Generalized Probabilistic Satisfiability

Glauber De Bona
Instituto de Matemática e Estatística
Universidade de São Paulo, Brazil
Email: debona@ime.usp.br

Fábio G. Cozman
Escola Politécnica
Universidade de São Paulo, Brazil
Email: fgcozman@usp.br

Marcelo Finger
Instituto de Matemática e Estatística
Universidade de São Paulo, Brazil
Email: mfinger@ime.usp.br

***Abstract*—This paper studies the Generalized Probabilistic Satisfiability (GPSAT) problem, where the probabilistic satisfiability problem is extended by allowing Boolean combinations of probabilistic assertions and nested probabilistic formulas. We introduce a normal form for this problem and show that nesting of probabilities does not increase the expressivity in GPSAT. An algorithm to solve GPSAT problems via mixed integer programming is proposed, and its implementation shows evidence of phase-transition phenomena.**

I. Introduction

Propositional logic and probability theory stand as major knowledge representation tools in many fields, and notably in artificial intelligence. Useful combinations of propositional logic and probability theory are already pursued by Boole [3, Chapter XVIII], who was concerned with problems where propositional formulas are associated with probability assertions. Loosely speaking, we have propositional sentences $\{\phi_i\}_{i=1}^q$, each containing a subset of atomic propositions $\{A_j\}_{j=1}^n$. We may associate one or more of these sentences with probabilities, writing for instance $P(\phi_i) = \alpha_i$. To establish semantics for these assessments, we consider a probability measure over the set of truth assignments. The *Probabilistic Satisfiability (PSAT)* problem is to determine whether it is possible to find a probability measure over truth assignments such that all assessments are satisfied [11]. PSAT problems have received attention in a variety of fields [4], [12], [13], [7], [15]; in artificial intelligence research, PSAT problems appear as a foundation for probabilistic rules [19] and first-order probabilistic logic [14], [18], [20].

In this paper we consider an extended version of PSAT problems. To understand our goal, note that the satisfiability of probabilistic formulas $P(\phi_i) = \alpha_i$, $1 \leq i \leq q$ obtains when there is a probability measure (over the truth assignments) that satisfies

$$P(\phi_1) = \alpha_1 \wedge P(\phi_2) = \alpha_2 \wedge \cdots \wedge P(\phi_q) = \alpha_q$$

. However flexible this may be, PSAT problems can only handle conjunction of probability assessments. One obvious generalization is to deal with disjunction and negation of probabilistic assertions to construct more complex formulas, such as

$$\neg(P(\phi_1) = \alpha_1) \wedge (P(\phi_2) = \alpha_2) \vee (P(\phi_3) = \alpha_3).$$

We will see that the same semantics used for PSAT can be adopted in such an extended probabilistic satisfiability problem.

We can move to an even more expressive language by allowing probabilistic formulas to be nested; that is, by allowing a subformula of ϕ in $P(\phi) = \alpha$ to be $P(\varphi) = \alpha'$. The problem of deciding satisfiability of these formulas is what we define as *Generalized Probabilistic Satisfiability (GPSAT)*. The resulting language can be viewed as the closure of probabilistic formulas with respect to Boolean and probabilistic operators.

To give meaning to nested probabilities, we use the assumption that "one believes the probability of ϕ is α if, and only if, one believes that the probability of the probability of ϕ being α is 1" [21]. Clearly, this assumption is restrictive, in the sense that it claims an agent may not have uncertainty about his own probabilistic beliefs. However, maybe surprisingly, under this premise we can prove that satisfiability of formulas with nested probabilities can be reduced to satisfiability without nesting.

There are no algorithms currently available to determine satisfiability of GPSAT problems, as far as we know. In this paper we present the first such algorithm. The most direct way to solve a PSAT problem is through its linear programming formulation [12], using column generation methods to handle the exponential number of columns [17]. A recent alternative approach reduces PSAT into logical satisfiability [9]. Neither of these approaches is easily extended to deal with disjunction of contraints, that is essential to solve GPSAT.

In this paper we present an approach to Generalized Probabilistic Satisfiability, where the original problem is written as an integer linear program of a size that is polynomial on the size of the original problem. This technique were firstly proposed to solve Probabilistic Satisfiability in [6], and this paper builds on that work to deal with GPSAT problems.

Section II summarizes necessary background on SAT and PSAT. The Generalized Probabilistic Satisfiability is defined in Section III, where a normal form is introducted. An algorithm for GPSAT is described in Section IV, building on a reduction from PSAT to mixed integer programming. Implementation and experiments, with a discussion of phase transitions, are presented in Section V.

II. SAT and PSAT

Consider n atomic propositions A_j and q sentences ϕ_i in propositional logic. If a truth assignment ω is such that sentence ϕ is True, write $\omega \models \phi$; if ϕ is False, we write $\omega \not\models \phi$. The Satisfiability (SAT) problem is to determine whether or not there exists a truth assignment to all variables such that all sentences evaluate to True [5], [10]. If every sentence ϕ_i is a conjunction of clauses, then we have a SAT problem in *CNF*. A SAT problem in CNF is a k-SAT problem when each clause has k literals. The 2-SAT problem has a polynomial solution, while k-SAT is NP-complete for $k > 2$.

Suppose now we have q propositional sentences, say ϕ_1 to ϕ_q, associated with probabilities through *assessments* of the form $P(\phi_i) \bowtie \alpha_i$, where $\bowtie$ is one of $\geq, =, \leq$. The semantics of such an assessment is as follows. Take the set of 2^n truth assignments that can be generated for the n propositions. A probability measure P over this set satisfies the assessments if, for each assessment $P(\phi_i) \bowtie \alpha_i$,

$$\sum_{\omega \models \phi_i} P(\omega) \bowtie \alpha_i. \tag{1}$$

The Probabilistic Satisfiability (PSAT) problem is to determine whether a given set of probabilistic assessments can be satisfied in the sense that there is a probability measure over truth assignments such that all assessments are satisfied. PSAT is know to be in NP, as it has a small model, and since SAT is a subproblem where all assigned probabilities are 1, PSAT is NP-complete [11]. A few polynomial special cases of PSAT are known [1].

There are many proposed algorithms for PSAT. The most obvious one is to write down q constraints of the form (1), one for each sentence, associated with assignments $P(\phi_i) \bowtie \alpha_i$. Each constraint can be written as

$$\sum_{j=1}^{2^n} I_{\phi_i}(\omega_j) P(\omega_j) \bowtie \alpha_i, \text{ where } I_{\phi_i}(\omega_j) = \begin{cases} 1 & \text{if } \omega_j \models \phi_i \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

while truth assignments ω_j are ordered from 1 to 2^n (say by the n-bit binary number obtained by writing 0 for False and 1 for True as assigned to $A_1, \ldots, A_n$). Probabilistic Satisfiability is then obtained when the resulting set of linear constraints has a solution. The challenge is that we have 2^n truth assignments, so the size of the linear constraints is exponential in the input.

The most efficient algorithms for PSAT apply linear programming tecniques to this set of contraints. As linear program, if there is a solution, there is one solution with no more than $q+1$ truth assignments with positive probability [11]. The two-phase simplex method can be used with the addition of $q+1$ artificial variables in order to find a feasible solution in the first phase. Starting with a basis with these $q+1$ artificial variables, at each iteration a new column (variable) enter the basis, keeping the solution feasible, until the basis has no artificial variables. If this point is reached, a solution was found; otherwise the linear program has no feasible solution and the PSAT problem is unsatisfiable. As the number of columns is exponential, column generation techniques are used. A good survey on this approach is [15]. Combining inference rules with linear programming techniques leads to the currently most efficient algorithms, as showed in [16].

III. Generalized Probabilistic Satisfiability

The language we contemplate is syntactically easy to specify: simply take Boolean operations over assessments, and allow a subformula in an assessment to be itself any well-formed formula of the language. That is, we allow sentences such as

$$(P(A_1 \wedge (A_2 \vee \neg A_3)) \geq 1/2) \vee (P(A_3) \leq 1/4) \vee (P(A_1) = 1/5),$$

$$(P(A_1) < 1/3) \vee P(A_2 \wedge (P(A_3) > 3/4)) > 1/2,$$

both of which contain a single "clause" of assessments (a disjunction of assessments). Another example is the following, where we find two "clauses" of assessments:

$$((P(A_1) = 1/3) \vee \neg(P(A_2) < 3/5)) \quad \wedge$$
$$((P(\neg A_1) \geq 2/3) \vee (P(A_1 \vee \neg A_3) \neq 3/7)).$$

To make sense of these sentences, we must establish an appropriate semantics. Note that in any PSAT problem the truth value of an assessment such as $P(\phi_i) = \alpha_i$ is derived from a probability measure on the truth assignments $\omega_1, \ldots, \omega_{2^n}$ (because the relation $\omega_j \models \phi_i$ is well defined). This is not the case when nested probabilities are allowed and ϕ_i may contain probabilistic subformulas. This next section shows how to define a semantics that characterizes when a probabilistic assessment is true in a possible world ω_j.

A. Syntax and Semantics

We start with an infinite set of atomic (or primitive) propositions $A = \{A_1, A_2, A_3, \ldots\}$. Our language is then defined recursively as the smallest set $\mathcal{L}$ such that:[1]

- If $\phi \in A$, then $\phi \in \mathcal{L}$;
- If $\phi \in \mathcal{L}$, then $\neg\phi \in \mathcal{L}$;
- If $\phi \in \mathcal{L}$ and $\theta \in \mathcal{L}$, then $(\phi \vee \theta) \in \mathcal{L}$;
- If $\phi \in \mathcal{L}$ and $\theta \in \mathcal{L}$, then $(\phi \wedge \theta) \in \mathcal{L}$;
- If $\phi \in \mathcal{L}$, then $(P(\phi) \bowtie \alpha) \in \mathcal{L}$, for $\bowtie \in \{\leq, \geq\}$ and $\alpha \in [0,1]$.

Parentheses are omitted whenever possible. As usual, $P(\phi) = \alpha$, $P(\phi) < \alpha$ and $P(\phi) > \alpha$ are abbreviations for $(P(\phi) \leq \alpha) \wedge (P(\phi) \geq \alpha)$, $\neg(P(\phi) \geq \alpha)$ and $\neg(P(\phi) \leq \alpha)$ respectively.

To give truth values to formulas in $\mathcal{L}$, we define a structure $\mathcal{M} = (\Omega, P(.))$, where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{2^n}\}$ is the set of propositional truth assignments (valuations), and $P : \Omega \rightarrow [0,1]$ is a probability distribution over Ω. With this machinery in hand, we now define when a pair $(\mathcal{M}, \omega)$ satisfies a formula $\phi \in \mathcal{L}$, or $(\mathcal{M}, \omega) \models \phi$:

- $(\mathcal{M}, \omega) \models A_i$ if $\omega \models A_i$;
- $(\mathcal{M}, \omega) \models \neg\phi$ if $(\mathcal{M}, \omega) \not\models \phi$;
- $(\mathcal{M}, \omega) \models \phi_1 \wedge \phi_2$ if $(\mathcal{M}, \omega) \models \phi_1$ and $(\mathcal{M}, \omega) \models \phi_2$;
- $(\mathcal{M}, \omega) \models \phi_1 \vee \phi_2$ if $(\mathcal{M}, \omega) \models \phi_1$ or $(\mathcal{M}, \omega) \models \phi_2$;
- $(\mathcal{M}, \omega) \models P(\phi) \bowtie \alpha$ if $\sum\{P(\omega_i) | (\mathcal{M}, \omega_i) \models \phi\} \bowtie \alpha$, for $\alpha \in \{\leq, \geq\}$;

This semantics makes propositional formulas and probabilistic formulas have their truth value associated to possible worlds $(\mathcal{M}, \omega)$. This means one can talk about the truth value of formulas such as

$$\phi_1 \wedge (P(\phi_2) \leq 0.5) \vee (P(\neg(P(\phi_3) \geq 0.7)) \leq 0.2)$$

[1]For simplicity, we left the connective $\rightarrow$ outside the language, without loss of generality.

in a given pair $(\mathcal{M}, \omega)$. However, as the truth value of $P(\phi) \bowtie \alpha$ in $(\mathcal{M}, \omega)$ does not depend on ω, $P(\phi) \bowtie \alpha$ will be true either in all worlds $(\mathcal{M}, \omega)$ or in none of them – this captures the intuition that nested probabilities collapse to either 1 or 0.

Given a formula $\phi \in \mathcal{L}$, we say that it is *satisfiable* if there is a pair $(\mathcal{M}, \omega)$, where $\mathcal{M} = \{\Omega, P(.)\}$ and $\omega \in \omega$, such that $(\mathcal{M}, \omega) \models \phi$; otherwise we say ϕ is *unsatisfiable*.

The *Generalized Probabilistic Satisfiability (GPSAT)* is then the problem of deciding whether or not a given $\phi \in \mathcal{L}$ is satisfiable.

The logic presented by Fagin, Halpern and Megido in [7] had alredy dealt with Boolean combinations of probabilistic assessments, but they kept the probabilities applied only to pure propositional formulas, hence avoiding nesting. Their semantics assigns truth values only to probability assessments through a whole structure $\mathcal{M}$, and not a pair $(\mathcal{M}, \omega)$. So their semantics is closer to PSAT's, even though they use probability "inside" the language. Another difference is that Fagin et al. allow more general assessments assessments of the form $a_1 P(\alpha_1) + a_1 P(\alpha_1) + \cdots + a_m P(\alpha_m) \geq \alpha$ (this enabled them to axiomatize their logic). They show that the corresponding satisfiability problem is NP-complete, but do not propose algorithms to solve it. One of our goals here is to provide concrete algorithms.

In another work, Fagin and Halpern [8] investigated a more general logic to reason about knowledge and probabilities. Its probabilistic semantics is similar to ours because the probabilitic formulas have truth value in specific states (worlds), but they introduce a probability distribution for each set of indistinguishable possible worlds (states). Additionally, their logic has an epistemic modal relation, so that multi-agent and linear combinations of probabilities are also allowed. Again, they axiomatize their logic, show the complexity of the related decision procedure, but do not provide algorithms.

The logic we propose here can be seen as a particular case of the logic in [8], where there is only one agent and all possible worlds are indistinguishable (all with identical probability distributions). It follows that GPSAT inherits an NP upper bound proved by Fagin and Halpern (Theorem 4.6 in [8]); and as PSAT is a subproblem of GPSAT, the latter is also NP-complete.

B. GPSAT Normal Form

The algorithm for GPSAT to be proposed decides satisfiability only for formulas without nested probabilities and with all propositional formulas in a probability assessment. That is, the algorithm cannot handle a formula such as $P(P(\phi))$ or a formula such as $\phi_1 \wedge P(\phi_2)$. As we now show, any GPSAT problem can be reduced in polynomial time to a normal form that complies with these constraints and keeps satisfiability.

We say a formula $\phi \in \mathcal{L}$ is in *normal form* if it is the conjunction of two formulas $\Psi \wedge \Gamma$, where

- Ψ is the conjunction of 3-clauses over probabilistic assessments in the form $P(A_i) \geq \alpha$, in which each assessment is over a different atomic proposition A_i, and
- Γ is a probabilitic assignment $P(\gamma) \geq 1$, in which γ is a 3-SAT problem,

and where probability values that are smaller than one can be assigned only to atomic propositions, and additionally where every atomic proposition occurrence is a subformula of no more than one probabilistic assessment in Ψ.

An example of a formula in normal form is:

$$((P(A_1) \geq 0.3) \vee (P(A_2) \geq 0.1) \vee (P(A_3) \geq 0.4)) \quad \wedge \\ (P((A_1 \vee \neg A_2 \vee A_3) \wedge (A_1 \vee A_2 \vee \neg A_3)) \geq 1).$$

This normal form is based on the PSAT normal form introduced by Finger and De Bona [9], and, although it may seem quite restrictive, we can show that all formulas in $\mathcal{L}$ can be brought to the normal form. Before we apply techniques from Finger and De Bona, a sequence of intermediate results is needed.

First, we investigate the case of Boolean propositions outside the scope of any probability assessment. Consider the formula $\phi = A_1 \wedge (P(A_1) \leq 0)$. Note that ϕ is satisfied by $(\mathcal{M}, \omega_1)$, where $\mathcal{M} = (\{\omega_1, \omega_2\}, P(.))$, $\omega_1 \models A_1$, $\omega_2 \not\models A_1$, $P(\omega_1) = 0$ and $P(\omega_2) = 1$. It may seem counterintuitive that A_1 is satisfied by a world with zero probability, but in the language we can have a formula simultaneously possible and with zero probability. For instance, $\theta = A_1 \wedge \neg A_1 \wedge (P(A_1) \leq 0)$ is unsatisfiable, as $A_1 \wedge \neg A_1$ is impossible. If one wants that ϕ be satisfiable only if A_1 has probability greater than zero, this can be expressed as $\phi' = \neg(P(A_1) \leq 0) \wedge (P(A_1) \leq 0)$. That being said, the following lemma shows how a formula ϕ can be transformed, in polynomial time, into a formula θ with no propositions outside any probability assignment.

Lemma 1: For every $\phi \in \mathcal{L}$, there exists $\theta \in \mathcal{L}$ whose all pure propositional subformulas are subformulas of a probability assignment $P(\psi) \geq \alpha$ (or $P(\psi) \leq \alpha$), such that ϕ is satisfiable if, and only if, θ is; furthermore, θ is computed in linear time.

Proof: We assume that ϕ has n atomic propositions $A_1, \ldots, A_n$. Let I be the set of indexes of all atomic propositions A_i ocurring in ϕ outside the scope of a probability assignment. To build θ, for all $i \in I$, substitute $p(B_i) \geq 1$, where B_i is a fresh atomic proposition, for all ocurrences of A_i out of a probability assessment; this is done in linear time in the size of ϕ. Now we need to prove that θ is satisfiable iff ϕ is.

($\leftarrow$) Suppose ϕ is satisfied by a pair $(\mathcal{M}, \omega_{j*})$, with $\mathcal{M} = (\Omega, P(.))$. Create a structure $\mathcal{M}' = (\Omega', P'(.))$, where for all $\omega_j \in \Omega$ there exists a $\omega'_j \in \Omega'$ such that $\omega_j \models A_i$ iff $\omega'_j \models A_i$. Make $\omega'_j \models B_i$ iff $\omega'_{j*} \models A_i$ and $P(\omega_j) = P'(\omega'_j)$, for all $i \in I$ and $1 \leq j \leq |\Omega|$. Clearly, $(\mathcal{M}', \omega'_{j*}) \models P(B_i) = 1$ iff $(\mathcal{M}, \omega_{j*}) \models A_i$, so $(\mathcal{M}', \omega'_{j*}) \models \theta$.

($\rightarrow$) Suppose now that $(\mathcal{M}, \omega_{j*}) \models \theta$, with $\mathcal{M} = (\Omega, P(.))$. Build $\Omega' = \{\omega'_1, \ldots, \omega'_{2^n}\}$ with an ω'_j for each truth assignment on $A_1, \ldots, A_n$. For each A_i and each ω'_j, let x_{ij} denote the literal, either A_i or $\neg A_i$, such that $\omega'_j \models x_{ij}$; analogously, let y_{ij} be A_i or its negation such that $\omega_j \models y_{ij}$. Make $P'(\omega'_j) = \sum\{P(\omega_k) | x_{ij} = y_{ik}$, for all $1 \leq i \leq n\}$. Finally, there is a ω'_{i*} such that $\omega'_{i*} \models A_i$ iff $(\mathcal{M}, \omega_{j*}) \models p(B_i) = 1$; thus $(\mathcal{M}', \omega'_{i*}) \models \phi$, with $\mathcal{M}' = (\Omega', P'(.))$. ■

We now show how to transform a formula with nested probabilities into one without nesting.

Lemma 2: For every $\phi \in \mathcal{L}$, there exists $\theta \in \mathcal{L}$ whose all probability assignments are not subformulas of another probability assignment, such that ϕ is satisfiable if, and only if, θ is; furthermore, θ is computed in polynomial time.

Proof: To prove by induction, we show how to decrease the number of nested probabilities, keeping the satisfiability. Given a formula ϕ with nested probabilities, construct ϕ' by substituting a new atomic proposition B_i for a expression $P(\psi) \geq \alpha$ (or $P(\psi) \leq \alpha$) inside the scope of probability assessment. Define $\phi'' = \phi' \wedge ((P(B_i) \geq 1) \vee P(B_i) \leq 0)) \wedge (\neg(P(B_i) \geq 1) \vee (P(\psi) \geq \alpha)) \wedge ((P(B_i) \geq 1) \vee \neg(P(\psi) \geq \alpha))$. Now we need to prove that ϕ'' is satisfiable iff ϕ is.

($\leftarrow$) Suppose $(\mathcal{M}, \omega_{j*} \models)\phi$, with $\mathcal{M} = (\Omega, P(.))$. We can extend $\mathcal{M}$ to satisfy ϕ''. For each $\omega_j \in \Omega$, make $\omega_j \models B_i$ iff $(\mathcal{M}, \omega_{j*}) \models P(\psi) \geq \alpha$ and insert the other possible truth assignments in Ω, with zero probability mass, to form Ω'. Create a structure $\mathcal{M}' = (\Omega', P'(.))$, in which P' extends P. As either $\omega_j \models B_i$ for all $\omega_j \in \Omega'$ with $P'(\omega_j) > 0$, or $\omega_j \models B_i$ for all $\omega_j \in \Omega'$ with $P'(\omega_j) > 0$, then either $(\mathcal{M}', \omega_{j*} \models)P(B_i) \leq 0$ or $(\mathcal{M}', \omega_j \models)P(B_i) \geq 1$. Thus $(\mathcal{M}', \omega_{j*} \models)\phi''$

($\rightarrow$) Note that the last two clauses in ϕ'' state that $P(B_i) \geq 1 \leftrightarrow P(\psi) \geq \alpha$. Hence, if $(\mathcal{M}, \omega_{i*} \models)\phi''$, $(\mathcal{M}, \omega_{i*}) \models P(B_i) \geq 1$ iff $(\mathcal{M}, \omega_{i*}) \models P(\psi) \geq \alpha$. Then, due to the clause $((P(B_i) \geq 1) \vee P(B_i) \leq 0))$, $(\mathcal{M}, \omega_{i*}) \models B_i$ iff $(\mathcal{M}, \omega_{i*}) \models P(\psi) \geq \alpha$ Finally, as $(\mathcal{M}, \omega_{i*} \models)\phi'$, $(\mathcal{M}, \omega_{i*} \models)\phi$.

By iterating the process of building ϕ'', a formula θ without nested probabilities is reached. As the number of nested probabilities is linear in the size of ϕ, and each iteration takes no more than linear time in the size of ϕ, the whole process of building θ is polynomial in time. ■

Note that the process of eliminating probabilities nesting does not generate subformulas out of the scope of any probability assignment. So, by using Lemma 1 and Lemma 2 we have a formula which is a Boolean combination of probability assingments over pure propositional formulas – we call them probabilistic atoms. If each probabilistic atom of a formula ϕ is replaced by a new atomic proposition B_i, then we have a formula θ from the classical propositional logic. Using standard techniques, by adding new atoms, we can build a 3-SAT problem θ' which is (Boolean) satisfiable iff θ is. Replace the atomic propositions B_i by the corresponding probabilistic atoms and the new atomic propositions by probabilistic atoms of the form $P(C_i) \geq 1$, where C_i a fresh atomic proposition. Now we have a GPSAT problem ϕ' that is satisfiable only if ϕ is. For our normal form transformation procedure, we start from a formula like ϕ', which is the conjunction of 3-clauses, each formed from probabilistic atoms or their negation using only $\geq$. Observe that a probability assessment in the form $P(\psi_i) \leq \alpha_i$ is equivalent to the assignment $P(\neg\psi_i) \geq 1 - \alpha_i$.

Theorem 1: For all fomulas $\phi \in \mathcal{L}$, there is a formula $\theta \in \mathcal{L}$ in normal form that is satisfiable iff ϕ is; θ can be computed in polynomial time.

Proof: Given a formula ϕ with 3-clauses of probabilistic atoms, we construct Ψ from ϕ e Γ from scratch. For each $P(\psi_i) \geq \alpha_i$ in ϕ, substitute a new atomic proposition B_i for ψ_i to construct Ψ. Let Γ be assignment $P(\gamma) \geq 1$, where γ is the conjunction of the clauses corresponding to $(\psi_i) \leftrightarrow B_i$, $(\neg\psi_i \vee B_i) \wedge (\psi_i \vee \neg B_i)$, for all B_i introduced. We have built $\theta = \Psi \wedge \Gamma$ in polynomial time, it remains to prove that θ is satisfiable iff ϕ is.

($\leftarrow$) Suppose $(\mathcal{M}, \omega_{j*} \models)\phi$, with $\mathcal{M} = (\Omega, P(.))$. We can extend $\mathcal{M}$ to satisfy θ. For each $\omega_j \in \Omega$, make $\omega_j \models B_i$ iff $(\mathcal{M}, \omega_{j*}) \models \psi_i$ and insert the other possible truth assignments in Ω, with zero probability mass, to form Ω'. Create a structure $\mathcal{M}' = (\Omega', P'(.))$, in which P' extends P. Observe that $(\mathcal{M}', \omega_{j*} \models)P(B_i) \geq \alpha_i$ iff $(\mathcal{M}', \omega_j \models)P(\psi_i) \geq \alpha_i$. Thus $(\mathcal{M}', \omega_{j*} \models)\theta$.

($\rightarrow$) Now suppose $(\mathcal{M}, \omega_{j*} \models)\theta$, with $\mathcal{M} = (\Omega, P(.))$. As $(\mathcal{M}, \omega_{j*} \models)P((\neg\psi_i \vee B_i) \wedge (\psi_i \vee \neg B_i)) \geq 1$, for all $\omega_j \in \Omega$ with $P(\omega_j > 0)$ $(\mathcal{M}, \omega_j \models B_i$ iff $(\mathcal{M}, \omega_j \models \psi_i$, for all i. Hence $(\mathcal{M}, \omega_{j*} \models P(B_i) \geq \alpha_i$ iff $(\mathcal{M}, \omega_{j*} \models P(\psi_i) \geq \alpha_i$, for all i. Finally, $(\mathcal{M}, \omega_{j*} \models \phi$. ■

Solutions to a GPSAT problem in normal form $\Psi \wedge P(\gamma) = 1$ can be seen as solutions to Ψ restricted by γ, because all truth assignments with positive probability must satisfy γ. We usually represent GPSAT problem in normal form simply by the pair (Ψ, γ).

IV. GPSAT through Integer Programming

Assume our GPSAT problem is in Normal Form with q clauses, which with 3 assessments $\{P(A_j) \geq \alpha_j\}$ and a sentence γ in CNF with m clauses, each clause with 3 literals. So our problem is parameterized by the number of atomic propositions n, the number of clauses with probability assessments q, and the number of clauses m. Such a parameterized Normal Form neatly separates the probabilistic and the propositional aspects of Probabilistic Satisfiability.

Call a probabilistic atom or its negation a probabilistic literal. To satisfy a GPSAT formula, at least one probabilistic literal in each clause in Ψ must be satisfied. Hence, for the same reasons of PSAT, if a formula is satisfiable, a solution exists with only $q+1$ truth assignments with positive probability.

Suppose, without loss of generality, the i^{th} clause contains assignments over A_{3i-2}, A_{3i-1} and A_{3i}. Our problem is: find the $(q+1)$ truth assignments ω_j satisfying γ and the following disjunction of restriction, for $1 \leq k \leq q$

$$\bigvee_{i=3k-2}^{i=3k} \sum_{j=1}^{q+1} I_{A_i}(\omega_j)P(\omega_j) \geq \alpha_i,$$

$$\text{where } I_{A_i}(\omega_j) = \begin{cases} 1 & \text{if } \omega_j \models \phi_i \\ 0 & \text{otherwise,} \end{cases}$$

Hence we have $3q(q+1)$ optimization variables (values of $I_{A_i}(\omega_j)$, denoted by $a_{i,j}$); all of them are binary with values 0 and 1. Furthermore, we have a probability measure over $q+1$ truth assignments, represented by the real-valued variables $p_1, \ldots, p_{q+1} \in [0, 1]$, that must sum up to 1. Following the approach of [6], we find $\{a_{i,j}\}$ and $\{p_j\}$ by solving an integer program.

Algorithm 1 GPSAT solution based on integer linear programming.

Input: Propositions $\{A_j\}_{j=1}^n$, Ψ with 3-clauses of (possibly negated) assignments over A_{3i-2}, A_{3i-1} and A_{3i} for $1 \leq i \leq q$, a sentence γ, from $\Gamma = P(\gamma) \geq 1$ in CNF with m 3-clauses.
Output: "Satisfiable" if $\Psi \wedge \Gamma$ is satisfiable; "Unsatisfiable" otherwise.

```
1:      ▷ Variables $a_{i,j}$ are binary; variables $b_{i,j}$ and $p_j$ are real-valued in $[0,1]$.
2:  for $j \in \{1,\dots,q+1\}$ and each clause $(\vee_{l'=1}^{k'} A_{i_{l'}}) \vee (\vee_{l''=1}^{k''} \neg A_{i_{l''}})$ of $\phi$ do
3:      Generate linear constraint $(\sum_{l'=1}^{k'} a_{i_{l'},j}) + (\sum_{l''=1}^{k''}(1 - a_{i_{l''},j})) \geq 1$.
4:  end for
5:  for each clause $\vee_{l'=1}^{k'} P(A_{l'}) \geq \alpha_{l'} \vee \vee_{l''=1}^{k''} \neg P(A_{l''}) \geq \alpha_{l''}$ do
6:      Generate disjunction of linear constraints $(\bigvee_{l'=1}^{k'} \sum_{j=1}^{q+1} b_{l',j} \geq \alpha_{l'}) \vee (\bigvee_{l''=1}^{k''} \sum_{j=1}^{q+1} b_{l'',j} < \alpha_{l''})$.
7:  end for
8:  for $i \in \{1,\dots,3q\}$ do
9:      for $j \in \{1,\dots,q+1\}$ do
10:         Generate linear constraints $0 \leq b_{i,j} \leq a_{i,j}$ and $a_{i,j} - 1 + p_j \leq b_{i,j} \leq p_j$.
11:     end for
12: end for
13: Generate linear constraint $\sum_{j=1}^{q+1} p_j = 1$.
14: return Satisfiable if linear constraints have a solution, Unsatisfiable otherwise.
```

The elements a_{ij} for a fixed j corresponds to a truth assignment that satisfies γ. We explore the well known connection between SAT and integer programming to find such a truth assignment [5]. Start by generating a vector $\mathbf{a}_j$ with n binary variables $\{a_{ij}\}_{i=1}^n$, all with values 0 and 1. Now take one clause of γ; suppose it is written as

$$(\vee_{l'=1}^{k'} A_{i_{l'}}) \vee (\vee_{l''=1}^{k''} \neg A_{i_{l''}}).$$

For this clause, generate the linear inequality:

$$\left(\sum_{l'=1}^{k'} a_{k,i_{l'}}\right) + \left(\sum_{l''=1}^{k''} (1 - a_{k,i_{l''}})\right) \geq 1. \tag{3}$$

Consider the m inequalities generated this way (one per clause). A vector $\mathbf{a}_j$ that satisfies these m inequalities yields a truth assignment ω_j for γ by assigning True to A_i when $a_{i,j}$ is one, and assigning False to A_i when $a_{i,j}$ is zero.

We generate the truth assignments by generating $(q+1)$ sets of variables $\mathbf{a}_j$ and their related inequalities. These valuations correspond to the possible worlds $\omega_j \in \Omega$ with positive probabiity. Now we have to force the probability measure over these worlds to satisfy a probabilistic literal in each clause of Ψ.

To do so, note that each probabilistic atom in Ψ that is not negated represents an inequality as follows:

$$\sum_{j=1}^{q+1} a_{i,j} p_j \geq \alpha_i, \tag{4}$$

where p_j denotes $P(\omega_j)$. When the atom is negated, the contraint becomes:

$$\sum_{j=1}^{q+1} a_{i,j} p_j < \alpha_i, \tag{5}$$

The inequalities in (4) and (5) do not have to hold for all $3q$ probabilistic atoms. Only one per clause must hold. So, we have the following disjunction of contraints for each clause $1 \leq k \leq q$:

$$\bigvee_{i=3k-2}^{3k} \sum_{j=1}^{q+1} a_{i,j} p_j \bowtie_i \alpha_i, \tag{6}$$

where $\bowtie_i$ denotes $<$ if the probabilistic atom $P(A_i) \geq \alpha_i$ is negated; otherwise it denotes $\geq$. The challenge is to reduce the bilinear term $a_{i,j}p_j$ to linear constraints. We do that by introducing a new fresh variable $b_{i,j}$ and the constraints:

$$0 \leq b_{i,j} \leq a_{i,j} \qquad \text{and} \qquad a_{i,j} - 1 + p_j \leq b_{i,j} \leq p_j. \tag{7}$$

Note that if $a_{i,j} = 0$, then $b_{i,j} = 0$; and if $a_{i,j} = 1$, then $b_{i,j} = p_j$.

A last restriction ensures $P(.)$ is a probability distribution:

$$\sum_{j=1}^{q+1} p_j = 1. \tag{8}$$

The whole procedure is presented in Algorithm 1; it basically collects constraints from Expressions (3), (6), (7), and (8). The algorithm produces an integer linear program that has a solution if and only if the original GPSAT problem is satisfiable.

V. Implementation, Experiments, and Phase Transition

We have coded our GPSAT method using the Java language with calls to CPLEX version 12, extending the implementation from [6] to solve PSAT, and run experiments in a computer with 2GBytes of memory running Linux.

We were particularly interested in investigating whether phase transition phenomena can be identified in our solutions to GPSAT. Until the recent work of Finger and De Bona [9], there was little evidence of phase transition for PSAT in the literature. Baiotelli et al. [2] have shown empirical results for the 2CPA problem – that is equivalent to the PSAT in Normal Form with only two literals per clause – in which a typical phase transition shape is seen. However, 2CPA is not exactly a normal form of PSAT, even though it is NP-complete. As our normal form extends Finger and De Bona's, and GPSAT is also NP-complete, we expect to find evidence of phase transition also for GPSAT. Consequently we examine the behavior of GPSAT for various values of n, m for a fixed q, looking for hard problems instead of randomly trying out large problems that in the end may be easy.

Following the approach from Finger and De Bona [9], we fix the number of clauses in Ψ, q, and the total number of probabilistic literals in Ψ, $3q$. For a total number of variables $n = 50$, we varied the number m of clauses in order to explore the complexity profile of GPSAT for m/n. For each

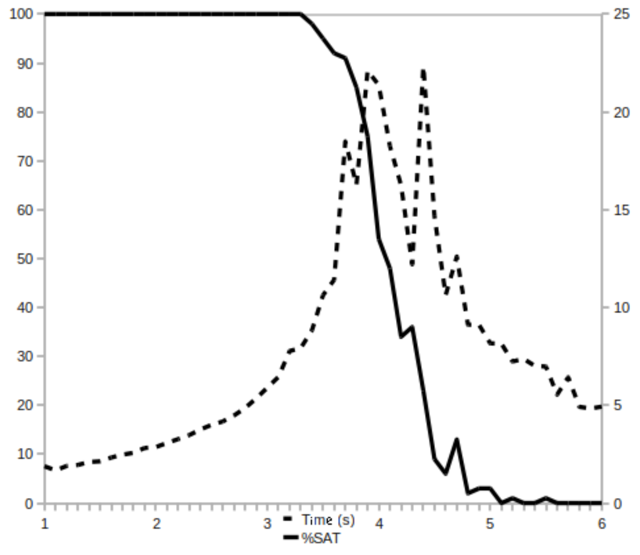

Fig. 1. GPSAT Phase Transition, for $n = 50$ and $q = 15$.

point m/n, 100 random GPSAT formulas in normal form were generated, submitted to the implementation of Algorithm 1, and the average computation time was registered together with the percentage of satisfiable formulas. Figure 1 shows the results of such experiment.

A easy-hard-easy pattern, typical from phase transition phenomena, is seen in Figure 1. The hardest problems were found close to the transition between regions of satisfiable and unsatisfible problems. If $q = 0$, the problem is Boolean satisfiability, and the point of approximately 50% of satisfiable instances is around $m/n = 4.3$ [10]. Here we add some probabilistic contraints, so the phase transition point has moved leftwards, as expected, to around $m/n = 4$.

Comparing to the results in [6] solving GPSAT seems harder than PSAT. This is not surprinsing, since it is known that disjunctive linear programming is considerably harder than linear programming.

VI. Conclusion

In this paper we have introduced the Generalized Probabilistic Satisfiability problem, a normal form for it, and an algorithm to solve it. GPSAT is considerably more expressive than PSAT, allowing negation and disjunction of probabilistic assessments and probabilistic nesting as well, but it is still NP-complete. Based on an integer linear programming solution to PSAT, we introduced an integer linear programming solution for GPSAT. Evidence for phase transition was found in our initial experiments, but an exhaustive investigation is needed to confirm and understand this phenomenom in GPSAT and PSAT.

For future work, it would be interesting to enlarge our language to embed linear combinations of probabilistic assignments, conditional probabilities and multi-agent probability measures. Another path would be to generalize the coherence checking problem, as we did for PSAT, from de Finetti's framework for probability.

References

[1] Kim Allan Andersen and Daniele Pretolani. Easy cases of probabilistic satisfiability. *Annals of Mathematics and Artificial Intelligence*, 33(1):69–91, 2001.

[2] Marco Baioletti, Andrea Capotorti, Paolo Tiberi and Sauro Tulipani. An empirical complexity for a 2CPA solver. In *10th International Conference IPMU*, pages 1857–1864. 2004.

[3] George Boole. *The Laws of Thought*. Dover edition, 1958.

[4] G. Bruno and A. Gilio. Applicazione del metodo del simplesso al teorema fondamentale per le probabilit nella concezione soggettivistica. *Statistica*, 40:337–344, 1980.

[5] Vijay Chandru and John Hooker. *Optimization Methods for Logical Inference*. John Wiley & Sons Inc., 1999.

[6] Fabio G. Cozman and Lucas F. di Ianni. Probabilistic satisfiability and coherence checking through integer programming. *To be published in ECSQARU2013*.

[7] Ronald Fagin, Joseph Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87:78–128, 1990.

[8] Ronald Fagin and Joseph Y. Halpern. Reasoning about knowledge and probability *Journal of the ACM*, 41(2):340–367, 1994.

[9] Marcelo Finger and Glauber De Bona. Probabilistic satisfiability: Logic-based algorithms and phase transition. In *IJCAI*, pages 528–533, 2011.

[10] I. P. Gent and T. Walsh. The SAT phase transition. In *European Conference on Artificial Intelligence*, pages 105–109, 1994.

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

[12] Theodore Hailperin. Best possible inequalities for the probability of a logical function of events. *American Mathematical Monthly*, 72:343–359, 1965.

[13] Theodore Hailperin. *Boole's Logic and Probability: a Critical Exposition from the Standpoint of Contemporary Algebra, Logic, and Probability Theory*. North-Holland, Amsterdam, 1976.

[14] Joseph Y. Halpern. *Reasoning about Uncertainty*. MIT Press, Cambridge, Massachusetts, 2003.

[15] Pierre Hansen and Brigitte Jaumard. Probabilistic Satisfiability. Technical Report G-96-31, Les Cahiers du GERAD, École Polytechique de Montréal, 1996.

[16] Pierre Hansen and Sylvain Perron. Merging the local and global approaches to probabilistic satisfiability. *International Journal of Approximate Reasoning*, 47(2):125–140, 2008.

[17] Brigitte Jaumard, Pierre Hansen, and Marcus Poggi de Aragão. Column generation methods for probabilistic logic. *ORSA Journal on Computing*, 3(2):135–148, Spring 1991.

[18] Thomas Lukasiewicz. Expressive probabilistic description logics. *Artificial Intelligence*, 172(6-7):852–883, April 2008.

[19] Raymond Ng and V. S. Subrahmanian. Probabilistic logic programming. *Information and Computation*, 101(2):150–201, 1992.

[20] N. J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28:71–87, 1986.

[21] Soshichi Uchii. Higher order probabilities and coherence. *Philosophy of Science*, 40(3):373–381, 1973.