}, X : \kappa \vdash_M \phi : \kappa}{\mathcal{K} \vdash_M \mu X.\phi : \kappa} \qquad \qquad \frac{\mathcal{K}, X : \kappa_1 \vdash_M \phi : \kappa_2}{\mathcal{K} \vdash_M \lambda X.\phi : \kappa_1 \to \kappa_2}$$

$$\frac{\mathcal{K} \vdash_M \phi_0 : Prop^{T_1,U_1} \to \cdots \to Prop^{T_k,U_k} \to Prop^{T,U} \qquad \mathcal{K} \vdash_M \phi_i : Prop^{T_i,U_i} \; (1 \le i \le k)}{\mathcal{K} \vdash_M \phi_0 \, \phi_1 \, \dots \, \phi_k : Prop^{T,U}}$$

■ **Figure 4** Type Derivation Rules for the PHFL Subclass. Here $\overline{X}$ means the complement $S \setminus X$.

▶ **Lemma 26.** *Let ϕ be a PHFL formula such that $\mathcal{K} \vdash_M \phi : \kappa$ in $\mathcal{T}_M$. Define the translation from the set of types in $\mathcal{T}_M$ to the set of types in PHFL by*

$$tr(Prop_{\{0,1\}}) = Prop_{\{0,1\}} \qquad tr(Prop^{T,U}) = Prop_{[0,1]} \qquad tr(\kappa_1 \to \kappa_2) = tr(\kappa_1) \to tr(\kappa_2)$$

and the translation of type environment $\mathcal{K}$ by $(tr(\mathcal{K}))(x) = tr(\mathcal{K}(x))$. Then we have $tr(\mathcal{K}) \vdash \phi : tr(\kappa)$.

The lemma above can be proved by induction on the structure of ϕ. Using the lemma, we can define the semantics of a type judgment of the type system $\mathcal{T}_M$ by $[\![\mathcal{K} \vdash_M \phi : \kappa]\!]_M = [\![tr(\mathcal{K}) \vdash \phi : tr(\kappa)]\!]_M$. As before, we often omit the type environment, the derived type and the subscript of the Markov chain in the notation of the semantics.

▶ **Example 27.** Let $p_1, p_2, p_3 \in AP$ be atomic propositions satisfying $\rho_{AP}(p_2) \cap \rho_{AP}(p_3) = \emptyset$. Consider the formula $\phi = \bigcirc((\{p_2\} \wedge \bigcirc\{p_1\}) \vee (\{p_3\} \wedge \bigcirc\{p_1\}))$. For each $s \in S$, the value $[\![\phi]\!](s)$ represents the probability that a two-step transition starting from s reaches a state satisfying p_1 through a state satisfying p_2 or p_3. We can derive $\emptyset \vdash_M \phi : Prop^{\emptyset,\emptyset}$ as follows. First, $\{p_1\}, \{p_2\}$, and $\{p_3\}$ have types $Prop^{\overline{\rho_{AP}(p_1)}, \rho_{AP}(p_1)}$, $Prop^{\overline{\rho_{AP}(p_2)}, \rho_{AP}(p_2)}$, and $Prop^{\overline{\rho_{AP}(p_3)}, \rho_{AP}(p_3)}$. It follows that $\{p_2\} \wedge \bigcirc\{p_1\}$ and $\{p_3\} \wedge \bigcirc\{p_1\}$ have types $Prop^{\overline{\rho_{AP}(p_2)}, \emptyset}$ and $Prop^{\overline{\rho_{AP}(p_3)}, \emptyset}$. Since $\overline{\rho_{AP}(p_2)} \cup \overline{\rho_{AP}(p_3)} = \overline{\rho_{AP}(p_2) \cap \rho_{AP}(p_3)} = \overline{\emptyset} = S$, the formula $(\{p_2\} \wedge \bigcirc\{p_1\}) \vee (\{p_3\} \wedge \bigcirc\{p_1\})$ has type $Prop^{\overline{\rho_{AP}(p_2)} \cap \overline{\rho_{AP}(p_3)}, \emptyset}$, from which we obtain $\emptyset \vdash_M \phi : Prop^{\emptyset,\emptyset}$. Note that the condition $L(p_2) \cap L(p_3) = \emptyset$ was crucial in the type derivation above. ◀

We have the following two theorems. The former one states the decidability result, and the latter one states that the restricted subclass of the PHFL model-checking problems is reasonably expressive. Proofs are found in Appendix B.

▶ **Theorem 28.** *Let M be a Markov chain, and ψ be a PHFL formula satisfying $\vdash_M \psi : Prop_{\{0,1\}}$. Then it is decidable whether $M \models \psi$.*

▶ **Theorem 29.** *There exists an algorithm that takes a recursive Markov chain R and a rational number r as input, and outputs an order-1 PHFL formula ϕ_R and a Markov chain M_R such that $\vdash_{M_R} [\phi_R]_{\geq r} : Prop_{\{0,1\}}$, and the termination probability of R is no less than r if and only if $M_R \models [\phi_R]_{\geq r}$.*

6   Related Work

As mentioned in Section 1, PHFL can be regarded as a probabilistic extension of the higher-order fixpoint logic, and as a higher-order extension of the μ^p-calculus. We thus compare our work with previous studies on (non-probabilistic) higher-order fixpoint logic and those on (non-higher-order) probabilistic logics. As already mentioned, for (non-probabilistic) HFL, model checking of finite-state systems is known to be decidable [22], and k-EXPTIME complete [1]. This is in a sharp contrast with our result that PHFL model checking is highly undecidable (both Π_1^1-hard and Σ_1^1-hard) even at order 1. As for studies on probabilistic logics, besides the μ^p-calculus, there are other probabilistic extensions of the modal μ-calculus [19, 9, 17]. To our knowledge, however, ours is the first *higher-order* and probabilistic extension of the modal μ-calculus.

Recently, Kobayashi et al. [10] introduced PHORS, a probabilistic extension of higher-order recursion schemes (HORS), which can also be viewed as a higher-order extension of recursive Markov chains (or probabilistic pushdown systems), and proved that the almost sure termination problem is undecidable. Although the problem setting is quite different (in our work, the *logic* is higher-order whereas the *system* to be verified is higher-order in their work), our encoding of the μ-arithmetic has been partially inspired by their undecidability proof; they also represented a natural number n as the probability $\frac{1}{2^n}$.

7   Conclusion

We have introduced PHFL, a probabilistic logic which can be regarded as both a probabilistic extension of HFL and a higher-order extension of the probabilistic logic μ^p-calculus. We have shown that the model-checking problem for PHFL for a finite Markov chain is undecidable for the μ-only and order-1 fragment. We have also shown that the model-checking problem for the full order-1 fragment of PHFL is Π_1^1-hard and Σ_1^1-hard. As positive results, we have introduced a decidable subclass of the PHFL model-checking problems, and showed that the termination problem of Recursive Markov Chains can be encoded in the subclass.

Finding an upper bound of the hardness of the PHFL model-checking problem is left for future work. It is also left for future work to find a larger, more natural decidable class of PHFL model-checking problems.

──── **References** ────

1   Roland Axelsson, Martin Lange, and Rafal Somla. The complexity of model checking higher-order fixpoint logic. *Logical Methods in Computer Science*, 3(2), 2007. `doi:10.2168/LMCS-3(2:7)2007`.

2   Pablo F. Castro, Cecilia Kilmurray, and Nir Piterman. Tractable probabilistic mu-calculus that expresses probabilistic temporal logics. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, volume 30 of *LIPIcs*, pages 211–223. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. `doi:10.4230/LIPIcs.STACS.2015.211`.

3 Kousha Etessami and Mihalis Yannakakis. Recursive markov chains, stochastic grammars, and monotone systems of nonlinear equations. *J. ACM*, 56(1):1:1–1:66, 2009. `doi:10.1145/1462153.1462154`.

4 Nathanaël Fijalkow. Undecidability results for probabilistic automata. *SIGLOG News*, 4(4):10–17, 2017. URL: `https://dl.acm.org/citation.cfm?id=3157833`.

5 Nathanaël Fijalkow, Hugo Gimbert, Edon Kelmendi, and Youssouf Oualhadj. Deciding the value 1 problem for probabilistic leaktight automata. *Logical Methods in Computer Science*, 11(2), 2015. `doi:10.2168/LMCS-11(2:12)2015`.

6 Hugo Gimbert and Youssouf Oualhadj. Probabilistic automata on finite words: Decidable and undecidable problems. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, volume 6199 of *Lecture Notes in Computer Science*, pages 527–538. Springer, 2010. `doi:10.1007/978-3-642-14162-1_44`.

7 Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Asp. Comput.*, 6(5):512–535, 1994. `doi:10.1007/BF01211866`.

8 David Harel. Effective transformations on infinite trees, with applications to high undecidability, dominoes, and fairness. *J. ACM*, 33(1):224–248, 1986. `doi:10.1145/4904.4993`.

9 Michael Huth and Marta Z. Kwiatkowska. Quantitative analysis and model checking. In *Proceedings, 12th Annual IEEE Symposium on Logic in Computer Science, Warsaw, Poland, June 29 - July 2, 1997*, pages 111–122. IEEE Computer Society, 1997. `doi:10.1109/LICS.1997.614940`.

10 Naoki Kobayashi, Ugo Dal Lago, and Charles Grellois. On the termination problem for probabilistic higher-order recursive programs. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–14. IEEE, 2019. `doi:10.1109/LICS.2019.8785679`.

11 Naoki Kobayashi, Étienne Lozes, and Florian Bruse. On the relationship between higher-order recursion schemes and higher-order fixpoint logic. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 246–259. ACM, 2017. URL: `http://dl.acm.org/citation.cfm?id=3009854`.

12 Naoki Kobayashi, Takeshi Tsukada, and Keiichi Watanabe. Higher-order program verification via HFL model checking. In Amal Ahmed, editor, *Programming Languages and Systems - 27th European Symposium on Programming, ESOP 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings*, volume 10801 of *Lecture Notes in Computer Science*, pages 711–738. Springer, 2018. `doi:10.1007/978-3-319-89884-1_25`.

13 Dexter Kozen. Results on the propositional mu-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983. `doi:10.1016/0304-3975(82)90125-6`.

14 Marta Z. Kwiatkowska, Gethin Norman, and David Parker. PRISM 4.0: Verification of probabilistic real-time systems. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*, pages 585–591. Springer, 2011. `doi:10.1007/978-3-642-22110-1_47`.

15 Étienne Lozes. A type-directed negation elimination. In Ralph Matthes and Matteo Mio, editors, *Proceedings Tenth International Workshop on Fixed Points in Computer Science, FICS 2015, Berlin, Germany, September 11-12, 2015*, volume 191 of *EPTCS*, pages 132–142, 2015. `doi:10.4204/EPTCS.191.12`.

16 Robert S. Lubarsky. mu-definable sets of integers. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89), Pacific Grove, California, USA, June 5-8, 1989*, pages 343–352. IEEE Computer Society, 1989. `doi:10.1109/LICS.1989.39189`.

17   Matteo Mio and Alex Simpson. Łukasiewicz mu-calculus. In David Baelde and Arnaud Carayol, editors, *Proceedings Workshop on Fixed Points in Computer Science, FICS 2013, Turino, Italy, September 1st, 2013*, volume 126 of *EPTCS*, pages 87–104, 2013. `doi:10.4204/EPTCS.126.7`.

18   Yo Mitani, Naoki Kobayashi, and Takeshi Tsukada. A probabilistic higher-order fixpoint logic, 2020. A full version, available from `http://www.kb.is.s.u-tokyo.ac.jp/~mitaniyo/papers/fscd2020-long.pdf`.

19   Carroll Morgan and Annabelle McIver. A probabilistic temporal calculus based on expectations. In *Proc. Formal Methods Pacific*, pages 4–22. Springer, 1997.

20   Azaria Paz. *Introduction to probabilistic automata*. Academic Press, 1971.

21   Michael O Rabin. Probabilistic automata. *Information and control*, 6(3):230–245, 1963.

22   Mahesh Viswanathan and Ramesh Viswanathan. A higher order modal fixed point logic. In Philippa Gardner and Nobuko Yoshida, editors, *CONCUR 2004 - Concurrency Theory, 15th International Conference, London, UK, August 31 - September 3, 2004, Proceedings*, volume 3170 of *Lecture Notes in Computer Science*, pages 512–528. Springer, 2004. `doi:10.1007/978-3-540-28644-8_33`.

Appendix

A   Proof of Theorem 24

We prove the theorem by induction on the structure of ϕ. In this proof, we omit the subscript M of $[\![-]\!]_M$ for simplicity. We discuss only the main cases; see [18] for more details.

- Case $\phi = X$.

  We have $tr(\phi) = X$ and $[\![\phi]\!]_\mu(\theta) = \theta(X) \sim_{\Gamma(X)} \rho(X) = [\![tr(\phi)]\!](\rho)$.

- Case $\phi = Z$.

  Then $tr(\phi) = \{p'_0\}$ and $A = N$. We have $[\![\phi]\!]_\mu(\theta) = 0 \sim_N (1, 0, 0, 0) = [\![tr(\phi)]\!](\rho)$.

- Case $\phi = S\,t$.

  Let $n = [\![t]\!]_\mu(\theta)$. By the induction hypothesis, we have

  $$[\![tr(t)]\!](\rho) = \left(\frac{1}{2^n},\ 1 - \frac{1}{2^n},\ —,\ — \right).$$

  By the definition of $tr(\phi)$ and calculation, we have

  $$[\![tr(\phi)]\!](\rho) = \left(\frac{1}{2^{n+1}},\ 1 - \frac{1}{2^{n+1}},\ —,\ — \right),$$

  which implies $[\![S\,t]\!]_\mu(\theta) = n + 1 \sim_N [\![tr(\phi)]\!](\rho)$.

- Case $\phi = (s \le t)$.

  Let $n = [\![s]\!]_\mu(\theta)$ and $m = [\![t]\!]_\mu(\theta)$. By the induction hypothesis, we have

  $$[\![tr(s)]\!](\rho) = (\frac{1}{2^n},\ ,1 - \frac{1}{2^n},\ —,\ —)$$
  $$[\![tr(t)]\!](\rho) = (\frac{1}{2^m},\ ,1 - \frac{1}{2^m},\ —,\ —).$$

  By the definition of $tr(s \le t)$ and calculation, we have

  $$[\![tr(s \le t)]\!](\rho) = \begin{cases} (1, —, —, —) & (\text{if } \frac{1}{2} \times \left(\frac{1}{2^n} + (1 - \frac{1}{2^m}) \right) \ge \frac{1}{2}, \text{ i.e. if } n \le m) \\ (0, —, —, —) & (\text{if } \frac{1}{2} \times \left(\frac{1}{2^n} + (1 - \frac{1}{2^m}) \right) < \frac{1}{2}, \text{ i.e., if } n > m). \end{cases}$$

  Thus, we have $[\![s \le t]\!]_\mu(\theta) \sim_\Omega [\![tr(s \le t)]\!](\rho)$ as required.

- Case $\phi = \lambda X.\psi$. In this case, A is of the form $B \to T$, with $\Gamma, X : B \vdash_\mu \psi : T$. By the induction hypothesis, ψ satisfies

$$[\![\Gamma, X : B \vdash_\mu \psi : T]\!]_\mu(\theta[X \mapsto v]) \sim_{B \to T} [\![tr(\Gamma, X : B) \vdash tr(\psi) : tr(T)]\!](\rho[X \mapsto u])$$

for any $v \in [\![B]\!]$ and $u \in [\![tr(B)]\!]$ such that $v \sim_B u$.
Therefore, by the definition of $\sim_{B \to T}$, we have

$$[\![\Gamma \vdash_\mu \phi : B \to T]\!]_\mu(\theta) \sim_{B \to T} [\![tr(\Gamma) \vdash tr(\phi) : tr(B \to T)]\!](\rho)$$

as required.

- Case $\phi = \psi_1 \psi_2$. We have $A = T$, with $\Gamma \vdash_\mu \psi_1 : B \to T$ and $\Gamma \vdash_\mu \psi_2 : B$.
By the induction hypothesis, we have $[\![\psi_1]\!]_\mu(\theta) \sim_{B \to T} [\![tr(\psi_1)]\!](\rho)$ and $[\![\psi_2]\!]_\mu(\theta) \sim_B [\![tr(\psi_2)]\!](\rho)$. Therefore by the definition of $\sim_{B \to T}$, we have

$$\begin{aligned}[\![\psi_1 \psi_2]\!]_\mu(\theta) &= [\![\psi_1]\!]_\mu(\theta) ([\![\psi_2]\!]_\mu(\theta)) \\ &\sim_A [\![tr(\psi_1)]\!](\rho) ([\![tr(\psi_2)]\!](\rho)) \\ &= [\![tr(\psi_1 \psi_2)]\!](\rho)\end{aligned}$$

as desired.

- Case $\phi = \mu X.\psi$.
In this case, $A = T$, with $\Gamma, X : T \vdash_\mu \psi : T$. By the induction hypothesis, for any $v \in [\![T]\!]_\mu$ and $u \in [\![tr(T)]\!]$ such that $v \sim_T u$, we have

$$[\![\psi]\!]_\mu(\theta[X \mapsto v]) \sim_T [\![tr(\psi)]\!](\rho[X \mapsto u]).$$

Since $tr(\mu X.\psi) = \mu X.tr(\psi)$, it suffices to show:

$$[\![\mu X.\psi]\!]_\mu(\theta) \sim_T [\![\mu X.tr(\psi)]\!](\rho).$$

Let $\mathcal{F} : [\![T]\!]_\mu \to [\![T]\!]_\mu$ and $\mathcal{G} : [\![tr(T)]\!] \to [\![tr(T)]\!]$ be the functions defined by:

$$\mathcal{F}(v) := [\![\psi]\!]_\mu(\theta[X \mapsto v]) \qquad \mathcal{G}(u) := [\![tr(\psi)]\!](\rho[X \mapsto u]).$$

By the reasoning above, we have $\mathcal{F} \sim_{T \to T} \mathcal{G}$. By the definitions of the semantics, we have $[\![\mu X.\psi]\!]_\mu(\theta) = \mathrm{LFP}(\mathcal{F})$ and $[\![\mu X.\psi]\!](\rho) = \mathrm{LFP}(\mathcal{G})$. Then there exists an ordinal α such that

$$\mathrm{LFP}(\mathcal{F}) = \mathcal{F}^\alpha(\bot_T) \qquad \text{and} \qquad \mathrm{LFP}(\mathcal{G}) = \mathcal{G}^\alpha(\bot_{tr(T)}),$$

where $f^\beta(x)$ is defined by $f^0(x) = x$, $f^{\beta+1} = f(f^\beta(x))$, and $f^\beta = \bigvee_{\gamma < \beta} f^\gamma(x)$ if β is a limit ordinal. We shall prove by (transfinite) induction on β that $\mathcal{F}^\beta(\bot_T) \sim_T \mathcal{G}^\beta(\bot_{tr(T)})$, which would imply

$$\mathrm{LFP}(\mathcal{F}) = \mathcal{F}^\alpha(\bot_T) \sim_T \mathcal{G}^\alpha(\bot_{tr(T)}) = \mathrm{LFP}(\mathcal{G})$$

as required.
The base case $\mathcal{F}^0(\bot_T) = \bot_T \sim_T \bot_{tr(T)} = \mathcal{G}^0(\bot_{tr(T)})$ follows by a straightforward induction on the structure of T. The case where β is a successor ordinal follows immediately from the induction hypothesis and $\mathcal{F} \sim_{T \to T} \mathcal{G}$. If β is a limit ordinal, then

$$\mathcal{F}^\beta(\bot_T) = \bigvee_{\gamma < \beta} \mathcal{F}^\gamma(\bot_T) \qquad \text{and} \qquad \mathcal{G}^\beta(\bot_T) = \bigvee_{\gamma < \beta} \mathcal{G}^\gamma(\bot_T).$$

By the induction hypothesis (of the transfinite induction),

$$\mathcal{F}^\gamma(\bot_T) \sim_T \mathcal{G}^\gamma(\bot_T)$$

for every $\gamma < \beta$. Since $\sim_T$ is preserved by the least upper-bound operation (which can be proved by an easy induction on T), we have

$$\mathcal{F}^\beta(\bot_T) \sim_T \mathcal{G}^\beta(\bot_T)$$

as required.

▬ Case $\phi = \nu X.\psi$. Similar to the case for $\phi = \mu X.\psi$ above.

B Proofs for Section 5

B.1 Proof of Theorem 28

We first give a matrix representation for each value of the semantics of types of $\mathcal{T}_M$. As mentioned before, we fix the underlying Markov chain M with the set of states $S = \{s_1, s_2, \ldots, s_n\}$. Henceforth, we identify the set of functions $S \to [0,1]$ with the set $[0,1]^n$.

We first give the formal definition of the semantics of types in $\mathcal{T}_M$. As explained in Section 5, the values of function types are restricted to affine functions.

▶ **Definition 30.** *For each type $\kappa \neq Prop_{\{0,1\}}$ in the type system $\mathcal{T}_M$, we define its semantics* $[\![\kappa]\!] = (D_\kappa, \sqsubseteq_\kappa)$ *by induction on κ as follows.*
1. *For $\kappa = Prop^{T,U}$, D_κ is the set $\{v \in [\![Prop_{[0,1]}]\!] \mid \forall s \in T.v(s) = 0, \forall s \in U.v(s) = 1\}$ and $f_1 \sqsubseteq_\kappa f_2$ if and only if $\forall s \in S.f_1(s) \leq f_2(s)$.*
2. *For $\kappa = Prop^{T_1,U_1} \to Prop^{T_2,U_2} \to \ldots Prop^{T_k,U_k} \to Prop^{T,U}$ ($k \geq 1$), D_κ is the set of affine functions $f : ([0,1]^n)^k \to [0,1]^n$ which belong to $[\![tr(\kappa)]\!]$ (with the identification between $[0,1]^S$ and $[0,1]^n$), and $f_1 \sqsubseteq_\kappa f_2$ if and only if for every tuple $(v_1, v_2, \ldots, v_k)$ in $[\![Prop^{T_1,U_1}]\!] \times \cdots \times [\![Prop^{T_k,U_k}]\!]$, the relation $f_1 \, v_1 \, v_2 \, \ldots \, v_k \sqsubseteq f_2 \, v_1 \, v_2 \, \ldots \, v_k$ holds.*

We now give a matrix representation $Mat_\kappa(f)$ for each type $\kappa \neq Prop_{\{0,1\}}$ of $\mathcal{T}_M$ and $f \in [\![\kappa]\!]$. For $v \in [\![Prop^{T,U}]\!]$, we write $Vec(v)$ for the $1 \times n$ matrix $(v(s_1) \, v(s_2) \, \ldots \, v(s_n))$.

▶ **Definition 31** (Matrix Representation). *For an element $f \in [\![\kappa]\!]$ where $\kappa = Prop^{T_1,U_1} \to Prop^{T_2,U_2} \to \ldots Prop^{T_k,U_k} \to Prop^{T,U}$ ($k \geq 0$), its matrix representation $Mat_\kappa(f)$ is the (unique) matrix $M = (m_{ij})_{ij}$ of size $(n+1) \times (kn+1)$ satisfying the following conditions.*
1. *For every tuple $(v_1, v_2, \ldots, v_k)$ where $v_i \in [\![Prop^{T_i,U_i}]\!]$ ($1 \leq i \leq k$), the following equality holds.*

$$M \begin{pmatrix} 1 & Vec(v_1) & Vec(v_2) & \ldots & Vec(v_k) \end{pmatrix}^\top = \begin{pmatrix} 1 & Vec(f \, v_1 \, v_2 \, \ldots \, v_k) \end{pmatrix}^\top$$

2. *For each i ($1 \leq i \leq k$), $s_j \in T_i \cup U_i$, and ℓ ($1 \leq \ell \leq n+1$), the equality $m_{\ell,(i-1)n+j+1} = 0$ holds.*
3. *For each j ($1 \leq j \leq kn+1$) and $s_i \in T$, the equality $m_{i+1,j} = 0$ holds. Also, for each j ($2 \leq j \leq kn+1$) and $s_i \in U$, the equalities $m_{i+1,1} = 1$ and $m_{i+1,j} = 0$ hold.*
4. *For each j ($2 \leq j \leq kn+1$), the equalities $m_{11} = 1$ and $m_{1j} = 0$ hold.*

The existence of M satisfying the first condition is obvious from the assumption that f is affine. The other conditions are imposed to ensure the uniqueness of M. We often omit the type annotation and just write Mat for Mat_κ.

When $k = 0$, the matrix representation $Mat(v)$ for $v \in [\![Prop^{T,U}]\!]$ is given by $Mat(v) = \begin{pmatrix} 1 & Vec(v) \end{pmatrix}^\top$.

Given a 2-dimensional matrix M, we write M_{ij} for the (i,j)-entry of M. The order $\leq$ between two matrices M and M' of the same size $(n+1) \times (kn+1)$ is defined as the pointwise order, i.e., $M \leq M' \overset{\text{def}}{\iff} \forall 1 \leq i \leq n+1, 1 \leq j \leq kn+1.M_{ij} \leq M'_{ij}$.

We define the matrix semantics of a type κ by $[\![\kappa]\!]_{Mat} = Mat([\![\kappa]\!]) = \{Mat(f) \mid f \in [\![\kappa]\!]\}$. For a type environment $\mathcal{K}$, its matrix semantics $[\![\mathcal{K}]\!]_{Mat}$ is the set of maps η_{Mat} satisfying $dom(\eta_{Mat}) = dom(\mathcal{K})$ and $\eta_{Mat}(X) \in [\![\mathcal{K}(X)]\!]_{Mat}$ for all $X \in dom(\mathcal{K})$. For a type derivation $\mathcal{K} \vdash_M \phi : \kappa$, we write $[\![\mathcal{K} \vdash_M \phi : \kappa]\!]_{Mat}$ for the map from $[\![\mathcal{K}]\!]_{Mat}$ to $Mat([\![\kappa]\!])$ defined by:

$$[\![\mathcal{K} \vdash_M \phi : \kappa]\!]_{Mat}(\eta_{Mat}) = Mat([\![\mathcal{K} \vdash_M \phi : \kappa]\!](\eta))$$

Here, η satisfies $\eta(X) = Mat^{-1}(\eta_{Mat}(X))$ for each $X \in dom(\mathcal{K})$. For the well-definedness of $[\![\mathcal{K} \vdash_M \phi : \kappa]\!]_{Mat}$ above, it must be the case that $[\![\mathcal{K} \vdash_M \phi : \kappa]\!](\eta) \in [\![\kappa]\!]$, which can be easily checked.

▶ **Example 32.** Let $M = (S, P, \rho_{AP}, s_{\text{in}})$ be a Markov chain such that
- $S = \{s_1, s_2, s_3\}$,
- P satisfies $P(s_1, s_2) = 0.4$, $P(s_1, s_3) = 0.6$, $P(s_2, s_1) = P(s_3, s_1) = 1$ and $P(s_i, s_j) = 0$ for all the other pairs $(s_i, s_j) \in S \times S$,
- there exist $p_1, p_2, p_3 \in AP$ such that $\rho_{AP}(p_i) = \{s_i\}$ for each $i \in \{1, 2, 3\}$, and
- $s_{\text{in}} = s_1$

Let us consider the the formula $\phi = \lambda X. \bigcirc (((\{p_1\} \vee \{p_2\}) \wedge \bigcirc X) \vee (\{p_3\} \wedge \bigcirc X))$. The matrix representation of the semantics of ϕ is

$$\left[\!\left[\phi : Prop^{\{s_3\}, \emptyset} \to Prop^{\emptyset, \emptyset} \right]\!\right]_{Mat} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.4 & 0 \\ 0 & 0 & 0.4 & 0 \end{pmatrix}$$

$$\left[\!\left[\phi : Prop^{\emptyset, \{s_3\}} \to Prop^{\emptyset, \emptyset} \right]\!\right]_{Mat} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0.6 & 0 & 0.4 & 0 \\ 0.6 & 0 & 0.4 & 0 \end{pmatrix}$$

Note that the matrix representation $[\![\phi : \kappa]\!]_{Mat}$ depends on the type κ. ◀

Henceforth, we assume that a formula is given in the form of a *hierarchical equation system* (HES) [11]: $\mathcal{E} = (X_1 =_\mu \phi_1; X_2 =_\mu \phi_2; \ldots; X_k =_\mu \phi_k)$, where ϕ_i does not contain fixpoint operators. The corresponding PHFL formula $toPHFL(\mathcal{E})$ is given by:

$$toPHFL(X =_\mu \phi) = \mu X.\phi \qquad toPHFL(\mathcal{E}; X =_\mu \phi) = toPHFL([\mu X.\phi/X]\mathcal{E}).$$

For an HES $\mathcal{E} = (X_1 =_\mu \phi_1; X_2 =_\mu \phi_2; \ldots; X_k =_\mu \phi_k)$, we define the fixpoint equation $toFP(\mathcal{E})$ by:

$$toFP(\mathcal{E}) := (M_1 = [\![\phi_1]\!]_{Mat}(\eta_{Mat}); M_2 = [\![\phi_2]\!]_{Mat}(\eta_{Mat}); \ldots; M_k = [\![\phi_k]\!]_{Mat}(\eta_{Mat})).$$

Here, η_{Mat} maps each variable X_i to the matrix M_i that contains variables that represent unknown values. The following theorem guarantees that the semantics of the formula can be effectively computed by using the matrix representation (see [18] for details).

▶ **Theorem 33.** *Let ϕ be a formula whose HES form is $(X_1 =_\mu \phi_1; X_2 =_\mu \phi_2; \ldots; X_k =_\mu \phi_k)$. Suppose $\emptyset \vdash_M \phi : Prop^{T,U}$. Let $(M_1 = m_1; M_2 = m_2; \ldots; M_k = m_k)$ be the least solution of the fixpoint equation $toFP(\mathcal{E})$, and v be the entry of the matrix m_1 which corresponds to the initial state s_{in} of the Markov chain. Then we have $[\![\phi]\!](s_{in}) = v$.*

Since a fixpoint equation on reals can be solved in PSPACE [3], we have the following result as a corollary of this theorem, which subsumes Theorem 28.

▶ **Corollary 34.** *Let M be a Markov chain. If $\emptyset \vdash_M \psi : Prop_{\{0,1\}}$, then whether $M \models \psi$ is decidable in space polynomial in $n(d + s)$, where n is the number of the states of M, d is the size of ψ and s is the sum of the arities of the order-1 variables bound by fixpoint operators.*

B.2   Proof of Theorem 29

Recursive Markov chains can be encoded as order-1 probabilistic HORS (PHORS) [10]. Thus, in this section, we show how the termination problem for PHORS can be encoded into a PHFL model-checking problem in the restricted class.

We transform an order-1 PHORS $\mathcal{G}$ to a pair of a Markov chain M and a PHFL formula ϕ typable in $\mathcal{T}_M$ where, for any $0 \le r \le 1$, the value $[\![\phi]_{\ge r}]\!](s_{\text{in}})$ over the Markov chain M equals 1 if and only if the termination probability of $\mathcal{G}$ is no less than r.

In the rest of this section we follow the notational conventions and definitions about PHORS and higher-order fixpoint equations from [10].

We first fix an order-1 PHORS $\mathcal{G} = (\mathcal{N}, \mathcal{R}, S)$ where $dom(\mathcal{N}) = \{S, F_1, F_2, \ldots, F_m\}$, $\mathcal{N}(F_i) = \underbrace{\mathsf{o} \to \mathsf{o} \to \cdots \to \mathsf{o}}_{k_i} \to \mathsf{o}$ (which is denoted by $\mathsf{o}^{k_i} \to \mathsf{o}$) and $\mathcal{R}$ is such that $F_i \, X_1 \, X_2 \, \ldots \, X_{k_i} = t_{i,L} \oplus_{p_i} t_{i,R}$ for each $1 \le i \le m$ and $S = t_S \oplus_1 \Omega$. Without loss of generality, we assume $p_1 \le p_2 \le \cdots \le p_m$. We write $\mathcal{P}(\mathcal{G})$ for the termination probability of the PHORS $\mathcal{G}$.

We define the Markov chain $M = (S, P, \rho_{AP}, s_{\text{in}})$ as follows.

- $S = \{s_0, s_1, \ldots, s_{m+1}\}$,
- P satisfies $P(s_0, s_1) = p_1$, $P(s_0, s_i) = p_i - p_{i-1}$ for $2 \le i \le m$, $P(s_0, s_{m+1}) = 1 - p_m$, $P(s_i, s_0) = 1$ for $1 \le i \le m+1$ and $P(s_i, s_j) = 0$ otherwise,
- $\rho_{AP}(P_i) = \{s_i\}$ for each $0 \le i \le m+1$, and
- $s_{\text{in}} = s_0$.

Before defining the formula ϕ, we define, for each applicative term t of PHORS, the PHFL formula $\langle t \rangle$ by induction on the structure of t as follows.

$$\langle halt \rangle = \{P_0\} \qquad\qquad \langle \Omega \rangle = \{\text{false}\}$$
$$\langle X \rangle = X \qquad\qquad \langle F_i \rangle = F_i$$
$$\langle f \, u_1 \, u_2 \, \ldots \, u_m \rangle = \langle f \rangle \, \langle u_1 \rangle \, \langle u_2 \rangle \, \ldots \, \langle u_k \rangle .$$

We also define the formula $br(\phi_L, \phi_R, i)$ for formulas ϕ_L, ϕ_R and an index $1 \le i \le m$ by:

$$br(\phi_L, \phi_R, i) = \{P_0\} \wedge \bigcirc \left(\left((\bigcirc \phi_L) \wedge \left(\bigvee_{1 \le j \le i} \{P_j\} \right) \right) \vee \left((\bigcirc \phi_R) \wedge \left(\bigvee_{i+1 \le j \le m+1} \{P_j\} \right) \right) \right) .$$

Then the desired formula ϕ is given by $\phi = toPHFL(\mathcal{E})$ where $\mathcal{E} = (S =_\mu \langle t_S \rangle \, ; F_1 =_\mu \lambda X_1 . \lambda X_2 . \ldots . \lambda X_{k_1} . br(\langle t_{1,L} \rangle, \langle t_{1,R} \rangle, 1) ; \ldots ; F_m =_\mu \lambda X_1 . \lambda X_2 . \ldots . \lambda X_{k_m} . br(\langle t_{m,L} \rangle, \langle t_{m,R} \rangle, m))$. Then ϕ has type $Prop^{\{s_1, s_2, \ldots, s_{m+1}\}, \emptyset}$ (thus belongs to the decidable subclass), and $\mathcal{P}(\mathcal{G}) = [\![\phi]\!](s_{\text{in}})$ holds; see [18] for details.