Probabilistic reasoning as Quadratic Unconstrained Binary Optimization

Marco Baioletti
marco.baioletti@unipg.it
University of Perugia
Perugia, Italy

ABSTRACT

Probabilistic reasoning is an important tool for using uncertainty in AI, especially for automated reasoning. Partial probability assessments are a way of expressing partial probabilistic knowledge on a set of events. These assessments contain only the information about "interesting" events (hence it can be easily assessed by an expert). On the other hand, partial assessments can cause consistency problems. In this paper we show how to formulate the main tasks of probabilistic reasoning on partial probability assessments, namely check of coherence, correction, and inference, as QUBO problems. This transformation allows to solve these problems with a quantum or a digital annealer and thus providing new computational methods to perform these tasks.

ACM Reference Format:
Marco Baioletti. 2022. Probabilistic reasoning as Quadratic Unconstrained Binary Optimization. In *Genetic and Evolutionary Computation Conference Companion (GECCO'22 Companion), July 9–13, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3520304.3534005

1 INTRODUCTION

Probabilistic models are an important tool in Artificial Intelligence, used in many research areas. In automated reasoning there many applications of probability theory: among them it is worth to mention probabilistic logic [14], Bayesian networks [15], and probabilistic logic programming [16].

Probability theory has a strong mathematical foundation and it is usually preferred as a basis for AI applications to other uncertainty models (for instance, belief functions).

Anyway, it is difficult to define and handle complete probabilistic assessments. In fact, a complete assessment on n events would require a probability value for each of all the 2^n composed propositions. Therefore, it would be unpractical for an expert to provide such a large number of probability values, most of them are difficult to be assessed because they are associated to "uninteresting" combinations. Moreover, most operations on complete assessments would be unfeasible due to high computational costs.

Hence, it is common to work with partial probabilistic assessments. A partial probabilistic assessment is a piece of information expressing the subjective belief about the probability values of some events of interest. An event can be usually described as a proposition in the language of propositional logic $\mathcal{L}$, defined in terms of the usual Boolean connectives and of a set of propositional variables.

In this paper, we use the formulation introduced in [10], where the probability values are assigned to the propositional variables, and the events are implicitly defined in terms of logical constraints.

Definition 1. *A probability assessment is defined as a quadruple* $\pi = (V, U, \mathbf{p}, C)$, *where*

- $V = \{X_1, \ldots, X_{\bar{n}}\}$ *is a finite set of propositional variables, denoting events,*
- $U = \{X_1, \ldots, X_n\}$ *is a subset of* V *that contains the effective events taken into consideration,*
- $\mathbf{p} = (p_1, \ldots, p_n)$ *is a list of probability values (a rational number in* $[0, 1]$*) for each variable in* U, *and*
- C *is a finite set of logical constraints which lie among all the variables in* V.

In Definition 1, a probability value is provided only for the elements of U, but the logical constraints can also be written in terms of all the existing events in V. This feature allows to extend an initial assessment to a larger domain without redefining the whole model. Moreover, by adding auxiliary variables in V, it is always possible to express C in Conjunctive Normal Form, where each clause $c \in C$ is the disjunction of at most three literals (either a variable or its negation). Let $m = |C|$ denote the number of clauses in C.

For instance, a probabilistic assessment can state that $P(X_1) = 0.1$, $P(X_2) = 0.2$ and $P(X_3) = 0.25$, where X_1, X_2, X_3 are propositional variables, related with the logical constraint $X_3 \equiv X_1 \vee X_2$, which can be translated in CNF as $C = \{\neg X_3 \vee X_1 \vee X_2, \neg X_1 \vee X_3, \neg X_2 \vee X_3\}$.

It is important to notice that a probabilistic assessment π assigns a probability value only to some events, and therefore it may or may not be coherent, i.e. consistent with a probability distribution. This concept is clearly the analogous of logical satisfiability for probabilistic assessments [14]. On the other hand, coherence would not be a problem for complete assessments: for them it is sufficient that all values are in $[0, 1]$ and their sum is 1.

Another important aspect of using partial assessment is that the expert is required to give only probability values for "interesting events", which can be a reasonable number, compared to all the possible events.

In the previous example, the assessment is clearly coherent. On the other hand, changing $P(X_3)$ to a value smaller than 0.2, the new

GECCO '22 Companion, July 9–13, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9268-6/22/07…$15.00
https://doi.org/10.1145/3520304.3534005

assessment would be incoherent, because the laws of probability calculus state that $P(X_3) = P(X_1 \vee X_2) \geq \max(P(X_1), P(X_2)) = 0.2$.

To formalize this concept we provide the following definitions.

Definition 2. *A truth assignment on V is a function $\alpha : V \rightarrow \{0,1\}$. We denote by 2^V the set of all truth assignments. We denote with $\alpha \models q$ the fact that the assignment $\alpha \in 2^V$ satisfies the proposition q.*

Definition 3. *A truth assignment $\alpha \in 2^V$ is called **atom** for a probability assessment $\pi = (V, U, \mathbf{p}, C)$ if α satisfies all the clauses $c \in C$. We denote by $Atm(\pi)$ the set of all atoms of π.*

Definition 4. *A probability assessment $\pi = (V, U, \mathbf{p}, C)$ is **coherent** if there exists a probability distribution $\mu : 2^V \longrightarrow [0,1]$ which satisfies the following properties*

(1) *for each $\alpha \in 2^V$, if $\mu(\alpha) > 0$, then $\alpha \in Atm(\pi)$;*
(2) $\displaystyle\sum_{\alpha \in 2^V} \mu(\alpha) = 1$;
(3) $\displaystyle\sum_{\alpha \in 2^V, \alpha \models X_i} \mu(\alpha) = p_i$, *for all $i = 1, \ldots, n$.*

The problem of checking the coherence of a probability assessment, called **CPA**, has been already studied in [1, 3–5], using the Disjunctive Normal Form, showing that it is a NP-complete problem.

The computational problem **CPA** is strictly related to the Probabilistic Satisfiabiliy problem **PSAT** [12], where the probability assessment is defined on a finite set of propositions, instead that on the propositional variables. It is possible to prove that every instance of **CPA** can be easily translated as a **PSAT** instance, and that every **PSAT** can be formulated in a normal form, which is essentially a particular **CPA** instance [9].

A naive way of solving this problem would require to find all the atoms $Atm(\pi) = \alpha_1, \ldots, \alpha_s$ for π and solve a linear systems of $n+1$ equations, corresponding to Properties (2) and (3) of Definition 4, and s unknowns, corresponding to the values $\mu(\alpha_i)$, for $i = 1, \ldots, n$. This approach is clearly unfeasible, due to the fact that, in general, $s = O(2^n)$.

Hence, there exist several algorithmical approaches to solve **CPA** and **PSAT** problems:

- a column-generation [12, 13] algorithm, where the problem is solved using linear programming techniques which exploit the sparsity of the solutions;
- the CPA algorithm [4, 5], which is based on a symbolic manipulation which, in some cases, needs a further linear programming procedure;
- a SAT-based approach [11], in which the problem is translated in a pure propositional satisfiability form (SAT);
- a MIP-based approach [9], in which the problem is formulated as a mixed integer programming problem (MIP).

In particular, the MIP-based approach has proved to be very effective as reported in [9], where their implementation was able to handle coherence testing instances up to 1000 variables and 1000 disjunctive clauses in average time ranging from some seconds to some minutes.

Other tasks in probabilistic reasoning are the correction and the extension of a probabilistic assessment.

The **correction** of a probabilistic assessment π [7] can be performed when π is incoherent. In that case, it would be interesting to change the probability values assigned to the events in U, obtaining a new probability vector $\mathbf{p}'$, such that the new probabilistic assessment $\pi' = (V, U, \mathbf{p}', C)$ is coherent and the distance $d(\mathbf{p}, \mathbf{p}')$ is minimal, according to a suitable distance between real vectors. In this way, the probability values are changed as least as possible, in order to obtain a coherent assessment.

In general, this computational problem is cast as a constrained mixed minimization problem. In [2], the correction is based on the L_1 distance, thus a MIP solver can be employed, while in other papers (for instance in [7]) the L_2 distance is used, hence a quadratic programming tool is required.

On the other hand, if π is coherent, it would be interesting to compute the probability value for a new event (which, without loss of generality, can be denoted by X_{n+1} and its probability by p_{n+1}), such that the extended assessment is still coherent. This task is called **inference** (of $P(X_{n+1})$) or **extension** (of π).

It is possible to prove that the set of values for p_{n+1} is a not empty interval $[p^L_{n+1}, p^U_{n+1}]$ [1], where $0 \leq p^L_{n+1} \leq p^U_{n+1} \leq 1$ [10]. Hence, this task reduces to compute the extremal values p^L_{n+1} and p^U_{n+1}, from π, which requires to solve two similar constrained mixed optimization problems.

The computational resources needed to solve these problems about probabilistic assessments are however large, although the recent implementations, using MIP and SAT solvers (the latter only for the check of coherence), are quite efficient.

Therefore, it would be very useful to formulate these tasks as Quadratic Unconstrained Binary Optimization (QUBO) problems, exploiting the presence of dedicated hardware devices, such as quantum annealer (like D-Wave 2000Q or the D-Wave Advantage) and digital annealers (like Fujitsu's Digital Annealer and other similar devices), whose performances are very good, if compared with classical computational devices.

A QUBO problem is the optimization problem of finding the minimum value of a quadratic objective function $F(y) = y^T A y$, where y is a vector of n binary variables (y^T is its transpose), and A is a symmetric square matrix of dimension n.

In the recent literature, there have been several attempts to formulate Artificial Intelligence problems as QUBO or Ising problems. Among them, it is worth to mention the study of how to solve SAT and MAX-SAT using a quantum annealer [6]. However, to the best of our knowledge, there are no previous attempts to cast probabilistic reasoning tasks as QUBO problems.

2 ENCODING OF THE COHERENCE PROBLEM

In this section, we show how to encode the problem of checking the coherence of a probabilistic assessment π as a QUBO problem.

This encoding is inspired by the MIP-approach used in [9] and in [2]. This approach exploits the fact that π is coherent if and only if there exists a probability distribution μ on 2^V which is non zero only on at most $n+1$ atoms (denoted by $\hat{\alpha}_1, \ldots, \hat{\alpha}_{n+1}$). Moreover, as shown in [9], the probability values for the atoms can be determined with a limited precision.

[1] which can collapse to $[0]$ if X_{n+1} is a contradiction, i.e., it is not satisfied by any atoms

Hence, all the rational numbers present in the encoding will be written in the fixed point form with r digits after the point.

The encoding uses four sets of binary variables.

The first set is composed by $\bar{n}(n+1)$ binary variables a_{ij}, for $i = 1, \ldots, \bar{n}$ and $j = 1, \ldots, n+1$, which represent the value of each Boolean variable X_i in every atom $\hat{\alpha}_j$.

Denoting by $q_j \in [0,1]$ the unknown probability $\mu(\hat{\alpha}_j)$, for $j = 1, \ldots, n+1$, the second set contains the variables ϕ_{js}, for $s = 1, \ldots, r$ and $j = 1, \ldots, n+1$, where ϕ_{js} is s-th binary digit of q_j.

The value of q_j can be recovered as

$$q_j = \sum_{s=1}^{r} \phi_{js} 2^{-s}$$

The constraint $\sum_{j=1}^{n+1} q_j = 1$, which corresponds to the Property (2) in Definition 4, can therefore be written as

$$\sum_{j=1}^{n+1} \sum_{s=1}^{r} \phi_{js} 2^{-s} = 1 \tag{1}$$

Let b_{ij} denote the unknown real number $q_j \cdot a_{ij}$, for $i = 1, \ldots, \bar{n}$ and $j = 1, \ldots, n+1$. It is easy to see that b_{ij} corresponds to q_j, if $a_{ij} = 1$, 0 otherwise.

The third set of binary variables contains the β_{ijs}, for $i = 1, \ldots, \bar{n}$, $j = 1, \ldots, n+1$, and $s = 1, \ldots, r$. β_{ijs} is the s-th binary digit of b_{ij}, so that

$$b_{ij} = \sum_{s=1}^{r} \beta_{ijs} 2^{-s}.$$

With some simple algebra, it can be shown that

$$\beta_{ijs} = \phi_{js} \wedge a_{ij} \tag{2}$$

must hold, where $\wedge$ is the AND operator.

Each constraint $P(X_i) = p_i$, for $i = 1, \ldots, n$, becomes

$$\sum_{j=1}^{n+1} b_{ij} = p_i,$$

which can be written as

$$\sum_{s=1}^{r} \sum_{j=1}^{n+1} \beta_{ijs} 2^{-s} = p_i \tag{3}$$

Finally, for each clause $c_h \in C$ and each atom $\hat{\alpha}_j$ there is a binary variable w_{hj}, which is used to encode the clause as a binary quadratic constraint, as shown later.

To summarize, let y be the binary vector comprising all the binary variables needed in the encoding, namely

- a_{ij}, for $i = 1, \ldots, \bar{n}$ and $j = 1, \ldots, n+1$;
- ϕ_{js}, for $j = 1, \ldots, n+1$ and $s = 1, \ldots, r$;
- β_{ijs}, for $i = 1, \ldots, \bar{n}$, $j = 1, \ldots, n+1$ and $s = 1, \ldots, r$;
- w_{hj}, for $h = 1, \ldots, m$ and for $j = 1, \ldots, n+1$.

The dimension of y is $\bar{n}(n+1) + r((n+1) + \bar{n}(n+1)) + m(n+1) = (n+1)(\bar{n}(r+1) + r + m)$, which is a polynomial of $n, \bar{n}, m$, and r.

The formulation of checking the coherence of π as a QUBO problem can be described by the following quadratic function, which is the objective function to be minimized:

$$F(y) = F_1(y) + F_2(y) + F_3(y) + F_4(y) \tag{4}$$

The first component F_1 corresponds to the constraint (1)

$$F_1(y) = \left(\sum_{j=1}^{n+1} \sum_{s=1}^{r} \phi_{js} 2^{-s} - 1 \right)^2$$

The second component F_2 corresponds to all the constraints (2)

$$F_2(y) = \sum_{i=1}^{\bar{n}} \sum_{j=1}^{n+1} \sum_{s=1}^{r} AND(\phi_{js}, a_{ij}, \beta_{ijs})$$

where $AND(x_1, x_2, z) = x_1 x_2 - 2(x_1 + x_2)z + 3z$ encodes the constraint that z is the logical conjunction of x_1 and x_2.

The third component F_3 corresponds to all the constraints (3)

$$F_3(y) = \sum_{i=1}^{\bar{n}} \left(\sum_{s=1}^{r} \sum_{j=1}^{n+1} \beta_{ijs} 2^{-s} - p_i \right)^2$$

The last component F_4, is the number of violations of the logical constraints C, i.e. the sum, for each atom $\hat{\alpha}_j$, of the number of clauses not satisfied by $\hat{\alpha}_j$.

$$F_4(y) = \sum_{j=1}^{n+1} \sum_{h=1}^{m} F_4^h(a_{1j}, \ldots, a_{nj}, w_{hj})$$

where

$$F_4^h(a_{1j}, \ldots, a_{nj}, w_{hj})$$

is a quadratic binary function whose minimum value (with respect to the dummy variable w_{hj}) is 1 if and only if the atom $\hat{\alpha}_j$ does not satisfy the clause c_h [17].

The definition of each F_4^h depends on the presence of the negated variables in c_h [18].

If all the literals in c_h are variables, namely $c_h = X_{k_{h1}} \vee X_{k_{h2}} \vee X_{k_{h3}}$, then

$$\begin{aligned} F_4^h(a_{1j}, \ldots, a_{nj}, w_{hj}) &= 1 - (1 + w_{hj})(a_{k_{h1},j} + a_{k_{h2},j} + a_{k_{h3},j}) \\ &\quad + a_{k_{h1},j} a_{k_{h2},j} + a_{k_{h1},j} a_{k_{h3},j} + a_{k_{h2},j} a_{k_{h3},j} + 2w_{hj} \end{aligned}$$

For the other seven possibilities, it is sufficient to replace $a_{k_{hl},j}$ with $1 - a_{k_{hl},j}$ in the definition of F_L^h, for all the literals which appear negated in c_h.

It is easy to show that π is coherent if and only if the minimum value of $F(y)$ is zero. In fact, if π is coherent, there exists a sparse probability distribution on 2^V, hence it is possible to find a binary vector y which satisfies all the constraints encoded in $F(y)$, hence $F(y) = 0$.

On the other hand, if $F(y) = 0$ for some binary vector y, it is possible to extract from y a sparse probability distribution on 2^V which satisfies Properties (1)–(3) of Definition 4.

3 ENCODING OF THE CORRECTION PROBLEM

As shown in the previous section, if π is not coherent, the minimum value of $F(y)$ is greater than 0. The related solution y violates some of the constraints imposed by the encoding, then it could not correspond to a meaningful probabilistic object. For instance y can lead to a set of logical assignments, which are not atoms of π, or to a non normalized probability distribution, or to other meaningless situations.

However, if y is such that $F_1(y) + F_2(y) + F_4(y) = 0$, then it is possible to extract a valid sparse probability distribution on 2^V, but which does not correspond to $\bar{p}$ on the events $X_1, \ldots, X_n$.

It is important to observe that in this case, the probability values for the events in U form a new vector $\mathbf{p}'$, such that the Euclidean distance $d(\mathbf{p}, \mathbf{p}')$, which corresponds exactly to $F_3(y)$, is minimal. If C is satisfiable, i.e., $Atm(\pi) \neq \emptyset$, it is always possible to find such a probability distribution, and hence the corresponding vector y.

Therefore, the formulation of the revision problem of an incoherent probabilistic assessment can be easily formulated as a QUBO problem. In fact, its objective function is a simple modification of the previously introduced $F(y)$, namely

$$\tilde{F}(y) = F_3(y) + K\left(F_1(y) + F_2(y) + F_4(y)\right)$$

where K is real positive number, large enough to force the minimization engine to choose only vectors y such that $F_1(y)+F_2(y)+F_4(y) = 0$.

4 ENCODING OF THE INFERENCE PROBLEM

The inference problem requires to find the minimum and the maximum values p^L_{n+1}, p^U_{n+1} for the probability of a new event X_{n+1}. The logical constraint set C should be enriched with the logical relations of the previous existing events with X_{n+1}.

It is possible to define two different objective functions for finding the extremal values as QUBO problems.

Namely, the value of p^L_{n+1} can be found by minimizing the following quadratic function

$$F_L(y) = G_{n+1}(y) + K_L F(y)$$

where

$$G_{n+1}(y) = \sum_{s=1}^{r} \sum_{j=1}^{n+1} \beta_{n+1,j,s} 2^{-s}$$

corresponds to the value of $P(X_{n+1})$.

The constant K_L should be large enough to obtain only solutions y which satisfies $F(y) = 0$.

Therefore, the minimum value for F_L is exactly the value for p^L_{n+1}.

In a similar way, the value p^U_{n+1} can be obtained by minimizing the quadratic function

$$F_U(y) = 1 - G_{n+1}(y) + K_L F_(y)$$

and taking as result $1 - F_{U,min}$, where $F_{U,min}$ is the minimum value of F_U.

5 CONCLUSION AND FUTURE WORK

In this paper we have seen how to formulate some important tasks in probabilistic reasoning as QUBO problems, in order to be solved with quantum or digital annealers, with the aim of reducing the computation times and being able to solve larger instances.

One of the key points which affects the efficiency of the encoding is the number of binary variables, i.e., the size of y. This parameter depends on the size of π and on the way the logical constraint are described. In particular, $\bar{n}$ is the sum of n (the number of events we are interested in) and the number of variables needed for expressing C in conjunctive clausal form, with at most three literals per clause. Of course, if C is at the beginning in that form, no additional variable is required and $\bar{n} = n$ (except that in the inference task).

A further reduction of the number of binary variables occurs when all the clauses have only one or two literals: in that case, the auxiliary variables w_h are not needed. This form of **CPA** is called **2-CPA** and is still a NP-complete problem, as shown in [5]. Moreover, **2-CPA** is expressive to represent some form of logical constraints, like incompatibility and implication relation among pairs of events.

Anyway, there could be also other ways of reducing the number of binary variables involved in the encoding. In particular, there exist cases where the variables added to produce the CNF form and the dummy variables w_{hj} can be unified, thus producing a consistent saving in the number variables.

As a future line of research we intend to test our QUBO encoding in real or simulated annealers, in order to evaluate the efficiency of our solutions. A prototypical implementation of the encoding is being developed and will be soon downloadable at `https://github.com/mbaioletti/probQUBO`.

Although the number of variables needed in the QUBO encoding is polynomial with respect to the size of the probabilistic assessment, it is however large and using QUBO formulations in fast annealers will be convenient as soon as the number of bits of these devices will increase.

In general, it would be interesting to look for other effective QUBO encoding methods for the problems at hand.

Another point to be investigated is how to assess the values for the constants K and K_L needed in the encoding for the correction and the inference tasks.

Finally, another point to be addressed in a future work is to apply these techniques to other probabilistic reasoning tasks, for instance handling conditional probabilistic assessments [7, 8], in which the values of the conditional probability $P(X_i|X_j)$ of some events are provided.

REFERENCES

[1] Kim Allan Andersen and Daniele Pretolani. 2001. Easy Cases of Probabilistic Satisfiability. *Ann. Math. Artif. Intell.* 33, 1 (2001), 69–91. https://doi.org/10.1023/A:1012332915908

[2] Marco Baioletti and Andrea Capotorti. 2019. A L1 based probabilistic merging algorithm and its application to statistical matching. *Appl. Intell.* 49, 1 (2019), 112–124. https://doi.org/10.1007/s10489-018-1233-z

[3] Marco Baioletti, Andrea Capotorti, and Sauro Tulipani. 2006. An empirical complexity study for a 2CPA solver. In *Modern Information Processing*, Bernadette Bouchon-Meunier, Giulianella Coletti, and Ronald R. Yager (Eds.). Elsevier Science, Amsterdam, 73–84. https://doi.org/10.1016/B978-044452075-3/50007-8

[4] Marco Baioletti, Andrea Capotorti, Sauro Tulipani, and Barbara Vantaggi. 2000. Elimination of Boolean variables for probabilistic coherence. *Soft Comput.* 4, 2 (2000), 81–88. https://doi.org/10.1007/s005000000040

[5] Marco Baioletti, Andrea Capotorti, Sauro Tulipani, and Barbara Vantaggi. 2002. Simplification Rules for the Coherent Probability Assessment Problem. *Ann. Math. Artif. Intell.* 35, 1-4 (2002), 11–28. https://doi.org/10.1023/A:1014585822798

[6] Zhengbing Bian, Fabián A. Chudak, William G. Macready, Aidan Roy, Roberto Sebastiani, and Stefano Varotti. 2020. Solving SAT (and MaxSAT) with a quantum annealer: Foundations, encodings, and preliminary results. *Inf. Comput.* 275 (2020), 104609. https://doi.org/10.1016/j.ic.2020.104609

[7] Andrea Capotorti, Giuliana Regoli, and Francesca Vattari. 2010. Correction of incoherent conditional probability assessments. *Int. J. Approx. Reason.* 51, 6 (2010), 718–727. https://doi.org/10.1016/j.ijar.2010.02.002

[8] Giulanella Coletti and Romano Scozzafava. 2002. *Probabilistic Logic in a Coherent Setting.* Dordrecht: Kluwer.

[9] Fábio Gagliardi Cozman and Lucas Fargoni di Ianni. 2015. Probabilistic satisfiability and coherence checking through integer programming. *Int. J. Approx. Reason.* 58 (2015), 57–70. https://doi.org/10.1016/j.ijar.2014.09.002

[10] Bruno de Finetti. 1974. *Theory of probability.* London: Wiley & Sons.

[11] Marcelo Finger and Glauber De Bona. 2011. Probabilistic Satisfiability: Logic-Based Algorithms and Phase Transition. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, Toby Walsh (Ed.). IJCAI/AAAI, 528–533. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-096

[12] George F. Georgakopoulos, Dimitris J. Kavvadias, and Christos H. Papadimitriou. 1988. Probabilistic satisfiability. *J. Complex.* 4, 1 (1988), 1–11. https://doi.org/10.1016/0885-064X(88)90006-4

[13] Brigitte Jaumard, Pierre Hansen, and Marcus Poggi de Aragão. 1991. Column Generation Methods for Probabilistic Logic. *INFORMS J. Comput.* 3, 2 (1991), 135–148. https://doi.org/10.1287/ijoc.3.2.135

[14] Nils J. Nilsson. 1986. Probabilistic Logic. *Artif. Intell.* 28, 1 (1986), 71–87. https://doi.org/10.1016/0004-3702(86)90031-7

[15] Judea Pearl. 1989. *Probabilistic reasoning in intelligent systems - networks of plausible inference.* Morgan Kaufmann.

[16] Luc De Raedt and Angelika Kimmig. 2015. Probabilistic (logic) programming concepts. *Mach. Learn.* 100, 1 (2015), 5–47. https://doi.org/10.1007/s10994-015-5494-z

[17] I.G. Rosenberg. 1975. Reduction of bivalent maximization to the quadratic case. *Cahiers du Centre d'Etudes de Recherche Opérationnelle* 17 (1975), 71–74.

[18] Amit Verma, Mark Lewis, and Gary Kochenberger. 2021. Efficient QUBO transformation for Higher Degree Pseudo Boolean Functions. https://doi.org/10.48550/ARXIV.2107.11695