

The Complexity of Non-Iterated Probabilistic Justification Logic

Ioannis Kokkinis

Institute of Computer Science, University of Bern, Switzerland

July 6, 2018

Abstract

The logic PJ is a probabilistic logic defined by adding (non-iterated) probability operators to the basic justification logic J. In this paper we establish upper and lower bounds for the complexity of the derivability problem in the logic PJ. The main result of the paper is that the complexity of the derivability problem in PJ remains the same as the complexity of the derivability problem in the underlying logic J, which is Π_2^P -complete. This implies that the probability operators do not increase the complexity of the logic, although they arguably enrich the expressiveness of the language.

Keywords: justification logic, probabilistic logic, complexity, derivability, satisfiability

1 Introduction

Traditional modal epistemic logic uses formulas of the form $\Box\alpha$ to express that an agent believes α . The language of justification logic [5] ‘unfolds’ the \Box -modality into a family of so-called *justification terms*, which are used to represent evidence for the agent’s belief. Hence, instead of $\Box\alpha$, justification logic includes formulas of the form $t : \alpha$ meaning

the agent believes α for reason t .

Artemov [2, 3] developed the first justification logic, the Logic of Proofs, to provide intuitionistic logic with a classical provability semantics. There, justification terms represent formal proofs in Peano Arithmetic. However, terms may also represent informal justifications. For instance, our belief in α may be justified by direct observation of α or by learning that a friend heard about α . This general reading of justification led to a big variety of epistemic justification logics for many different applications [6, 7, 19]. In [15, 16] we extended justification logic with probability operators in order to accommodate the idea that

different kinds of evidence for α lead to different degrees of belief in α .

For example it could be the case that the agent learns α from some unreliable source (e.g. from some friend of his) or that the agent reads about α in some reliable newspaper. In both cases the agent has a justification for α : in the first case he has the statement of his friend and in the second case the text of the newspaper. However, it is natural that the agent does not want to put the same credence in both sources of information. This differentiation in credulity cannot be expressed in classical justification logic. So, the main contribution of justification logics with probability operators (probabilistic justification logics [15, 16]) is the ability to compare different sources of information. Uncertain reasoning in justification logic has also been studied in [21, 12, 11]. See [15, 16] for an extended comparison between our approach and the ones from [21, 12, 11].

Probabilistic logics are logics than can be used to model uncertain reasoning. Although the idea of probabilistic logic was first proposed by Leibnitz, the modern development of this topic started only in the 1970s and 1980s in the papers of H. Jerome Keisler [13] and Nils Nilsson [22]. Following Nilsson's research, Fagin, Halpern and Meggido [10] introduced a logic with arithmetical operations built into the syntax so that Boolean combinations of linear inequalities of probabilities of formulas can be expressed. The probabilistic logic of [10] can be considered as a probabilistic logic with classical base. The derivability problem in this logic is proved to be *coNP*-complete, the same as that of classical propositional logic. Following the lines of [10], Ognjanović, Rašković and Marković [23] defined the logic LPP_2 , which is also a probabilistic logic with classical base. The logic LPP_2 makes use of an infinitary rule which makes the proof of strong completeness possible (as opposed to the finitary system of [10] which is only simply complete). The LPP_2 -derivability problem is again *coNP*-complete.

Following the lines of [23] the logic PJ was defined in [15]. PJ is a probabilistic logic defined over the basic justification logic J.¹ The language of PJ contains formulas of the form $P_{\geq s}\alpha$ meaning

the probability of truthfulness of the justification formula α is at least s .

So, in the logic PJ, statements like “evidence t serves as a justification for α with probability at least 30%” can be expressed. PJ does not allow iterations of the probability operator. In [16] we study an extension of PJ, the logic PPJ,² where iterations of the probability operator as well as justification operators over probability operators are allowed.

The results of [17, 20, 8, 1] showed that, under some reasonable assumptions, the derivability problem for the justification logic J is Π_2^P -complete, i.e. it is complete in the second level of the polynomial hierarchy. In this paper we show that under the same assumptions the derivability problem for the probabilistic justification logic PJ remains in the class Π_2^P -complete. We achieve this, by showing that the satisfiability problem for the logic PJ, which is dual to the

¹J stands for justification, whereas PJ stands for probabilistic justification.

²the two P's stand for iterations of the probability operator.

derivability problem, belongs to the class Σ_2^P -complete. The methods we use are adaptations from [10] and [17]. As it is the case in [23] and [10] we also make use of some well known results from the theory of linear programming. The main result of the paper is that the probability operators do not increase the complexity of the logic, although they arguably enrich the expressiveness of the logical framework.

The rest of the paper is organized as follows. In section 2 we briefly recall the justification logic J and the probabilistic justification logic PJ. In section 3 we establish a small model theorem for PJ. In section 4 we present an algorithm that decides the satisfiability problem for the logic PJ and evaluate its complexity. We close the paper in section 5 with some final observations.

An earlier version of the present paper is available in arXiv [14].

2 The logics J and PJ

In this section we briefly recall the basic justification logic J [5] and the probabilistic justification logic PJ [15].

Justification terms are built according to the following grammar:

$$t ::= c \mid x \mid (t \cdot t) \mid (t + t) \mid !t$$

where c is a constant and x is a variable. \mathbf{Tm} denotes the set of all terms. For any term t and any non-negative integer n we define:

$$!^0 t := t \quad \text{and} \quad !^{n+1} t := ! (!^n t)$$

Terms are used to provide justifications for formulas. Constants are used as justifications for axioms, whereas variables are used as justifications for arbitrary formulas. The operator \cdot can be used by the agents to apply modus ponens (see axiom (J) in Figure 1), the operator $+$ is used for concatenation of proofs (see axiom (+) in Figure 1) and the operator $!$ is used for stating positive introspection (see rule (AN!) in Figure 2). That is, if the agent has a justification c for α then he has a justification $!c$ for the fact that c is a justification for α and so on.

Let \mathbf{Prop} denote a countable set of atomic propositions. Formulas of the language \mathcal{L}_J (justification formulas) are built according to the following grammar:

$$\alpha ::= p \mid \neg \alpha \mid \alpha \wedge \alpha \mid t : \alpha$$

where $t \in \mathbf{Tm}$ and $p \in \mathbf{Prop}$. Any formula of the form $t : \alpha$ for $t \in \mathbf{Tm}$ and $\alpha \in \mathcal{L}_J$ will be called a *justification assertion*. We will use the letter p possibly primed or with subscripts to represent an element of \mathbf{Prop} and lower-case Greek letters like $\alpha, \beta, \gamma, \dots$ for \mathcal{L}_J -formulas. In Figure 1 we present the axioms schemes of the logic J.

In order to build justifications for arbitrary formulas in the logic J we need to start by some justifications for the axioms. That is why we need the notion

(P)	finitely many axiom schemes for classical propositional logic in the language of \mathcal{L}_J
(J)	$\vdash u : (\alpha \rightarrow \beta) \rightarrow (v : \alpha \rightarrow u \cdot v : \beta)$
(+)	$\vdash (u : \alpha \vee v : \alpha) \rightarrow u + v : \alpha$

Figure 1: Axioms Schemes of J

of a constant specification. A *constant specification* is any set \mathbf{CS} that satisfies the following condition:

$$\mathbf{CS} \subseteq \{(c, \alpha) \mid c \text{ is a constant and } \alpha \text{ is an instance of some axiom scheme of the logic J}\}$$

A constant specification \mathbf{CS} will be called:

axiomatically appropriate: if for every instance of a J-axiom, α , there exists some constant c such that $(c, \alpha) \in \mathbf{CS}$, i.e. every instance of a J-axiom scheme is justified by at least one constant.

schematic: if for every constant c the set

$$\{\alpha \mid (c, \alpha) \in \mathbf{CS}\}$$

consists of all instances of several (possibly zero) axiom schemes, i.e. if every constant specifies certain axiom schemes and only them.

decidable: if the set \mathbf{CS} is decidable. In this paper when we refer to a decidable \mathbf{CS} , we will always imply that \mathbf{CS} is decidable in *polynomial time*.

finite: if \mathbf{CS} is a finite set.

almost schematic: if $\mathbf{CS} = \mathbf{CS}_1 \cup \mathbf{CS}_2$ where $\mathbf{CS}_1 \cap \mathbf{CS}_2 = \emptyset$, \mathbf{CS}_1 is a schematic constant specification and \mathbf{CS}_2 is a finite constant specification.

total: if for every constant c and every instance α of a J-axiom scheme, $(c, \alpha) \in \mathbf{CS}$.

Let \mathbf{CS} be any constant specification. The deductive system $\mathbf{J}_{\mathbf{CS}}$ is presented in Figure 2.

axioms schemes of J	
+	
(AN!)	$\vdash !^{n+1}c : !^nc : \dots : !c : c : \alpha$, where $(c, \alpha) \in \mathbf{CS}$ and $n \in \mathbb{N}$
(MP)	if $T \vdash \alpha$ and $T \vdash \alpha \rightarrow \beta$ then $T \vdash \beta$

Figure 2: System $\mathbf{J}_{\mathbf{CS}}$

As usual $T \vdash_L \alpha$ means that the formula α is provable from the set of formulas T using the rules and axioms of the logic L . When L is clear from the context it will be omitted.

Now we present the semantics for the logic J . The models for a J_{CS} are the so called J_{CS} -evaluations (see Definition 1). We use \top to represent the truth value “true” and \bot to represent the truth value “false”. Let $\mathcal{P}(W)$ denote the powerset of the set W .

Definition 1 (J_{CS} -Evaluation). Let CS be any constant specification. A J_{CS} -evaluation is a function $*$ such that $*$: $\mathbf{Prop} \rightarrow \{\top, \bot\}$ and $*$: $\mathbf{Tm} \rightarrow \mathcal{P}(\mathcal{L}_J)$ and for $u, v \in \mathbf{Tm}$, for a constant c and $\alpha, \beta \in \mathcal{L}_J$ we have:

- (1) $(\alpha \rightarrow \beta \in u^* \text{ and } \alpha \in v^*) \implies \beta \in (u \cdot v)^*$
- (2) $u^* \cup v^* \subseteq (u + v)^*$
- (3) if $(c, \alpha) \in CS$ then for all $n \in \mathbb{N}$ we have³:

$$!^{n-1}c : !^{n-2}c : \dots : !c : c : \alpha \in (!^n c)^*$$

We will usually write t^* and p^* instead of $*(t)$ and $*(p)$ respectively.

Now we will define the binary relation \Vdash .

Definition 2 (Truth under a J_{CS} -Evaluation). We define what it means for an \mathcal{L}_J -formula to hold under a J_{CS} -evaluation $*$ inductively as follows:

$$\begin{aligned} * \Vdash p &\iff p^* = \top \\ * \Vdash \neg \alpha &\iff * \not\Vdash \alpha \\ * \Vdash \alpha \wedge \beta &\iff (* \Vdash \alpha \text{ and } * \Vdash \beta) \\ * \Vdash t : \alpha &\iff \alpha \in t^* \end{aligned}$$

We have the following theorem.

Theorem 3 (Completeness of J [4, 19]). *Let CS be any constant specification. Let $\alpha \in \mathcal{L}_J$. Then we have:*

$$\vdash_{J_{CS}} \alpha \iff \Vdash_{CS} \alpha.$$

where $\Vdash_{CS} \alpha$ means that α holds under any J_{CS} -evaluation.

Let S be the set of all rational numbers from the interval $[0, 1]$. The formulas of the language \mathcal{L}_P (the so called probabilistic formulas) are built according to the following grammar:

$$A ::= P_{\geq s} \alpha \mid \neg A \mid A \wedge A$$

³We agree to the convention that the formula $!^{n-1}c : !^{n-2}c : \dots : !c : c : \alpha$ represents the formula α for $n = 0$.

where $s \in \mathbb{S}$, and $\alpha \in \mathcal{L}_J$. We use capital Latin letters like A, B, C, \dots for \mathcal{L}_P -formulas. We employ the standard abbreviations for classical connectives. Additionally, we set:

$$\begin{aligned} P_{<_s}\alpha &\equiv \neg P_{\geq_s}\alpha & P_{\leq_s}\alpha &\equiv P_{\geq_{1-s}}\neg\alpha \\ P_{>_s}\alpha &\equiv \neg P_{\leq_s}\alpha & P_{=_s}\alpha &\equiv P_{\geq_s}\alpha \wedge P_{\leq_s}\alpha \end{aligned}$$

The axioms schemes of PJ are presented in Figure 3. For any constant spec-

(P)	finitely many axiom schemes for classical propositional logic in the language of \mathcal{L}_P
(PI)	$\vdash P_{\geq_0}\alpha$
(WE)	$\vdash P_{\leq_r}\alpha \rightarrow P_{<_s}\alpha$, where $s > r$
(LE)	$\vdash P_{<_s}\alpha \rightarrow P_{\leq_s}\alpha$
(DIS)	$\vdash P_{\geq_r}\alpha \wedge P_{\geq_s}\beta \wedge P_{\geq_1}\neg(\alpha \wedge \beta) \rightarrow P_{\geq_{\min(1, r+s)}}(\alpha \vee \beta)$
(UN)	$\vdash P_{\leq_r}\alpha \wedge P_{<_s}\beta \rightarrow P_{<_{r+s}}(\alpha \vee \beta)$, where $r + s \leq 1$

Figure 3: Axioms Schemes of PJ

ification CS the deductive system PJ_{CS} is presented in Figure 4. Definitions 4–6 describe the semantics for the logic PJ.

axiom schemes of PJ	
+	
(MP)	if $T \vdash A$ and $T \vdash A \rightarrow B$ then $T \vdash B$
(CE)	if $\vdash_{JCS} \alpha$ then $\vdash_{PJCS} P_{\geq_1}\alpha$
(ST)	if $T \vdash A \rightarrow P_{\geq_{s-\frac{1}{k}}}\alpha$ for every integer $k \geq \frac{1}{s}$ and $s > 0$ then $T \vdash A \rightarrow P_{\geq_s}\alpha$

Figure 4: System PJ_{CS}

Definition 4 (Algebra over a set). Let W be a non-empty set and let H be a non-empty subset of $\mathcal{P}(W)$. H will be called an *algebra over W* iff the following hold:

- $W \in H$
- $U, V \in H \implies U \cup V \in H$
- $U \in H \implies W \setminus U \in H$

Definition 5 (Finitely Additive Measure). Let H be an algebra over W and $\mu : H \rightarrow [0, 1]$. We call μ a *finitely additive measure* iff the following hold:

- (1) $\mu(W) = 1$

(2) for all $U, V \in H$:

$$U \cap V = \emptyset \implies \mu(U \cup V) = \mu(U) + \mu(V)$$

Definition 6 (PJ_{CS}-Model). Let CS be any constant specification. A PJ_{CS}-model, or simply a model, is a structure $M = \langle W, H, \mu, * \rangle$ where:

- W is a non-empty set of objects called worlds.
- H is an algebra over W .
- $\mu : H \rightarrow [0, 1]$ is a finitely additive measure.
- $*$ is a function from W to the set of all J_{CS}-evaluations, i.e. $*(w)$ is a J_{CS}-evaluation for each world $w \in W$. We will usually write $*_w$ instead of $*(w)$.

Definition 7 (Measurable model). Let $M = \langle W, H, \mu, * \rangle$ be a model and $\alpha \in \mathcal{L}_J$. We define the following set:

$$[\alpha]_M = \{w \in W \mid *_w \Vdash \alpha\}$$

We will omit the subscript M , i.e. we will simply write $[\alpha]$, if M is clear from the context. A PJ_{CS}-model $M = \langle W, H, \mu, * \rangle$ is *measurable* iff $[\alpha]_M \in H$ for every $\alpha \in \mathcal{L}_J$. The class of measurable PJ_{CS}-models will be denoted by PJ_{CS,Meas}.

Definition 8 (Truth in a PJ_{CS,Meas}-model). Let CS be any constant specification. Let $M = \langle W, H, \mu, * \rangle$ be a PJ_{CS,Meas}-model. We define what it means for an \mathcal{L}_P -formula to hold in M inductively as follows⁴:

$$\begin{aligned} M \models P_{\geq s} \alpha &\iff \mu([\alpha]_M) \geq s \\ M \models \neg A &\iff M \not\models A \\ M \models A \wedge B &\iff (M \models A \text{ and } M \models B) \end{aligned}$$

In the sequel we may refer to PJ_{CS,Meas}-models simply as models if there is no danger for confusion. We have the following theorem.

Theorem 9 (Strong Completeness for PJ [15]). *Any PJ_{CS} is sound and strongly complete with respect to PJ_{CS,Meas}-models, i.e. for any $T \subseteq \mathcal{L}_P$ and any $A \in \mathcal{L}_P$:*

$$T \vdash_{PJ_{CS}} A \iff T \models_{PJ_{CS}} A$$

Let CS be any constant specification. A formula $A \in \mathcal{L}_P$ is satisfied in $M \in PJ_{CS,Meas}$ iff $M \models A$. A will be called PJ_{CS,Meas}-satisfiable or simply satisfiable if there is a PJ_{CS,Meas}-model that satisfies A . We define the PJ_{CS,Meas}-satisfiability problem to be the decision problem defined as follows:

“For a given $A \in \mathcal{L}_P$ and a given CS is A PJ_{CS,Meas}-satisfiable?”

⁴Observe that the satisfiability relation of a J_{CS}-evaluation is represented with \Vdash whereas the satisfiability relation of a model is represented with \models .

A formula $\alpha \in \mathcal{L}_J$ is satisfied in a J_{CS} -evaluation $*$ iff $* \models \alpha$. α will be called J_{CS} -satisfiable or simply satisfiable if there is some J_{CS} -evaluation $*$ that satisfies α . We define the J_{CS} -satisfiability problem to be the decision problem defined as follows:

“For a given $\alpha \in \mathcal{L}_J$ and a given CS is α J_{CS} -satisfiable?”

3 Small Model Property

The goal of this section is to prove a small model property for the logic PJ. The small model property will be the most important tool for establishing the upper bound for the complexity of PJ.

Definition 10 (Subformulas). The set $\text{subf}(\cdot)$ is defined recursively as follows:

For \mathcal{L}_J -formulas:

- $\text{subf}(p) := \{p\}$
- $\text{subf}(t : \alpha) := \{t : \alpha\} \cup \text{subf}(\alpha)$
- $\text{subf}(\neg\alpha) := \{\neg\alpha\} \cup \text{subf}(\alpha)$
- $\text{subf}(\alpha \wedge \beta) := \{\alpha \wedge \beta\} \cup \text{subf}(\alpha) \cup \text{subf}(\beta)$

For \mathcal{L}_P -formulas:

- $\text{subf}(P_{\geq s}\alpha) := \{P_{\geq s}\alpha\} \cup \text{subf}(\alpha)$
- $\text{subf}(\neg A) := \{\neg A\} \cup \text{subf}(A)$
- $\text{subf}(A \wedge B) := \{A \wedge B\} \cup \text{subf}(A) \cup \text{subf}(B)$

Observe that for $A \in \mathcal{L}_P$ we have that $\text{subf}(A) \subseteq \mathcal{L}_P \cup \mathcal{L}_J$.

Definition 11 (Atoms). Let A be an \mathcal{L}_P - or an \mathcal{L}_J -formula. Let X be the set that contains all the atomic propositions and the justification assertions from the set $\text{subf}(A)$. An atom of A is any formula of the following form:

$$\bigwedge_{B \in X} \pm B \tag{1}$$

where $\pm B$ denotes either B or $\neg B$. We will use the lowercase Latin letter a for atoms, possibly with subscripts.

Let A be an \mathcal{L}_P - or an \mathcal{L}_J -formula. Assume that A is either of the form $\bigwedge_i B_i$ or of the form $\bigvee_i B_i$. Then $C \in A$ means that for some i , $B_i = C$.

Definition 12 (Sizes). The size function $|\cdot|$ is defined as follows:

For \mathcal{L}_P -formulas: (recursively)

- $|P_{\geq s}\alpha| := 2$
- $|\neg A| := 1 + |A|$
- $|A \wedge B| := |A| + 1 + |B|$

For sets:

Let W be a set. $|W|$ is the cardinal number of W .

For non-negative integers:

Let r be a non-negative integer. We define the size of r to be equal to the length of r written in binary, i.e.:

$$|r| := \begin{cases} 1 & , r = 0 \\ \lfloor \log_2(r) + 1 \rfloor & , r \geq 1 \end{cases}$$

where $\lfloor \cdot \rfloor$ is the function that returns the greatest integer that is less than or equal to its argument.

For non-negative rational numbers:

Let $r = \frac{s_1}{s_2}$, where s_1 and s_2 are relatively prime non-negative integers with $s_2 \neq 0$, be a non-negative rational number. We define:

$$|r| := |s_1| + |s_2|$$

Let $A \in \mathcal{L}_P$ we define:

$$||A|| := \max \{ |s| \mid P_{\geq s} \alpha \in \text{subf}(A) \}$$

Lemma 13 was originally proved in [23] for the logic LPP₂. The proof for the logic PJ is given in [15].

Lemma 13. *For any constant specification CS, we have:*

$$\vdash_{\text{JCS}} \alpha \leftrightarrow \beta \iff \vdash_{\text{PJCS}} P_{\geq s} \alpha \leftrightarrow P_{\geq s} \beta$$

A proof for Theorem 14 can be found in [9, p. 145].

Theorem 14. *Let \mathcal{S} be a system of r linear equalities. Assume that the vector⁵ \mathbf{x} is a solution of \mathcal{S} such that all of \mathbf{x} 's entries are non-negative. Then there is a vector \mathbf{x}^* such that:*

- (1) \mathbf{x}^* is a solution of \mathcal{S} .
- (2) all the entries of \mathbf{x}^* are non-negative.
- (3) at most r entries of \mathbf{x}^* are positive.

Theorem 15 establishes some properties for the solutions of a linear system.

Theorem 15. *Let \mathcal{S} be a linear system of n variables and of r linear equalities and/or inequalities with integer coefficients each of size at most l . Assume that the vector $\mathbf{x} = x_1, \dots, x_n$ is a solution of \mathcal{S} such that for all $i \in \{1, \dots, n\}$, $x_i \geq 0$. Then there is a vector $\mathbf{x}^* = x_1^*, \dots, x_n^*$ with the following properties:*

- (1) \mathbf{x}^* is a solution of \mathcal{S} .
- (2) for all $i \in \{1, \dots, n\}$, $x_i^* \geq 0$.

⁵We will always use bold font for vectors.

- (3) at most r entries of \mathbf{x}^* are positive.
- (4) for all $i \in \{1, \dots, n\}$, if $x_i^* > 0$ then $x_i > 0$.
- (5) for all i , x_i^* is a non-negative rational number with size bounded by

$$2 \cdot (r \cdot l + r \cdot \log_2(r) + 1) .$$

Proof. In \mathcal{S} we replace the variables that correspond to the entries of \mathbf{x} that are equal to zero (if any) with zeros. This way we obtain a new linear system \mathcal{S}_0 , with r linear equalities and/or inequalities and $m \leq n$ variables. \mathbf{x} is a solution⁶ of \mathcal{S}_0 . It also holds that any solution of \mathcal{S}_0 is a solution⁷ of \mathcal{S} .

Assume that the system \mathcal{S}_0 contains an inequality of the form

$$b_1 \cdot y_1 + \dots + b_m y_m \diamond c \tag{2}$$

for $\diamond \in \{<, \leq, \geq, >\}$ where y_1, \dots, y_m are variables of \mathcal{S} and b_1, \dots, b_m, c are constants that appear in \mathcal{S} . \mathbf{x} is a solution of (2). We replace the inequality (2) in \mathcal{S}_0 with the following equality:

$$b_1 \cdot y_1 + \dots + b_m y_m = b_1 \cdot x_1 + \dots + b_l \cdot x_m$$

We repeat this procedure for every inequality of \mathcal{S}_0 . This way we obtain a system of linear equalities which we call \mathcal{S}_1 . It is easy to see that \mathbf{x} is a solution of \mathcal{S}_1 and that any solution of \mathcal{S}_1 is also a solution of \mathcal{S}_0 and thus of \mathcal{S} .

Now we will transform \mathcal{S}_1 to another linear system by applying the following algorithm.

Algorithm:

We set $i = 1$, $e_i = r$, $v_i = m$, $\mathbf{x}^i = \mathbf{x}$ and we execute the following steps:

- (i) If $e_i = v_i$ then go to step (ii). Otherwise go to step (iii).
- (ii) If the determinant of \mathcal{S}_i is non-zero then stop. Otherwise go to step (v).
- (iii) If $e_i < v_i$ then go to step (iv), else go to step (v).
- (iv) We know that the vector \mathbf{x}^i is a non-negative solution for the system \mathcal{S}_i . From Theorem 14 we obtain a solution \mathbf{x}^{i+1} for the system \mathcal{S}_i which has at most e_i entries positive. In \mathcal{S}_i we replace the variables that correspond to zero entries of the solution \mathbf{x}^{i+1} with zeros. We obtain a new system which we call \mathcal{S}_{i+1} with e_{i+1} equalities and v_{i+1} variables. \mathbf{x}^{i+1} is a solution of \mathcal{S}_{i+1} and any solution of \mathcal{S}_{i+1} is a solution of \mathcal{S}_i . We set $i := i + 1$ and we go to step (i).

⁶In the proof of Theorem 15 all vectors have n entries. The entries of the vectors are assumed to be in one to one correspondence with the variables that appear in the original system \mathcal{S} .

Let \mathbf{y} be a solution of a linear system \mathcal{T} . If \mathbf{y} has more entries than the variables of \mathcal{T} we imply that entries of \mathbf{y} that correspond to variables that appear in \mathcal{T} compose a solution of \mathcal{T} .

⁷Assume that system \mathcal{T} has less variables than system \mathcal{T}' . When we say that any solution of \mathcal{T} is a solution of \mathcal{T}' we imply that the missing variables are set to 0.

- (v) From any set of equalities that are linearly dependent we keep only one equation. We obtain a new system which we call \mathcal{S}_{i+1} with e_{i+1} equalities and $v_{i+1} := v_i$ variables. We set $i := i + 1$ and $\mathbf{x}^{i+1} := \mathbf{x}^i$. We go to step (i).

Let I be the final value of i after the execution of the algorithm. Since the only way for our algorithm to terminate is through step (ii) it holds that system \mathcal{S}_I is an $e_I \times e_I$ system of linear equalities with non-zero determinant (for $e_I \leq r$). System \mathcal{S}_I is obtained from system \mathcal{S}_1 by replacing some variables that correspond to zero entries of the solution with zeros. So any solution of \mathcal{S}_I is also a solution of system \mathcal{S}_1 and thus a solution of \mathcal{S} . From the algorithm we have that \mathbf{x}^I is a solution of \mathcal{S}_I . Since \mathcal{S}_I has a non-zero determinant Cramer's rule can be applied. Hence the vector \mathbf{x}^I is the unique solution of system \mathcal{S}_I . Let x_i^I be an entry of \mathbf{x}^I . x_i^I will be equal to the following rational number

$$\frac{\begin{vmatrix} a_{11} & \dots & a_{1e_I} \\ & \ddots & \\ a_{e_I 1} & \dots & a_{e_I e_I} \end{vmatrix}}{\begin{vmatrix} b_{11} & \dots & b_{1e_I} \\ & \ddots & \\ b_{e_I 1} & \dots & b_{e_I e_I} \end{vmatrix}}$$

where all the a_{ij} and b_{ij} are integers that appear in the original system \mathcal{S} . By properties of the determinant we know that the numerator and the denominator of the above rational number will each be at most equal to $r! \cdot (2^l - 1)^r$. So we have that:

$$\begin{aligned} |x_i^I| &\leq 2 \cdot (\log_2(r! \cdot (2^l - 1)^r) + 1) && \implies \\ |x_i^I| &\leq 2 \cdot (\log_2(r^r \cdot 2^{l \cdot r}) + 1) && \implies \\ |x_i^I| &\leq 2 \cdot (r \cdot \log_2(r) + l \cdot r + 1) \end{aligned}$$

As we already mentioned the final vector \mathbf{x}^I is a solution of the original linear system \mathcal{S} . We also have that all the entries of \mathbf{x}^I are non-negative, at most r of its entries are positive and the size of each entry of \mathbf{x}^I is bounded by $2 \cdot (r \cdot \log_2 r + r \cdot l + 1)$. Furthermore, since the variables that correspond to zero entries of the original vector \mathbf{x} were replaced by zeros, we have that for every i , if the i -th entry of \mathbf{x}^I is positive then the i -th entry of \mathbf{x} is positive too. So \mathbf{x}^I is the requested vector \mathbf{x}^* . \square

The following theorem is an adaptation of the small model theorem from [10]. Similar techniques have also been used in [23] to obtain decidability for the logic LPP_2 .

Theorem 16 (Small Model Property). *Let CS be any constant specification and let $A \in \mathcal{L}_P$. If A is $\text{PJ}_{\text{CS}, \text{Meas}}$ -satisfiable then it is satisfiable in a $\text{PJ}_{\text{CS}, \text{Meas}}$ -model $M = \langle W, H, \mu, * \rangle$ such that:*

(1) $|W| \leq |A|$

(2) $H = \mathcal{P}(W)$

(3) For every $w \in W$, $\mu(\{w\})$ is a rational number with size at most

$$2 \cdot (|A| \cdot ||A|| + |A| \cdot \log_2(|A|) + 1)$$

(4) For every $V \in H$

$$\mu(V) = \sum_{w \in V} \mu(\{w\})$$

(5) For every atom of A , a , there exists at most one $w \in W$ such that $*_w \Vdash a$.

Proof. Let \mathbf{CS} be any constant specification and let $A \in \mathcal{L}_{\mathbf{P}}$. Let a_1, \dots, a_n be all the atoms of A . By propositional reasoning (in the logic $\mathbf{PJ}_{\mathbf{CS}}$) we can prove that:

$$\mathbf{PJ}_{\mathbf{CS}} \vdash A \leftrightarrow \bigvee_{i=1}^K \bigwedge_{j=1}^{l_i} P_{\Diamond_{ij} s_{ij}}(\beta^{ij})$$

where all the $P_{\Diamond_{ij} s_{ij}}(\beta^{ij})$ appear in A and $\Diamond_{ij} \in \{\geq, <\}$.

By using propositional reasoning again (but this time in the logic $\mathbf{J}_{\mathbf{CS}}$) we can prove that each β^{ij} is equivalent to a disjunction of some atoms of A . So, by using Lemma 13 we have that:

$$\mathbf{PJ}_{\mathbf{CS}} \vdash A \leftrightarrow \bigvee_{i=1}^K \bigwedge_{j=1}^{l_i} P_{\Diamond_{ij} s_{ij}}(\alpha^{ij})$$

where each α^{ij} is a disjunction of some atoms of A . By Theorem 9 we have that for any $M \in \mathbf{PJ}_{\mathbf{CS}, \text{Meas}}$:

$$M \models A \iff M \models \bigvee_{i=1}^K \bigwedge_{j=1}^{l_i} P_{\Diamond_{ij} s_{ij}}(\alpha^{ij}) \quad (3)$$

Assume that A is satisfiable. By (3) there must exist some i such that

$$\bigwedge_{j=1}^{l_i} P_{\Diamond_{ij} s_{ij}}(\alpha^{ij})$$

is satisfiable. Let $M' = \langle W', H', \mu', *' \rangle$ be a model such that:

$$M' \models \bigwedge_{j=1}^{l_i} P_{\Diamond_{ij} s_{ij}}(\alpha^{ij}) \quad (4)$$

For every $k \in \{1, \dots, n\}$ we define:

$$x_k = \mu'([a_k]_{M'}) \quad (5)$$

In every world of M' some atom of A must hold. Thus, we have:

$$W' = \bigcup_{k=1}^n [a_k]_{M'}$$

And since $\mu'(W') = 1$ we get:

$$\mu' \left(\bigcup_{k=1}^n [a_k]_{M'} \right) = 1 \quad (6)$$

The a_k 's are atoms of the same formula, so we have:

$$k \neq k' \implies [a_k]_{M'} \cap [a_{k'}]_{M'} = \emptyset \quad (7)$$

By (6), (7) and the fact that μ' is a finitely additive measure we get:

$$\sum_{k=1}^n \mu'([a_k]_{M'}) = 1$$

and by (5):

$$\sum_{k=1}^n x_k = 1 \quad (8)$$

Let $j \in \{1, \dots, l_i\}$. From (4) we get:

$$M' \models P_{\Diamond_{ij} s_{ij}}(\alpha^{ij}).$$

This implies that $\mu'([\alpha^{ij}]_{M'}) \Diamond_{ij} s_{ij}$, i.e.

$$\mu' \left(\left[\bigvee_{a_k \in \alpha^{ij}} a_k \right]_{M'} \right) \Diamond_{ij} s_{ij}$$

which implies that

$$\mu' \left(\bigcup_{a_k \in \alpha^{ij}} [a_k]_{M'} \right) \Diamond_{ij} s_{ij}$$

By (7) and the additivity of μ' we have that:

$$\sum_{a_k \in \alpha^{ij}} \mu'([a_k]_{M'}) \Diamond_{ij} s_{ij}$$

and by (5):

$$\sum_{a_k \in \alpha^{ij}} x_k \Diamond_{ij} s_{ij} .$$

So we have that

$$\text{for every } j \in \{1, \dots, l_i\}, \sum_{a_k \in \alpha^{ij}} x_k \Diamond_{ij} s_{ij} \quad (9)$$

Let \mathcal{S} be the following linear system:

$$\begin{aligned} \sum_{k=1}^n z_k &= 1 \\ \sum_{a_k \in \alpha^{i1}} z_k \Diamond_{i1} s_{i1} \\ &\vdots \\ \sum_{a_k \in \alpha^{il_i}} z_k \Diamond_{il_i} s_{il_i} \end{aligned}$$

where the variables of the system are z_1, \dots, z_n . We have the following:

- (i) By (8) and (9) the vector $\mathbf{x} = x_1, \dots, x_n$ is a solution of \mathcal{S} .
 - (ii) From (5) every x_k is non-negative.
 - (iii) Every s_{ij} is a rational number with size at most $\|A\|$.
 - (iv) System \mathcal{S} has at most $|A|$ equalities and inequalities.
- From (i)-(iv) and Theorem 15 we have that there exists a vector $\mathbf{y} = y_1, \dots, y_n$ such that:
- (I) \mathbf{y} is a solution of \mathcal{S} .
 - (II) every y_i is a non-negative rational number with size at most

$$2 \cdot (|A| \cdot \|A\| + |A| \cdot \log_2(|A|) + 1) .$$

(III) at most $|A|$ entries of \mathbf{y} are positive.

(IV) for all i , if $y_i > 0$ then $x_i > 0$.

Assume that y_1, \dots, y_N are the positive entries of \mathbf{y} where

$$N \leq |A| \quad (10)$$

We define the quadruple $M = \langle W, H, \mu, * \rangle$ as follows:

- (a) $W = \{w_1, \dots, w_N\}$, for some w_1, \dots, w_N .
- (b) $H = \mathcal{P}(W)$.
- (c) for all $V \in H$:

$$\mu(V) = \sum_{w_k \in V} y_k .$$

- (d) Let $i \in \{1, \dots, N\}$. We define $*_{w_i}$ to be some JCS-evaluation that satisfies the atom a_i . Since y_i is positive, by (IV), x_i is positive too, i.e. $\mu'([a_i]_{M'}) > 0$, which means that $[a_i]_{M'} \neq \emptyset$, i.e. that the atom a_i is JCS-satisfiable.

It holds:

$$\begin{aligned} \mu(W) &= \sum_{w_k \in W} y_k \\ &= \sum_{k=1}^n y_k \\ &\stackrel{(I)}{=} 1 \end{aligned}$$

Let $U, V \in H$ such that $U \cap V = \emptyset$. It holds:

$$\begin{aligned} \mu(U \cup V) &= \sum_{w_k \in U \cup V} y_k \\ &= \sum_{w_k \in U} y_k + \sum_{w_k \in V} y_k \\ &= \mu(U) + \mu(V) \end{aligned}$$

Thus μ is a finitely additive measure. By Definitions 6 and 7 we have that $M \in \text{PJCS}_{\text{Meas}}$.

We will now prove the following statement:

$$(\forall 1 \leq k \leq n)[w_k \in [\alpha^{ij}]_M \iff a_k \in \alpha^{ij}] \quad (11)$$

Let $k \in \{1, \dots, n\}$. We prove the two directions of (11) separately.

(\implies ;) Assume that $w_k \in [\alpha^{ij}]_M$. This means that $*_{w_k} \models \alpha^{ij}$. Assume that $a_k \notin \alpha^{ij}$. Then, since α^{ij} is a disjunction of atoms of A , there must exist some $a_{k'} \in \alpha^{ij}$, with $k \neq k'$, such that $*_{w_k} \models a_{k'}$. However, by definition we have that $*_{w_k} \models a_k$. But this is a contradiction, since a_k and $a_{k'}$ are different atoms of the same formula, which means that they cannot be satisfied by the same JCS-evaluation. Hence, $a_k \in \alpha^{ij}$.

(\impliedby ;) Assume that $a_k \in \alpha^{ij}$. We know that $*_{w_k} \models a_k$, which implies that $*_{w_k} \models \alpha^{ij}$, i.e. $w_k \in [\alpha^{ij}]_M$.

Hence, (11) holds. Now, we will prove the following statement:

$$(\forall 1 \leq j \leq l_i)[M \models P_{\Diamond_{ij} s_{ij}} \alpha^{ij}] \quad (12)$$

Let $j \in \{1, \dots, l_i\}$. It holds

$$\begin{aligned}
M &\models P_{\Diamond_{ij} s_{ij}}(\alpha^{ij}) && \Longleftrightarrow \\
\mu([\alpha^{ij}]_M) &\Diamond_{ij} s_{ij} && \Longleftrightarrow \\
\sum_{w_k \in [\alpha^{ij}]_M} y_k &\Diamond_{ij} s_{ij} && \stackrel{(11)}{\Longleftrightarrow} \\
\sum_{a_k \in \alpha^{ij}} y_k &\Diamond_{ij} s_{ij}
\end{aligned}$$

The last statement holds because of (I). Thus, (12) holds.

By (12) we have that $M \models \bigwedge_{j=1}^{l_i} P_{\Diamond_{ij} s_{ij}}(\alpha^{ij})$, which implies that

$$M \models \bigvee_{i=1}^K \bigwedge_{j=1}^{l_i} P_{\Diamond_{ij} s_{ij}}(\alpha^{ij}),$$

which, by (3), implies that $M \models A$.

Let $w_k \in W$. It holds:

$$\mu(\{w_k\}) = \sum_{w_i \in \{w_k\}} y_i = y_k \quad (13)$$

Now we will show that conditions (1)–(5) in the theorem's statement hold.

- Condition (1) holds because of (a) and (10).
- Condition (2) holds because of (b).
- Condition (3) holds because of (13) and (II).
- For every $V \in H$, because of (13), we have:

$$\mu(V) = \sum_{w_k \in V} y_k = \sum_{w_k \in V} \mu(\{w_k\})$$

Hence condition (4) holds.

- By (d) every world of M satisfies a unique atom of α . Thus condition (5) holds.

So M is the model in question. \square

4 Complexity

Lemmata 17 and 18 can be proved by straightforward induction on the complexity of the formula. Lemma 17 tells us that if two J_{CS} -evaluations agree on some atom of a justification formula then they agree on the formula itself.

Lemma 17. *Let CS be any constant specification. Let $\alpha \in \mathcal{L}_J$ and let a be an atom of α . Let $*_1, *_2$ be two JCS -evaluations and assume that*

$$*_1 \Vdash a \iff *_2 \Vdash a .$$

Then we have:

$$*_1 \Vdash \alpha \iff *_2 \Vdash \alpha .$$

Lemma 18. *Let $\alpha \in \mathcal{L}_J$ and let a be an atom of α . Let $*$ be a JCS -evaluation and assume that $* \Vdash a$. The decision problem*

does $$ satisfy α ?*

belongs to the complexity class P .

Kuznets [17] presented an algorithm for the JCS -satisfiability problem for a total constant specification CS . Kuznets' algorithm is divided in two parts: the saturation algorithm and the completion algorithm. Let $\alpha \in \mathcal{L}_J$ be the formula that is tested for satisfiability.

- The saturation algorithm produces a set of requirements that should be satisfied by any JCS -evaluation that satisfies α . The saturation algorithm operates in NP -time⁸.
- The completion algorithm determines whether a JCS -evaluation that satisfies α exists or not. The completion algorithm operates in $coNP$ -time.

If the saturation and the completion algorithm are taken together, then we obtain a Σ_2^P -algorithm for the JCS -satisfiability problem (for a total CS). The completion algorithm (adjusted to our notation) is stated in Theorem 19.

Theorem 19. *Let CS be a total constant specification. Let a be an atom of some \mathcal{L}_J -formula. The decision problem*

is a JCS -satisfiable?

belongs to the complexity class $coNP$.

Now we are ready to prove the upper bound for the complexity of the $\text{PJ}_{\text{CS}, \text{Meas}}$ -satisfiability problem.

Theorem 20. *Let CS be a total constant specification. The $\text{PJ}_{\text{CS}, \text{Meas}}$ -satisfiability problem belongs to the complexity class Σ_2^P .*

Proof. First we will describe an algorithm that decides the problem in question and we will explain its correctness. Then we will evaluate the complexity of the algorithm.

⁸A reader unfamiliar with notions of computational complexity theory may consult a text-book on the field, like [24].

Algorithm:

Let $A \in \mathcal{L}_P$. It suffices to guess a small model $M = \langle W, H, \mu, * \rangle$ that satisfies A and also satisfies the conditions (1)–(5) that appear in the statement of Theorem 16. We guess M as follows: we guess n atoms of A , call them a_1, \dots, a_n , and we also choose n worlds, w_1, \dots, w_n , for $n \leq |A|$. Using Theorem 19 we verify that for each $i \in \{1, \dots, n\}$ there exists a J_{CS} -evaluation $*_i$ such that $*_i \models a_i$. We define $W = \{w_1, \dots, w_n\}$. For every $i \in \{1, \dots, n\}$ we set $*_{w_i} = *_i$. Since we are only interested in the satisfiability of justification formulas that appear in A , by Lemma 17, the choice of the $*_{w_i}$ is not important (as long as $*_{w_i}$ satisfies a_i).

We assign to every $\mu(\{w_i\})$ a rational number with size at most:

$$2 \cdot (|A| \cdot ||A|| + |A| \cdot \log_2(|A|) + 1) .$$

We set $H = \mathcal{P}(W)$. For every $V \in H$ we set:

$$\mu(V) = \sum_{w_i \in V} \mu(\{w_i\}) .$$

It is then straightforward to see that the conditions (1)–(5) that appear in the statement of Theorem 16 hold.

Now we have to verify that our guess is correct, i.e. that $M \models A$. Assume that $P_{\geq s}\alpha$ appears in A . In order to see whether $P_{\geq s}\alpha$ holds we need to calculate the measure of the set $[\alpha]_M$ in the model M . The set $[\alpha]_M$ will contain every $w_i \in W$ such that $*_{w_i} \models \alpha$. Since $*_{w_i}$ satisfies an atom of A it also satisfies an atom of α . So, by Lemma 18, we can check whether $*_{w_i}$ satisfies α in polynomial time. If $\sum_{w_i \in [\alpha]_M} \mu(\{w_i\}) \geq s$ then we replace $P_{\geq s}\alpha$ in A with the truth value \top , otherwise with the truth value \bot . We repeat the above procedure for every formula of the form $P_{\geq s}\alpha$ that appears in A . At the end we have a formula that is constructed only from the connectives \neg, \wedge and the truth constants \top and \bot . Using a truth table we can verify in polynomial time that the formula is true. This, of course implies that $M \models A$.

Complexity Evaluation:

All the objects that are guessed in our algorithm have size that is polynomial on A . Also the verification phase of our algorithm can be made in polynomial time. Furthermore the application of Theorem 19 is possible with an NP -oracle (an NP -oracle can obviously decide $coNP$ problems too). Thus our algorithm is an NP^{NP} algorithm and since $\Sigma_2^P = NP^{NP}$ the claim of the Theorem follows. \square

5 Final Remarks and Conclusion

As a continuation of [15] and [16] we showed that results for justification logic and probabilistic logic can be nicely combined. Recall that the probabilistic justification logic PJ is obtained by adding probability operators to the justification logic J. In [17] it was proved that under some assumptions on the constant specification the complexity of the satisfiability problem for the logic J belongs to

the class Σ_2^p . By Theorem 20 we have that, under the same assumptions on the constant specification, the complexity of the satisfiability problem for the logic PJ remains in the same complexity class. Hence, the probabilistic operators do not increase the complexity of the satisfiability problem, although they increase the expressiveness of the language.

As it is pointed out in [18], Theorem 19 holds for a decidable almost schematic constant specification. Theorem 20 uses Theorem 19 as an oracle. So, obviously Theorem 20 holds for a decidable almost schematic constant specification too.

The upper complexity bound we established is tight. By a result from [20] which was later strengthened in [8] and [1] we have that for a decidable, schematic and axiomatically appropriate constant specification CS the J_{CS} -satisfiability problem is Σ_2^p -hard. For any $\alpha \in \mathcal{L}_J$ it is not difficult to prove that:

$$\alpha \text{ is } J_{CS}\text{-satisfiable} \iff P_{\geq 1}\alpha \text{ is } PJ_{CS, Meas}\text{-satisfiable}$$

Hence, the J_{CS} -satisfiability problem can be reduced to the $PJ_{CS, Meas}$ -satisfiability problem, which implies that the $PJ_{CS, Meas}$ -satisfiability problem is Σ_2^p -hard too. Thus the J_{CS} -satisfiability problem as well as the $PJ_{CS, Meas}$ -satisfiability problem are Σ_2^p -complete.

Observe that by Theorem 9 and our previous remarks we have that, for a decidable schematic and axiomatically appropriate constant specification, the derivability problem for the logic PJ_{CS} is Π_2^p -complete.

In [16] the probabilistic justification logic PPJ is defined. PPJ is a natural extension of PJ that supports iterations of the probability operator as well as justifications over probabilities. An interesting open problem related to the present work is to determine complexity bounds for PPJ.

Funding:

The author is supported by the SNSF project 153169, *Structural Proof Theory and the Logic of Proofs*.

Acknowledgements:

The author is grateful to Antonis Achilleos, Thomas Studer and the anonymous referees for valuable comments and remarks that helped him improve the quality of the paper substantially.

References

- [1] Achilleos, A.: Nexp-completeness and universal hardness results for justification logic (2015), cSR 2015: 27-52
- [2] Artemov, S.N.: Operational modal logic. Tech. Rep. MSI 95-29, Cornell University (Dec 1995)
- [3] Artemov, S.N.: Explicit provability and constructive semantics. Bulletin of Symbolic Logic 7(1), 1-36 (Mar 2001)

- [4] Artemov, S.N.: The ontology of justifications in the logical setting. *Studia Logica* 100(1–2), 17–30 (Apr 2012), published online February 2012
- [5] Artemov, S.N., Fitting, M.: Justification logic. In: Zalta, E.N. (ed.) *The Stanford Encyclopedia of Philosophy*. Fall 2012 edn. (2012), <http://plato.stanford.edu/archives/fall2012/entries/logic-justification/>
- [6] Bucheli, S., Kuznets, R., Studer, T.: Justifications for common knowledge. *Journal of Applied Non-Classical Logics* 21(1), 35–60 (Jan–Mar 2011)
- [7] Bucheli, S., Kuznets, R., Studer, T.: Partial realization in dynamic justification logic. In: Beklemishev, L.D., de Queiroz, R. (eds.) *Logic, Language, Information and Computation, 18th International Workshop, WoLLIC 2011, Philadelphia, PA, USA, May 18–20, 2011, Proceedings, Lecture Notes in Artificial Intelligence*, vol. 6642, pp. 35–51. Springer (2011)
- [8] Buss, S.R., Kuznets, R.: Lower complexity bounds in justification logic. *Annals of Pure and Applied Logic* 163(7), 888–905 (Jul 2012)
- [9] Chvátal, V.: *Linear programming*. W. H. Freeman and Company, New York (1983)
- [10] Fagin, R., Halpern, J., Megiddo, N.: A logic for reasoning about probabilities. *Information and Computation* 87, 78–128 (1990)
- [11] Fan, T., Liau, C.: A logic for reasoning about justified uncertain beliefs. In: Yang, Q., Wooldridge, M. (eds.) *Proc. IJCAI 2015*. pp. 2948–2954. AAAI Press (2015)
- [12] Ghari, M.: Justification logics in a fuzzy setting. *ArXiv e-prints* (Jul 2014)
- [13] Keisler, J.: Hyperfinite model theory. In: Gandy, R.O., Hyland, J.M.E. (eds.) *Logic Colloquium 1976*, p. 5–10. North-Holland (1977)
- [14] Kokkinis, I.: On the complexity of probabilistic justification logic (2015), *arXiv e-prints*
- [15] Kokkinis, I., Maksimović, P., Ognjanović, Z., Studer, T.: First steps towards probabilistic justification logic. *Logic Journal of the IGPL* 23(4), 662–687 (2015)
- [16] Kokkinis, I., Ognjanović, Z., Studer, T.: Probabilistic justification logic. In: Artemov, S., Nerode, A. (eds.) *Symposium on Logical Foundations in Computer Science 2016* (2016), to appear
- [17] Kuznets, R.: On the complexity of explicit modal logics. In: Clote, P.G., Schwichtenberg, H. (eds.) *Computer Science Logic, 14th International Workshop, CSL 2000, Annual Conference of the EACSL, Fischbachau, Germany, August 21–26, 2000, Proceedings, Lecture Notes in Computer Science*, vol. 1862, pp. 371–383. Springer (2000)

- [18] Kuznets, R.: Complexity Issues in Justification Logic. Ph.D. thesis, City University of New York (May 2008), <http://gradworks.umi.com/33/10/3310747.html>
- [19] Kuznets, R., Studer, T.: Justifications, ontology, and conservativity. In: Bolander, T., Braüner, T., Ghilardi, S., Moss, L. (eds.) *Advances in Modal Logic*, Volume 9, pp. 437–458. College Publications (2012)
- [20] Milnikel, R.S.: Derivability in certain subsystems of the Logic of Proofs is Π_2^p -complete. *Annals of Pure and Applied Logic* 145(3), 223–239 (Mar 2007)
- [21] Milnikel, R.S.: The logic of uncertain justifications. *Annals of Pure and Applied Logic* 165(1), 305–315 (Jan 2014)
- [22] Nilsson, N.: Probabilistic logic. *Artificial Intelligence* 28, 71–87 (1986)
- [23] Ognjanović, Z., Rašković, M., Marković, Z.: Probability logics. *Zbornik radova*, subseries “Logic in Computer Science” 12(20), 35–111 (2009)
- [24] Papadimitriou, C.H.: *Computational Complexity*. Addison-Wesley (1994)