Solutions for Hard and Soft Constraints Using Optimized Probabilistic Satisfiability

Marcelo Finger[⋆], Ronan Le Bras, Carla P. Gomes, and Bart Selman

Department of Computer Science
Cornell University

Abstract. Practical problems often combine real-world hard constraints with soft constraints involving preferences, uncertainties or flexible requirements. A probability distribution over the models that meet the hard constraints is an answer to such problems that is in the spirit of incorporating soft constraints.

We propose a method using SAT-based reasoning, probabilistic reasoning and linear programming that computes such a distribution when soft constraints are interpreted as constraints whose violation is bound by a given probability. The method, called Optimized Probabilistic Satisfiability (oPSAT), consists of a two-phase computation of a probability distribution over the set of valuations of a SAT formula. Algorithms for both phases are presented and their complexity is discussed.

We also describe an application of the oPSAT technique to the problem of combinatorial materials discovery.

1 Introduction

There are many proposals in the literature that combine logical and probabilistic reasoning, e.g. [23,22,5]. Perhaps the earliest such proposal was made by Boole himself, as a natural extension of Boolean satisfiability [1]. This framework is now called *probabilistic satisfiability (PSAT)*. The semantics is given by assigning a probability distribution π over the set of all 2^n truth assignments of n variables. Given π, one can now assign a probability P to each compound formula by considering the sum of the probabilities of all truth assignments (or models) that satisfy the formula. It has been shown that such a formalization has a number of desirable properties, such as the fact that it satisfies Kolmogorov's probability axioms [19,11].

A set of logical formulas, each assigned some probability value or a probability bound (e.g., $P(A \wedge B) \geq 0.6$), can be viewed as a set of probabilistic constraints. A natural question is whether there exist any probability distribution over all truth assignments that satisfies the probabilistic constraints. This is the consistency problem for probabilistic satisfiability. Note that by assigning probability 0 or 1 to some of the logical formulas, they effectively act as standard logical constraints. So, we can have a mix of logical and probabilistic constraints.

[⋆] On leave from Department of Computer Science, University of Sao Paulo.

In the mid eighties and early nineties, the consistency problem for PSAT became the focus of much attention because, in principle, it could be used to determine whether expert system sets of rules (hard and soft constraints) were consistent [19,7,12]. Unfortunately, the consistency problem for PSAT appeared to be extremely hard [20]. In particular, since the probability distribution ranges over all truth assignments, it was not even clear how to get a polynomial size witness for a consistent set of PSAT formulas. However, there have been several major breakthroughs in dealing with the complexity of this problem, such as polynomial size witness [7], linear programming algorithms [11] and SAT-based algorithms and the detection of PSAT phase-transition [4].

So, the recent progress has made PSAT a potentially relevant formalism for practical applications, providing an alternative to other approaches. One advantage of the PSAT framework is that its foundations are quite natural and well-grounded.

The goals of this work are three-fold:

(a) To enhance PSAT and introduce a method, called oPSAT, as a modeling framework to deal with mixtures of (hard) logical and soft probabilistic constraints.
(b) To propose a practical algorithmic strategy for solving oPSAT problems.
(c) To demonstrate the practical effectiveness of our proposed approach on a real-world reasoning task in the domain of Materials Discovery.

In this approach, formulas that encode the existence of a *soft violation* in the solution (sometimes called a *defect*) will be modeled by probabilistic constraints. Consider the following underspecified example.

Example 1. There are m students and k summer courses. Each student has a set of potential teammates, with whom coursework will be developed. We want the course enrollment to respect the following constraints:

Hard. For each course, students must decide to develop their coursework alone or in teams of 2 (pairs). A student may have different teammates in different courses, and may work alone in some course and have a teammate in others. Students must enroll in at least one and at most three courses. There is a limit of ℓ students per course.

Soft. Avoid having students with no teammate. A student's enrollment in a course with no team mate is seen as a "soft violation." □

Example 1 clearly shows the presence of hard and soft constraints. That problem also has some other implicit, data-dependent hard constraints, such as the number of students, courses and list of possible pairs of team mates. An important implicit hard constraint is the definition of a soft violation ("student in a course with no teammates") in terms of the variables present in the hard constraints. There may be no solutions to the hard constraints; or there may be several ones, in which case we are interested in computing a probability distribution over them, which will allow one to answer questions such as "what is the expected number of enrollments?" or "what is the probability that two students will be teammates?".

Yet, Example 1 is underspecified, as no clear way to deal with the soft constraints has been provided. In our method, this additional specification will correspond to a set of probabilistic constraints of the form $P(\mathit{softViolation}_i) \leq p_i$, $1 \leq i \leq m$, where for each student i there is a maximum probability p_i that i enrolls in some course with no teammate. Where do these probabilities come from? There are three main sources:

(a) The probabilities are stipulated or given. In Example 1, the student may be asked with which probability he or she accepts to be with no teammate. In this work, we will assume that this is the case.
(b) The probabilities are learned. For instance, compute p_i from previous editions of the summer course.
(c) The probabilities are minimal. Assume that all p_i are the same and compute the minimal value for which the hard and soft probabilistic constraints are satisfiable. This topic remains for further investigation.

Our method, called *Optimized Probabilistic Satisfiability (oPSAT)* consists of two phases[1]. The first phase is the PSAT problem, which determines if the hard constraints and probability constraints can be jointly satisfied; Section 2 will formalize PSAT and briefly describe a solver method. The output of such a problem, when satisfiable, is a probability distribution over a (small) class of models of the hard constraints. As this solution may not be unique, a second phase is needed to find a "reasonable" or "balanced" solution. By that we mean a distribution with minimal variance of soft violation occurrences. In Section 3, a novel SAT-based column generation method to compute such a distribution is presented.

Then in Section 4 we demonstrate the effectiveness of this approach on a complex real-world application involving the identification of crystallographic structures from high-intensity X-ray diffraction patterns [16,3,15]. The problem arises in the area of so-called combinatorial materials discovery [18].

2 Probabilistic Satisfiability

The PSAT problem is formalized as follows. Let $\mathcal{L}$ be the language of classical propositional formulas. A *PSAT instance* is a set $\Sigma = \{P(\alpha_i) \bowtie_i p_i | 1 \leq i \leq k\}$, where $\alpha_1, \ldots, \alpha_k \in \mathcal{L}$ are classical propositional formulas defined on n logical variables $\mathcal{P} = \{x_1, \ldots, x_n\}$, which are restricted by probability assignments $P(\alpha_i) \bowtie_i p_i$, $\bowtie_i \in \{=, \leq, \geq\}$ and $1 \leq i \leq k$.

There are 2^n possible propositional valuations v over the logical variables, $v : \mathcal{P} \rightarrow \{0, 1\}$; each such valuation is extended, as usual, to all formulas, $v : \mathcal{L} \rightarrow \{0, 1\}$. A *probability distribution over propositional valuations* $\pi : V \rightarrow [0, 1]$, is a function that maps every valuation to a value in the real interval $[0, 1]$ such

[1] This method should not be confused with OPTSAT [8].

that $\sum_{i=1}^{2^n} \pi(v_i) = 1$. The probability of a formula α according to distribution π is given by $P_\pi(\alpha) = \sum\{\pi(v_i)|v_i(\alpha) = 1\}$. We simply write $P(\alpha)$ when the distribution is clear from the context.

The definition of PSAT involves linear algebraic notation. We assume a vector b to be a column-vector and b' its transpose, a row-vector. If A is an $m \times n$ matrix, A^j represents its j-th column, and if b is an m-dimensional column, $A[j := b]$ represents the matrix obtained by substituting b for A^j; if A is square matrix, $|A|$ is A's determinant.

From a PSAT instance, construct a $k \times 2^n$ matrix $A = [a_{ij}]$ such that $a_{ij} = v_j(\alpha_i)$. The *probabilistic satisfiability problem* is to decide if there is a probability vector $\pi_{2^n \times 1}$ subject to:

$$\begin{aligned} A\pi &\bowtie p \\ \sum \pi_i &= 1 \\ \pi &\geq 0 \end{aligned} \tag{1}$$

A PSAT instance Σ is *satisfiable* if its associated PSAT restriction (1) has a solution π; in that case, we say that π satisfies Σ. The last two conditions of (1) force π to be a probability distribution. Usually the first two conditions of (1) are combined: A is a $(k+1) \times 2^n$ $\{0,1\}$-matrix with 1's at its first line, p_1 is set to 1 in vector $p_{(k+1)\times 1}$, and the $\bowtie_1$-relation corresponds to "=". In this case, each column A^j, excluding its first position that is always 1, can be seen as a Boolean valuation.

Example 2. We continue Example 1 and for simplicity assume that there is only one course, three students whose enrollment is represented by variables x, y and z, and two potential partnerships of the first student with either of the others, represented by p_{xy} and p_{xz}. These partnerships are mutually exclusive, as x can only have one partner. So we have the hard constraint

$$P(x \wedge y \wedge z \wedge \neg(p_{xy} \wedge p_{xz})) = 1$$

and the soft constraints are probability restriction on the enrollment of a student without partners, set for this example as:

$$P(x \wedge \neg p_{xy} \wedge \neg p_{xz}) \leq 0.25, \qquad P(y \wedge \neg p_{xy}) \leq 0.6, \qquad P(z \wedge \neg p_{xz}) \leq 0.6.$$

Of all the 2^5 valuations, we consider π such that $\pi(x, y, z, \neg p_{xy}, \neg p_{xz}) = 0.1$, $\pi(x, y, z, p_{xy}, \neg p_{xz}) = 0.4$, $\pi(x, y, z, \neg p_{xy}, p_{xz}) = 0.5$ and $\pi(v) = 0$ for the remaining 29 valuations. This distribution satisfies the PSAT instance. □

It is no coincidence that only a small number of valuations in Example 2 receive non-zero probability. In fact, satisfiable PSAT instances always have a "small" witness.

Proposition 1 ([7]). *If an instance $\Sigma = \{P(\alpha_i) = p_i | 1 \leq i \leq k\}$ has a solution $\pi \geq 0$, then there is a solution $\pi' \geq 0$ with at most $k+1$ non-zero elements.*

From Proposition 1, it follows that PSAT is in NP. As SAT is a special case of PSAT when all $p_i = 1$, PSAT is NP-hard. As a result, PSAT is NP-complete.

There have been several proposed algorithms for PSAT [14,11,13], but its general applicability in practice has only been established with the demonstration that PSAT presents a phase transition [4], just like the SAT problem [17,6].

As in SAT, to display a phase transition the problem must be brought to a normal form. A PSAT instance is in *normal form* if it is partitioned in two sets, $\langle \Gamma, \Psi \rangle$, where $\Gamma = \{P(\alpha_i) = 1 | 1 \leq i \leq m\}$ and $\Psi = \{P(y_i) = p_i | y_i \text{ is a variable}, 1 \leq i \leq k\}$. Every PSAT instance can be transformed in a normal form instance $\langle \Gamma, \Psi \rangle$ in polynomial time, such that satisfiability is preserved [4]. The set $\Psi(y_1, \ldots, y_k)$ contains probabilistic restrictions over variables $y_1, \ldots, y_k$ only, and the set $\Gamma(y_1, \ldots, y_k; x_1, \ldots, x_n)$ is a SAT formulas. A valuation v over $y_1, \ldots, y_k$ is *Γ-consistent* if there is an extension of v over $y_1, \ldots, y_k, x_1, \ldots, x_n$ such that $v(\Gamma) = 1$. The following refines Proposition 1.

Proposition 2 ([4]). *A normal form instance $\langle \Gamma, \Psi \rangle$ is satisfiable iff there is a $(k+1) \times (k+1)$-matrix A and $\pi \geq 0$ such that $A\pi = p$ and whenever $\pi_j > 0$ then column j of A is Γ-consistent.*

In this work, we will always consider instances to be in normal form. Based on Proposition 2, two algorithms for PSAT solving were proposed in [4], and here we are interested in the one that solves the following optimization problem

$$\begin{array}{ll} \text{minimize} & c'\pi \\ \text{subject to} & A\pi = p \text{ and } \pi \geq 0 \end{array} \tag{2}$$

where A is a $(k+1) \times 2^n$ $\{0,1\}$-matrix in (1), π is the probability distribution and c is a $2^n \times 1$ *cost vector*; $c_j = 1$ if A's column j is a Γ-inconsistent valuation, and $c_j = 0$ otherwise. The PSAT instance is satisfiable iff the optimization leads to a cost $c'\pi = 0$.

As A is exponentially large, we do not generate it explicitly. Instead, we use a SAT-solver to generate Γ-consistent columns as the linear optimization *simplex algorithm* requires [21]. The problem is solved iteratively; at each iteration step i, Proposition 2 allows for storing $A_{(i)}$ with $k+1$ columns and a *column generation* method is employed in which a SAT-based *auxiliary problem* generates a Γ-consistent column that replaces some column in $A_{(i)}$ and decreases the objective function; this method is detailed in Section 2.1. Accordingly, only the components of c and π corresponding to the columns of $A_{(i)}$ are stored.

A *feasible solution* $A_{(i)}$ is a $\{0,1\}$-matrix for which there exists a $\pi_{(k+1)\times 1} \geq 0$ such that $A_{(i)}\pi = p$. It is shown in [4] that an initial feasible solution $A_{(0)}$ always exists and can be easily computed. The simplex method guarantees that the cost function always decreases at each step, by computing the *reduced cost* $\bar{c}_b$ of inserting a column b in a feasible solution A and forcing it to be non-positive [14]:

$$\bar{c}_b = c_b - c_A A^{-1} b \leq 0 \tag{3}$$

Algorithm 2.1. PSATsolver($\langle \Gamma, \Psi \rangle$)

Input: A normal form PSAT instance $\langle \Gamma, \Psi \rangle$.
Output: Total solution A; or "No", if unsatisfiable.

```
1: $A_{(0)}$ := initial feasible solution; $i := 0$; compute $cost^{(i)}$;
2: while $\mathrm{cost}^{(\mathrm{i})} > 0$ do
3:     $b^{(i)} = GenerateColumn(A_{(i)}, p, \Gamma)$; /* Described in Section 2.1 */
4:     return "No" if $b_1^{(i)} < 0$; /* PSAT instance is unsat */
5:     $A_{(i+1)} = merge(A_{(i)}, b^{(i)})$;
6:     increment $i$; compute $cost^{(i)}$;
7: end while
8: return $A_{(i)}$; /* PSAT instance is satisfiable */
```

where c_b and c_A are, respectively, the component of the cost vector corresponding to the column b and the columns of A. In our case, $c_b = 0$, so the goal is to find a column b such that $c_A A^{-1} b \geq 0$.

Algorithm 2.1 presents a method that decides whether a PSAT instance is satisfiable by solving Problem (2), with a positive answer if minimum cost is 0. Let us see an example of Algorithm 2.1 at work.

Example 3. We express the instance of Example 2 in normal form $\langle \Gamma, \Psi \rangle$ by adding variables for each soft violation: s_x, s_y, s_z. Thus

$$\Gamma = \left\{ \begin{array}{c} x,\ y,\ z,\ \neg p_{xy} \vee \neg p_{xz}, \\ (x \wedge \neg p_{xy} \wedge \neg p_{xz}) \to s_x,\ (y \wedge \neg p_{xy}) \to s_y,\ (z \wedge \neg p_{xz}) \to s_z \end{array} \right\}$$
$$\Psi = \{\ P(s_x) = 0.25,\ P(s_y) = 0.6,\ P(s_z) = 0.6\ \}$$

Note that the existence of a soft violation implies the truth of the corresponding variable in s_x, s_y, s_z, but the truth of some of these variables *does not* necessarily imply the occurrence of a soft violation. We now apply Algorithm 2.1.

$$A_{(0)} = \begin{bmatrix} \mathit{1} & 1 & 1 & 1 \\ \mathit{0} & 0 & 0 & 1 \\ \mathit{0} & 0 & 1 & 1 \\ \mathit{0} & 1 & 1 & 1 \end{bmatrix} \qquad A_{(1)} = \begin{bmatrix} \mathit{1} & 1 & 1 & 1 \\ \mathit{0} & 0 & 0 & 1 \\ \mathit{0} & 0 & 1 & 1 \\ \mathit{0} & 1 & 0 & 1 \end{bmatrix} \qquad A_{(2)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} \pi^{(0)} &= [0.4\ 0\ 0.35\ 0.25]' & \pi^{(1)} &= [0.05\ 0.35\ 0.35\ 0.25]' & \pi^{(2)} &= [0.05\ 0.35\ 0.4\ 0.2]' \\ \mathrm{cost}^{(0)} &= 0.4 & \mathrm{cost}^{(1)} &= 0.05 & \mathrm{cost}^{(2)} &= 0 \\ b^{(0)} &= [1\ 0\ 1\ 0]' : \mathrm{col}\ 3 & b^{(1)} &= [1\ 1\ 0\ 1]' : \mathrm{col}\ 1 & & \end{aligned}$$

The initial feasible solution $A_{(0)}$ is a line permutation of an upper 1-triangular matrix, has all but its first column Γ-consistent, with lines 2,3,4 corresponding to Ψ-variables s_x, s_y, s_z and leads to $\pi^{(0)}$ and cost 0.4. The first line is always 1 to force the probabilities to add up to 1. Column generation (Section 2.1) produces $b^{(0)}$ which the simplex merging determines to substitute $A_{(0)}$'s third column. This generates $A_{(1)}$, $\pi^{(1)}$ and decreasing cost 0.05; column generation yields $b^{(1)}$

that substitutes $A_{(1)}$'s first column. In $A_{(2)}$ there are no Γ-inconsistent columns and the cost is 0, so the problem is satisfiable. At each step i, $A_{(i)} \cdot \pi^{(i)} = p$.

The distribution here is distinct from that in Example 2, as here we consider only the variables in Ψ; this also illustrates that the satisfying distribution is not unique. □

2.1 SAT-Based Column Generation

The following describes procedure *GenerateColumn*$(A_{(i)}, p, \Gamma)$ used in Algorithm 2.1 and adapted for optimization Algorithm 3.1.

A Γ-consistent column b that never increases the value of the objective function is obtained by solving a SAT problem as follows. Consider $x_1, \ldots, x_k$ taking values in $\{0, 1\}$, $a_1, \ldots, a_k, c \in \mathbb{Q}$ and

$$a_1 \cdot x_1 + \cdots a_k \cdot x_k \quad \bowtie \quad c \qquad \bowtie \in \{<, \leq, >, \geq, =, \neq\} \tag{LR}$$

Linear restriction (LR) can be seen as a propositional formula Δ_{LR}, in the sense that a valuation $v : x_i \mapsto \{0, 1\}$ *satisfies* Δ_{LR} iff v makes (LR) a true condition. Δ_{LR} can be obtained from (LR) in time $O(k)$ [25].

Suppose $1, \ldots, q \leq k + 1$ are the Γ-inconsistent columns of feasible solution A. By (3), a column $b = [1\ y_1 \ldots y_k]'$ that substitutes some A^j and enforces a decreasing *cost* satisfies

$$\textstyle\sum_{i=1}^{q} A_i^{-1} \cdot [1\ y_1\ \cdots\ y_k]' \geq 0 \tag{LR_{cost}}$$

A valuation that satisfies $\Gamma \wedge \Delta_{LR_{cost}}$ instantiates b. If that formula is satisfiable, $A[j := b]$ is a feasible solution and *cost* never increases.

With respect to the termination of the simplex method, one must ensure that Bland's rule for fixed order of insertion/removal of columns is respected, and thus termination of the simplex optimization is guaranteed [21].[2]

2.2 The Practical Feasibility of PSAT

Prior to the development of very efficient SAT solvers, PSAT was considered "completely impractical" [20]. But the work of [4] has shown that PSAT presents, in practice, the hard/easy phase transition behavior similar to that of SAT [17,6]. Among other things, this means that there are predominantly "easy" cases of satisfiable and unsatisfiable PSAT instances. Of course, PSAT is still several times slower than SAT due to the fact that a PSAT solver invokes a SAT solver several times.

With the current technology of SAT solvers, an auxiliary formula $\Gamma \wedge \Delta_{LR_{cost}}$ with tens or even hundreds of thousands of variables can be mostly dealt without problems. To keep the number of iterations of Algorithm 2.1 under control, it is advisable to keep a small number k of probability restrictions. Several dimensionality reduction techniques may be employed, such as the one described in Example 4.

[2] In practice, some SAT solvers, such as zchaff, have an internal behavior that obeys Bland's rule; others, such as minisat, need extra coding precautions to avoid loops.

Example 4. Reconsider Example 1, assuming there are $k > 1$ courses for m students to enroll, but with a limit of ℓ students per course. Consider as a soft violation now a course having any students with no partners, reducing the number of probabilistic constraints from m to $k \ll m$. The probability of a violating course, p_c, can be obtained from the previous one, adopting a simplifying assumption of independence between soft violations, thus obtaining the probability $p_c = 1 - (1 - p_i)^{\ell}$. □

3 Optimizing Probability Distributions with oPSAT

Solutions to a PSAT problem are not unique, and a second phase is needed to obtain a distribution with desirable properties. This, in some sense, mirrors the two steps of a linear optimization problem using the simplex algorithm. The first phase searches for a feasible solution for the initial constraints, which is what PSAT does; the second phase produces a solution to the constraints that optimizes an objective function.

A first candidate for this objective function is the minimization of the expected value of S, the number of soft violations:

$$E(S) = \sum_{v_i | v_i(\Gamma)=1} S(v_i)\pi(v_i), \qquad \text{where } S(v_i) = \sum_{j=1}^{k} v_i(y_j)$$

However, due to the following result, this initial idea is not applicable. Define a (PSAT) *model linear function* over Ψ-variables $y_1, \ldots, y_k \in \{0, 1\}$:

$$f(y_1, \ldots, y_k) = a_1 y_1 + \cdots + a_k y_k, \qquad \text{where } a_j \in \mathbb{Q}, 1 \le j \le k \tag{4}$$

It is important that *only* variables in Ψ are arguments of f. Note that $E(S)$ is a model linear function with all $a_j = 1$. Also note that the *expected value* of a linear function f according to a probability distribution π is $E_\pi(f) = \sum_j (a_1 v_j(y_1) + \cdots + a_k v_j(y_k))\, \pi(v_j)$.

Lemma 1. *Consider a satisfiable normal form PSAT instance* $\langle \Gamma(y_1, \ldots, y_k; x_1, \ldots, x_n), \Psi\{P(y_j) = p_j | 1 \le j \le k\}\rangle$*; let* $f(y_1, \ldots, y_k)$ *be a model linear function. Then for every satisfying probability distribution* π*, the expected value of* f *with respect to* π *is fixed,* $E_\pi(f) = \sum a_j p_j$*.*

Proof. Directly from the definition of $E_\pi(f)$ and using linearity of E_π:

$$\begin{aligned} E_\pi(f) &= \sum_v (a_1 v(y_1) + \cdots + a_k v(y_k))\pi(v) \\ &= a_1 \sum_v v(y_1)\pi(v) + \cdots + a_k \sum_v v(y_k)\pi(v) \\ &= a_1 P_\pi(y_1) + \cdots + a_k P_k(y_k) = \sum_{j=1}^{k} a_j p_j. \end{aligned}$$

Note that the use of normal form helped considerably to obtain this result. □

Lemma 1 shows that there is no point in minimizing the expected number of soft violations, which is a constant for a given PSAT instance.

3.1 Variance Minimization

Lemma 1 implies that the model function to be minimized to obtain a "balanced" probability distribution must be non-linear. The idea is to choose a function that prioritizes assigning higher probability mass to distributions with smaller number of soft violations.

One possible choice is then to minimize the expected value of the square number of soft violations, $E(S^2)$. The minimal value of the expected value of this function tends to assign more weight, that is, a greater probability, to the models with smaller number of soft violations. It also seems a good choice of function that a distribution must minimize to obtain a "balanced" distribution due to the following property.

Theorem 1 (Minimal Variance). *The probability distribution that minimizes $E(S^2)$ is also the probability distribution that minimizes the variance of the number of soft violations,* $\mathrm{Var}(S)$.

Proof. We know from basic statistics that the variance of a function is given by

$$Var(S) = E\left((S - E(S))^2\right) = E(S^2) - (E(S))^2 \tag{5}$$

But, by Lemma 1, $E(S)$ is fixed, so the distribution that minimizes $E(S^2)$, by (5), is also the distribution that minimizes $Var(S)$. □

So we take the view that a "balanced" distribution that respects soft constraints is one that minimizes the variance of the number of soft violations.

To implement it, we also use a SAT-based column generation to minimize the objective function. The generation of a column b is based on the encoding of the reduced cost given by (3) as $\bar{c}_b = c_b - c_A A^{-1} b < 0$, where c is the cost vector and A is a feasible solution. In the PSAT case, the cost of the new column is $c_b = 0$, but here we do not know a priori its value.

However, there are only a few possible values of $c_b = (S(b))^2$. Thus we iterate $i = 0$ to k, $c_b = i^2$, at each step generating a Γ-consistent SAT formula encoding of (3) with at most i soft violations. Assume $\mathit{VarianceDecreasingColumn}(i, A, p, \Gamma)$ is a column generation function that performs such encoding and submits it to a SAT-solver, obtaining b; again a value $b_1 < 0$ indicates unsatisfiability.

Algorithm 3.1 implements variance minimization and is a variation of Algorithm 2.1. It takes as input the first phase solution to a satisfiable PSAT instance. It contains two nested loops. The outermost one iterates over the computation step (from 0 to k), to be able to compute columns that generate a reduced cost. The inner loop actually performs the column generation optimization step; this loop stops when it is not possible to further minimize the cost for a given number of soft violations set by the outer loop, which may occur if no satisfiable instance for the column generated is obtained.

Algorithm 3.1. MinimizeVariance(Γ, Ψ, A, π)

```
Input: A PSAT instance $\langle \Gamma, \Psi \rangle$, satisfied by $A\pi = p$.
Output: $\langle A', \pi' \rangle$ such that $\pi'$ has minimal variance of all solutions to $\langle \Gamma, \Psi \rangle$.
 1: $A_{(0)} := A$; $\pi^{(0)} = \pi$; $cost^{(0)} = E_\pi(S^2)$;
 2: for $i = 0$ to $k$ do
 3:   repeat
 4:     $b^{(i)} = VarianceDecreasingColumn(i, A_{(i)}, p, \Gamma)$;
 5:     if $b_1^{(i)} \geq 0$ then
 6:       $A_{(i+1)} = merge(A_{(i)}, b^{(i)})$;
 7:       compute $\pi^{(i+1)}$ and $cost^{(i+1)} = E_{\pi^{(i+1)}}(S^2)$;
 8:     end if
 9:   until $b_1^{(i)} < 0$ /* cost cannot be further minimized */
10: end for
11: return $\langle A_{(k+1)}, \pi^{(k+1)} \rangle$;
```

Example 5. We continue Example 3, optimizing its output, which had $E(S) = 2{\cdot}0.05 + 1{\cdot}0.35 + 1{\cdot}0.4 + 3{\cdot}0.2 = 1.45$ and $E(S^2) = 4{\cdot}0.05 + 1{\cdot}0.35 + 1{\cdot}0.4 + 9{\cdot}0.2 = 2.75 = cost^{(0)}$. According to Algorithm 3.1, we iterate over the number of soft violations allowed ($i = 0$ to 3). For $i = 0$ and $i = 1$, the computed SAT formula is unsatisfiable; for $i = 2$, a new column is obtained to substitute the third column:

$$A_{(2)} = \begin{bmatrix} 1&1&1&1 \\ 1&0&0&1 \\ 0&0&1&1 \\ 1&1&0&1 \end{bmatrix} \qquad A_{(2)'} = \begin{bmatrix} 1&1&1&1 \\ 1&0&0&0 \\ 0&0&1&1 \\ 1&1&0&1 \end{bmatrix}$$

$$\begin{aligned} \pi^{(2)} &= [0.05\ 0.35\ 0.4\ 0.2]' & \pi^{(2)'} &= [0.25\ 0.15\ 0.4\ 0.2]' \\ \text{cost}^{(2)} &= 2.75 & \text{cost}^{(2)'} &= 2.35 \\ b_{(2)} &= [1\ 0\ 1\ 1]' : \text{col } 3 \end{aligned}$$

The remaining iterations all generate unsatisfiable formulas, so the minimum variance obtained for $i = 2$ is $Var(S) = 2.35 - 1.45^2 = 0.2475$. □

4 oPSAT and Combinatorial Materials Discovery

In this section, we present an application of the proposed oPSAT approach to a practical problem in materials discovery. We first provide some background on this motivating application, before formally defining the problem. Finally, we present an oPSAT encoding for this problem and the experimental results for it.

4.1 Background

In combinatorial materials discovery, the goal is to find intermetallic compounds with desirable physical properties by obtaining measurements on samples from a *thin film* composition spread. This approach has been successfully applied for example to speed up the discovery of new materials with improved catalytic activity for fuel cell applications [24,9]. Nevertheless, the analysis of these measurements, also called the phase-field identification problem, requires a manual and laborious data interpretation component, and our goal is to automate it and reduce its processing time.

Combinatorial materials discovery, and in particular the problem of ternary phase-field identification addressed in this paper, provides unique computational and modeling challenges. While statistical methods and machine learning are important components to address this challenge, they fail to incorporate relationships that are inherent to the basic physics and chemistry of the underlying materials. In fact, a successful approach to materials discovery requires a tight integration of statistical methods, to deal with noise and uncertainty in the measurement data, and optimization and inference techniques, to incorporate a rich set of constraints arising from the underlying materials physics and chemistry. As a consequence, the proposed oPSAT framework seems particularly suited to address this problem.

4.2 Problem Definition

In the composition spread approach, three metals (or oxides) are sputtered onto a silicon wafer using guns pointed at three distinct locations, resulting in a so-called *thin film* (Fig. 1). Different locations (or samples) on the thin film correspond to different concentrations of the sputtered materials, based on their distance to the gunpoints. X-ray diffraction (XRD) is then used to characterize a number of samples on the thin film. For each sample point, it provides the intensity of the electromagnetic waves as a function of the angle of diffraction. The observed diffraction pattern is closely related to the underlying crystal structure, which provides important insights into chemical and physical properties of the corresponding composite material.

The goal of the phase-field identification problem is to identify regions of the thin film that share the same underlying crystal structure. Intuitively, the XRD patterns observed across the *thin film* can be explained as combinations of a small set of basis patterns called *phases*. Finding the phase field corresponds to identifying these *phases* as well as their concentration on the thin film. The main challenge is to model the complex crystallographic process that these phases are subject to (such as the expansion of the lattice, which results in a 'shift' of the XRD pattern), while taking into account the imperfection of the silicon wafer as well as experimental noise of the data.

While it is natural to study the phase-field identification problem on the basis of full XRD curves, constructive interference of the scattered X-rays occurs, by nature, at *specific* angles and creates spikes (or *peaks*) of intensity. In addition, experimental noise combined with variations of the Silicon substrate make the measured intensity of the beam not reliable. As a result, materials scientists mostly rely on peak angles when tackling the phase-field identification problem. Therefore, we use a specialized peak detection algorithm [10] to extract the set of peak angles $\mathcal{Q}(i)$ in the XRD pattern of a sample point i.

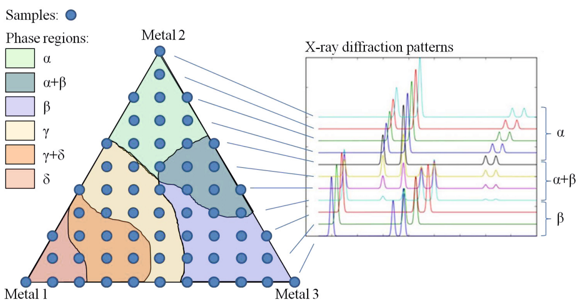

Fig. 1. Example of a *thin film*. Each sample on the silicon wafer corresponds to a different composition, and has an associated measured x-ray diffraction pattern. Colors correspond to different combinations of the basis patterns $\alpha, \beta, \gamma, \delta$. On the right, diffraction patterns of the sample points along the right side of the thin film illustrate how the patterns combine and shift as one moves from one point to a neighboring one.

The goal is then to find a set of peak angles $\mathcal{E}_k$ for each phase k, as well as phase-presence Boolean variables $a_{i,k}$ and scaling factors $s_{i,k} \in \mathbb{R}$ for each sample i and phase k, such that each observed set of peaks $\mathcal{Q}(i)$ is explained. Namely, for each peak $q \in \mathcal{Q}(i)$ we want to have at least one phase k and one peak $e \in \mathcal{E}_k$ of that phase that can explain it, i.e. $\forall q \in \mathcal{Q}(i)\ \exists e \in \mathcal{E}_k\ s.t.\ (a_{i,k} \wedge\ |q - s_{i,k} \cdot e| \leq \epsilon)$ where ϵ is a parameter that depends on the accuracy of the peak detection algorithm.

Moreover, no more than 3 basis patterns can be used to explain the peaks at sample point i, which translates to $|\{k | a_{i,k} = 1\}| \leq 3$. Finally, the sample points are embedded into a graph $\mathcal{G}$, such that there is a vertex for every sample and edges connect samples that are close on the *thin film* (eg. based on the grid). Given this graph, we require that the subgraph induced by $\{i | a_{i,k} = 1\}$ is connected in order for the basis patterns to appear in contiguous locations on the *thin film*. In addition, the scaling factors $s_{i,k}$ should be monotonic along the paths of this graph, and cannot exceed a given value S_{max}.

An analogy with the student enrollment example would be to consider a sample as a student who is enrolling in at most 3 courses (phases assigned to peaks of the sample) and is teaming up with other students (a peak paired with a neighboring peak).

4.3 oPSAT Encoding

We now formulate the phase-field identification problem as an oPSAT encoding. Let K be the set of phases. Also, let G be the set of sample points embedded in a grid, such that each sample has neighbors in one or more of the four directions $\{N, E, S, W\}$. We denote $G(i)$ the peaks of sample point i and l_p the angle of peak $p \in G(i)$. For a peak $p \in G(i)$, we define $N_{p,D} \subseteq G(i')$ the subset of peaks of sample i', where i' is the sample in direction D from i (denoted $i' \in D(i)$), and such that $p' \in N_{p,D}$ if $l_p \leq l'_p \leq l_p.S_{max}$. In other words, $N_{p,D}$ is the set of p's neighbor peaks that can be matched with p according to the direction D and without exceeding the maximum allowed shift (see Fig. 2).

Variables. We define a Boolean variable $x_{p,k}$, for $p \in G(i)$, $i \in G$, $k \in K$, to indicate whether peak p belongs to phase k. Similarly, $z_{i,k}$ indicates whether sample point i contains some peak in phase k, i.e. $z_{i,k} = \bigvee_{p \in G(i)} x_{p,k}$. In addition, a Boolean variable $y_{pp'k}$ indicates that peak p is paired with peak p' for phase k. Therefore, we have $y_{pp'k} \rightarrow x_{pk} \wedge x_{p'k}$. Furthermore, we introduce two directions $D_{1k} \in \{N, S\}$ and $D_{2k} \in \{E, W\}$ for each phase k. The direction of a phase is used to impose that any peak of that phase shifts according to that direction. Accordingly, we have: $\bigvee_{p' \notin N_{p,D_{1k}} \cup N_{p,D_{2k}}} y_{pp'k} = 0$ for all $i \in G, p \in G(i), k \in K$. Moreover, in order to introduce probability restrictions on the number of unmatched peaks, we define a Boolean variable d_p that corresponds to whether peak p is paired with a peak of the neighboring samples. Similarly, d_i denotes whether all peaks of sample i are paired, and are channeled to the d_p variables through the following propositional formula: $\neg d_p \vee d_i$ for all $i \in G, p \in G(i)$.

Propositional Formulas. A peak is assigned to at most one phase, i.e. $\sum_k x_{pk} \leq 1$. An unassigned peak is considered unmatched (as illustrated by p_0 in Fig. 2). Namely, $(\bigvee_k x_{pk}) \vee d_p$ for all $i \in G, p \in G(i)$. If a peak is assigned to a phase, then it needs to be paired with a neighboring peak, otherwise it is considered unmatched (see p_1 in Fig. 2). This constraint translates to: $x_{pk} \rightarrow \left(\bigvee_{p'} y_{pp'k} \vee d_p\right)$ for all $i \in G, p \in G(i), k \in K$. In addition, a phase should be consistent among the samples in which this phase is involved. Namely, if two adjacent samples share a phase, each peak of one must be paired with a peak of the other, otherwise it is considered unmatched (as illustrated by p_2 in Fig. 2). This translates to: $x_{pk} \wedge z_{i'k} \rightarrow \left(\bigvee_{p' \in G(i')} y_{pp'k} \vee d_p\right)$ for all $i \in G, p \in G(i), k \in K, i' \in D_{1k}(i) \cup D_{2k}(i)$. Moreover, we enforce a relaxed form of convex connectivity of a phase on the thin film, requiring that if any two samples that are two or more columns (or rows) apart involve a given phase, then there should be a sample in between them that involves this phase as well. In other words, we require $(x_{pk} \wedge x_{p'k}) \rightarrow \bigvee_{i'' \in N_C(i,i'), p'' \in G(i'')} x_{p''k}$, where $N_C(i, i')$ (resp. $N_R(i, i')$) is the set of samples on the grid between the columns (resp. rows) of i and i'. Finally, we impose that a peak cannot be paired with more than one neighboring peak, i.e. $\sum_{k,p' \neq p} y_{pp'k} \leq 1$, for all $i \in G, p \in G(i)$ and $\sum_{k,p' \neq p} y_{p'pk} \leq 1$, for all $i \in G, p \in G(i)$.

Probability Restrictions. We limit the probability that all peaks of a sample i remain unmatched by requiring $P(d_i) \leq p_i$, where p_i is either given or refined by dichotomy search.

Inference Method. For the experimental results described in the following, we computed a probability distribution using oPSAT with variance minimization and used, in order to obtain the accuracy of the computation, the model of the hard (SAT) constraints in that distribution with the highest probability.

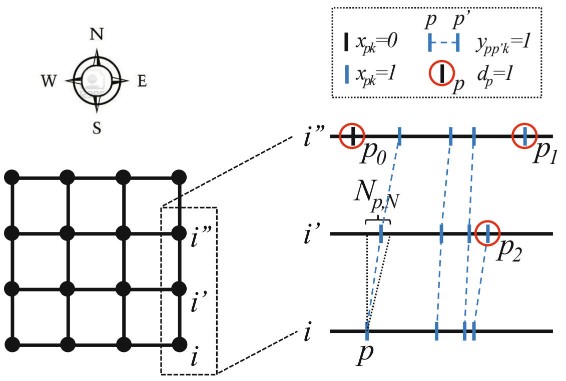

Fig. 2. Examples of soft violations in the oPSAT encoding. Left: Grid of sample points. Right: Pairing the peaks of sample points i, i' and i''. In the case of phase k of direction *North*, the peak p of sample i can only be paired with peaks of i' in $N_{p,N}$. Also, this example illustrates the three possible soft violations for peaks: 1) p_0 is not assigned a phase (assuming one single phase), 2) p_1 is not paired with any other peak, and 3) p_2, assigned to phase k, has no matching peak in i'', although i'' involves phase k.

4.4 Experimental Validation

We evaluate the oPSAT approach on the synthetic data used in [3] and compare with the SMT approach. Note that data from real experiments has to be manually labeled, which unfortunately is not yet available. Data was synthesized based on a known underlying phase map for the Al-Li-Fe system [16], a ternary system composed of 6 phases ($|K| = 6$).

All experiments were conducted on the same machine and using the same C++ implementation of an oPSAT solver, using minisat as the auxiliary SAT solver. The SMT solver used in these experiments was *Z3*[2]. For the oPSAT approach, the model with highest probability in the computed distribution was used to obtain the accuracy results. The maximum probability of a peak to be unmatched, that is, a peak with no phase assigned, was fixed as 2%, and a soft violation was defined as a sample point with some unmatched peak. This soft violation probability was computed over all peaks at that sample point, assuming that the probability of one peak to be unmatched is independent from that of any other peak. Table 1 shows the results of the experiments.

In all cases, the accuracy of the model computed by the oPSAT solver, defined as the percentage of peaks predicted with the same phase as in the synthetic data set, was above 80% (compared to 100% for SMT). On the other hand, the oPSAT implementation presents a dramatic increase of efficiency, of at least two orders of magnitude in all cases, and of about 2,000 times in one case.

Overall, while materials scientists currently proceed to a manual analysis of the high-throughput experimental data, our results provide solutions that are good and useful from the point of view of materials scientists, especially as these solutions are, by design of hard constraints, physically meaningful and

Table 1. Runtime (in seconds) for both SMT and oPSAT approaches on 5 datasets, as well as the accuracy of oPSAT (the accuracy of SMT being 100%). P is the number of sample points, L^* is the average number of peaks per phase, K is the number of basis patterns, and $\#Peaks$ is the overall number of peaks.

Dataset					**SMT**	**oPSAT**	
System	P	L^*	K	$\#Peaks$	**Time**(s)	**Time**(s)	**Accuracy**
Al/Li/Fe	28	6	6	170	346	5.3	84.7%
Al/Li/Fe	28	8	6	424	10076	8.8	90.5%
Al/Li/Fe	28	10	6	530	28170	12.6	83.0%
Al/Li/Fe	45	7	6	651	18882	121.1	82.0%
Al/Li/Fe	45	8	6	744	46816	128.0	80.3%

comply with the crystallographic process. In addition, our approach is the first automated method to exhibit short running times, and has great potential to be used within an online setting that guides the data collection itself. Therefore, the acceptable loss of accuracy is made up by a significant gain in speed. Finally, these results advocate the practical feasibility of oPSAT for real applications involving hard and soft constraints.

5 Conclusions

In this work we have described how to use the optimized probabilistic satisfiability (oPSAT) method to deal with problems that combine hard and soft restrictions. We have shown how a probability distribution can be computed to satisfy logic and probabilistic constraints and how it can be optimized to display balanced properties via variance minimization. The technique was then applied to the non-trivial problem of materials discovery with acceptable precision and superior run times than existing methods.

Future work should address the computation of probability constraints that minimize the expected value of soft violations, as well as inference methods that employ the probability distribution computed by the oPSAT method, instead of just considering the model with the largest probability in that distribution. With respect to experimental results, we plan to measure the efficiency of the oPSAT solver on real data, once a manually annotated data set becomes available. The application of oPSAT to other problems combining hard and soft constraints is also a direction to be explored.

Acknowledgments. The first author acknowledges the support of Fapesp-Brazil grant 2011/19860-4 and CNPq grant PQ 302553/2010-0. This work was supported by an NSF Expeditions in Computing Award on Computational Sustainability (Award Number 0832782).

References

1. Boole, G.: An Investigation on the Laws of Thought. Macmillan, London (1854), available on project Gutemberg at `http://www.gutenberg.org/etext/15114`
2. de Moura, L., Bjørner, N.: Z3: An efficient SMT solver. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 337–340. Springer, Heidelberg (2008)
3. Ermon, S., Le Bras, R., Gomes, C.P., Selman, B., van Dover, R.B.: SMT-aided combinatorial materials discovery. In: Cimatti, A., Sebastiani, R. (eds.) SAT 2012. LNCS, vol. 7317, pp. 172–185. Springer, Heidelberg (2012)
4. Finger, M., Bona, G.D.: Probabilistic satisfiability: Logic-based algorithms and phase transition. In: Walsh, T. (ed.) IJCAI, pp. 528–533. IJCAI/AAAI (2011)
5. Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: IJCAI, pp. 1300–1309. Springer (1999)
6. Gent, I.P., Walsh, T.: The SAT phase transition. In: ECAI 1994 – Proceedings of the Eleventh European Conference on Artificial Intelligence, pp. 105–109. John Wiley & Sons (1994)
7. Georgakopoulos, G., Kavvadias, D., Papadimitriou, C.H.: Probabilistic satisfiability. Journal of Complexity 4(1), 1–11 (1988)
8. Giunchiglia, E., Maratea, M.: Solving optimization problems with dll. In: Brewka, G., Coradeschi, S., Perini, A., Traverso, P. (eds.) ECAI 2006, pp. 377–381. IOS Press (2006)
9. Gregoire, J.M., Tague, M.E., Cahen, S., Khan, S., Abruna, H.D., DiSalvo, F.J., van Dover, R.B.: Improved fuel cell oxidation catalysis in pt1-xtax. Chem. Mater. 22(3), 1080 (2010)
10. Gregoire, J.M., Dale, D., van Dover, R.B.: A wavelet transform algorithm for peak detection and application to powder x-ray diffraction data. Review of Scientific Instruments 82(1), 015105–015105 (2011)
11. Hansen, P., Jaumard, B.: Probabilistic satisfiability. In: Handbook of Defeasible Reasoning and Uncertainty Management Systems, vol. 5, p. 321. Springer, Netherlands (2000)
12. Hansen, P., Jaumard, B., Nguetsé, G.B.D., de Aragão, M.P.: Models and algorithms for probabilistic and bayesian logic. In: IJCAI, pp. 1862–1868 (1995)
13. Hansen, P., Perron, S.: Merging the local and global approaches to probabilistic satisfiability. Int. J. Approx. Reasoning 47(2), 125–140 (2008)
14. Kavvadias, D., Papadimitriou, C.H.: A linear programming approach to reasoning about probabilities. Annals of Mathematics and Artificial Intelligence 1, 189–205 (1990), `http://dx.doi.org/10.1007/BF01531078`
15. Le Bras, R., Bernstein, R., Gomes, C.P., Selman, B.: Crowdsourcing backdoor identification for combinatorial optimization. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, IJCAI 2013 (2013)
16. LeBras, R., Damoulas, T., Gregoire, J.M., Sabharwal, A., Gomes, C.P., van Dover, R.B.: Constraint reasoning and kernel clustering for pattern decomposition with scaling. In: Lee, J. (ed.) CP 2011. LNCS, vol. 6876, pp. 508–522. Springer, Heidelberg (2011), `http://dl.acm.org/citation.cfm?id=2041160.2041202`
17. Mitchell, D., Selman, B., Levesque, H.: Hard and easy distributions of SAT problems. In: AAAI 1992 – Proceedings of the 10th National Conference on Artificial Intelligence, pp. 459–465 (1992)
18. Narasimhan, B., Mallapragada, S., Porter, M.: Combinatorial Materials Science. Wiley (2007), `http://books.google.com/books?id=tRdvxlL7mL0C`

19. Nilsson, N.: Probabilistic logic. Artificial Intelligence 28(1), 71–87 (1986)
20. Nilsson, N.: Probabilistic logic revisited. Artificial Intelligence 59(1-2), 39–42 (1993)
21. Papadimitriou, C., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity. Dover (1998)
22. De Raedt, L., Frasconi, P., Kersting, K., Muggleton, S. (eds.): Probabilistic Inductive Logic Programming. LNCS (LNAI), vol. 4911. Springer, Heidelberg (2008)
23. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62(1-2), 107–136 (2006)
24. Van Dover, R.B., Schneemeyer, L., Fleming, R.: Discovery of a useful thin-film dielectric using a composition-spread approach. Nature 392(6672), 162–164 (1998)
25. Warners, J.P.: A linear-time transformation of linear inequalities into conjunctive normal form. Inf. Process. Lett. 68(2), 63–69 (1998)