Classical Generalized Probabilistic Satisfiability *

Carlos Caleiro
SQIG - Instituto de Telecomunicações
DMath, Instituto Superior Técnico
Universidade de Lisboa, Portugal
carlos.caleiro@tecnico.ulisboa.pt

Filipe Casal
CMAF-CIO, Portugal
DMath, Instituto Superior Técnico
Universidade de Lisboa, Portugal
filipe.casal@tecnico.ulisboa.pt

Andreia Mordido
INOV INESC Inovação, Portugal
andreia.mordido@inov.pt

Abstract

We analyze a classical generalized probabilistic satisfiability problem (GGenPSAT) which consists in deciding the satisfiability of Boolean combinations of linear inequalities involving probabilities of classical propositional formulas. GGenPSAT coincides precisely with the satisfiability problem of the probabilistic logic of Fagin et al. and was proved to be NP-complete. Here, we present a polynomial reduction of GGenPSAT to SMT over the quantifier-free theory of linear integer and real arithmetic. Capitalizing on this translation, we implement and test a solver for the GGenPSAT problem. As previously observed for many other NP-complete problems, we are able to detect a phase transition behavior for GGenPSAT.

1 Introduction

The starting point of a deep analysis of the propositional satisfiability (SAT) problem was due to Cook in [Cook, 1971], where it was shown that this problem is NP-complete. Given its simplicity and expressiveness, the SAT problem has become the standard NP-complete problem to study and, because of that, SAT solvers became extremely efficient. Due to this, several extensions and generalizations have been developed, taking advantage of the referred solvers. An example of this is the satisfiability modulo theories problem (SMT) [De Moura and Bjørner, 2011] where instead of working in propositional logic, one tries to decide if a formula is valid in some specific first-order theory. This area has had a great impact in industry, especially in hardware and software verification. One other direction for generalization of propositional satisfiability consists in the introduction of probabilities into the classical reasoning, allowing one to express quantitative assertions about propositional formulas.

In this sense, there was an effort to extend propositional logic in order to handle probabilistic reasoning. Fagin et al. [Fagin *et al.*, 1990] developed a widely used probabilistic logic and showed that its satisfiability problem is NP-complete. Recently, several satisfiability solvers were proposed for fragments of this probabilistic logic. Finger and Bona developed a PSAT solver [Finger and Bona, 2011; 2015] in the context of the probabilistic satisfiability problem (PSAT) [Boole, 1853; Nilsson, 1986], which consists in deciding the satisfiability of a set of assignments of probabilities to propositional formulas. Afterwards, the PSAT problem was generalized to handle Boolean combinations of assignments of probabilities to propositional formulas leading to GPSAT in [Bona *et al.*, 2015]. After that, in [Caleiro *et al.*, 2016a], Caleiro et al. introduced the generalized probabilistic satisfiability problem (GenPSAT) which consists in deciding the satisfiability of linear inequalities involving probabilities of classical propositional formulas. Given this, it is only natural to think about the extension of this problem to Boolean combinations of probabilistic formulas. This is our goal for this paper: extend the GenPSAT problem to allow Boolean combinations of probabilistic formulas as well as present a solver for this more expressive problem.

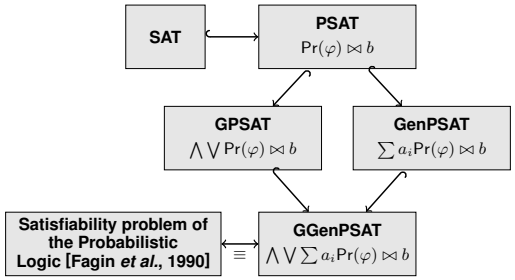

Figure 1: Inclusion diagram of several fragments of the probabilistic logic

In this paper, we present the classical generalized probabilistic satisfiability problem GGenPSAT, which consists in deciding the satisfiability of Boolean combinations of linear inequalities involving probabilities of classical propositional formulas. This problem was proved to be NP-complete in [Fagin *et al.*, 1990]. We stress that the formulas expressible in GGenPSAT are precisely the formulas in the probabilistic logic by Fagin et al. We develop an algo-

*Work done under the scope of Project UID/EEA/50008/2013, financed by the applicable financial framework (FCT/MEC through national funds and when applicable co-funded by FEDER–PT2020) and partially supported by Fundação para a Ciência e a Tecnologia by way of grant UID/MAT/04561/2013 to Centro de Matemática, Aplicações Fundamentais e Investigação Operacional of Universidade de Lisboa (CMAF-CIO). AM was supported by FCT under the grant SFRH/BD/77648/2011 and by the Calouste Gulbenkian Foundation under *Programa de Estímulo à Investigação* 2011. FC acknowledges the support from the DP-PMI and FCT (Portugal) through scholarship SRFH/BD/52243/2013. CC acknowledges the support of EU FP7 Marie Curie PIRSES-GA-2012-318986 project GeTFun: Generalizing Truth-Functionality.

rithm for the GGenPSAT problem by constructing a polynomial reduction to the quantifier-free theory of linear integer and real arithmetic (QF_LIRA) [Barrett *et al.*, 2016; King, 2014]. We also provide an implementation of the algorithm and study its phase transition behavior.

As the main contribution of this work, we develop the theoretical framework that allows the translation between GGenPSAT and SMT problems, which then allows the implementation of a provably correct solver for GGenPSAT. With the GGenPSAT solver in hands, we are able to detect and study the phase transition behavior.

This paper is outlined as follows: in Section 2 we recall some basic notions on probabilistic satisfiability; in Section 3, we introduce the GGenPSAT problem; in Section 4, we present the polynomial reduction to SMT and prove the correctness of the algorithm; in Section 5, we describe the implemented tool and study the phase transition behavior of the GGenPSAT problem; finally, Section 6 concludes the paper and discusses avenues for further research.

2 Preliminaries

Let us begin by fixing a set of propositional variables $\mathcal{P} = \{x_1, \ldots, x_n\}$. The set of *classical propositional formulas* is defined, as usual, by

$$\mathrm{L_{CPL}} ::= \mathcal{P} \mid \neg \mathrm{L_{CPL}} \mid \mathrm{L_{CPL}} \wedge \mathrm{L_{CPL}} \ .$$

A *propositional literal* is either a propositional variable or its negation. A *propositional clause* is a non-empty disjunction of one or more propositional literals. A *propositional valuation* is a map $v : \mathcal{P} \to \{0, 1\}$, which is extended to propositional formulas as usual. A set of valuations $\mathcal{V}$ *satisfies* a propositional formula φ if, for each $v \in \mathcal{V}$, $v(\varphi) = 1$. This notion is naturally extended to sets of propositional formulas. Let $\mathcal{V}^* = \{v_1, \ldots, v_{2^n}\}$ be the set of all valuations defined over variables of $\mathcal{P}$. We define a *probability distribution* π over $\mathcal{V}^*$ as a probability vector of size 2^n.

We recall from [Fagin *et al.*, 1990] the set of *probabilistic atoms* (used herein to define probabilistic formulas) composed by linear inequalities of probabilities of propositional formulas with rational coefficients:

$$\mathsf{PAt} ::= \mathbb{Q} \cdot \mathsf{Pr}(\mathrm{L_{CPL}}) + \ldots + \mathbb{Q} \cdot \mathsf{Pr}(\mathrm{L_{CPL}}) \geq \mathbb{Q} \ .$$

The set of *probabilistic formulas* is defined as a Boolean combination of probabilistic atoms as follows:

$$\mathsf{Prob} ::= \mathsf{PAt} \mid \neg \mathsf{Prob} \mid \mathsf{Prob} \wedge \mathsf{Prob} \ .$$

Observe that the other relational symbols $\{<, >, \leq, =, \neq\}$ can be defined by abbreviation, as well as the logical connectives $\rightarrow, \vee, \leftrightarrow$.

To interpret probabilistic formulas, we consider a probability distribution π over $\mathcal{V}^*$. The satisfaction relation is inductively defined as:

- $\pi \Vdash q_1 \cdot \mathsf{Pr}(\varphi_1) + \ldots + q_\ell \cdot \mathsf{Pr}(\varphi_\ell) \geq q$ iff $\sum_{i=1}^{\ell} \left(q_i \left(\sum_{j=1}^{2^n} v_j(\varphi_i) \cdot \pi_j \right) \right) \geq q$;
- $\pi \Vdash \neg\delta$ iff $\pi \nVdash \delta$;
- $\pi \Vdash \delta_1 \wedge \delta_2$ iff $\pi \Vdash \delta_1$ and $\pi \Vdash \delta_2$,

where $\delta, \delta_1, \delta_2 \in \mathsf{Prob}$, $q, q_i \in \mathbb{Q}$ and $\varphi_i \in \mathrm{L_{CPL}}$ where $i \in \{1, \ldots, \ell\}$. A probability distribution π *satisfies* $\delta \in \mathsf{Prob}$ if $\pi \Vdash \delta$ and satisfies a set of probabilistic formulas if it satisfies each one of them.

2.1 The PSAT Problem

A *simple probabilistic formula* is a probabilistic formula of the form $\mathsf{Pr}(c) \bowtie q$ where $q \in \mathbb{Q}$, $0 \leq q \leq 1$, $\bowtie \in \{=, \leq, \geq\}$ and $c \in \mathrm{L_{CPL}}$ is a propositional clause. Note that a probability distribution π satisfies a formula $\mathsf{Pr}(c) \bowtie q$ if

$$\sum_{j=1}^{2^n} (v_j(c) \cdot \pi_j) \bowtie q \ .$$

We now recall the PSAT problem [Nilsson, 1986; Georgakopoulos *et al.*, 1988; Finger and Bona, 2011].

Definition 1 (PSAT problem)**.** *Given a set of propositional variables $\mathcal{P}$ and a set of simple probabilistic formulas $\Sigma = \{\mathsf{Pr}(c_i) \bowtie p_i \mid 1 \leq i \leq k\}$, the Probabilistic Satisfiability problem (PSAT) consists in determining whether there exists a probability distribution π over $\mathcal{V}^*$ that satisfies Σ.*

The PSAT problem for $\{\mathsf{Pr}(c_i) \bowtie_i p_i \mid 1 \leq i \leq k\}$ can be formulated algebraically as the problem of finding a solution π for the system of inequalities

$$\begin{cases} V\pi \bowtie p \\ \sum \pi_i = 1 \\ \pi \geq 0 \end{cases}$$

where V is the $k \times 2^n$ matrix such that $V_{ij} = v_j(c_i)$, i.e., $V_{ij} = 1$ iff the j-th valuation satisfies the i-th clause, $p = [p_i]$ is the k vector of all p_i and $\bowtie = [\bowtie_i]$ is the k vector of all $\bowtie_i$.

A SAT problem can be modeled as a PSAT instance where the entries p_i of the probability vector are all identical to 1. The PSAT problem was shown to be NP-complete [Georgakopoulos *et al.*, 1988; Fagin *et al.*, 1990], even when the clauses consist of the disjunction of only two literals, 2PSAT.

2.2 The GenPSAT Problem

In [Caleiro *et al.*, 2016a], probabilistic satisfiability was extended to handle linear inequalities involving assignments of values to propositional formulas.

An *instance* of GenPSAT is a pair (Γ, Σ) where Γ is a set of propositional clauses (also called hard constraints) and Σ is a set of probabilistic atoms (soft constraints). We say that a probability distribution π *satisfies* a GenPSAT instance (Γ, Σ) if it satisfies the set of probabilistic atoms

$$\Xi_{(\Gamma,\Sigma)} = \Sigma \cup \{\mathsf{Pr}(\gamma) = 1 \mid \gamma \in \Gamma\} \ . \tag{1}$$

Definition 2 (GenPSAT problem)**.** *Given a GenPSAT instance (Γ, Σ), the Generalized Probabilistic Satisfiability problem (GenPSAT) consists in determining if there exists a probability distribution π over $\mathcal{V}^*$ that satisfies (Γ, Σ).*

3 The GGenPSAT Problem

We now aim to extend the GenPSAT problem in order to cope with Boolean combinations of probabilistic atoms.

An *instance* of GGenPSAT is a pair (Γ, Ψ) where Γ is a set of classical propositional formulas (also called hard constraints) and Ψ is a set of probabilistic formulas (soft constraints). We say that a probability distribution π *satisfies* a GGenPSAT instance (Γ, Ψ) if it satisfies the set of probabilistic formulas

$$\Xi_{(\Gamma,\Psi)} = \Psi \cup \{\mathsf{Pr}(\gamma) = 1 \mid \gamma \in \Gamma\} \ . \tag{2}$$

Despite the similarities between a GenPSAT and a GGenPSAT instance, the latter allows more expressive probabilistic formulas by allowing Boolean combinations of probabilistic atoms.

Definition 3 (GGenPSAT problem)**.** *Given a* GGenPSAT *instance* (Γ, Ψ)*, the Classical Generalized Probabilistic Satisfiability problem consists in determining if there exists a probability distribution* π *over* $\mathcal{V}^*$ *that satisfies* (Γ, Ψ)*.*

GGenPSAT extends the scope of PSAT and GenPSAT by dealing with Boolean combinations of probabilistic formulas. In this way, we are not only able to assign values to probabilities of propositional variables or linear inequalities involving them, but also able to express powerful probabilistic assertions. For instance, we can easily model and reason about a framework where a variable x is either true or false with probability 1 but we do not know which is the case:

$$\mathsf{Pr}(x) = 0 \vee \mathsf{Pr}(x) = 1 \ .$$

In this way, we can, in a sense, model the universal quantification of propositional formulas. To notice the impact of this generalization on the available models of a formula, consider the following example.

Example 1. *Consider a game of* Odds and Evens *where players* A *and* B *play* $x, y \in \mathbb{Z}_2$*, respectively, and player* A *wins iff* $x \oplus y = 0$*. We can easily study the effectiveness and existence of strategies for this game in* GGenPSAT*: the term* $\mathsf{Pr}(\neg(x \oplus y))$ *represents the probability that player* A *wins the game; with this, we can determine that there is a strategy in which player* B *wins sometimes,* $\mathsf{Pr}(x \oplus y) > 0$*, and that player* A *wins twice as much as player* B*,* $\mathsf{Pr}(\neg(x \oplus y)) \geq 2 \cdot \mathsf{Pr}(x \oplus y)$ *by checking the satisfiability of such formulas.*

However, if we additionally assume that player A *always plays* 0*,* $\mathsf{Pr}(x) = 0$*, and that player* B *always plays the same,* $\mathsf{Pr}(y) = 0 \vee \mathsf{Pr}(y) = 1$*, then the above formulas are no longer satisfiable.*

In these examples, we studied the existence of strategies *by determining the satisfiability of formulas. Additionally, we could also determine if some strategies are always better than others, by determining if a certain formula is* valid*.* ♢

A GenPSAT instance is also a GGenPSAT instance.

Notice that GGenPSAT has been studied in the context of the decision problem for the probabilistic logic introduced by Fagin, Halpern and Megiddo in [Fagin *et al.*, 1990]. Hence, the computational complexity of this problem is known and addressed in the following theorem.

Theorem 1 ([Fagin *et al.*, 1990])**.** GGenPSAT *is* NP*-complete.*

4 Reducing GGenPSAT to Satisfiability Modulo Theories

Our goal now is to effectively build a decision procedure for this problem. In [Caleiro *et al.*, 2016a], Caleiro et al. constructed an effective procedure for the GenPSAT problem by a polynomial reduction to Mixed-Integer Programming (MIP). However, in that framework, one cannot handle Boolean combinations of linear inequalities, at least intuitively. A framework where this problem is naturally expressed is in Satisfiability Modulo Theories (SMT) with respect to the theory of Quantifier-Free Linear Integer and Real Arithmetic, QF_LIRA, which is also in NP [Barrett *et al.*, 2016; King, 2014]. We will now explore the NP-completeness of GGenPSAT and provide a polynomial reduction to QF_LIRA. The variables of the theory can be of one of three sorts: Boolean, integer or real, and the signature is composed of the function symbols $\mathcal{F} = \{0, 1, +, \cdot\}$ and the usual predicates $\mathcal{P} = \{\geq, \leq, <, >, =, \neq\}$. The atoms of the theory are either Boolean variables or linear inequalities involving real and integer variables.

We explore some preliminary steps that lead to an algorithm to solve the GGenPSAT problem.

On the details of the GGenPSAT **instance:** Assume we are given a GGenPSAT instance (Γ, Ψ), where Γ is a set of classical propositional formulas (hard constraints), $\Gamma = \{\varphi_1, \ldots, \varphi_k\}$, and Ψ is a set of probabilistic formulas (soft constraints), $\Psi = \{\delta_1, \ldots, \delta_s\}$. Recall that a formula δ_j is a Boolean combination of probabilistic atoms of the form

$$q_1 \cdot \mathsf{Pr}(\psi_1) + \ldots + q_\ell \cdot \mathsf{Pr}(\psi_\ell) \bowtie q \ ,$$

where $\bowtie \in \{\geq, \leq, <, >, =, \neq\}$.

When ghosts attack: Driven by GenPSAT and PSAT developments, it is simpler to deal with probabilities of propositional variables than with probabilities of propositional formulas. To this end, we introduce *propositional ghost variables* which will represent the propositional formulas occurring inside the probabilistic formulas. Let us define the set of fresh variables that will be used. For this purpose, collect in $\mathsf{InsidePr}(\delta) \subseteq \mathrm{L}_{\mathsf{CPL}}$ all the propositional formulas occurring inside the probabilistic formula $\delta \in \mathsf{Prob}$, which is defined inductively on the structure of δ:

- $\mathsf{InsidePr}(q_1 \cdot \mathsf{Pr}(\psi_1) + \ldots + q_\ell \cdot \mathsf{Pr}(\psi_\ell) \bowtie q) = \{\psi_1, ..., \psi_\ell\}$;
- $\mathsf{InsidePr}(\neg\delta) = \mathsf{InsidePr}(\delta)$;
- $\mathsf{InsidePr}(\delta_1 \wedge \delta_2) = \mathsf{InsidePr}(\delta_1) \cup \mathsf{InsidePr}(\delta_2)$.

This notion is extended for a set Δ of probabilistic formulas as usual, $\mathsf{InsidePr}(\Delta) = \bigcup_{\delta \in \Delta} \mathsf{InsidePr}(\delta)$. According to this, and recalling that the propositional formulas in Γ need to be satisfied with probability 1, we consider the set of relevant propositional formulas, RelF defined by:

$$\mathsf{RelF} = \Gamma \cup \mathsf{InsidePr}(\Psi) \ .$$

Consider the set of propositional ghost variables corresponding to each element of RelF:

$$\mathfrak{G} = \{\mathfrak{p}_\psi \mid \psi \in \mathsf{RelF}\} \ .$$

Furthermore, we will use the real $[0, 1]$-variable α_ψ to represent the probability of $\psi \in \mathsf{RelF}$.

For ease of notation, we denote by $\mathfrak{G}_i$ the i-th element of $\mathfrak{G}$ and ψ_i the corresponding propositional formula in RelF. We also denote by $|\mathfrak{G}|$ the cardinality of a set $\mathfrak{G}$. The set of propositional variables of interest is $\mathcal{B} = \mathcal{P} \cup \mathfrak{G}$.

Algebraic formulation: Motivated by the algebraic formulation of PSAT and GenPSAT, we express the probabilistic assertions about the elements in RelF algebraically as follows:

$$\begin{cases} V\pi = \alpha \\ \sum \pi_j = 1 \\ \pi \geq 0 \end{cases} \tag{3}$$

where:

- $V = [V_{ij}]$ is a matrix of size $|\mathfrak{G}| \times 2^n$, where V_{ij} is defined from the j^{th} valuation $v_j \in \mathcal{V}^*$ and from the i^{th} propositional ghost variable $\mathfrak{G}_i$, by $V_{ij} = v_j(\psi_i)$;
- $\pi = [\pi_j]$ is a vector of size 2^n, where each π_j is a real $[0, 1]$-variable representing the probability valuation v_j;
- $\alpha = [\alpha_i]$ is a vector of size $|\mathfrak{G}|$ and each α_i is a real $[0, 1]$-variable that represents the probability of ψ_i.

Using the following Lemma by Chvátal [Chvátal, 1983], we can take a step forward in the choice of the right valuations.

Lemma 1 ([Chvátal, 1983; Fagin *et al.*, 1990])**.** *If a system of ℓ linear inequalities with integer coefficients has a nonnegative solution, then it has a nonnegative solution with at most ℓ positive entries.*

Lemma 1 tells us that a system with $|\mathfrak{G}|+1$ linear inequalities has a solution iff it has a solution with $|\mathfrak{G}|+1$ nonnegative entries. Furthermore, if a GGenPSAT instance is satisfiable then system (3) has a solution. Let us collect in $H = [h_{ij}]$, the $|\mathfrak{G}|+1$ columns of V given by Lemma 1, where $h_{|\mathfrak{G}|+1,j} = 1$ for each j, and consider the corresponding probability assignments in π:

$$\begin{cases} H\pi = \alpha \\ \pi \geq 0 \end{cases} . \tag{4}$$

When variables multiply: Inspired by the previous arguments, we consider $|\mathfrak{G}| + 1$ copies of each propositional variable of interest in $\mathcal{B}$. Each copy is intended to represent the valuations underlying the columns of matrix H. We represent them by

$$\mathcal{B}^{(k)} = \{x^{(k)} \mid x \in \mathcal{P}\} \cup \{\mathfrak{p}^{(k)} \mid \mathfrak{p} \in \mathfrak{G}\} \ .$$

We extend this notation to propositional formulas as expected – given a propositional formula ψ, $\psi^{(k)}$ represents the formula ψ where each of its variables x was replaced by its appropriate copy, $x^{(k)}$. Denote by $\widetilde{\mathcal{B}} = \bigcup_{k=1}^{|\mathfrak{G}|+1} \mathcal{B}^{(k)}$ the set of all copies of all propositional variables.

Probabilistic formulas seen as linear restrictions: To handle probabilistic formulas in the QF_LIRA formalism, we make use of linear inequalities. Since the variable α_ψ represents the probability of each $\psi \in \mathsf{InsidePr}(\Psi)$, we can represent a probabilistic atom $q_1 \cdot \mathsf{Pr}(\psi_1) + \ldots + q_\ell \cdot \mathsf{Pr}(\psi_\ell) \bowtie q$ as a linear arithmetic formula of the form $q_1 \cdot \alpha_{\psi_1} + \ldots + q_\ell \cdot \alpha_{\psi_\ell} \bowtie q$. This translation, denoted by PrToLIRA, can be inductively extended to probabilistic formulas (which are Boolean combinations of probabilistic atoms):

- $\mathsf{PrToLIRA}(q_1 \cdot \mathsf{Pr}(\psi_1) + \ldots + q_\ell \cdot \mathsf{Pr}(\psi_\ell) \bowtie q)$ is the assertion $q_1 \cdot \alpha_{\psi_1} + \ldots + q_\ell \cdot \alpha_{\psi_\ell} \bowtie q$;
- $\mathsf{PrToLIRA}(\neg\delta)$ is the assertion $\neg\mathsf{PrToLIRA}(\delta)$;
- $\mathsf{PrToLIRA}(\delta_1 \wedge \delta_2)$ is the assertion $\mathsf{PrToLIRA}(\delta_1) \wedge \mathsf{PrToLIRA}(\delta_2)$.

All together now: To verify the satisfiability of the GGenPSAT instance, we will need to satisfy the following constraints:

(hard_constr) $\bigwedge_{\varphi \in \Gamma} \alpha_\varphi = 1$;

(soft_constr) $\bigwedge_{\delta \in \Psi} \mathsf{PrToLIRA}(\delta)$;

(cons) $h_{ik} = 1 \leftrightarrow \mathfrak{G}_i^{(k)}$ for each $i \in \{1, ..., |\mathfrak{G}|\}$, $k \in \{1, ..., |\mathfrak{G}| + 1\}$;

(val1) $\sum_{j=1}^{|\mathfrak{G}|+1} b_{ij} = \alpha_{\psi_i}$ for each $i \in \{1, \ldots, |\mathfrak{G}|\}$;

(val2) $(0 \leq b_{ij} \leq h_{ij}) \wedge (h_{ij} - 1 + \pi_j \leq b_{ij} \leq \pi_j)$ for each $i \in \{1, \ldots, |\mathfrak{G}|\}$, $j \in \{1, \ldots, |\mathfrak{G}| + 1\}$;

(sums1) $\sum_{j=1}^{|\mathfrak{G}|+1} \pi_j = 1$;

(prop_prob) $\bigwedge_{k=1}^{|\mathfrak{G}|+1} \left(\mathfrak{G}_i^{(k)} \leftrightarrow \psi_i^{(k)}\right)$ for each $i \in \{1, ..., |\mathfrak{G}|\}$.

All these restrictions amount to:

- 3 assertions from (hard_constr), (soft_constr) and (sums1);
- $2 \cdot |\mathfrak{G}| \cdot (|\mathfrak{G}| + 1)$ assertions from (cons) and (val2);
- $2 \cdot |\mathfrak{G}|$ assertions from (val1) and (prop_prob).

Hence, we have a total of $\mathcal{O}(|\mathfrak{G}| \cdot (|\mathfrak{G}| + 1))$ assertions, each of polynomial size on the length of (Γ, Ψ) over:

- $|\mathfrak{G}| \cdot (|\mathfrak{G}| + 1)$ binary variables h_{ij};
- $|\mathfrak{G}| \cdot (|\mathfrak{G}| + 1)$ real variables b_{ij};
- $|\mathfrak{G}|$ real variables $0 \leq \alpha_{\psi_i} \leq 1$;
- $|\mathfrak{G}| + 1$ real variables $0 \leq \pi_j \leq 1$;
- $(|\mathfrak{G}| + 1) \cdot (|\mathfrak{G}| + n)$ propositional variables in $\widetilde{\mathcal{B}}$.

With this, we can conclude that the presented procedure translates a GGenPSAT instance into a problem in QF_LIRA of polynomial size.

The solver: We test the satisfiability of a GGenPSAT instance (Γ, Ψ) by translating it to a QF_LIRA problem and then solving the latter appropriately. The procedure presented in Algorithm 1, begins by initializing an empty QF_LIRA problem and uses the following auxiliary procedures:

- assert$(\cdot)$ introduces an assertion to the QF_LIRA problem;
- PrToLIRA$(\cdot)$ translates probabilistic formulas into QF_LIRA assertions;
- qf_lira_solver() returns SAT or UNSAT depending on whether the problem is satisfiable or not.

When the resulting QF_LIRA problem is satisfiable, we conclude that (Γ, Ψ) is a satisfiable GGenPSAT instance.

Algorithm 1 GGenPSAT solver based on SMT−QF_LIRA

procedure GGenPSAT($\{x_i\}_{i=1}^n$, Γ, Ψ)
  assume: $\mathfrak{G} = \{\mathfrak{p}_\psi \mid \psi \in \mathsf{RelF}\}$
  declare: propositional variables: $\widetilde{\mathcal{B}} = \bigcup_{k=1}^{|\mathfrak{G}|+1} \mathcal{B}^{(k)}$
    binary variables: h_{ij}
    $[0, 1]$-variables: α_{ψ_i}, π_j, b_{ij}
  for $i = 1$ to $|\mathfrak{G}|$ **do**
    assert$(\sum_j b_{ij} = \alpha_{\psi_i})$ ▷ (val1)
    assert$(\bigwedge_k (\mathfrak{G}_i^{(k)} \leftrightarrow \psi_i^{(k)}))$ ▷ (prop_prob)
    for $j = 1$ to $|\mathfrak{G}| + 1$ **do**
      assert$(h_{ij} = 1 \leftrightarrow \mathfrak{G}_i^{(j)})$ ▷ (cons)
      assert$(0 \leq b_{ij} \leq h_{ij})$ ▷ (val2)
      assert$(h_{ij} - 1 + \pi_j \leq b_{ij} \leq \pi_j)$ ▷ (val2)
  assert$(\bigwedge_\varphi \ \alpha_\varphi = 1)$ ▷ (hard_constr)
  assert$(\bigwedge_\delta \ \mathsf{PrToLIRA}(\delta))$ ▷ (soft_constr)
  assert$(\sum \pi_i = 1)$ ▷ (sums1)
  return qf_lira_solver()

Proposition 1. *A* GGenPSAT *instance* (Γ, Ψ) *is satisfiable iff Algorithm 1 returns* Sat.

Proof. Assume that a GGenPSAT instance (Γ, Ψ) is satisfiable. Then, there exists a probability distribution ρ over the set of valuations $\mathcal{V}^*$ satisfying (Γ, Ψ). Our goal is to present a model for the QF_LIRA problem obtained by the translation of (Γ, Ψ) describe above. We denote the obtained solutions by $\alpha^*_\psi, \pi^*_j, b^*_{ij}$ and h^*_{ij} and construct a valuation $\widetilde{v}$ over the extended set of propositional variables $\widetilde{\mathcal{B}}$.

For each $\mathfrak{p}_\psi \in \mathfrak{G}$, let α^*_ψ be the probability of the propositional variable $\mathfrak{p}_\psi$ induced by the probability distribution ρ in the following manner:

$$\alpha^*_\psi = \sum_{v:v(\psi)=1} \rho(v) \ . \tag{5}$$

Then, consider the algebraic formulation as in (3):

$$\begin{cases} V\pi = \alpha^* \\ \sum \pi_j = 1 \\ \pi \geq 0 \end{cases} \tag{6}$$

where now the vector $\alpha^* = [\alpha^*_{\psi_i}]$ is defined as in (5) and

- $V = [V_{ij}]$ is a matrix of size $|\mathfrak{G}| \times 2^n$, where V_{ij} is defined from the j^{th} valuation $v_j \in \mathcal{V}^*$ and from the i^{th} propositional ghost variable $\mathfrak{G}_i$, by $V_{ij} = v_j(\psi_i)$;
- $\pi = [\pi_j]$ is a vector of size 2^n, where each π_j is a real $[0, 1]$-variable representing the probability valuation v_j.

Note that $\rho^* = [\rho_j]$ where $\rho_j = \rho(v_j)$ is a solution for (6). By Lemma 1, there exists a matrix $H = [h^*_{ij}]$ composed by $|\mathfrak{G}| + 1$ columns of V such that

$$\begin{cases} H\pi^* = \alpha^* \\ \pi^* \geq 0 \end{cases} \tag{7}$$

where $h^*_{|\mathfrak{G}|+1,j} = 1$ for each j and π^* corresponds to the appropriate entries of ρ^*.

Since π^* is also a probability distribution, the assertion (sums1) is satisfied and, considering $b^*_{ij} = h^*_{ij} \cdot \pi^*_j$, the assertions (val1) and (val2) are also satisfied.

The propositional valuation $\widetilde{v}$ of the variables in $\widetilde{\mathcal{B}}$ inherent to the QF_LIRA model is defined in the following manner:

- $\widetilde{v}(x_i^{(k)}) = v_k(x_i)$;
- $\widetilde{v}(\mathfrak{p}_i^{(k)}) = v_k(\psi_i)$.

This implies that (prop_prob) is satisfied since the valuation $\widetilde{v}$ atributes the same truth value to $\mathfrak{p}_i^{(k)}$ and $\psi_i^{(k)}$. Furthermore, the assertion (cons) is also satisfied as the truth value of $\mathfrak{G}_i^{(k)}$ is given by h^*_{ik}.

Provided that the original probability distribution ρ satisfies the GGenPSAT instance, we immediately conclude that (hard_constr) and (soft_constr) are satisfied, which concludes the proof of the direct implication.

Reciprocally, assume that the associated QF_LIRA problem is satisfiable, and consider the components of its model: a valuation $\widetilde{v}$ of the variables in $\widetilde{\mathcal{B}}$, and define by y^* the value that the model gives to the variable y. Our aim, is to define a probability distribution ρ over the set of valuations $\mathcal{V}^*$ of the variables in $\mathcal{P}$.

With this purpose, we will refine the valuation $\widetilde{v}$, reduce it to valuations over $\mathcal{B}$, and finally define the probability distribution ρ. Since $\widetilde{v}$ is a valuation over $\widetilde{\mathcal{B}} = \bigcup_{k=1}^{|\mathfrak{G}|+1} \mathcal{B}^{(k)}$, define its reduct to each copy of $\mathcal{B}$, $v_k(p) = \widetilde{v}(p^{(k)})$, for each $p \in \mathcal{B}$. Let $W = \{v_1, \ldots, v_{|\mathfrak{G}|+1}\}$ be the set of such valuations. Then, consider the probability distribution $\pi : W \to [0, 1]$ defined as $\pi(v_k) = \pi^*_k$. The probability distribution $\rho : \mathcal{V}^* \to [0, 1]$ we seek is now easily defined recalling that $\mathcal{B} = \mathcal{P} \cup \mathfrak{G}$:

$$\begin{cases} \rho(v_{i|_\mathcal{P}}) = \pi^*_i & \text{for each } i \in \{1, \ldots, |\mathfrak{G}| + 1\} \\ \rho(v) = 0 & \text{otherwise} \end{cases}$$

We now need to check that this valuation is well defined, i.e., that if $v_{i|_\mathcal{P}} = v_{j|_\mathcal{P}}$ then $\pi^*_i = \pi^*_j$. We will do this by showing that if $v_i \neq v_j$ then $v_{i|_\mathcal{P}} \neq v_{j|_\mathcal{P}}$: if for some $x \in \mathcal{P}$, $v_i(x) \neq v_j(x)$ then obviously their reducts to $\mathcal{P}$ will also differ in x; on the other hand, if for every $x \in \mathcal{P}$, $v_i(x) = v_j(x)$, and there is a $\mathfrak{p} \in \mathfrak{G}$ such that $v_i(\mathfrak{p}) \neq v_j(\mathfrak{p})$ we obtain a contradiction – let ψ be the propositional formula corresponding to $\mathfrak{p}$. Since $v_i(x) = v_j(x)$ for every $x \in \mathcal{P}$, then $v_i(\psi) = v_j(\psi)$ which means that $\widetilde{v}(\psi^{(i)}) = \widetilde{v}(\psi^{(j)})$. Since $\widetilde{v}$ satisfies (prop_prob), this means that $\widetilde{v}(\mathfrak{p}^{(i)}) = \widetilde{v}(\mathfrak{p}^{(j)})$ and so $v_i(\mathfrak{p}) = v_j(\mathfrak{p})$.

Since (sums1) is satisfied, ρ constitutes a well-defined probability distribution. To conclude that the hard constraints are satisfied, observe that for each $\varphi \in \Gamma$, since $\alpha^*_\varphi = 1$ it follows that $h^*_{ij} = 1$ for each j such that $\pi^*_j > 0$ and by (cons) and (prop_prob) it means that ρ satisfies φ with probability 1. For soft constraints, the reasoning is similar – observe that (cons) and (prop_prob) links the algebraic reasoning with the valuations. To show that ρ satisfies Ψ, i.e., the soft constraints, since the assertion (soft_constr) is satisfied, it is enough to show that $\mathsf{Pr}(\psi)$ coincides with α^*_i, where ψ is the i-th formula of $\mathsf{InsidePr}(\Psi)$. In fact, the probability of ψ

$$\begin{aligned} \mathsf{Pr}(\psi) &= \sum_{v \in \mathcal{V}^*} v(\psi) \cdot \rho(v) = \sum_{v \in W} v(\psi) \cdot \rho(v) \\ &= \sum_{j=1}^{|\mathfrak{G}|+1} v_j(\psi) \cdot \pi^*_j = \sum_{j=1}^{|\mathfrak{G}|+1} \widetilde{v}(\psi^{(j)}) \cdot \pi^*_j \\ &= \sum_{j=1}^{|\mathfrak{G}|+1} h^*_{ij} \cdot \pi^*_j \overset{\triangleright}{=} \sum_{j=1}^{|\mathfrak{G}|+1} b^*_{ij} = \alpha^*_i \ . \qquad \triangleright \text{ by (val2)} \end{aligned}$$

We detail the second to last step of the deduction: either h^*_{ij} is 0 and by the first (val2) assertion then b^*_{ij} is also 0 and so $h^*_{ij} \cdot \pi^*_j = 0 = b^*_{ij}$; or h^*_{ij} is 1 and by the (val2) assertions we obtain that b^*_{ij} equal to π^*_j and so $h^*_{ij} \cdot \pi^*_j = 1 \cdot \pi^*_j = b^*_{ij}$.

This shows that ρ satisfies the formulas in Ψ and so it is a model for the GGenPSAT instance (Γ, Ψ). □

5 Phase Transition

In this section we describe the open-source tool [Caleiro *et al.*, 2016c] developed to implement the algorithm that solves the GGenPSAT problem. With this in hands, we generate batches of random GGenPSAT instances and study the behavior of the implemented solver, in terms of time and satisfiability.

For this, we measure the proportion of satisfiable instances as well as the average time the solver spent to solve them. The software was written in Python, and we used Yices [Dutertre, 2014], version 2.5.1, to solve the SMT problem. Our tool takes as input a GGenPSAT written in an (smt-lib)-style notation enriched with the probability operator `(pr` φ`)`. As an example, the second problem from Example 1 can be formulated as follows:

```
(define x::bool)
(define y::bool)
(assert (>= (pr (not (xor x y))) (* 2 (pr (xor x y)))))
(assert (> (pr (xor x y)) 0))
(assert (or (= (pr y) 0) (= (pr y) 1)))
(assert (= (pr x) 0))
(check)
```

The machine used for the tests was a Mac Pro at 3.33 GHz 6-Core Intel Xeon with 6 GB of memory.

A phase transition behavior is characterized by a sharp transition between two clearly distinct states. Regarding satisfiability problems, these states correspond to states in which the problems are either satisfiable or not satisfiable. In [Gent and Walsh, 1994], this behavior was studied for random 3SAT problems and, heuristically, shown that the ratio m/n of number of clauses over the number of variables characterizes the phase transition. That is, there is a value (close to 4.3 on 3SAT) of m/n for which the random problems rapidly transition from being satisfiable to not being satisfiable. It is also noteworthy that the harder random instances lie in this critical area, as we can heuristically observe a peak in time taken to solve problems.

We now present some results regarding the behavior of random instances of 3SATwhen embedded in GGenPSAT. The embedding of PSAT and GenPSAT was also studied but not included due to size limitations. Besides revealing the phase transition behavior, our results show that the presented solver is more efficient than the solver for GenPSAT, [Caleiro *et al.*, 2016b]. This is not unexpected since the translation used here for the GGenPSAT problem is much more natural and concise than the one used for GenPSAT.

We denote by n the number of variables and m the number of propositional clauses. In random 3SAT instances, we detect the phase transition when m/n is close to 4.3 as previously detected in [Gent and Walsh, 1994], see Figure 2.

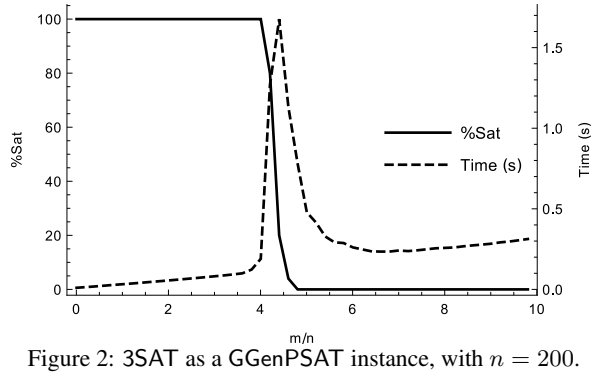

Figure 2: 3SAT as a GGenPSAT instance, with $n = 200$.

As expected, being an NP-complete problem, the full GGenPSAT problem also exhibits a phase transition behavior. We generated the random GGenPSAT instances as follows: a random 3SAT instance with n variables and m clauses is generated and then, each variable x_i is replaced by a problem G_i which is a conjunction of k random probabilistic atoms over n variables. The results are seen in Figure 3.

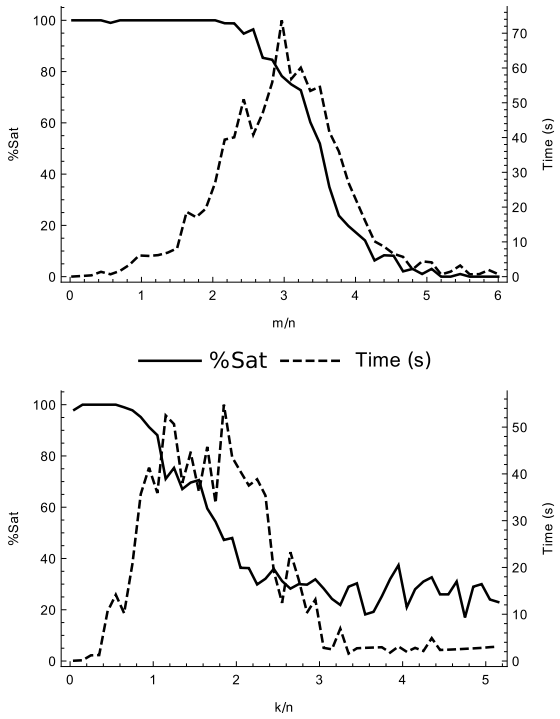

Figure 3: Random GGenPSAT instances with $n = 30$ and $k = 2$ (above) and $n = 20$ and $m = 10$ (below).

In summary, the developed tool (due to the size of the translation) is able to solve reasonably sized instances, surpassing as well the GenPSAT dedicated tool developed in [Caleiro *et al.*, 2016a] for those instances. Given this, we are able to detect the phase transition behavior and heuristically determine parameters for which random instances are hard to solve.

6 Conclusions and Future Work

In this paper, we aimed to study a generalization of the probabilistic satisfiability problem. The GGenPSAT problem naturally models Boolean combinations of linear inequalities involving probabilities of propositional formulas. We developed a satisfiability procedure by a polynomial reduction to the quantifier-free theory of integer and real arithmetic, and proved its correctness. We also implemented a tool that translates problems in GGenPSAT to QF_LIRA and solves them with an off-the-shelf SMT solver such as Yices. With this tool in hands we are able to detect and study the phase-transition behavior of this problem. It is also worth noting that since the expressiveness of the GGenPSAT problem coincides with the probabilistic logic of Fagin et al. [Fagin *et al.*, 1990], this tool also serves as a satisfiability procedure for the logic.

We believe the study of this problem and subsequent tool implementation provides a sound foundational basis to the development of applications where probabilistic reasoning is required. As future work, we aim to develop applications of this tool for the analysis of cryptographic protocols, specially related to the existence of side-channel attacks.

References

[Barrett *et al.*, 2016] C. Barrett, P. Fontaine, and C. Tinelli. The Satisfiability Modulo Theories Library (SMT-LIB). `www.SMT-LIB.org`, 2016.

[Bona *et al.*, 2015] G.D. Bona, F. G. Cozman, and M. Finger. Generalized probabilistic satisfiability through integer programming. *Journal of the Brazilian Computer Society*, 21(1):1–14, 2015.

[Boole, 1853] G. Boole. *Investigation of The Laws of Thought On Which Are Founded the Mathematical Theories of Logic and Probabilities*. 1853.

[Caleiro *et al.*, 2016a] C. Caleiro, F. Casal, and A. Mordido. Generalized probabilistic satisfiability. In *LSFA 2016 - 11th Workshop on Logical and Semantic Frameworks, with Applications*, 2016.

[Caleiro *et al.*, 2016b] C. Caleiro, F. Casal, and A. Mordido. GenPSAT solver, 2016. Available online at https://github.com/fcasal/genpsat.git.

[Caleiro *et al.*, 2016c] C. Caleiro, F. Casal, and A. Mordido. GGenPSAT solver, 2016. Available online at https://github.com/fcasal/ggenpsat.git.

[Chvátal, 1983] V. Chvátal. *Linear programming*. Macmillan, 1983.

[Cook, 1971] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[De Moura and Bjørner, 2011] L. De Moura and N. Bjørner. Satisfiability modulo theories: introduction and applications. *Communications of the ACM*, 54(9):69–77, 2011.

[Dutertre, 2014] B. Dutertre. Yices 2.2. In Armin Biere and Roderick Bloem, editors, *CAV'2014*, volume 8559 of *Lecture Notes in Computer Science*, pages 737–744. Springer, 2014.

[Fagin *et al.*, 1990] R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Inf. Comput.*, 87(1-2):78–128, 1990.

[Finger and Bona, 2011] M. Finger and G.D. Bona. Probabilistic satisfiability: Logic-based algorithms and phase transition. In *IJCAI*, pages 528–533. IJCAI/AAAI, 2011.

[Finger and Bona, 2015] M. Finger and G.D. Bona. Probabilistic satisfiability: algorithms with the presence and absence of a phase transition. *Annals of Mathematics and Artificial Intelligence*, 75(3):351–389, 2015.

[Gent and Walsh, 1994] I. P. Gent and T. Walsh. *The hardest random SAT problems*. Springer, 1994.

[Georgakopoulos *et al.*, 1988] G. Georgakopoulos, D. Kavvadias, and C. H Papadimitriou. Probabilistic satisfiability. *Journal of Complexity*, 4(1):1 – 11, 1988.

[King, 2014] T. King. *Effective Algorithms for the Satisfiability of Quantifier-Free Formulas Over Linear Real and Integer Arithmetic*. PhD thesis, Courant Institute of Mathematical Sciences New York, 2014.

[Nilsson, 1986] N. J. Nilsson. Probabilistic logic. *Artif. Intell.*, 28(1):71–88, 1986.