THE SET OF FORMULAS OF PrAL$^+$ VALID IN A FINITE STRUCTURE IS UNDECIDABLE

Anna Borowska

Faculty of Computer Science, Bialystok University of Technology, Białystok, Poland

Abstract: We consider a probabilistic logic of programs. In [6] it is proved that the set of formulas of the logic PrAL, valid in a finite structure, is decidable with respect to the diagram of the structure. We add to the language L_P of PrAL a sign $\bigcup$ and a functor lg. Next we justify that the set of formulas of extended logic, valid in a finite at least 2-element structure (for L_P^+) is undecidable.

Keywords: Probabilistic Algorithmic Logic, existential iteration quantifier

1. Introduction

In [6] the Probabilistic Algorithmic Logic PrAL is considered, constructed for expressing properties of probabilistic algorithms understood as iterative programs with two probabilistic constructions $x :=$ **random** and **either**$_p$ ... **or** ... **ro**. In order to describe probabilities of behaviours of programs a sort of variables (interpreted as real numbers) and symbols $+, -, *, 0, 1, <$ (interpreted in the standard way in the ordered field of real numbers) was added to the language L_P of PrAL.

In the paper [5] the changes of information which depend on realizations of probabilistic program was considered. That's why the language L_P was extended by adding the sign $\bigcup$ (called the existential iteration quantifier) and the functor lg (for the one-argument operation of a logarithm with a base 2 interpreted in the real ordered field). The new language was denoted by L_P^+.

The paper [6] contains an effective method of determining probabilities for probabilistic programs interpreted in a finite structure. The effectiveness of the method leads to the decidability of the set of formulas of L_P, valid in a fixed finite structure (provided that we have at our disposal a suitable finite part of the diagram of the structure). Here we shall justify that the set of probabilistic algorithmic formulas of L_P^+,

Advances in Computer Science Research, vol. 10, pp. 5-18, 2013.

valid in an arbitrary, finite at least 2-element structure, is undecidable with respect to its diagram.

We shall start from a presentation of the syntax and the semantics of the language L_P^+. We use the syntax and the semantics of L_P proposed by W. Danko in [6].

2. Syntax and Semantics of L_P^+

A language L_P is an extension of a first-order language L and includes three kinds of well-formed expressions: terms, formulas and programs. As mentioned above, the alphabet of L_P^+ contains two additional elements: the arithmetic one-argument functor lg and the sign $\bigcup$ (the existential iteration quantifier). An interpretation of L_P^+ relies on an interpretation of the first-order language L in a structure $\Im$ (We take into consideration only finite structures. By finite structure we mean a structure with a finite, at least 2-element set A.) and on the standard interpretation of the language $L_{\Re}$ in the ordered field of real numbers (cf. [6]).

The alphabet of the language L_P^+ contains

- a set of constants C_P, which consists of a finite subset $C = \{c_1, \ldots, c_u\}$ of symbols for each element of the set $A = \{a_1, \ldots, a_u\}$, a subset $C_{\Re}$ of real constant symbols and a subset C_L of logical constant symbols,
- an enumerable set $V_P = \{V \cup V_{\Re} \bigcup V_0\}$ of variables, where a subset $V = \{v_0, v_1, \ldots\}$ consists of non-arithmetic individual variables, a subset $V_{\Re} = \{x_0, x_1, \ldots\}$ contains real variables and a subset $V_0 = \{q_0, q_1, \ldots\}$ contains propositional variables,
- a set of signs of relations $\Psi_P = \{\Psi \cup \Psi_{\Re}\}$, where the subset Ψ consists of non-arithmetic predicates and the subset $\Psi_{\Re} = \{<_{\Re}, =_{\Re}\}$ contains arithmetic predicates,
- an enumerable set of functors $\Phi_P = \{\Phi \cup \Phi_{\Re}\}$, which consists of the subset $\Phi_{\Re} = \{+, -, *, \mathrm{lg}\}$ of symbols for arithmetic operations and the subset Φ of symbols for non-arithmetic operations,
- the set $\{\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow\}$ of logical connectives,
- the set {**if**, **then**, **else**, **fi**, **while**, **do**, **od**, **either**, **or**, **ro**, $\mathbf{random}_l$[1] } of symbols for program constructions,
- the set $\{\exists, \forall\}$ of symbols for classical quantifiers (for real variables only),
- the existential iteration quantifier $\bigcup$,

[1] For each probability distribution defined on a set A we generate a different random assignment. We use a number l to distinguish them.

– the set $\{(,)\}$ of auxiliary symbols.

In the language L_P^+ we distinguish two kinds of terms (arithmetic and non-arithmetic), formulas (classical and algorithmic) and programs.

The set of terms $T_P = \{T \cup T_{\mathfrak{R}}\}$ of L_P^+ consists of a subset of non-arithmetic terms T and a subset $T_{\mathfrak{R}}$ of arithmetic terms.

Definition 2.1 The set T of *non-arithmetic terms* is defined as the smallest set of expressions satisfying the following conditions:
– each constant of C and each variable of V belongs to T,
– if $\phi_i \in \Phi$ (ϕ_i – an n_i-argument functor ($n_i \geq 0$)) and $\tau_1, \ldots, \tau_{n_i} \in T$ then an expression $\phi_i(\tau_1, \ldots, \tau_{n_i})$ belongs to T.

Definition 2.2 The set $T_{\mathfrak{R}}$ of *arithmetic terms* is the smallest set such that:
– each constant of $C_{\mathfrak{R}}$ and each real variable of $V_{\mathfrak{R}}$ belongs to $T_{\mathfrak{R}}$,
– if $t_1, t_2 \in T_{\mathfrak{R}}$ then expressions $t_1 + t_2$, $t_1 - t_2$, $t_1 * t_2$, $\lg t_1$ belong to $T_{\mathfrak{R}}$,
– if α is a formula of L then $P(\alpha)$ belongs to $T_{\mathfrak{R}}$. (We read the symbol P as follows "probability that".)

Definition 2.3 The set F_O of *open formulas* is the smallest set such that:
– if $\tau_1, \ldots, \tau_{m_j} \in T$ and $\psi_j \in \Psi$ (ψ_j – an m_j-argument predicate) then $\psi_j(\tau_1, \ldots, \tau_{m_j}) \in F_O$,
– if $\alpha, \beta \in F_O$ then expressions $\neg\alpha$, $\alpha \vee \beta$, $\alpha \wedge \beta$, $\alpha \Rightarrow \beta$, $\alpha \Leftrightarrow \beta$ belong to F_O.

Definition 2.4 The set Π of all *programs* is defined as the smallest set of expressions satisfying the following conditions:
– each expression of the form $v := \tau$ or $v :=$**random**$_l$, where $v \in V$, $\tau \in T$ is a program,
– if $\gamma \in F_O$ and $M_1, M_2 \in \Pi$ then expressions $M_1; M_2$, **if** γ **then** M_1 **else** M_2 **fi**, **while** γ **do** M_1 **od**, **either**$_p$ M_1 **or** M_2 **ro** (p is a real number) are programs.

We establish that in an expression $\bigcup K\alpha$ (where K is a program) the letter α denotes a formula which does not contain any iteration quantifiers.

Definition 2.5 The set F_P of all *formulas* of the language L_P^+ is the smallest extension of the set F_O such that:
– if $t_1, t_2 \in T_{\mathfrak{R}}$ then $t_1 =_{\mathfrak{R}} t_2$, $t_1 <_{\mathfrak{R}} t_2$ belong to F_P,
– if $\alpha, \beta \in F_P$ then the expressions $\neg\alpha$, $\alpha \vee \beta$, $\alpha \wedge \beta$, $\alpha \Rightarrow \beta$, $\alpha \Leftrightarrow \beta$ belong to F_P,

– if $\alpha \in F_P$ and $x \in V_{\Re}$ is a free variable in α then $\exists x\alpha$, $\forall x\alpha$ belong to F_P,
– if $K \in \Pi$ and $\alpha \in F_P$ then $K\alpha$ is a formula of F_P,
– if $K \in \Pi$ and $\alpha \in F_P$ then $\bigcup K\alpha$ belongs to F_P.

A variable x is *free* in a formula α if x is not bounded by any quantifier.

Let L_P^+ be a fixed algorithmic language of the type $< \{n_k\}_{\phi_k \in \Phi_P}, \{m_l\}_{\psi_l \in \Psi_P} >$ and let a relational system $\Im =< A \cup R;\ \{\phi_{k\Im}\}_{\phi_k \in \Phi_P},\ \{\psi_{l\Im}\}_{\psi_l \in \Psi_P} >$ (which consists of the fixed, finite, at least 2-element set A, the set R of real numbers, operations and relations) be a fixed data structure for L_P^+.

We interpret non-arithmetic individual variables of L_P^+ as elements of A. Real variables are interpreted as elements of the set R of real numbers.

Let's denote the set of possible valuations w of non-arithmetic variables by W.

Definition 2.6 By *the interpretation of a non-aritmetic term* τ of L_P in the structure $\Im$ we mean a function $\tau_\Im : W \mapsto A$ which is defined recursively.
– If τ is a variable $v \in V$ then $v_\Im(w) \stackrel{df}{=} w(v)$.
– If τ is of the form $\phi(\tau_1, \ldots, \tau_n)$, where $\tau_1, \ldots, \tau_n \in T$ and $\phi \in \Phi$ is an n-argument functor then $\phi(\tau_1, \ldots, \tau_n)_\Im(w) \stackrel{df}{=} \phi_\Im(\tau_{1\Im}(w), \ldots, \tau_{n\Im}(w))$, where $\tau_{1\Im}(w), \ldots, \tau_{n\Im}(w)$ are defined earlier.

To interpret random assignments (i.e. constructions of the form $v :=$ **random**$_l$) in a probabilistic way we assume that there exists a fixed probability distribution defined on A

$$\rho_l : A \mapsto [0,1], \qquad \sum_{i=1}^{u} \rho_l(a_i) = 1.$$

Definition 2.7 (cf. [6]) A pair $< \Im, \rho >$, where ρ is a set of fixed probability distributions ρ_l defined on A and $\Im$ is a structure for L_P^+, is called *a probabilistic structure*. In this structure we interpret probabilistic programs.

By $\mathcal{M}$ we denote the set of all probability distributions defined on the set W of valuations of non-arithmetic variables such that

$$\mu : W \mapsto [0,1], \qquad \sum_{w_i \in W} \mu(w_i) \leq 1.$$

By S we mean the set of all *states*, i.e. all pairs $s =< \mu, w_{\Re} >$, where μ is a probability distribution of valuations of non-arithmetic variables and $w_{\Re}$ is a valuation of real variables of $V_{\Re}$.

Definition 2.8 (cf. [6]) A probabilistic program K is interpreted in the structure $<\Im,\rho>$ as a partial function transforming the set of states into the set of states

$$K_{<\Im,\rho>} : S \mapsto S.$$

Let $K(v_1,\ldots,v_h)$ represent a fixed program in L_P^+. An arbitrary program K contains only a finite number of non-arithmetic variables. We denote this set of variables by $V = \{v_1,\ldots,v_h\}$. Since $A = \{a_1,\ldots,a_u\}$ is also a finite set, then a set of all possible valuations of program variables will be also finite. We denote it by $\{w_1,\ldots,w_n\}$, where $n = u^h$.

Let's notice that programs do not operate on variables of $V_{\Re}$. Thus we can interpret an arbitrary program K as partial functions transforming probability distributions defined on the set of valuations of program variables (cf. [6])

$$K_{<\Im,\rho>} : \mathcal{M} \mapsto \mathcal{M}.$$

If μ is the input probability distribution of valuations of program variables (input probability distribution for short) then a realization of a program K leads to a new output probability distribution $\mu^{'}$ of valuations of program variables (output probability distribution for short). A distribution μ ($\mu^{'}$) associates with each valuation w of program variables a corresponding probability of its appearance.

The interpretation of program constructions (used in this paper) can be found in the Appendix.

An arithmetic term of the form $\mathrm{P}(\alpha)$ denotes the probability, that the formula α of L is satisfied at a distribution μ (cf. [6])

$[\mathrm{P}(\alpha)]_{\Im}(s) = \sum_{w \in W^{\alpha}} \mu(w)$, where $W^{\alpha} = \{w \in W : \Im, w \models \alpha\}$.

Let $s = <\mu, w_{\Re}>$ be a state and let $s^{'} = <\mu^{'}, w_{\Re}>$ represent the state $s^{'} = K_{<\Im,\rho>}(s)$.

Given below is the interpretation of a formula $K\alpha$ ($\alpha \in F_P$ and $K \in \Pi$).

$$(K\alpha)_{<\Im,\rho>}(s) = \begin{cases} \alpha_{<\Im,\rho>}(s^{'}) & \text{if } K_{<\Im,\rho>}(s) \text{ is defined and } s^{'} = K_{<\Im,\rho>}(s) \\ \text{is not defined otherwise} & \end{cases}$$

The satisfiability of a formula $K\alpha$, where $\alpha \in F_P$ and $K \in \Pi$, is defined in the following way (cf. [6])

$<\Im,\rho>, s \models K\alpha$ iff $<\Im,\rho>, s^{'} \models \alpha$, where $s^{'} = K_{<\Im,\rho>}(s)$.

The next definition establishes the meaning of the existential iteration quantifier ($K \in \Pi$, $\alpha \in F_P$).

$(\bigcup K\alpha)_{<\Im,\rho>}(s) \overset{df}{=} \underset{i\in N}{\text{l.u.b.}} (K^i\alpha)_{<\Im,\rho>}(s).$

We can informally express the formula $\bigcup K\alpha$ in the following way $\alpha \vee K\alpha \vee K^2\alpha \vee \ldots$

The satisfiability of a formula $\bigcup K\alpha$ ($K \in \Pi$, $\alpha \in F_P$) is defined as an infinite alternative of formulas $(K^i\alpha)$ for $i \in N$.

Example 2.10 Now we shall present a formula which contains the iteration quantifier. Let's consider the formula $\beta :\ K_0 \bigcup K\alpha$ such that

K_0: $v_1 := 0$;
K : **if** $(v_1 = 0)$ **then** v_1 :=**random**$_1$; $v_2 := 0$; **else** $v_2 := 1$; **fi**
α: $x = \mathrm{P}(v_1 = 1 \vee v_2 = 0)$

where K_0 and K are programs interpreted in the structure $< \Im, \rho >$ with a 2-element set $A = \{0,1\}$. For a random assignment v_1 :=**random**$_1$ we define the probability distribution $\rho_1 = [0.5, 0.5]$. The set of possible valuations of program variables contains 4 elements: $w_1 = (0,0)$, $w_2 = (0,1)$, $w_3 = (1,0)$, $w_4 = (1,1)$. We carry out computations for the input probability distribution $\mu = [0.25, 0.25, 0.25, 0.25]$. $\mathrm{P}(\gamma)$ denotes the probability that γ is satisfied (at a distribution μ). Let's notice, that formula β describes the following fact

$(x = 0) \vee (x = 0.5) \vee (x = 0.5 * 0.5) \vee (x = 0.5 * 0.5 * 0.5) \vee \ldots.$

3. The proof of the main lemma

As we have mentioned (it is proved in [6]), the set of probabilistic algorithmic formulas of PrAL valid in a finite structure for L_P is decidable with respect to the diagram of the structure. By the diagram $D(\Im)$ of the structure $\Im$ we understand the set of all atomic or negated atomic formulas $\phi(c_{i_1}, \ldots, c_{i_m}) = c_{i_0}$ (ϕ is a functor of L) and $\psi(c_{i_1}, \ldots, c_{i_m})$ (ψ is a predicate symbol of L), which are valid in $\Im$.

The proof of decidability of PrAL essentially uses the Lemma which reduces the problem of validity of sentences of L_P to the (decidable) problem of the validity of sentences of the first-order arithmetic of real numbers. Finally, it appears that the set of formulas of PrAL, valid in all at most u-element structures for L_P, is decidable.

We shall show that if the language L_P^+ contains additionally the sign $\bigcup$ and the functor lg (for the operation of a logarithm) we can define natural numbers and operations of addition and multiplication for natural numbers.

Let's assume that 0.5^i abbreviates the expression $\underbrace{0.5 * 0.5 * \ldots * 0.5}_{i\ times}$.

Lemma 3.1 Let $< \Im, \rho >$ be an arbitrary fixed probabilistic structure (for L_P^+) with a finite set $A = \{a_1, a_2, \ldots, a_u\}$, where $u > 1$. Let K_0 and K be as follows

K_0: $v_1 := a_u$;
K : **if** $(v_1 = a_u)$ **then**
  either$_{0.5}$ $v_1 := a_u$; $v_2 := a_u$; **or** $v_1 := a_{u-1}$; $v_2 := a_u$; **ro**
 else $v_1 := a_1$; $v_2 := a_u$; **fi**

For an arbitrary natural number $i > 0$, if $\mu = [\mu_1, \mu_2, \ldots, \mu_{u^2}]$ is an input probability distribution then as a result of realization of program $K_0; K^i$ we obtain the following output probability distribution

$$\mu' = K_0 K^i_{<\Im,\rho>}(\mu) = [\underbrace{0,\ldots,0}_{u-1\ times}, 1-0.5^{(i-1)}, \underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5^i, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5^i].$$

Proof. Let us assume that $< \Im, \rho >$ is a fixed probabilistic structure (for L_P^+) with a finite at least 2-element set $A = \{a_1, a_2, \ldots, a_u\}$. Let's consider an arbitrary program $K_0; K^i$ ($i \in N_+$). The set of possible valuations of program variables contains u^2 elements: $w_1 = (a_1, a_1)$, $w_2 = (a_1, a_2)$, ..., $w_u = (a_1, a_u)$, $w_{u+1} = (a_2, a_1)$, $w_{u+2} = (a_2, a_2)$, ..., $w_{2u} = (a_2, a_u)$, ...,$w_{u^2-u+1} = (a_u, a_1)$, $w_{u^2-u+2} = (a_u, a_2)$, ..., $w_{u^2} = (a_u, a_u)$. We carry out computations for the input probability distribution $\mu = [\mu_1, \mu_2, \ldots, \mu_{u^2}]$. The proof of the Lemma 3.1 will proceed by induction on the length of programs.

(A) The base of induction.

First we shall justify that the realization of the program $K_0; K$ leads to the probability distribution

$$\mu' = K_0 K_{<\Im,\rho>}(\mu) = [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 0.5, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5].$$

We shall determine the necessary probability distributions (cf. the Appendix).

$$[v_1 := a_1]_{<\Im,\rho>}(\mu) = [\mu_1 + \mu_{u+1} + \ldots + \mu_{u^2-u+1}, \mu_2 + \mu_{u+2} + \ldots + \mu_{u^2-u+2}, \ldots, \mu_u + \mu_{2u} + \ldots + \mu_{u^2}, \underbrace{0,\ldots,0}_{u^2-u\ times}]$$

$$[v_1 := a_{u-1}]_{<\Im,\rho>}(\mu) = [\underbrace{0,\ldots,0}_{u^2-2u\ times}, \mu_1 + \mu_{u+1} + \ldots + \mu_{u^2-u+1}, \mu_2 + \mu_{u+2} + \ldots + \mu_{u^2-u+2}, \ldots, \mu_u + \mu_{2u} + \ldots + \mu_{u^2}, \underbrace{0,\ldots,0}_{u\ times}]$$

$[v_1 := a_u]_{<\Im,\rho>}(\mu) = [\underbrace{0,\ldots,0}_{u^2-u\ times}, \mu_1 + \mu_{u+1} + \ldots + \mu_{u^2-u+1}, \mu_2 + \mu_{u+2} + \ldots + \mu_{u^2-u+2}, \ldots, \mu_u + \mu_{2u} + \ldots + \mu_{u^2}]$

$[v_2 := a_u]_{<\Im,\rho>}(\mu) = [\underbrace{0,\ldots,0}_{u-1\ times}, \mu_1 + \mu_2 + \ldots + \mu_u, \underbrace{0,\ldots,0}_{u-1\ times}, \mu_{u+1} + \mu_{u+2} + \ldots + \mu_{2u}, \underbrace{0,\ldots,0}_{u-1\ times}, \ldots, \underbrace{0,\ldots,0}_{u-1\ times}, \mu_{u^2-u+1} + \mu_{u^2-u+2} + \ldots + \mu_{u^2}]$

Let's denote the subprogram $v_1 := a_u;\ v_2 := a_u;$ by N_1.

$N_{1<\Im,\rho>}(\mu) = [v_2 := a_u]_{<\Im,\rho>}([v_1 := a_u]_{<\Im,\rho>}(\mu)) =$
$= [\underbrace{0,\ldots,0}_{u^2-1\ times}, (\mu_1 + \mu_{u+1} + \ldots + \mu_{u^2-u+1}) + (\mu_2 + \mu_{u+2} + \ldots + \mu_{u^2-u+2}) + \ldots + (\mu_u + \mu_{2u} + \ldots + \mu_{u^2})] =$
$= [\underbrace{0,\ldots,0}_{u^2-1\ times}, \mu_1 + \mu_2 + \ldots + \mu_{u^2}] = [\underbrace{0,\ldots,0}_{u^2-1\ times}, 1]$

By N_2 we denote the subprogram $v_1 := a_{u-1};\ v_2 := a_u;$.

$N_{2<\Im,\rho>}(\mu) = [v_2 := a_u]_{<\Im,\rho>}([v_1 := a_{u-1}]_{<\Im,\rho>}(\mu)) =$
$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, (\mu_1 + \mu_{u+1} + \ldots + \mu_{u^2-u+1}) + (\mu_2 + \mu_{u+2} + \ldots + \mu_{u^2-u+2}) + \ldots + (\mu_u + \mu_{2u} + \ldots + \mu_{u^2}), \underbrace{0,\ldots,0}_{u\ times}] =$
$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, \mu_1 + \mu_2 + \ldots + \mu_{u^2}, \underbrace{0,\ldots,0}_{u\ times}] = [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 1, \underbrace{0,\ldots,0}_{u\ times}]$

The subprogram $v_1 := a_1;\ v_2 := a_u;$ we denote by N_3.

$N_{3<\Im,\rho>}(\mu) = [v_2 := a_u]_{<\Im,\rho>}([v_1 := a_1]_{<\Im,\rho>}(\mu)) =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, (\mu_1 + \mu_{u+1} + \ldots + \mu_{u^2-u+1}) + (\mu_2 + \mu_{u+2} + \ldots + \mu_{u^2-u+2}) + \ldots + (\mu_u + \mu_{2u} + \ldots + \mu_{u^2}), \underbrace{0,\ldots,0}_{u^2-u\ times}] =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, \mu_1 + \mu_2 + \ldots + \mu_{u^2}, \underbrace{0,\ldots,0}_{u^2-u\ times}] = [\underbrace{0,\ldots,0}_{u-1\ times}, 1, \underbrace{0,\ldots,0}_{u^2-u\ times}]$

Let's denote the subprogram **either**$_{0.5}$ N_1 **or** N_2 **ro** by E.

$E_{<\Im,\rho>}(\mu) = 0.5 * (N_{1<\Im,\rho>}(\mu)) + 0.5 * (N_{2<\Im,\rho>}(\mu)) =$
$= 0.5 * [\underbrace{0,\ldots,0}_{u^2-1\ times}, \mu_1 + \mu_2 + \ldots + \mu_{u^2}] + 0.5 * [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, \mu_1 + \mu_2 + \ldots + \mu_{u^2}, \underbrace{0,\ldots,0}_{u\ times}] =$

$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 0.5*(\mu_1+\ldots+\mu_{u^2}), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5*(\mu_1+\ldots+\mu_{u^2})] =$
$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 0.5, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5]$

$[(v_1 = a_u)?]_{<\Im,\rho>}(\mu) = [\underbrace{0,\ldots,0}_{u^2-u\ times}, \mu_{u^2-u+1}, \mu_{u^2-u+2},\ldots,\mu_{u^2}]$
$[\neg(v_1 = a_u)?]_{<\Im,\rho>}(\mu) = [\mu_1,\mu_2,\ldots,\mu_{u^2-u}, \underbrace{0,\ldots,0}_{u\ times}]$

$K_{<\Im,\rho>}(\mu) = E_{<\Im,\rho>}([(v_1 = a_u)?]_{<\Im,\rho>}(\mu)) + N_{3<\Im,\rho>}([\neg(v_1 = a_u)?]_{<\Im,\rho>}(\mu)) =$
$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 0.5*(\mu_{u^2-u+1}+\mu_{u^2-u+2}+\ldots+\mu_{u^2}), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5*(\mu_{u^2-u+1}+$
$\mu_{u^2-u+2}+\ldots+\mu_{u^2})] + [\underbrace{0,\ldots,0}_{u-1\ times}, (\mu_1+\mu_2+\ldots+\mu_{u^2-u}), \underbrace{0,\ldots,0}_{u^2-u\ times}] =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, (\mu_1+\mu_2+\ldots+\mu_{u^2-u}), \underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5*(\mu_{u^2-u+1}+\mu_{u^2-u+2}+\ldots+$
$\mu_{u^2}), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5*(\mu_{u^2-u+1}+\mu_{u^2-u+2}+\ldots+\mu_{u^2})]$

Finally

$K_{<\Im,\rho>}(K_{0<\Im,\rho>}(\mu)) = K_{<\Im,\rho>}([v_1 := a_u]_{<\Im,\rho>}(\mu)) =$
$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 0.5*((\mu_1+\mu_{u+1}+\ldots+\mu_{u^2-u+1})+(\mu_2+\mu_{u+2}+\ldots+\mu_{u^2-u+2})+\ldots+$
$(\mu_u+\mu_{2u}+\ldots+\mu_{u^2})), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5*((\mu_1+\mu_{u+1}+\ldots+\mu_{u^2-u+1})+(\mu_2+\mu_{u+2}+$
$\ldots+\mu_{u^2-u+2})+\ldots+(\mu_u+\mu_{2u}+\ldots+\mu_{u^2}))] =$
$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 0.5*(\mu_1+\mu_2+\ldots+\mu_{u^2}), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5*(\mu_1+\mu_2+\ldots+\mu_{u^2})] =$
$= [\underbrace{0,\ldots,0}_{u^2-u-1\ times}, 0.5, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5].$

(B) The inductive step.

The inductive assumption. For a certain natural number k, if $\mu = [\mu_1,\mu_2,\ldots,\mu_{u^2}]$ is an input probability distribution then as a result of realization of the program $K_0;K^k$ we obtain the following output probability distribution

$K_0 K^k < \Im, \rho > (\mu) =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, (1-0.5^{(k-1)}) * (\mu_1+\mu_2+\ldots+\mu_{u^2}), \underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5^k * (\mu_1+\mu_2+\ldots+$
$\mu_{u^2}), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5^k * (\mu_1+\mu_2+\ldots+\mu_{u^2})] =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, (1-0.5^{(k-1)}), \underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5^k, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5^k]$

We shall apply the inductive assumption to show that if we take $\mu = [\mu_1, \mu_2, \ldots, \mu_{u^2}]$ as the input probability distribution then after the execution of the program $K_0; K^{k+1}$ we obtain the following output probability distribution
$K_0 K^{k+1}_{<\Im,\rho>}(\mu) = [\underbrace{0,\ldots,0}_{u-1\ times}, (1-0.5^k) * (\mu_1+\mu_2+\ldots+\mu_{u^2}), \underbrace{0,\ldots,0}_{u^2-2u-1\ times},$
$0.5^{(k+1)} * (\mu_1+\mu_2+\ldots+\mu_{u^2}), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5^{(k+1)} * (\mu_1+\mu_2+\ldots+\mu_{u^2})] =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, (1-0.5^k), \underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5^{(k+1)}, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5^{(k+1)}]$

We can express a composition of programs in the following way (cf. the Appendix)
$K_0 K^{k+1}_{<\Im,\rho>}(\mu) = K_{<\Im,\rho>}(K_0 K^k_{<\Im,\rho>}(\mu))$

Hence by the inductive assumption
$K_{<\Im,\rho>}(K_0 K^k_{<\Im,\rho>}(\mu)) = K_{<\Im,\rho>}([\underbrace{0,\ldots,0}_{u-1\ times}, (1-0.5^{(k-1)}) * (\mu_1+\mu_2+\ldots+\mu_{u^2}),$
$\underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5^k * (\mu_1+\mu_2+\ldots+\mu_{u^2}), \underbrace{0,\ldots,0}_{u-1\ times}, 0.5^k * (\mu_1+\mu_2+\ldots+\mu_{u^2})]) =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, (1-0.5^{(k-1)}+0.5^k), \underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5*0.5^k, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5*0.5^k] =$
$= [\underbrace{0,\ldots,0}_{u-1\ times}, (1-0.5^k), \underbrace{0,\ldots,0}_{u^2-2u-1\ times}, 0.5^{(k+1)}, \underbrace{0,\ldots,0}_{u-1\ times}, 0.5^{(k+1)}]$
which accomplishes the inductive proof.

□

Lemma 3.2 Let $< \Im, \rho >$ be an arbitrary fixed structure (for L_P^+) with a finite set $A = \{a_1, a_2, \ldots, a_u\}$, where $u > 1$. The set of formulas of PrAL$^+$ valid in $< \Im, \rho >$ is undecidable.

Proof. Let $<\Im,\rho>$ be an arbitrary fixed structure (for L_P^+) with a finite at least 2-element set $A=\{a_1,\ldots,a_u\}$. Let's consider the formula β of the form $K_0\bigcup K\alpha$, where K_0, K are the programs considered in the Lemma 3.1 and α is as follows

α: $x=P(v_1=a_{u-1}\wedge v_2=a_u)$.

The computations are carried out for the input probability distribution $\mu=[\mu_1,\mu_2,\ldots,\mu_{u^2}]$ and for programs K_0 and $K_0;K^i$, where $i\in N_+$. Let's denote $K_{0<\Im,\rho>}(\mu)$ by η. We know that

$\eta=K_{0<\Im,\rho>}(\mu)=[v_1:=a_u]_{<\Im,\rho>}(\mu)=[\underbrace{0,\ldots,0}_{u^2-u\ times},\mu_1+\mu_{u+1}+\ldots+\mu_{u^2-u+1},\mu_2+\mu_{u+2}+\ldots+\mu_{u^2-u+2},\ldots,\mu_u+\mu_{2u}+\ldots+\mu_{u^2}]$.

By the Lemma 3.1 we obtain that for an arbitrary number $i>0$

$\mu'=K_0K^i_{<\Im,\rho>}(\mu)=[\underbrace{0,\ldots,0}_{u-1\ times},(1-0.5^{(i-1)}),\underbrace{0,\ldots,0}_{u^2-2u-1\ times},\underline{0.5^i},\underbrace{0,\ldots,0}_{u-1\ times},0.5^i]$.

We recall, that $P(v_1=a_{u-1}\wedge v_2=a_u)=\mu'(w_{u^2-u})$, where $w_{u^2-u}=(a_{u-1},a_u)$. We can notice that for $i\in N_+$ we have $\mu'(w_{u^2-u})=\underline{0.5^i}$ and additionally $\eta(w_{u^2-u})=\underline{0}$. Therefore the formula β: $K_0\bigcup K\alpha$ describes the following fact

$(x=0)\vee(x=0.5)\vee(x=0.25)\vee(x=0.125)\vee\ldots\vee(x=0.5^i)\vee\ldots$

Let's notice, that we can define an arbitrary natural number k in the following way. Let k be a real number

$N(k)$ iff $<\Im,\rho>\models\ (k=0\vee\exists x((k=-\lg x)\wedge K_0\bigcup K\alpha))$.

Since the natural numbers were generated among real numbers and operations of addition and multiplication exist in the structure $\Re=<R;+,-,*,0,1,<>$, we can define these operations for constructed natural numbers. For arbitrary x_0, x_1, x_2

$x_0\underline{+}x_1=x_2$ iff $<\Im,\rho>\models N(x_0)\wedge N(x_1)\wedge x_2=x_0+x_1$,
$x_0\underline{*}\,x_1=x_2$ iff $<\Im,\rho>\models N(x_0)\wedge N(x_1)\wedge x_2=x_0*x_1$.

Since $Th(<N;\underline{+},\underline{*},0,1>)$ is undecidable (cf. [2,11,7]), the set of formulas of considered algorithmic logic, valid in a fixed, finite at least 2-element structure (for L_P^+) is also undecidable.

$\square$

4. Appendix (cf. [6])

By the interpretation of a program K of L_P^+ in the structure $< \Im, \rho >$ we mean a function $K_{<\Im,\rho>} : \mathcal{M} \mapsto \mathcal{M}$ which is defined recursively.

– If K is *an assignment instruction* of the form $v_r := \tau$ (for $v_r \in V$, $r = 1, \ldots, h$ and $\tau \in T$) then
$[v_r := \tau]_{<\Im,\rho>}(\mu) = \mu^{'}$,where
$\mu^{'}(w_j) = \sum_{w \in W^{r,\tau}} \mu(w)$ for $j = 1, \ldots, n$ and
$W^{r,\tau} = \{w \in W : w(v_r) = \tau_{\Im}(w_{in}) \wedge \forall_{v \in V \setminus \{v_r\}} w(v) = w_{in}(v)\}$.
w_{in} denotes an input valuation of program variables.

– If K is *a random assignment* of the form $v_r :=$**random**$_l$ (for $v_r \in V$, $r = 1, \ldots, h$ and ρ_l being a probability distribution defined on A) then
$[v_r :=\textbf{random}_l]_{<\Im,\rho>}(\mu) = \mu^{'}$, where
$\mu^{'}(w_j) = \rho_l(w_j(v_r)) * \sum_{w \in W^r} \mu(w)$ and
$W^r = \{w \in W : \forall_{v \in V \setminus \{v_r\}} w(v) = w_{in}(v)\}$.

– We interpret the program **while** $\neg\gamma$ **do** $v := v$ **od** (for $v \in V$ and $\gamma \in F_O$) in the following way
$[\gamma?]_{<\Im,\rho>}(\mu) = [\textbf{while } \neg\gamma \textbf{ do } v := v \textbf{ od}]_{<\Im,\rho>}(\mu) = \mu^{'}$, where
$\mu^{'}(w_j) = \begin{cases} \mu(w_i) \text{ for } w_i = w_j \wedge \Im, w_i \models \gamma \\ 0 \quad\quad \text{otherwise} \end{cases}$
We denote this program construction by $[\gamma?]$.

– If K is *a composition of programs* M_1, M_2 and $M_{1<\Im,\rho>}(\mu)$, $M_{2<\Im,\rho>}(\mu)$ are defined then
$[M_1;\, M_2]_{<\Im,\rho>}(\mu) = M_{2<\Im,\rho>}(M_{1<\Im,\rho>}(\mu))$.

– If K is *a branching between the two programs* M_1, M_2 and $M_{1<\Im,\rho>}(\mu)$, $M_{2<\Im,\rho>}(\mu)$ are defined then
$[\textbf{if } \gamma \textbf{ then } M_1 \textbf{ else } M_2 \textbf{ fi}]_{<\Im,\rho>}(\mu) =$
$= M_{1<\Im,\rho>}([\gamma?]_{<\Im,\rho>}(\mu)) + M_{2<\Im,\rho>}([\neg\gamma?]_{<\Im,\rho>}(\mu))$.

– If K is *a probabilistic branching*, $p \in R$, $0 < p < 1$ and $M_{1<\Im,\rho>}(\mu)$, $M_{2<\Im,\rho>}(\mu)$ are defined then
$[\textbf{either}_p\ M_1 \textbf{ or } M_2 \textbf{ ro}]_{<\Im,\rho>}(\mu) = p * M_{1<\Im,\rho>}(\mu) + (1-p) * M_{2<\Im,\rho>}(\mu)$.

References

[1] L. Banachowski, Investigations of Properties of Programs by Means of the Extended Algorithmic Logic, Fundamenta Informatica, 1 (1977), p. 167–193.

[2] J. Barwise, Handbook of Mathematical Logic, Elsevier Science Publishers B.V., 1983.

[3] A. Borowska, Algorithmic Information Theory, Computer Information Systems and Industrial Management Applications, 2003, p. 5–31.

[4] A. Borowska, Algebraic Methods of Determining of Transition Probabilities in Probabilistic Algorithms, Conference Materials of XV FIT, 2004, p. 148–157.

[5] A. Borowska, Selected Problems of the Probabilistic Algorithms and the Markov Chains. Reduction of Valuations, Ph.D. Thesis, Technical University of Bialystok, 2010.

[6] W. Danko, The set of Probabilistic Algorithmic Formulas Valid in a Finite Structure is Decidable with Respect to its diagram, Fundamenta Informatica, Vol. 19 (1993), p. 417–431.

[7] A. Grzegorczyk, An Outline of Theoretical Arithmetics, PWN, Warsaw 1971.

[8] J. Koszelew, The Methods of Analysis of Properties of Probabilistic Programs Interpreted in Finite Fields, Ph.D. Thesis, PAN, 2000.

[9] A. Kreczmar, Effectivity problems of algorithmic logic, Automata, Languages and Programming, Lecture Notes in Computer Science, Vol. 14 (1974), Publisher Springer Berlin Heidelberg, p. 584–600.

[10] A. Kreczmar, The Set of all Tautologies of Algorithmic Logic is Hyperarithmetical, Bull. Acad. Polon. Sci., Ser. Math. Astron. Phys., 21 (1971), p. 781–783.

[11] R. C. London, About Mathematical Logic, PWN, Warsaw, 1968.

[12] G. Mirkowska, A. Salwicki, Algorithmic Logic, PWN-Polish Scientific Publishers, 1987.

[13] G. Mirkowska, A. Salwicki, Algorithmic Logic for Programmers, WN-T, Warsaw, 1992.

[14] A. Rutkowski, Elements of Mathematical Logic, WSiP, Warsaw, 1978.

Anna Borowska

ZBIÓR FORMUŁ LOGIKI PrAL^{+} PRAWDZIWYCH W SKOŃCZONEJ STRUKTURZE JEST NIEROZSTRZYGALNY

Streszczenie Rozważamy probabilistyczną logikę algorytmiczną. W pracy [6] znajduje się uzasadnienie, że zbiór formuł logiki PrAL, prawdziwych w skończonej strukturze, jest rozstrzygalny ze względu na diagram struktury. Dodajemy do języka L_P logiki PrAL znak $\bigcup$ i funktor lg. Następnie uzasadniamy, że zbiór formuł rozszerzonej logiki, prawdziwych w skończonej co najmniej 2-elementowej strukturze (dla L_P^+), nie jest już rozstrzygalny.

Słowa kluczowe: probabilistyczna logika algorytmiczna, egzystencjalny kwantyfikator iteracji