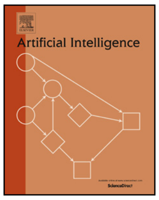
Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint



Probabilistic sentence satisfiability: An approach to PSAT

T.C. Henderson [a,*], R. Simmons [a], B. Serbinowski [a], M. Cline [a], D. Sacharny [a], X. Fan [b], A. Mitiche [c]

[a] University of Utah, SLC, UT USA
[b] Swansea University, Wales, UK
[c] INRS, EMT, Montreal, Canada

ARTICLE INFO

Article history:
Received 16 August 2018
Received in revised form 5 August 2019
Accepted 29 October 2019
Available online 4 November 2019

Keywords:
PSAT
Probabilistic knowledge base
Nonlinear systems

ABSTRACT

Information analysis often involves heterogeneous sources expressed as logical sentences, numerical models, sensor data, etc. Each of these has its own way to describe uncertainty or error; e.g., frequency analysis, algorithmic truncation, floating point roundoff, Gaussian distributions, etc. A unifying framework is proposed here to represent this information as logical sentences with associated probabilities in order to allow the inference of the probability of a query sentence.
Given such a knowledge base in Conjunctive Normal Form (CNF) for use by an intelligent agent, with probabilities assigned to the conjuncts, the probability of any new query sentence can be determined by solving the Probabilistic Satisfiability Problem (PSAT). This involves finding a consistent probability distribution over the atoms (if they are independent) or complete conjunction set of the atoms. For each sentence in the knowledge base, we propose to produce an equation in terms of atoms and conditional probabilities. This system of equations is then solved numerically to get a solution consistent with the sentence probabilities. Finding such a solution is called the Probabilistic Sentence Satisfiability (PS-SAT) problem. In particular, findings include:

1. For independent logical variables:
   (a) atom probabilities which solve PS-SAT also provide a PSAT solution.
   (b) numerical experiments demonstrate a q-superlinear convergence rate for most test cases.
   (c) problems with up to 1,000 variables and 300 sentences are solved.
2. For general knowledge bases (i.e., variables not independent):
   (a) both atom and a subset of conditional probabilities must be found,
   (b) a solution to PS-SAT does not guarantee a solution to PSAT, but most empirical results provide such a solution.
   (c) The convergence rate for equations with non-independent variables also appears q-superlinear.

* Corresponding author.
E-mail address: tch@cs.utah.edu (T.C. Henderson).

https://doi.org/10.1016/j.artint.2019.103199
0004-3702/© 2019 Elsevier B.V. All rights reserved.

1. Introduction

A major motivation for the work presented here is to provide a framework for uncertainty quantification in geospatial intelligence systems as exemplified by *BRECCIA* [1] which receives information from humans (as logical statements), simulations (e.g., weather or platform physics), and sensors (e.g., cameras, weather instruments, microphones, etc.), where each piece of information has an associated certainty. *BRECCIA* then provides coherent responses to user queries concerning UAV flight missions based on the PS-SAT methods described here. Example information (expressed as logical sentences) might include:

Sentence 1: Raven_1_Platform_Available [0.93]
Sentence 2: Wind_Less_Than_17_Knots [0.87]
Sentence 3: Smoke_Obscures_BLDG_21 [0.70]

Each of these sentences has an associated probability (shown in square brackets after the sentence) based in this case on: (1) maintenance history, (2) sensor error model, and (3) human determination. The PS-SAT approach presented here is particularly well-suited to this probabilistic knowledge base formulation, and moreover, allows the determination of the best allocation of information acquisition resources to increase certainty in a given query. Moreover, it is possible to validate both knowledge base clause probabilities, as well as probabilities assigned to queries by measuring the appropriate uncertainties related to the query variables.

The paper is presented as follows. First, the background information required to technically define the problem is given. Next, related work is discussed as well as how the proposed method differs from current approaches. Next, the proposed method is described in terms of systems of equations that are generated from the probabilistic knowledge base (including the difference between handling independent variables and non-independent variables), and numerical techniques are provided to solve these systems of equations It is shown that the method is *Fixed-Parameter Tractable* (FPT) (e.g., is polynomial once the maximum clause length is fixed; see [2]). Experiments are then described which characterize the performance of the methods, and it is demonstrated that they outperform the most related works (e.g., that of Hansen and Perron [3]). Finally, future work is described and includes: (1) a deeper examination of numerical methods for solving the equations, (2) discovery of better initial starting points, and (3) the incorporation of a more technical argumentation framework.

2. Background

The effective and efficient confluence of logic and probability has long been a goal of mathematics and artificial intelligence; e.g., see [4] for an early study that undergirds most modern approaches. Here we follow Bacchus' development of defining probabilities on propositions (for details see [5]). First, it is necessary to define a suitable algebraic structure so that probabilities may be correctly defined.

Define a collection of sets, $\mathcal{F}$, as a *field of sets* if:

1. $\mathcal{F}$ contains a universal set $V \ni \forall S \in \mathcal{F}, S \subseteq V$,
2. $H \in \mathcal{F}$ implies $\overline{H} \in \mathcal{F}$, i.e., the complement of H wrt V is in $\mathcal{F}$, and
3. $H, G \in \mathcal{F}$ implies $H \cup G \in \mathcal{F}$.

Given such an $\mathcal{F}$, a probability function, P, may be defined over $\mathcal{F}$:

1. *Total Probability*: $P(V) = 1$ where V is the universal set in $\mathcal{F}$,
2. *Positivity*: $P(H) \geq 0$, if $H \in \mathcal{F}$, and
3. *Additivity*: $H \cap G = \emptyset$ implies $P(H \cup G) = P(H) + P(G)$.

Next, propositional logic is defined as a set of atomic variables (atoms), $A = \{A_1, A_2, \ldots, A_n\}$, and a set of logical connectives or operators, $C = \{\neg, \vee, \wedge\}$. The language, $\mathcal{L}$, is the set of all well-formed formulas and is defined recursively as:

1. (A_i), where $A_i \in A$,
2. $(\neg\sigma)$, where $\sigma \in \mathcal{L}$,
3. $(\sigma_1 \vee \sigma_2)$, where $\sigma_1, \sigma_2 \in \mathcal{L}$,
4. $(\sigma_1 \wedge \sigma_2)$, where $\sigma_1, \sigma_2 \in \mathcal{L}$,

$\mathcal{L}$ is the closure of the atomic variables using these connectives to generate formulas.

Atoms are assigned truth values, i.e., $v(A_i) \leftarrow$ *true* or *false*, and the standard truth value functions are given for the logical connectives. Given a logical sentence $\sigma \in \mathcal{L}$, then $v(\sigma)$ is defined using the connective truth functions applied to the atomic truth assignments. Ω is defined as the set of complete conjunctions (CC), i.e., all possible truth assignments to the atoms; although this is a set, we will consider it in the order of the elements as binary numbers. The truth value of formula σ given a truth assignment $\omega \in \Omega$ is denoted by $v(\sigma, \omega)$. If τ is a tautology, then $v(\tau, \omega)$ is *true* for all $\omega \in \Omega$. A contradiction is the negation of a tautology. An *equivalence class*, $\mathcal{E}$, is the set of all formulas such that $\sigma_1, \sigma_2 \in \mathcal{E}$ implies that $\forall \omega \in \Omega \;\; v(\sigma_1, \omega) = v(\sigma_2, \omega)$.

Bacchus showed that a Boolean algebra may be defined over the set of equivalence classes using the logical connectives as operators. If [0] denotes the smallest element in the algebra (i.e., contradictions), and [1] denotes the largest element (tautologies), then a probability function, μ, can be defined as follows:

1. $\mu([1]) = 1$, and
2. if $[\sigma_1 \wedge \sigma_2] \equiv [0]$, then

$$\mu([\sigma_1 \vee \sigma_2]) = \mu([\sigma_1]) + \mu([\sigma_2])$$

Then given a logical language, its associated Boolean algebra (in this case, the Lindenbaum-Tarksi algebra), and a probability function, the probability of any formula may be determined.

Several initial observations are in order. The logical language and the structures necessary to assign probabilities to formulas are now in place, but as Bacchus points out, the probabilities are given at the semantic level, and not at the syntactic level. That is, the language cannot use these probabilities. Bacchus goes on to describe an extension of Halpern's probability logic [6] which includes the power to represent and make inferences over the probabilities of sentences. That is not the goal in the current work.

Consider a knowledge base, K, from propositional calculus expressed in Conjunctive Normal Form (CNF) over n logical variables:

$$K = C_1 \wedge C_2 \wedge \ldots \wedge C_m$$

where $C_i, i = 1 \ldots m$ is a *conjunct.* and:

$$C_i = L_{i,1} \vee L_{i,2} \vee \ldots \vee L_{i,k_i}$$

where $L_{i,j}$ is a *literal*, i.e., either a logical atom (variable) or its negation. This is a sublanguage of that defined above.

The *Satisfiability Problem* (SAT) is to determine if there exists a truth assignment, ω, to the n atoms such that K is true; if so, we say that ω satisfies K (denoted $\omega \models K$). A simple method to solve SAT is to generate and test all 2^n truth assignments; i.e., the *complete conjunction set*. This approach is guaranteed to find a solution if it exists, but has $O(2^n)$ complexity.

Each conjunct in K is called a *clause* or *sentence*. K is *consistent* if K has a SAT solution; otherwise, it is *inconsistent*. For example, $A \vee \neg A$ is consistent, while $A \wedge \neg A$ is inconsistent. We note that SAT solvers are very efficient, and large formulas can be solved, even though SAT is NP-complete.

Suppose that a probability is assigned to each conjunct. That is:

$$K = C_1[p_1] \wedge C_2[p_2] \wedge \ldots \wedge C_m[p_m]$$

where p_i is the probability of C_i.

Definition. A function $\bar{\pi} : \Omega \to [0, 1]$ is a *consistent probability distribution* with respect to K for Ω iff:

1. $\forall i, 0 \leq \bar{\pi}(\omega_i) \leq 1$.
2. $\sum_{i=0}^{2^n-1} \bar{\pi}(\omega_i) = 1$.
3. $p_i = \sum_{\omega_k \in \Omega, \omega_k \models C_i} \bar{\pi}(\omega_k)$.

Definition. The *Probabilistic Satisfiability (PSAT) Problem* is to determine if for a given knowledge base, K, there exists a consistent probability distribution for Ω with respect to K.

Nilsson [7] proposed methods to find a consistent assignment of probabilities to logical clauses (essentially rediscovering Boole's method – also analyzed by Hailperin [8,9]). Georgakopoulos [10] showed this problem to be NP-complete and exponential in n. Nilsson posed this problem in terms of probabilities over the possible worlds as follows: given K, determine whether there exists an $m \times 2^n$ binary matrix A and a probability distribution function $\bar{\pi}$ such that:

$$A\bar{\pi} = \bar{p}$$

where $\bar{p} = [p_1, p_2, \ldots, p_m]^T$ from K, and A is defined as:

$$A(i, j) = \begin{cases} 1 & \text{if } \omega_j \text{ satisfies } C_i \\ 0 & \text{otherwise} \end{cases}$$

Note that the probabilities of the complete conjunction set form a basis for probabilities of the set of clauses as follows: for any clause, C_i:

$$P(C_i) = \sum_{\omega_k \models C_i} \bar{\pi}(\omega_k)$$

This is true because the complete conjunction set forms a partition of the event space, and the probability of any clause is the sum of the probabilities of the complete conjunctions that satisfy the clause.

[7] solves for $\bar{\pi}$, the probabilities of the complete conjunction set, and shows how probabilistic inference can be performed for an arbitrary query, so that the probability of any clause over the n variables can be computed. However, for a given K, PSAT, just like SAT, may have more than one solution, and may in fact, have an uncountable number of solutions. To see this, consider Nilsson's example (Modus Ponens):

$$K = P[0.7] \wedge (\neg P \vee Q)[0.7]$$

Then:

$$\Omega = \{\neg A \wedge \neg B, \neg A \wedge B, A \wedge \neg B, A \wedge B\}$$
$$\bar{p} = [0.7, 0.7]^T$$
$$A = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$
$$\bar{\pi} = [0, 0.3, 0.3, 0.4]^T$$

where $\bar{\pi}$ is found using least squares methods. (Note that an extra row of 1's is added to A as is an extra row with 1 to the $\bar{p}$ vector so as to force the sum of the elements of $\bar{\pi}$ to be 1.) It turns out that $\bar{\pi} = [0.3, 0, 0.3, 0.4]^T$ is also a solution for this problem. In fact, if we define:

$$\bar{\pi}(0,0) \in [0, 0.3]$$
$$\bar{\pi}(0,1) = 0.3 - \bar{\pi}(0,0)$$
$$\bar{\pi}(1,0) = 0.3$$
$$\bar{\pi}(1,1) = 0.4$$

then these equations describe an uncountable set of solutions for this problem.

Bacchus [5] states that Nilsson's method is equivalent to the algebraic field of sets approach described above where in this case the probability distribution is defined over the field of sets of possible worlds (complete conjunctions) of the language. The possible world semantics provides a standard denotational semantics, and a probabilistic knowledge base is just a set of constraints on possible world probabilities.

Definition. A knowledge base K is *probabilistically consistent* if there exists a consistent probability distribution $\bar{\pi}$ for the Ω arising from K. Otherwise, K is *probabilistically inconsistent*.

Consider:

$$K = P[0.7] \wedge (\neg P \vee Q)[0.7] \wedge Q[0.2]$$

Then K is probabilistically inconsistent (as shown by Nilsson). Nilsson also considered a geometric approach in the sentence probability space where a query clause was added as a last dimension to the clause set. For more detailed discussion of this approach, including the geometric approach, see [11]. This method produces a matrix A that is exponential size in the number of variables.

3. Related work

For broader discussions of this problem and approaches to solving it, see [5,12–16]. Adams early work focused on the probability of conditionals and its relation to measurement uncertainty. Hailperin proposed a way to allow the expression of statistical knowledge as well as constraints on the probability of the truthfulness of formulas in a formal first order logical framework. Unfortunately, the "set of valid formulas of the logic with probabilities on possible worlds is not recursively enumerable" (see [17]). Thus, it is not possible to find a finitary axiomatization. Belle and Lakemeyer have recently extended the work of Bacchus et al. [18] to $\mathcal{DS}$ (**D**egrees of belief in the **S**ituational calculus) which "captures a family of only knowing logics" [19]. We note that Belle gives as future directions the implementation (possibly) of a propositional version of $\mathcal{DS}$ as well as a way to "allow both discrete and continuous probabilities, although it is not clear how this can be achieved."

The results achieved in probabilistic first-order logic have been exploited to extend logic programming environments to include probabilities. For example, ProbLog is one Prolog-based language [20]. Such languages are interesting, but face some serious challenges, including: (1) a need for efficient inference methods for broader language feature support, (2) an analysis of their relative computational complexity, and (3) how to learn probabilistic programs (currently learned from entailment, although some attempts have been made to learn from interpretations).

Chavira and Darwiche [21] consider the problem of probabilistic inference in terms of weighted model counting on a propositional knowledge base. A given Bayesian network (and consequently a known full joint probability distribution) is converted to Conjunctive Normal Form, and the network probabilities are used to assign weights to the CNF variables. These are then used to obtain weights on the models. Finally, the evidence is used to select consistent models and sum their weights. This approach assumes model probabilities are available, i.e., $Pr(x_1, x_2, \ldots, x_n) = \prod Pr(x_i \mid u_i)$. Another approach based on Bayesian networks (Bayesian Logic – BLOG) is described by Milch and Russell [22,23], wherein the PSAT solution is assumed. Chakraborty et al. [24] have proposed distribution-aware sampling to achieve weighted model counting, and assume an NP-oracle (they use a SAT solver as the black box oracle for weights of truth assignments – i.e., possible world probabilities). The PS-SAT approach does not require a PSAT solution as input, and does not involve any explicit representation of such a solution (either as a function or set of probabilities). Moreover, direct comparison to their benchmark set is not possible if the knowledge bases do not have independent logical variables. Moreover, approximately counting the models of a CNF formula is known to be NP-hard [25]. Their algorithm uses a SAT solver, and moreover, requires bounded *tilt* to succeed, where *tilt* is $\omega_{max}/\omega_{min}$, the maximum and minimum probabilities of worlds making the formula *true*. However, there are problem classes for which the tilt increases exponentially with n. For instance, consider:

$$F_n = A_1 \vee A_2 \vee \ldots \vee A_n$$

and let:

$$\omega(\sigma) = 2^{-s}$$

where s is the number of true variables in σ; this weight function gives higher weight when there are less true variables because it is easier to verify. Then ω_{max} is $\frac{1}{2}$ when just one variable is true, and ω_{min} is $\frac{1}{2^n}$ so that $tilt = 2^{n-1}$ and grows exponentially with n. Knowledge compilation may be used to allow the application of these methods to a much larger range of Bayesian networks [26,27].

Another approach is that of the Statistical Relational Learning (SRL) community [28] which extends graphical models (e.g., Bayesian or Markov networks, see [29] for a detailed account of probabilistic graphical models) to allow relations and logical statements. The main goal is to develop models of object-relational structures of data that has some amount of uncertainty. It is key to build such models so as to allow efficient learning and inference. Current models combine graphical, probability and logical structures. Markov Logic Networks (MLN) [30–32] constitute one main SRL approach. A Markov network models the joint probability distribution of a set of variables. A Markov Logic Network (MLN) is a template for constructing a Markov network from a set of logical formulas (clauses). Moreover, this representation allows the computation of the probability of a possible world given a KB by simply assigning a weight to each KB formula. There are however, some drawbacks to this approach. First, Proposition 2.5 (p. 15) in [31] proves that propositional knowledge bases can be handled by MLN's, but requires a consistent probability distribution (i.e., a PSAT solution) as the potential functions on the one maximal n-clique with n variables. In general, the complexity of solving a query is exponential in the number of cliques in the graph. e.g., consider the simple k-Modus Ponens problem with:

$$K = \{A_1 \wedge (\neg A_1 \vee A_2) \wedge \ldots \wedge (\neg A_{k-1} \vee A_k)\}$$

Then this KB has $O(2^n)$ maximal cliques. In addition, MLN's take a maximum entropy approach to the probabilities of the possible worlds, and it has been shown that maximum entropy distributions are dependent on the representation used [33], which means solutions are ambiguous.

The method proposed here can also be viewed as a probabilistic form of logical argumentation in that it finds a solution even though there may be probabilistic and logical inconsistencies in the knowledge base; i.e., is a form of argumentation in that it finds a (perhaps locally minimal) solution (in terms of minimal sentence probability error) which best fits the given data. Others have also explored inconsistencies in probabilistic knowledge bases [34,35], but not in this direction.

Finally, there has been some work in characterizing and improving linear solver methods for the PSAT problem as posed by Nilsson. Hansen and Jaumard [36] proposed the use of the column generation technique of linear programming in order to increase the effectiveness of their solvers. More recently, Hansen and Perron [3] have given a merged local and global solver strategy for which KB's with up to 200 variables and 800 sentences have been solved. Others [37,38] have empirically studied the solution complexity distribution of the PSAT problem (number of solvable cases and time required as a function of $\frac{m}{n}$ where n is the number of variables and m is the number of sentences. There appears to be a phase transition at about $\frac{m}{n} = 4.3$. The notion of generalized probabilistic satisfiability has also been recently proposed [39]; that is, "deciding the satisfiability of linear inequalities involving probabilities of classical propositional formulas" and is shown to be NP-complete.

The in-depth relationship between the proposed method and existing methods is provided in Table 1. The following acronyms are used:

Table 1
Advantages and Disadvantages of Various Methods. The column headings indicate: (col 1): each possible grounding of a first-order formula is considered; (col 2): the underlying computational complexity is exponential or not; (col 3): the computation requires knowledge of the full joint probability distribution.

	Propositional Version	Efficient Inference	Require Full Joint Prob Distribution
NILS	Yes	Yes	No
Nilsson	Yes	No	No
DS	No	No	No
ProbLog	Yes	No	Yes
CD	Yes	No	Yes
BLOG	Yes	No	Yes
DA	Yes	No	Yes
MLN	Yes	No	Yes
HP	Yes	Somewhat	Yes

- NILS: method proposed here
- Nilsson: Nilsson's linear system formulation
- DS: Bacchus' Degrees of belief in Situational calculus
- ProbLog: Fierens' ProbLog system
- CD: Chavira-Darwiche weighted model counting
- BLOG: Bayesian Logic
- DA: Chakraborty's distribution-aware method
- MLN: Markov Logic Networks
- HP: Hansen and Perron method (improvement of Nilsson)

Thus, it can be seen that NILS offers distinct advantages over the other methods in terms of efficiency and not requiring the full joint probability distribution. An empirical comparison is given between NILS and HP in the experiments section.

4. Method

The application here is a decision support system expressed in CNF form where each sentence (clause) has an associated probability, and decision makers pose queries in terms of variables in the knowledge base. The method proposed here, called *NonlInear Probabilistic Logic Solver (NILS)* involves conversion of the probabilistic CNF to a set of equations (1)–(4) as described above, and using numerical solvers. Having provided the context of the work in terms of languages and representations, the current approach may now be given. Using the laws of probability structure described above, the probability of a disjunctive clause $C = L_1 \vee L_2 \vee \ldots \vee L_k$ can be expressed as:

$$P(C) = P(L_1) + P(R) - P(L_1 \wedge R) \tag{1}$$

$$P(C) = P(L_1) + P(R) - P(L_1 \mid R)P(R) \tag{2}$$

$$P(C) = P(L_1) + P(R) - P(R \mid L_1)P(L_1) \tag{3}$$

$$P(C) = P(L_1) + P(R) - P(L_1)P(R) \tag{4}$$

where $R = L_2 \vee \ldots \vee L_k$; note that the formula must be applied recursively and will have $2^k - 1$ terms. Eqn (1) is the conjunction form and makes a linear equation in the unknown probabilities. Eqns (2) and (3) are expressed with conditional probabilities and are nonlinear. Eqn (4) assumes independent variables and is nonlinear. Then given a knowledge base, equations (1)–(4) contain a set of unknown probabilities over atoms, conditions on atoms, or the logical *and* of atoms.

Definition. The *Probabilistic Sentence Satisfiability (PS-SAT)* problem is to find a set of values for the unknown probabilities in (1)–(4) which produce the given sentence (clause) probabilities on the left hand side.

Here we advance a new method based on converting the probabilistic knowledge base into a set of nonlinear equations which are then solved using some variant of gradient descent (e.g., Newton's method). This is strictly speaking an approximation method, and it aims to produce an assignment of probabilities to the logical expressions in the equations so as to obtain the given clause probabilities. This may or may not lead to a consistent probability distribution. For example, given a knowledge base (KB) with $A[0.6] \wedge \neg A[0.6]$ which clearly has no consistent probability, the solution produced is P(A) = 0.5. In solving PS-SAT, the complexity is determined by the solution procedure and will, in general, be bounded by $O(2^k)$, where k is the maximum number of literals in a sentence.

The *NILS* method consists then of the following steps:

1. Convert K to a system of nonlinear equations expressing the clause probabilities in terms of one of Equations (1) to (4).
2. Solve numerically, constraining each unknown to be in the [0,1] interval.

Given a CNF clause of length k, the resulting formula will have $2^k - 1$ terms. In order to avoid this complexity, we assume that an arbitrary PSAT clause can be converted to a 3PSAT set of clauses; otherwise, the complexity of the method is $O(2^k)$, where k is the number of literals in the longest clause. At the present time, it is straightforward to transform a single clause of length four or more to a 3PSAT KB which maintains the probability of the initial clause (i.e., when queried); however, a general proof is not yet known. The value k provides a parameterization of this problem, so that it is, in fact, *fixed-parameter tractable* (see [40]). This means that to determine if (K, k) is in the PS-SAT language (K a probabilistic CNF knowledge base, and k as above) is decidable in $f(k)\cdot |x|^{O(1)}$, where $f(k) = 2^k$, and, thus, is in class FPT (at least for independent variable KB's).

4.1. Independent variables

Assuming the variables are independent allows use of Eqn (4). Thus, clauses with one, two or three literals produce the following equations, respectively:

$$P(C) = P(L)$$
$$P(C) = P(L_1) + P(L_2) - P(L_1)P(L_2)$$
$$P(C) = P(L_1) + P(L_2)P(L_3) - P(L_1)P(L_2) -$$
$$P(L_1)P(L_3) - P(L_2)P(L_3) + P(L_1)P(L_2)P(L_3)$$

Disjunctions of length k will have 2^k terms. Now consider Nilsson's Modus Ponens example on two variables:

$$K = A_1[0.7] \wedge (\neg A_1 \vee A_2)[0.7]$$

This gives rise to the following equations:

$$0.7 = P(A_1) \tag{5}$$
$$0.7 = P(A_1) + P(A_2) - P(A_1)P(A_2) \tag{6}$$

Substituting the value for $P(A_1)$ from (5) into (6), we find that $P(A_2) = 0.571$. This specific case requires no search, and unlike Nilsson's method is not exponential. However, in general the complexity of the method is related to the convergence properties of the numerical solver.

An important question is whether or not this solves PSAT for a given K. Suppose the method produces probabilities for the n atoms in K such that the clause probabilities are produced. Then we can show that there exists a consistent probability distribution for K which can be computed from the atom probabilities. It is not necessary to generate these 2^n values since the probability of any clause can be computed just using the atom probabilities.

Theorem. *Given K with n independent logical atoms, and an assignment of probabilities, a_k, to the atoms such that the nonlinear equations produce the sentence probabilities, $\bar{p}$, then $\exists \bar{\pi} : \Omega \to [0, 1] \ni \bar{\pi}$ is a consistent probability distribution.*

Proof. By induction on n.

Case $n = 2$: Consider the $\omega_k \in \Omega$:

$$P(\neg A \wedge \neg B) = P(\neg A)P(\neg B)$$
$$P(\neg A \wedge B) = P(\neg A)P(B)$$
$$P(A \wedge \neg B) = P(A)P(\neg B)$$
$$P(A \wedge B) = P(A)P(B)$$

In the sum of these, the first and third yield $P(\neg B)$ while the second and fourth result in $P(B)$. These sum to 1.

Case n: Consider the $\omega_k \in \Omega$; each has a counterpart:

$$P(\omega_k) = P(\neg A_k \wedge X) = P(\neg A_k)P(X)$$
$$P(\omega_{k+2^{n-1}}) = P(A_k \wedge X) = P(A_k)P(X)$$

which sums to: P(X). Altogether these produce the sum of the complete conjunction set of $n - 1$ variables, which by induction is 1. □

It would be great if all consistent knowledge bases had an independent variable solution, but that is not the case. For example:

$$K = A_1[0.7] \wedge (\neg A_1 \vee A_2)[0.5] \wedge (\neg A_2 \vee A_3)[0.5]$$

has a consistent probability distribution over Ω, but not for independent variables. This can be seen as follows: from the first clause, $P(A_1) = 0.7$. Substituting that into the equation for the second clause yields: $P(A_2) = 0.2857$. If this value is substituted into the equation for the third clause, we get $P(A_3) = -0.7501$ which is not in [0,1]. On the other hand, the assignment

$$\pi = [0, 0, 0.2999, 0, 0.25, 0.25, 0.2001, 0]$$

is a consistent probability assignment, and $P(A \wedge B) = 0.2001$ while $P(A)P(B) = 0.7 \cdot 0.5 = 0.35$, so that $P(A \wedge B) \neq P(A)P(B)$. The question is then: if a consistent probability distribution exists for a knowledge base with independent variables, will the gradient descent method find it?

4.2. Non-independent variables

Consider variables which are not necessarily independent. In this case we use the disjunction probability Eqn (2). Unlike the independent case where only the atom probabilities were computed (or needed), now, there is a set of conditional probabilities which expands the set of variables to be solved. E.g.:

$$K = (A_1 \vee A_2)[0.7]$$

gives rise to:

$$0.7 = x(1) + x(2) - x(3)x(2)$$

where $x(1)$ is $P(A_1)$, $x(2)$ is $P(A_2)$, and $x(3)$ is $P(A_1 \mid A_2)$, and which has solution: $x(1) = 0.4663$, $x(2) = 0.4834$, and $x(3) = 0.5166$. However, now the equations from the clauses do not include all the necessary constraints arising from the conditional variables, and equations must be added to include these. In particular, a conditional $P(A_1 \mid A_2)$ gives rise to three additional constraints expressed in terms of the conditional and related variables:

$$P(\overline{A}_1 \mid A_2) = 1 - P(A_1 \mid A_2) \quad \text{(Complement rule)} \tag{7}$$

$$\begin{aligned} P(A_1 \mid \overline{A_2}) &= \frac{P(A_1 \cap \overline{A_2})}{P(\overline{A_2})} \\ &= \frac{P(A_1) - P(A_1 \cap A_2)}{P(\overline{A_2})} \\ &= \frac{P(A_1)}{P(\overline{A_2})} - \frac{P(A_1 \cap A_2)}{P(\overline{A_2})} \\ &= \frac{P(A_1)}{1 - P(A_2)} - \frac{P(A_1 \cap A_2)}{1 - P(A_2)} \cdot \frac{P(A_2)}{P(A_2)} \\ &= \frac{P(A_1) - P(A_2)P(A_1 \mid A_2)}{1 - P(A_2)} \end{aligned} \tag{8}$$

$$\begin{aligned} &P(\overline{A_1} \mid \overline{A_2}): \\ &\quad \text{Let } \overline{A_1} = C \text{ and use above} \\ P(C \mid \overline{A_2}) &= \frac{P(C) - P(A_2)P(C \mid A_2)}{1 - P(A_2)} \\ &= \frac{1 - P(A_1) - P(A_2)(1 - P(A_1 \mid A_2)}{1 - P(A_2)} \end{aligned} \tag{9}$$

Another issue that arises with non-independent variables is that the solution may not allow the direct determination of the probability of a query. This may happen if the query involves a conditional probability not included in the original knowledge base. For example:

$$K = A[0.8] \wedge B[0.9]$$

NILS produces $P(A) = 0.8$ and $P(B) = 0.9$ for K. Given the query $A \vee B$, the equation to be solved is:

$$P(A \vee B) = P(A) + P(B) - P(A \mid B)P(B)$$

The result is then $P(A \vee B) \in [0.8, 1]$ with $P(A \mid B) \in [0.7778, 1]$. The Matlab function *lsqlin* returns the solution $P(A \vee B) = 0.8885$ and $P(A \mid B) = 0.9$ right in the center of the interval. Thus, the determination of the probability of a query generally requires solving a new system (using the known values for the knowledge base).

4.3. Numerical solutions

Given a set of nonlinear equations resulting from a CNF KB and the associated sentence probabilities, it is necessary to create the sentence error function, find an initial guess at a solution, and then apply Newton's method or some other technique. We have applied two methods: (1) Newton's method, and (2) gradient descent using the Jacobian. The stability of these methods has been shown for solving nonlinear systems [41].

4.4. Sentence error function

Given a clause, $C_i[p_i]$, the ith element of a sentence error function is defined as:

$$E(i) = -p_i + P(C_i)$$

where $P(C_i)$ is formed from one of the disjunction probability equations. The scalar sentence error is $\|E\|$. A solution is found by choosing an initial solution estimate, and then using the sentence error function to perform gradient descent.

4.4.1. Newton's method

Given the vector function E defined above (a vector function of m elements), Newton's method iterates the following until within tolerance of a solution:

1. Produce next step vector
   $H_{E(\bar{x}^k)}\bar{s} = \nabla E$
2. Move toward solution
   $\bar{x}^{k+1} = \bar{s} + \bar{x}^k$

where H_E is the Hessian matrix for E. The development of the Hessian is done symbolically then solved numerically in Matlab. This imposes constraints of the application of this method to larger problems. However, we give results below for reasonably large KB's.

4.4.2. Gradient descent using the Jacobian

Gradient descent using the Jacobian should have q-quadratic convergence when starting not too far from a solution, but may hit a local (non-solution) minimum otherwise. The method iterates until within tolerance of a solution as follows:

1. Determine the Jacobian
   $J = \nabla E$
2. Move toward lower sentence error
   $\bar{x}^{k+1} = \alpha * J(\bar{x}^k) + \bar{x}^k$

Consider Nilsson's Modus Ponens example:

$$K = A_1[0.7] \wedge (\neg A_1 \vee A_2)[0.7]$$

which gives rise to:

$$E(1) = -0.7 + x(1)$$

$$E(2) = -0.7 + (1 - x(1)) + x(2) - (1 - x(1))x(2)$$

Fig. 1 shows the convergence path for some random initial values (all ending at $[0.7; 0.571]^T$ for the atom probabilities).

Given a knowledge base and an assignment of probabilities to the set of atom and conditional probability variables, its quality can be determined by the final sentence error. If it is low, say less than 0.01, then it is a useful approximation. Of course, the gradient descent method can be run from several initial starting vectors and the best result selected. Better methods for initial point selection (other than random) are discussed below.

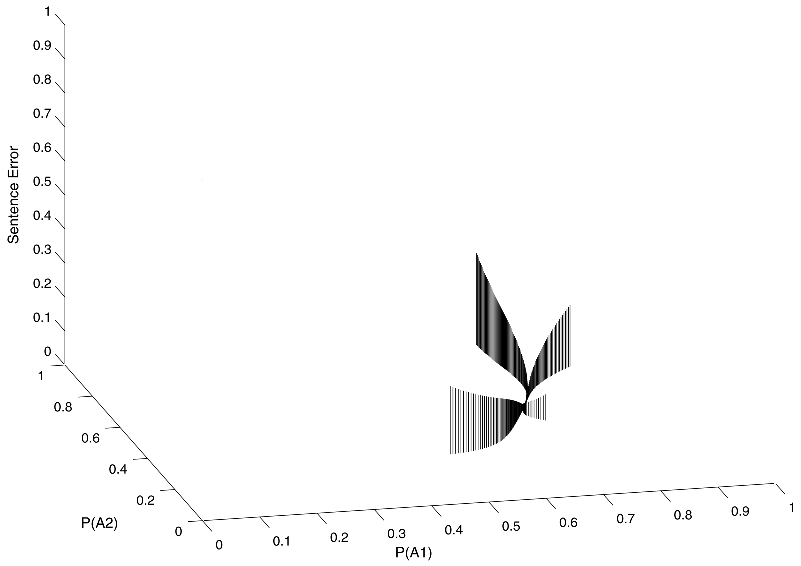


Fig. 1. Convergence Tracks for 4 Random Starting Points for Modus Ponens; This Assumes Independent Variables.

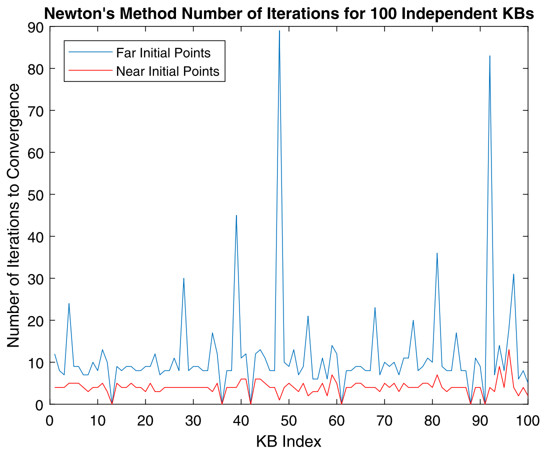


Fig. 2. Newton's Method Results for 100 KB's.

5. Experiments and results

We have tested this approach on sets of randomly generated knowledge bases. This involves selecting a number of variables (n), specifying a maximum number of sentences to generate, as well as the maximal length of any one sentence. A set of sentences is generated which satisfies these constraints, and then a set of probabilities is produced for the complete conjunction set, and from these the sentence probabilities are computed. This ensures that there is a solution, although it does not preclude the existence of other solutions (generally a non-zero measure subset of the unit hypercube). We have applied this to both independent and non-independent variable knowledge bases.

5.1. *Independent variables*

A set of 100 KB's was generated with independent variables, with $n = 5$, the maximum number of clauses 30, and the maximal clause length of 5. Fig. 2 shows the number of iterations required by Newton's method to solve PSAT; the blue trace shows when initial points are far from the known solution, and red when they are near (within 0.5 vector norm). The mean number of iterations is 4.26 when starting near, and 12.63 when starting far. The method fails on 6 of the 100 KB's. As for gradient descent, Fig. 3 shows the number of required iterations. Although a few KB's require over 1000 iterations, the mean number of iterations required when starting near a solution is 21.19, and when starting far is 171.58. Note that the search is terminated when a sentence error of less than 0.01 is reached. We have also generated larger KB's and run the method on those and solved problems with up to 1000 variables and 300 sentences. These are produced by generating a set

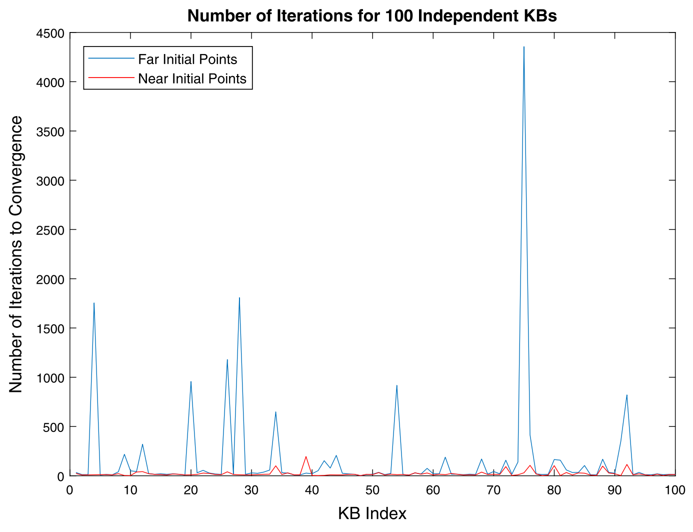


Fig. 3. Gradient Descent Results for 100 KB's. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

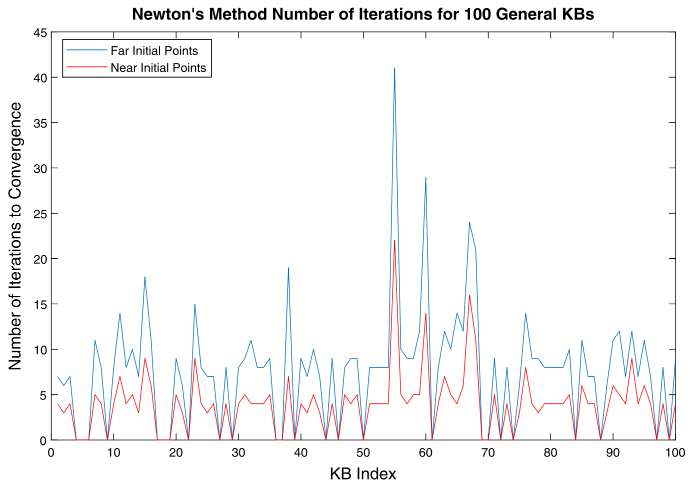


Fig. 4. Newton's Method Results for 100 KB's.

of atom probabilities and computing clause probabilities from those, then finding the independent solution and comparing the atom probabilities. This avoids the necessity of producing a representation for the full PSAT probability distribution over possible worlds.

5.2. Non-independent variables

The equations must include variables for whatever conditional probabilities arise from the sentences, and are thus a bit more complicated. Fig. 4 shows the number of iterations required for Newton's method on 100 random general KB's with the same parameters as above, except that $len_{max} = 3$. In this case, solutions were found for 75 of the 100 KB's, and the mean number of iterations was 3.91 when starting near the known solution (within 0.1 of any atom probability), and 10.27 when starting far from it. Fig. 5 shows the results for gradient descent (which found solutions for all the KB's) and had mean number of iterations 662.17 for far starting points and mean number of iterations 638.61 for near points.

What these results indicate is that Newton's method should be tried first given the low iteration cost, and then gradient descent used if Newton's Method fails. Also, note that even though in the case of failure (i.e., local minimum found), the methods were allowed to re-start at new random initial locations. Gradient descent was re-started this way and then only tried 2 alternate points. When Newton's Method finds a solution is does so with the initial guess; when it failed, it did so for both near and far initial starting points.

Finally, Fig. 6 shows the maximal individual atom probability error comparing the atom probabilities from the actual 100 general KB's to the atom probabilities found by the numerical solver. The mean of the max atom probability error for near

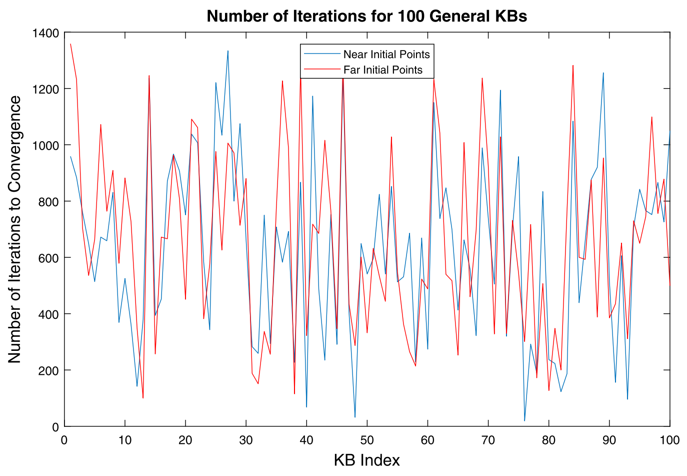


Fig. 5. Gradient Descent Results for 100 KB's.

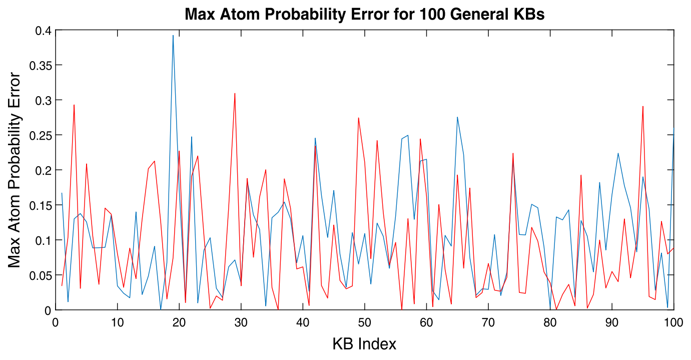


Fig. 6. Maximum Atom Probability Error for 100 General KB's (Near Starting Points in red and Far Starting Points in blue).

starting points is 0.09, while for far starting points is 0.10. This is very promising in that the discovered solutions are near the actual underlying solution for most KB's.

5.3. *Trajectory visualization and finding good initial guesses*

As pointed out above, if the initial guess is too far from a solution, these methods may not converge. Thus, it would help to be able to identify good starting points. In order to get insight into the convergence sequence, we have developed a visualization method which maps n-D points to 2-D points. Given a point, $\bar{a}$, in n-D, define the corresponding 2-D coordinates as follows:

$$x = \sum_{i=1}^{n} (a_i cos(\frac{(i-1)\pi}{n})) \tag{10}$$

$$y = \sum_{i=1}^{n} (a_i sin(\frac{(i-1)\pi}{n})) \tag{11}$$

Fig. 7 shows the convergence trajectories for four different initial points. The q-convergence of the method can be estimated by determining the c_k's in the following equation:

$$\mid \bar{x}^{k+1} - \bar{x}^* \mid \leq c_k \mid \bar{x}^k - \bar{x}^* \mid \tag{12}$$

Fig. 8 shows these values for the 100 tracks for gradient descent on the general KB's. The plots indicate that the method is q-superlinear/quadratic.

Another interesting aspect of this visualization method is its use to find good starting points. Given fixed x and y in the plane, we have developed a method to obtain a unique point in the pre-image of Eqns (1) and (2). Each equation defines a hyperplane in n-space; taken together they represent a hyperplane of dimension $n-2$. One way to understand the map defined by Eqns (1) and (2) is as an n-joint prismatic manipulator, where joint k translates in the direction $\theta = \frac{(k-1)\pi}{n}$.

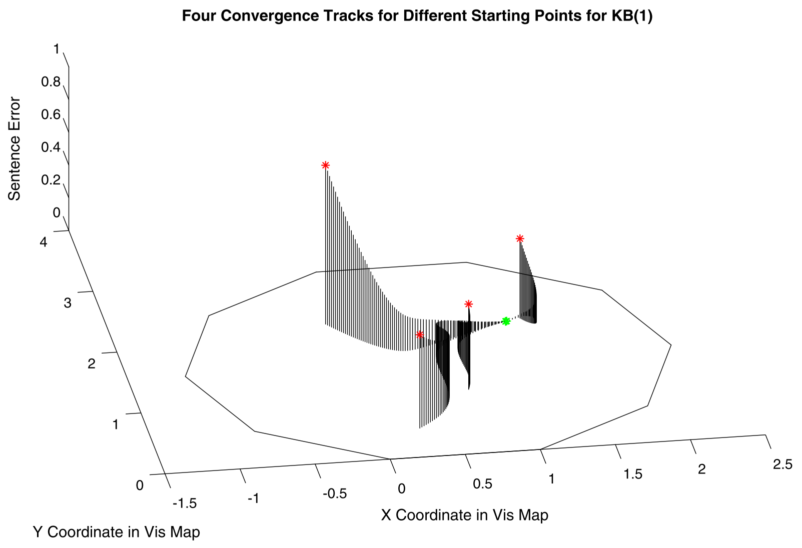


Fig. 7. Convergence Tracks for 4 Random Starting Points for 5-D Problem; x and y Values are Projections of 5-D Points to 2-D, and z Value is Sentence Error Value.

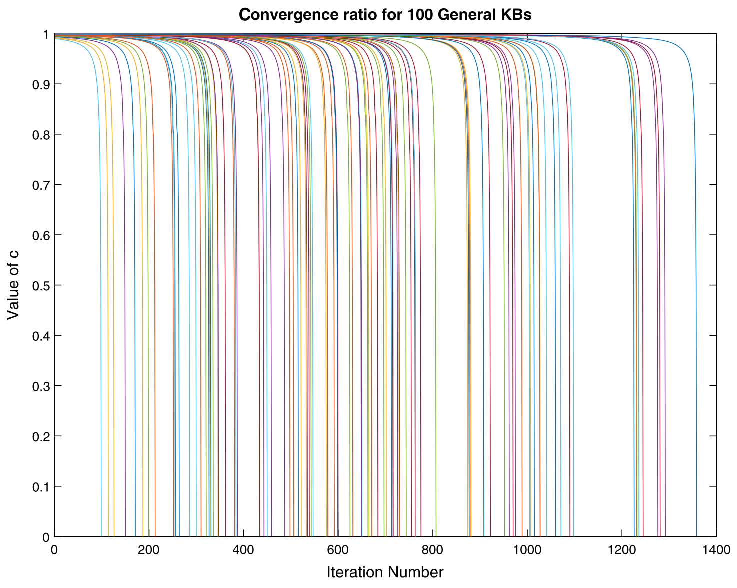


Fig. 8. Convergence step ratio for 100 general KB's using Gradient Descent.

The manipulator's workspace is a $2n$-gon (as shown in Fig. 7). By uniformly sampling this workspace, and then finding pre-image points in n-D, the sentence error can be found, and then the lowest such value used to pick the initial point. Of course, since there is a potentially infinite number of pre-image points for each x and y location, other methods can be used to sample that subspace to find better starting points.

6. Conclusions and future work

We propose a novel approach to solve PS-SAT which avoids the computational complexity of previous methods as well as the error introduced using MC-SAT methods. Instead we solve a system of nonlinear equations derived directly from the meaning of the probability of the logical sentences. The experiments reported here show that solving these systems is possible and not overly complex (evidence shows q-superlinear/quadratic convergence). The number of variables and sentences used in these experiments is beyond current state-of-the-art work on directly solving PSAT (e.g., [3]). Moreover, the method is Fixed-Parameter Tractable, where the fixed-parameter k is the length of the longest clause, and the associated function is 2^k.

Other future work includes the investigation of:

1. The problem encountered with Newton's Method. It is possible that the Hessian as computed does not remain positive definite which can cause failure. It may be possible to address this with SVD methods.
2. The discovery of good initial starting points. For this, the trajectory visualization method will be studied; i.e., the inverse kinematics of the planar n-joint prismatic manipulator.
3. The exploitation of the method to support a knowledge base providing probabilistic logic and in the future, argumentation. Such a capability will provide decision makers and analysts a robust estimate of the confidence of a statement or the consequences of an action. The application domain for this is geospatial knowledge bases [1]. Given a query for a KB with independent variables, the solution to any logical formula may be found from the atom probabilities. However, for KB's with non-independent variables, the equations resulting from the query may involve new conditional probabilities, and thus, requires the use of a solver when there is more than one unknown. Queries in the current version of NILS are restricted to disjunctions with less than four literals.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work.

Acknowledgements

This research supported in part by Dynamic Data Driven Application Systems AFOSR grant FA9550-17-1-0077.

References

[1] D. Sacharny, T. Henderson, R. Simmons, A. Mitiche, T. Welker, X. Fan, BRECCIA: a novel multi-source fusion framework for dynamic geospatial data analysis, in: Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, Daegu, South Korea, 2017.
[2] J. Flum, M. Grohe, Parameterized Complexity Theory, Springer Verlag, Berlin, Germany, 2006.
[3] P. Hansen, S. Perron, Merging the local and global approaches to probabilistic satisfiability, Int. J. Approx. Reason. 47 (2008) 125–140.
[4] G. Boole, An Investigation of the Laws of Thought, Walton and Maberly, London, UK, 1857.
[5] F. Bacchus, Representing and Reasoning with Probabilistic Knowledge, MIT Press, Cambridge, MA, 1990.
[6] J. Halpern, An analysis of first-order logics of probability, Artif. Intell. J. 46 (3) (1990) 311–350.
[7] N. Nilsson, Probabilistic logic, Artif. Intell. J. 28 (1986) 71–87.
[8] T. Hailperin, Best possible inequalities for the probability of a logical function of events, Am. Math. Mon. 72 (4) (1965) 343–359.
[9] T. Hailperin, Probability logic, Notre Dame J. Form. Log. 25 (3) (1984) 198–212.
[10] G. Georgakopoulos, D. Kavvadias, C. Papadimitriou, Probabilistic satisfiability, J. Complex. 4 (1988) 1–11.
[11] T. Henderson, A. Mitiche, R. Simmons, X. Fan, A Preliminary Study of Probabilistic Argumentation, Tech. Rep. UUCS-17-001, University of Utah, February 2017.
[12] E. Adams, A Primer of Probability Logic, CLSI Publications, Stanford, CA, 1998.
[13] T. Hailperin, Boole's Logic and Probability, North Holland Publishing Company, Amsterdam, the Netherlands, 1976.
[14] T. Hailperin, Sentential Probability Logic, Lehigh University Press, Cranbury, NJ, 1996.
[15] A. Hunter, A probabilistic approach to modelling uncertain logical arguments, Int. J. Approx. Reason. 54 (2013) 47–81.
[16] Z. Ognjanovic, M. Raskovic, Some first-order probability logics, J. Theor. Comput. Sci. 247 (2000) 191–212.
[17] M. Abadi, J. Halpern, Decidability and expresiveness for first-order logics of probability, J. Inf. Comput. 112 (1994) 1–36.
[18] F. Bacchus, J. Halpern, H. Levesque, Reasoning about noisy sensors and effectors in the situation calculus, Artif. Intell. J. 111 (1–2) (1999) 171–208.
[19] V. Belle, G. Lakemeyer, Reasoning about probabilities in unbounded first-order dynamical domains, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia, 2017.
[20] D. Fierens, G. van den Broack, J. Renkens, D. Shterionov, B. Gutmann, I. Thon, G. Janssens, L. de Raedt, Inference and learning in probabilistic logic programs using weighted Boolean formulas, Artif. Intell. J. Theory Pract. Logic Program. 15 (3) (2015) 358–401.
[21] M. Chavira, A. Darwiche, On probabilistic inference by weighted model counting, Artif. Intell. J. 172 (2008) 772–799.
[22] B. Milch, B. Marthi, D. Sontag, S. Russell, D. Ong, A. Kolobov, BLOG: probabilistic models with unknown objects, in: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, 2005.
[23] B. Milch, Probabilistic Models with Unknown Objects, Ph.D. thesis, University of California, Berkeley, Berkeley, CA, 2006.
[24] S. Chakraborty, D. Fremont, K. Meel, S. Seshia, M. Vardi, Distribution-aware sampling and weighted model counting for SAT, in: Proceedings of the Twenty-Eigth AAAI Conference on Artificial Intelligence, Quebec City, Canada, 2014, pp. 1722–1730.
[25] D. Roth, On the hardness of approximate reasoning, Artif. Intell. J. 82 (1996) 273–302.
[26] G. Dal, S. Michels, P. Lucas, Reducing the cost of probabilistic knowledge compilation, Proc. Mach. Learn. Res. 73 (2017) 141–152.
[27] A. Darwiche, P. Marquis, A knowledge compilation map, J. Artif. Intell. Res. 17 (2002) 229–264.
[28] L. Getoor, B. Taskar, Introduction to Statistical Relational Learning, MIT Press, Cambridge, MA, 2007.
[29] D. Koller, N. Friedman, Proabilistic Graphical Models: Principles and Techniques, MIT Press, Cambridge, MA, 2009.
[30] M. Biba, Integrating Logic and Probability: Algorithmic Improvements in Markov Logic Networks, Ph.D. thesis, University of Bari, Bari, Italy, 2009.
[31] P. Domingos, D. Lowd, Markov Logic: An Interface Layer for Artificial Intelligence, Morgan and Claypool, San Rafael, CA, 2009.
[32] V. Gogate, P. Domingos, Probabilistic theorem proving, Commun. ACM 59 (7) (2016) 107–115.
[33] T. Seidenfeld, Why I am not an objective Bayesian, Theory Decis. 11 (1979) 413–449.
[34] M. Thimm, Measuring inconsistency in probabilistic knowledge bases, in: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, Montreal, Quebec, Canada, 2009, pp. 530–537.
[35] M. Thimm, A probabilistic semantics for abstract argumentation, in: Proceedings of the 20th European Conference on Artificial Intelligence, Montpellier, France, 2012.
[36] P. Hansen, B. Jaumard, Probabilistic Satisfiability, March 1996.

[37] M. Finger, G. de Bona, Probabilistic satisfiability: logic-based algorithms and phase transitions, in: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, Barcelona, Spain, 2011.
[38] M. Finger, G. de Bona, Probabilistic satisfiability: algorithms with the presence and absence of a phase transitions, Ann. Math. Artif. Intell. 75 (3–4) (2015) 351–389.
[39] C. Caleiro, F. Casal, A. Mordido, Generalized probabilistic satisfiability, Electron. Notes Theor. Comput. Sci. 332 (2017) 39–56.
[40] R. Downey, M. Fellow, Parameterized Complexity, Springer Verlag, Berlin, Germany, 1999.
[41] H. Wozniaskowski, Numerical stability for solving nonlinear equations, Numer. Math. 27 (4) (1976) 373–390.