Almost No News on the Complexity of MAP in Bayesian Networks

Cassio P. de Campos C.DECAMPOS@ACM.ORG
TU Eindhoven, The Netherlands

Abstract

This article discusses the current state of the art in terms of computational complexity for the problem of finding the most probable configuration of a subset of variables in a multivariate domain modelled by probabilistic graphical models such as Markov networks (random fields) or Bayesian networks. It contains complexity proofs and an algorithm for the problem and shows empirical results for Boolean trees which may suggest tractability of the task in some special cases.

Keywords: Bayesian and Markov networks, Maximum a Posterior, Most Probable Explanation.

1. Introduction

A Bayesian network (BN) is a probabilistic graphical model that relies on a structured dependency among random variables to represent a joint probability distribution in a compact manner (Pearl, 1988). One of the hardest inference tasks in BNs is the *maximum a posteriori* (or MAP) problem, where one looks for states of some variables that maximise their joint probability, given some other variables as evidence (there may exist variables that are neither queried nor part of the evidence and thus may need marginalisation). This problem is also called marginal or partial MAP, while the version without variables requiring marginalisation is also called Most Probable Explanation (MPE). In this paper we discuss on the state of the art in computational complexity of such tasks. We revise some known and show some new theoretical results, in particular related to marginal MAP in Bayesian and Markov network trees. It is disappointing, however, that the main goal of closing the question about the complexity of marginal MAP in Boolean trees remains unreachable (to my best knowledge). This paper provides further steps towards closing relevant complexity questions.

The reader is assumed to be familiar with complexity classes and reductions, and with MAP and MPE in Bayesian networks (de Campos, 2011). Of use are the classes NP (non-deterministic polynomial time) and PP (probabilistic polynomial time), as well as oracle constructions $\mathrm{A}^{\mathrm{B}[m]}$ to represent the class A with access to a B-complete oracle such that the use of this oracle of class B is restricted to at most m calls over all computations (m typically is a constant or a function of input size; the notation without $[m]$ means the oracle may be invoked as many times as desired).

Definition 1 *A Bayesian network $\mathcal{N}$ is a triple $(\mathcal{G}, \mathcal{X}, \mathcal{P})$, where $\mathcal{G} = (\mathbf{V}_\mathcal{G}, \mathbf{A}_\mathcal{G})$ is a directed acyclic graph with nodes $\mathbf{V}_\mathcal{G}$ associated (one-to-one) to random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ over discrete domains $\{\Omega_{X_1}, \ldots, \Omega_{X_n}\}$ and $\mathcal{P}$ is a collection of probability values $p(x_i|\pi_{X_i}) \in \mathcal{Q}$, the non-negative rational numbers defined by fractions of integers, with $\sum_{x_i \in \Omega_{X_i}} p(x_i|\pi_{X_i}) = 1$, where $x_i \in \Omega_{X_i}$ is a state of X_i and $\pi_{X_i} \in \times_{X \in \mathbf{PA}_{X_i}} \Omega_X$ a complete instantiation of states for the parents $\mathbf{PA}_{X_i}$ of X_i in $\mathcal{G}$. Furthermore, every variable is conditionally independent of its non-descendant non-parents given its parents (Markov condition).*

The joint probability distribution represented by a BN $\mathcal{N} = (\mathcal{G}, \mathcal{X}, \mathcal{P})$ is obtained by the factorisation implied by the Markov condition: $p(\mathbf{x}) = \prod_i p(x_i|\pi_{X_i})$, where $\mathbf{x} \in \Omega_\mathcal{X}$ and all states x_i, π_{X_i}

(for every i) agree with $\mathbf{x}$. As abuse of notation, singletons $\{X_i\}$ and $\{x_i\}$ may be respectively denoted as X_i and x_i, and events such as $\{X_i = \mathrm{T}\}$ as x_i^T. Nodes of the graph and their associated random variables are used interchangeably. Uppercase letters are used for random variables and lowercase letters for their corresponding states. Bold letters are employed for vectors/sets. The input size of a BN, called simply b here, is defined by the length of the bit string needed to specify all information in the network, including the local conditional probability distributions.

An important concept is called treewidth. Treewidth measures the similarity of the graph with a tree, and strongly relates to the hardness of inferences in BNs (Kwisthout et al., 2010).

Definition 2 *Given a BN $\mathcal{N} = (\mathcal{G}, \mathcal{X}, \mathcal{P})$ with $\mathbf{A}_{\mathcal{G}} \neq \emptyset$, its minimum treewidth $k \geq 0$ is obtained by first moralising $\mathcal{G}$ (that is, connecting using an edge every pair of nodes sharing a common child and then dropping the direction of all arcs) into $\mathcal{G}'$ and then computing the minimum treewidth k of $\mathcal{G}'$, which is such that $\mathcal{G}'$ is a subgraph of a k-tree, but not of any $(k-1)$-tree.*

Note that a BN whose graph has at most one parent per node has minimum treewidth $k = 1$ and is called a BN tree, while a BN polytree also has no cycles in the underlying undirected graph, but they appear because of moralisation.

The MAP problem for input $(\mathcal{N}, \mathbf{X}, \mathbf{e})$ is to find an instantiation $\mathbf{x}_{\text{opt}} \in \Omega_{\mathbf{X}}$, with $\mathbf{X} \subseteq \mathcal{X} \setminus \mathbf{E}$ ($\mathbf{E}$ are evidence variables given along their states $\mathbf{e}$), such that its probability is maximised:

$$\mathbf{x}_{\text{opt}} = \operatorname*{argmax}_{\mathbf{x} \in \Omega_{\mathbf{X}}} p(\mathbf{x}|\mathbf{e}) = \operatorname*{argmax}_{\mathbf{x} \in \Omega_{\mathbf{X}}} p(\mathbf{x}, \mathbf{e}), \tag{1}$$

because $p(\mathbf{e})$ is a constant with respect to the maximisation (if $p(\mathbf{e}) = 0$, then all $\mathbf{x}$ may be considered equally good/bad). In spite of that, the complexity of the decision version of the problem may depend on whether one conditions or not, so I will define both variations here.

Definition 3 *Given a BN $\mathcal{N} = (\mathcal{G}, \mathcal{X}, \mathcal{P})$ chosen among those with maximum cardinality of any variable at most z and minimum treewidth at most w, $\mathbf{X} \subseteq \mathcal{X} \setminus \mathbf{E}$, a rational r and an instantiation $\mathbf{e} \in \Omega_{\mathbf{E}}$, Decision-MAPE-$z$-$w$ is the problem of deciding if there is $\mathbf{x} \in \Omega_{\mathbf{X}}$ such that $p(\mathbf{x}|\mathbf{e}) > r$. If $\mathbf{E} \cup \mathbf{X} = \mathcal{X}$, we call the problem Decision-MPEE-z-w. If the query is changed to deciding whether there is $\mathbf{x}$ such that $p(\mathbf{x}|\mathbf{e}) < r$, then we call it minimum a posterior and Decision-MINAPE.*

Definition 4 *Given a BN $\mathcal{N} = (\mathcal{G}, \mathcal{X}, \mathcal{P})$ chosen among those with maximum cardinality of any variable at most z and minimum treewidth at most w, $\mathbf{X} \subseteq \mathcal{X} \setminus \mathbf{E}$, a rational r and an instantiation $\mathbf{e} \in \Omega_{\mathbf{E}}$, Decision-MAP-$z$-$w$ is the problem of deciding if there is $\mathbf{x} \in \Omega_{\mathbf{X}}$ such that $p(\mathbf{x}, \mathbf{e}) > r$. If $\mathbf{E} \cup \mathbf{X} = \mathcal{X}$, we call the problem Decision-MPE-z-w. If the query is changed to deciding whether there is $\mathbf{x}$ such that $p(\mathbf{x}, \mathbf{e}) < r$, then we call it minimum a posterior and Decision-MINAP.*

Table 1 shows a summary of known results. Before proceeding, it is worth noting that a BN can be trivially cast as a Markov network by using the mass functions in $\mathcal{P}$ as potentials of the Markov network and imposing the same factorisation through the product of potentials.

Definition 5 *A Markov network (or random field) is a pair $(\mathcal{X}, \mathcal{P})$, where $\mathcal{P}$ is a collection of potentials over domains of subsets $\{X_{j_1}, \cdots, X_{j_d}\}$ of $\mathcal{X}$: $f_{X_{j_1},\cdots,X_{j_d}} : \Omega_{X_{j_1},\cdots,X_{j_d}} \to \mathcal{Q}$, with variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ over discrete domains $\{\Omega_{X_1}, \ldots, \Omega_{X_n}\}$ such that for every $\mathbf{x} \in \Omega_{\mathcal{X}}$:*

$$p(\mathbf{x}) \propto \prod_{f_{X_{j_1},\cdots,X_{j_d}} \in \mathcal{P}} f_{X_{j_1},\cdots,X_{j_d}}(x_{j_1}, \cdots, x_{j_d}),$$

where every $x_{j_i} \in \Omega_{X_{j_i}}$ is compatible with $\mathbf{x}$.

Table 1: Complexity situation of different MAP/MPE versions. tw means minimum treewidth, and z means (a bound on) the (maximum) cardinality of any variable. (E) indicates that the result holds both with and without the conditioning (see Definitions 3 and 4).

Problem	Constraint	z	Result	References
MAP(E)		2	NP$^{\text{PP}}$-complete	Park and Darwiche (2004)
MAP(E)	$tw = 1$	5	NP-complete	de Campos (2011)
MAP(E)	$tw = 1$	b	exp-APX-hard	de Campos (2011)
MAP(E)	Naive Bayes	b	NP-complete	de Campos (2011)
MAP(E)	$tw = 1$	3	NP-complete	de Campos (2013), this paper
MINAP(E)	$tw = 1$	2	NP-complete	this paper
MAP(E)	polytree $tw = 2$	2	NP-complete	Park (2002)
MAP(E)	$tw \in O(1)$	$O(1)$	FPTAS	de Campos (2011)
MPE		2	NP-complete	Shimony (1994)
MPEE		2	PP-hard, in PP$^{\text{NP}[b]}$	de Campos (2011), this paper
MPE(E)	$tw \in O(1)$	b	in P	well-known, see Darwiche (2009)
MPE		2	non approximable	Kwisthout (2015), this paper

It is straightforward to use potentials f in Definition 5 to accommodate the functions of a BN obtaining the very same joint distribution. In the special case of BN trees (at most one parent per node), this translation yields an undirected tree with pairwise potentials (involving a node and possibly its sole parent). Therefore, the hardness results for BNs often translate into hardness results for Markov networks without major efforts.

It is also worth mentioning that work has been done on multiple variations of the problems, and different complexity results for exact and approximate MPE/MAP inferences have been obtained, see for example (Mauá et al., 2015; Kwisthout, 2011a,b). It is also worth noting that MPE cannot be approximated in polynomial time by any multiplicative factor (even as function of input size). I believe this is a well-known result, see for instance the discussion by Kwisthout (2015) and also by Abdelbar and Hedetniemi (1998), but I have missed to find a proper citation for this precise result. I invite the reader to point out where this result was first published and to cite the appropriate source.

Theorem 6 *It is NP-hard to approximate MPE-z-b to any multiplicative ratio function, even if a(n exponential) function of the input size and even for $z = 2$.*

Proof It it is NP-hard to decide whether $p(\mathbf{e}) > 0$ (Cooper, 1990; Kwisthout, 2018), hence it is NP-hard to decide if exists $\mathbf{x}$ such that $p(\mathbf{x}, \mathbf{e}) > 0$. Since an answer of zero is not an approximate value for any multiplicative ratio function, the result follows. ■

Therefore, note that a common "mild" assumption that the evidence which is provided as input to the problem has positive probability changes the complexity class of MPE. Another common "mild" assumption regards Markov networks, which is often defined using an exponential function for the potentials. Again, this would imply strictly positive values everywhere, and hence MPE

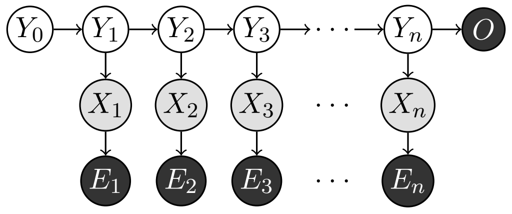

Figure 1: Tree used to prove Theorem 7.

Table 2: Probability values used in the proof of Theorem 7.

$p(Y_i \mid Y_{i-1})$	y_{i-1}^T	y_{i-1}^F	y_{i-1}^*
y_i^T	$\frac{1+t_i}{2}$	0	0
y_i^F	$\frac{1-t_i}{2}$	0	0
y_i^*	0	1	1

$p(E_i \mid X_i)$	x_i^T	x_i^F
e_i^T	t_i	1
e_i^F	$1-t_i$	0

$p(X_i \mid Y_i)$	y_i^T	y_i^F	y_i^*
x_i^T	$\frac{t_i}{1+t_i}$	1	$\frac{1}{2}$
x_i^F	$\frac{1}{1+t_i}$	0	$\frac{1}{2}$

becomes approximable too. On the other hand, MPEE is not approximable regardless of these considerations, as it is an PP-hard problem (de Campos and Cozman, 2005) (Theorem 2 there, which has a flaw that will be discussed in the continuation, but which hardness result stands).

2. Hardness of Decision-MAP-3-1

The decision version of MAP is hard even in trees with cardinality five (de Campos, 2011). The next theorem strengthens that result, and was first presented in a technical report but never published elsewhere (de Campos, 2013). Pertinence in NP is trivial and well-known.

Theorem 7 *Decision-MAP-z-w and Decision-MAPE-z-w are NP-hard even if $z = 3$, $w = 1$.*

Proof We use a reduction from the partition problem (Garey and Johnson, 1979), which is the problem of deciding whether a list of positive integer numbers $s_1, \ldots, s_n$ can be partitioned in a way such that $\sum_{i \in I} s_i = \sum_{i \notin I} s_i$, where $I \subseteq N = \{1, \ldots, n\}$ (the notation $i \notin I$ means that $i \in N \setminus I$). We say that the partition problem is a *yes-instance* if there is such a I, otherwise we call it a *no-instance*. Instead of working with the partition problem in that form, we define $S := \sum_i s_i/2$ and $v_i := s_i/S$, $i = 1, \ldots, n$, and work with the partition problem using v_i instead of s_i (note that S is an integer, otherwise it would be trivially a no-instance, and that the hardness of the problem remains the same, as there is a polynomial-time computable bijection between solutions and decisions of the two variants).

We build a tree over variables $X_1, \ldots, X_n$, $Y_0, \ldots, Y_n$, O, $E_1, \ldots, E_n$ with graph as in Figure 1. The root node is associated to the ternary variable Y_0 taking values in $\{y_0^T, y_0^F, y_0^*\}$ such that $p(y_0^T) = 1$. For variable O, we have $p(o^T|y_n^F) = p(o^T|y_n^*) = 1$ and $p(o^T|y_n^T) = 0$. The remaining variables are defined in Table 2.

Consider the computation of MAP where the variables of interest are $\mathbf{X} = \{X_1, \ldots, X_n\}$ (the gray ones in Figure 1) and evidence $(\mathbf{e}^T, o^T) = \{\forall i : E_i = e_i^T\} \cup \{O = o^T\}$ (dark nodes in the figure). It follows that

$$p(\mathbf{x}, \mathbf{e}^T, o^T) = \sum_{y_n} p(o^T|y_n) p(y_n, \mathbf{x}, \mathbf{e}^T) = p(y_n^F, \mathbf{x}, \mathbf{e}^T) + p(y_n^*, \mathbf{x}, \mathbf{e}^T) =$$

$$= p(\mathbf{x}, \mathbf{e}^T) - p(y_n^T, \mathbf{x}, \mathbf{e}^T) = p(\mathbf{e}^T|\mathbf{x})\left(p(\mathbf{x}) - p(y_n^T, \mathbf{x})\right).$$

By calculations with this specification of the network, one obtains $p(\mathbf{x}) = 2^{-n}$ (no matter the actual states of $\mathbf{x}$), $p(y_n^T, \mathbf{x}) = 2^{-n}p(\mathbf{e}^T|\mathbf{x}) = 2^{-n}\prod_{i\in I} t_i$, where $I \subseteq N$ is the set of indices of the elements such that X_i is at the state x_i^T. Denote $\mathbf{t} = \prod_{i\in I} t_i$. Then $p(\mathbf{x}, \mathbf{e}^T, o^T) = 2^{-n}\mathbf{t}(1-\mathbf{t})$. This is a concave quadratic function on $0 \le \mathbf{t} \le 1$ with maximum at 2^{-1} such that $\mathbf{t}(1-\mathbf{t})$ monotonically increases when $\mathbf{t}$ approaches one half (from both sides). If we could set t_i to be exactly 2^{-v_i}, then $\frac{1}{2^n}\mathbf{t}(1-\mathbf{t}) = \frac{1}{2^n}2^{-\sum_{i\in I} v_i}(1 - 2^{-\sum_{i\in I} v_i})$, which achieves the maximum of $\frac{1}{2^n}2^{-1}(1-2^{-1})$ if and only if $\sum_{i\in I} v_i = 1$, that is, if and only if there is an even partition.

It remains to show that we can specify values t_i using only polynomially many bits in b such that they are very close to 2^{-v_i}, and hence yes-instances of partition are separated from no-instances. For that, one just needs to follow the results in (Mauá et al., 2013; de Campos and Cozman, 2013) regarding errors introduced by using rationals in place of real numbers, or the very same approach as in Theorem 3 of (de Campos, 2011), which we copy here for completeness.

Compute each t_i to be equal to 2^{-v_i} with $4b+3$ bits of precision and by rounding it up (if necessary), that is, $t_i = 2^{-v_i} + \text{error}_i$, where $0 \le \text{error}_i < 2^{-(4b+3)}$. Clearly t_i can be computed in polynomial time and space in b (this ensures that the specification of the Bayesian network, which requires rational numbers, is polynomial in b). Note that $2^{-v_i} \le t_i \le 2^{-v_i} + \text{error}_i < 2^{-v_i} + 2^{-(4b+3)} \le 2^{-v_i + 2^{-4b}}$ (in short, this holds because 2^{-4b} in the exponent makes the value grow faster than the linear addition of $2^{-(4b+3)}$).

If I is not an even partition, then we know that one of the two conditions hold: (i) $\sum_{i\in I} s_i \le S - 1 \Rightarrow \sum_{i\in I} v_i \le 1 - \frac{1}{S}$, or (ii) $\sum_{i\in I} s_i \ge S + 1 \Rightarrow \sum_{i\in I} v_i \ge 1 + \frac{1}{S}$, because the original numbers s_i are integers. Consider these two cases.

If $\sum_{i\in I} s_i \ge S+1$, then $\mathbf{t} < \prod_{i\in I} 2^{-v_i+2^{-4b}}$ equals to

$$2^{\sum_{i\in I}(-v_i + 2^{-4b})} \le 2^{\frac{n}{2^{4b}} - (1+\frac{1}{S})} \le 2^{-1-(\frac{1}{2^b} - \frac{1}{2^{3b}})} = l,$$

by using $S \le 2^b$ and $n \le b < 2^b$. On the other hand, if $\sum_{i\in I} s_i \le S-1$, then $\mathbf{t} \ge \prod_{i\in I} 2^{-v_i}$ equals to

$$2^{-\sum_{i\in I} v_i} \ge 2^{-(1-\frac{1}{S})} = 2^{-1+\frac{1}{S}} \ge 2^{-1+\frac{1}{2^b}} = u.$$

Now suppose I' is an even partition. Then we know that the corresponding $\mathbf{t}'$ satisfies $2^{-1} \le \mathbf{t}'$ and

$$\mathbf{t}' < \prod_{i\in I'} 2^{-v_i+2^{-4b}} = 2^{\sum_{i\in I'}(-v_i+2^{-4b})} \le 2^{-1+\frac{1}{2^{3b}}} = a.$$

To complete the proof, we show that the distance between $\mathbf{t}'$ and 2^{-1} is always less than the distance between $\mathbf{t}$ and 2^{-1} of a non-even partition plus a gap, that is,

$$\begin{aligned} |\mathbf{t}' - 2^{-1}| + 2^{-(3b+2)} &\le a - 2^{-1} + 2^{-(3b+2)} \\ &< \min\{u - 2^{-1}, 2^{-1} - l\} \le |\mathbf{t} - 2^{-1}|, \end{aligned} \tag{2}$$

which can be proved by analysing the two elements of the min. The first term holds because

$$\begin{aligned} a + 2^{-(3b+2)} - 2^{-1} &< a \cdot 2^{\frac{1}{2^{2b}}} - 2^{-1} \\ &= 2^{-1+\frac{2^{-b}+2^{-2b}}{2^b}} - 2^{-1} < 2^{-1+\frac{1}{2^b}} - 2^{-1} = u - 2^{-1}. \end{aligned}$$

The second comes from the fact that the function $h(b) = a+l+2^{-(3b+2)} = 2^{-1+\frac{1}{2^{3b}}}+2^{-1-(\frac{1}{2^b}-\frac{1}{2^{3b}})}+2^{-(3b+2)}$ is less than 1 for $b = 1, 2$ (by inspection), it is a monotonic increasing function for $b \geq 2$ (the derivative is always positive), and it has $\lim_{b\to\infty} h(b) = 1$. Hence, we conclude that $h(b) < 1$, which implies

$$a + l + 2^{-(3b+2)} < 1 \iff a - 2^{-1} + 2^{-(3b+2)} < 2^{-1} - l.$$

This concludes that there is a gap of at least $2^{-(3b+2)}$ between the worst value of $\mathbf{t}'$ (relative to an even partition) and the best value of $\mathbf{t}$ (relative to a non-even partition), which will be used next to specify the threshold of the MAP problem:

$$\max_{\mathbf{x}} p(\mathbf{x}, \mathbf{e}^T, o^T) > r = c \cdot \frac{1}{2^n}, \tag{3}$$

where c is defined as $a' \cdot (1 - a')$, with a' equals a evaluated up to $3b + 2$ bits and rounded up, which implies that $2^{-1} < a \leq a' < a + 2^{-(3b+2)}$. By Eq. (2), a' is closer to one half than any $\mathbf{t}$ of a non-even partition, so the value c is certainly greater than any value that would be obtained by a non-even partition. On the other hand, a' is farther from 2^{-1} than a, so we can conclude that c separates even and non-even partitions, that is, $\mathbf{t} \cdot (1 - \mathbf{t}) < c \leq a \cdot (1 - a) < \mathbf{t}' \cdot (1 - \mathbf{t}')$ for any $\mathbf{t}$ corresponding to a non-even partition and any $\mathbf{t}'$ of an even partition. Thus, a solution of the MAP problem obtains $p(\mathbf{x}, \mathbf{e}^T, o^T) > r$ if and only there is an even partition. The conditional version Decision-MAPE-3-1 is also hard by including the term $\frac{1}{p(\mathbf{e}^T, o^T)}$ in r, which can be computed in polynomial time by a BN propagation (Pearl, 1988). ■

3. Complexity of Decision-MPEE

It is known that MPE and MPEE are both solvable in polynomial time if the treewidth of the BN is bounded by a constant using join-trees and/or variable elimination algorithms and given that knowing the treewidth bound allows one to find a good elimination order in polynomial time (Bodlaender, 1993). If treewidth is unbounded, then the situation changes: Decision-MPE-z-b is NP-complete even if $z = 2$. Previously, we have attempted to demonstrate that Decision-MPEE-2-b is PP-complete in Theorem 2 of (de Campos and Cozman, 2005). That proof is correct for hardness, but unfortunately wrong for the pertinence in PP. The result remains open, but can be further tightened.

Theorem 8 *Decision-MPEE-2-b is PP-hard and belongs to PP$^{NP[b]}$.*

Proof PP-hardness comes from Theorem 2 in (de Campos and Cozman, 2005) (ignoring the mistakenly pertinence in PP claimed there). Pertinence in PP$^{\mathrm{NP}[b]}$ is obtained by the following procedure:

1. Using an NP-complete problem (e.g. Decision-MPE-2-b), compute $v = \max_{\mathbf{x}} p(\mathbf{x}, \mathbf{e})$ exactly by discovering its bits one by one through calls to Decision-MPE-2-b (we know that v requires at most $O(b)$ since it cannot be larger than all integers, numerators and denominators, in the input multiplied, and thus all bits can be discovered by a sort of bisection approach).

2. If $v = 0$, then use the PP machine to check whether $p(\mathbf{e}) > 0$. If so, answer zero, otherwise answer as desired (based on your understanding of $0/0$).

3. Use the PP machine to check whether $p(\mathbf{e}) < v/r \iff v/p(\mathbf{e}) > r$ (note that PP is closed under complement, so the direction of the call does not matter).

Note that PP $=$ $\text{P}^{\text{PP}[\log b]}$, so it is fine to run two calls of PP without changing complexity class. ■

To the best of my knowledge, it is unknown whether $\text{PP}^{\text{NP}[b]}$=PP, though such equality would not surprise this author (neither would the inequality) (de Campos et al., 2020).

4. Decision-MINAP-2-1 is NP-complete

The minimisation version of the MAP/MPE problem has been arguably less interesting, but it is nevertheless an valid theoretical question. Moreover, as an example, it could be used to check the "support" of the probability distribution by checking if $p(\mathbf{x}) > 0$ everywhere (respectively by checking if there is $\mathbf{x}$ such that $p(\mathbf{x}) \leq 0$ (or less than $2^{-O(b)}$ to avoid using zero). I prove that such task is NP-complete even if the BN is a tree and all variables are Boolean.

Theorem 9 *Decision-MINAP-2-1 and Decision-MINAPE-2-1 are NP-Complete even in a Naive Bayes structure.*

Proof Pertinence in NP for both cases is trivial, as we can compute, for given $\mathbf{x}$, the values $p(\mathbf{x}, \mathbf{e})$ and $p(\mathbf{e})$ in linear time in BN trees. Hardness comes with a reduction from the partition problem (see beginning of proof of Theorem 7 for its definition – the same problem is employed here).

Table 3: Probability values used in the proof of Theorem 9.

$p(X_i \mid Y)$	y^T	y^F
x_i^T	$\frac{t_i}{1+t_i}$	$\frac{1}{1+t_i}$
x_i^F	$\frac{1}{1+t_i}$	$\frac{t_i}{1+t_i}$

Define a Naive Bayes structure with Boolean class variable Y and $p(y^T) = p(y^F) = 1/2$, while $X_1, \ldots, X_n$ are the Naive Bayes features with mass functions defined in Table 3. Let $t_i = 2^{-v_i}$, with $v_i = s_i/S$, where $S = \sum_i s_i/2$ and $s_1, \ldots, s_n$ are the naturals in the input of the partition problem. Those will be the MINAP(E) queried variables. Therefore, for any given $\mathbf{x}$, we have

$$p(\mathbf{x}) = \frac{1}{2\prod_i(1+t_i)} \left(\prod_{i:\, x_i^T \in \mathbf{x}} t_i + \prod_{i:\, x_i^F \in \mathbf{x}} t_i \right) . \tag{4}$$

Note that $\mathbf{x}$ defines which elements are put in each of the two terms. Moreover,

$$\mathbf{t} = \prod_{i:\, x_i^T \in \mathbf{x}} t_i + \prod_{i:\, x_i^F \in \mathbf{x}} t_i = 2^{-\sum_{i\in I} v_i} + 2^{-\sum_{i \in I^C} v_i} , \tag{5}$$

with $I \subseteq \{1, \ldots, n\}$ and $I^C = \{1, \ldots, n\} \setminus I$ defined as the indices of $\mathbf{x}$ with true and false states, respectively. Expression (5) is a convex function in the terms, so it obtains its minimum at $\sum_{i\in I} v_i = \sum_{i\in I^C} v_i = 1$ (by construction $\sum_i v_i = 2$), and in such case Expression (5) becomes 1. Therefore, a MINAP query (no evidence is required) of whether $p(\mathbf{x}) \leq \frac{1}{2\prod_i(1+t_i)}$ would solve partition. The only remaining task is to show that we can encode the probability values described in Table 3 succinctly while keeping the result correct. For that, we shall note that there is a gap between a partition solution yielding a *yes* result and the best possible *no* result. This gap will increase $\mathbf{t}$ by at least $1/S$ in the exponent of one of its two terms, that is, if I is not an even partition, because the original numbers s_i are integers, we have

$$\mathbf{t} \geq 2^{-1-1/S} + 2^{-1+1/S} > 1 + 2^{-O(b)} .$$

Therefore, it is enough to encode all values t_i with a precision of $h(b)$ bits, for some polynomial h, such that yes and no results do not get overlapped even with small "errors" introduced by the approximate encode of $t_i = 2^{-v_i}$. The gap of $2^{-O(b)}$ is enough to avoid such overlapping so we can query MINAP whether $p(\mathbf{x}) < \frac{1}{2\prod_i(1+t'_i)} + 2^{-h'(b)}$ with an appropriate polynomial h' and appropriate approximations $t'_i \approx t_i$ used to define the BN. For a more detailed discussion, one can follow the same ideas as in the proof of Theorem 7. It is also possible to arrive to this result by employing results about the approximation of BNs by similar networks with rational parameters, e.g. through a crafted use of Lemma 1 in (Mauá et al., 2014). ■

Despite numerous attempts, unfortunately I have not been able to prove the NP-hardness (or create a provably tractable algorithm) for Decision-MAP-2-1, which remains an open problem. This paper is an invitation for researchers who may have or may come up with a proof to close this problem. Because of the lack of a proof in either direction (tractability or NP-hardness), I have created a small software with the approach to solve MAP as presented in (Mauá and de Campos, 2012) in order to test the empirical characteristics of the problem.

5. An Efficient Implementation for MAP in Trees

This section builds an algorithm for MAP inferences. First, it is assumed that every node has a "list" of potentials which is initialised with the potentials that regard itself and its parent (the root node can be seen as having an imaginary parent to facilitate the description). The potentials are stored in a list, with one potential for each state of the node's variable, and each potential vector coordinate value refers to the conditional on a particular state of the parent node (so these vectors have dimension equal to the number of states of the parent node). Now, we start the algorithm by calling it for the root node of the tree: MAPinTree(root).

The algorithm MAPinTree assumes the following about the MAP problem: (1) All MAP variables (those to be explained) are in the leave nodes, while all internal nodes are to be marginalised. This is possible without loss of generality because one can, for every internal MAP node (processed from the deepest to the shallowest), recursively solve the MAP problem by calling the algorithm for all subtrees of such internal MAP node independently for each possible value of the MAP node, and then linearly combine the results (making such internal MAP node become just like a leaf node for all later purposes, as its subtrees were already resolved and their results integrated into its potentials). Such recursion is efficient because it breaks the tree into smaller independent subproblems. (2) All leaves are MAP nodes: If they are not, then they can be marginalised beforehand.

Algorithm MAPinTree(node): // Leaves are MAP nodes, others are to be marginalised

1. If node is leaf, return.

2. For each child of the node: call MAPinTree(child).

3. While node has at least 2 children:

   (a) Merge two children of node by multiplying (value-wise) every potential $(v_i)_{\forall i}$ of one child with every potential $(u_i)_{\forall i}$ of the other, obtaining a new potential $(v_i \cdot u_i)_{\forall i}$. Here i runs over the states of the current node, while the different potentials represent different states of the child. This generates $L_1 \cdot L_2$ new potentials if these children had L_1 and L_2 potentials in their lists.

   (b) Run Pareto-pruning and/or convex hull algorithm(s) on the new list of $L_1 \cdot L_2$ potentials to potentially reduce its size and store in the merged child (discard the other child).

4. (Here node has only one child.) Cross-multiply each potential $(v_i)_{\forall i}$ of the sole child by the potentials $\{p_i : (w_j)_{\forall j}\}$ of the current node, obtaining the new potential $(\sum_i v_i \cdot p_{i,j})_{\forall j}$. Here i runs over states of the current node, while j runs over states of the parent of the current node. The new potentials become the list of potentials of the current node. If we are at the root node, the vectors become one-dimensional (as if there was a single option j).

5. Run Pareto-pruning and/or convex hull algorithm(s) on the new list of potentials of the current node and return. (At the root node, this will become a single number.)

It is easy to keep track of the configurations so to return the MAP configuration in the end by labelling potentials with states of the MAP variables as potentials are created. The correctness of this algorithm is discussed in previous work (Mauá and de Campos, 2011; Maua et al., 2013; Mauá et al., 2012; Mauá and de Campos, 2012), though in a more elaborate presentation because the work there was not restricted to trees. The simplicity of this algorithm in the case of trees allows one to implement the Pareto pruning in $O(L^2)$ regardless of the size of the state space of the variables in the problem, where L is the number of potentials in the list, while the convex hull can be done in $O(L \cdot \log L)$ in the Boolean case, but grows quickly as we move to larger state spaces of the variables (this is why it is particularly efficient in Boolean trees).

The algorithm trivially becomes a Fully Polynomial-Time Approximation Scheme (FPTAS) if at each iteration we only keep one potential for each cell in a grid in the log-space translation of the potentials (which are simply vectors with dimension equal to the number of states of the parent node). Such a grid in a log-space gives us a multiplicative approximation. More details can be found in (de Campos, 2011; Mauá et al., 2011). Under a clever implementation, this can be achieved, for example, by limiting the precision of the numbers in some of the computations. In order to understand better Decision-MAP-2-1 for which no complexity proof exists, I have run some simple experiments with the algorithm and report the results here. The code works with unnormalised Boolean tree BNs (Markov networks), is implemented in an efficient manner using the C programming language and GNU multiple precision libraries. I have used three types of input during the experiments: random structures with random parameters, Naive Bayes structure with random parameters, and Naive Bayes structure with parameters mapping the partition problem (with random positive integer input values). Results are presented in Table 4. All experiments used 256 bits of precision for mantissa (around five times a standard double precision number) and runs were allowed 8GB of memory. Results involving the partition problem indicate the difficulty of solving large instances. Random results suggest that MINAP complexity exponentially grows with the size of the input, while MAP stays stable. This may suggest that the MAP problem in Boolean trees is tractable, or is simply a reflection of domain sizes and limited numeric precision.

Table 4: MAPinTree results on Boolean trees. Max. value regards the integer numbers in the input (fractions are numerator / denominator; for partition, they regard the integers of the input before translation to a BN). Memory is the max. number of potentials used. Median and maximum are over 20 random runs. (*These cases have always run out of 8GB memory).

				Median		Maximum	
Problem	Description	Max Value	Nodes	Memory	Time(s)	Memory	Time(s)
MINAP	naive Bayes w/	10^6	50	59	0.06	84	0.08
	random params		100	125	0.198	200	0.285
			200	396	1.328	1238	1.893
			300	1103	2.793	20863	9.893
MAP	naive Bayes w/	10^6	50	5	0.01	7	0.015
	random params		100	5	0.017	6	0.023
			200	5	0.04	7	0.047
			300	5	0.043	7	0.049
MINAP	partition	10^4	10	512	0.034	512	0.039
	problem		20	91857	11.553	100842	17.42
			30	236979	77.09	264638	82.81
		10^5	10	512	0.036	512	0.045
			20	347065	27.599	372670	31.10
			30	2046264	532.318	2237859	586.4
		10^6	10	512	0.035	512	0.038
			20	501347	34.672	510413	38.13
			30*	> 10Mln	> 600	> 10Mln	> 600
MINAP	random struct.	10^6	50	57	0.046	101	0.073
	and parameters		100	143	0.21	197	0.326
			200	417	1.288	713	1.761
			300	1129	2.509	10403	14.53
MAP	random struct.	10^6	50	5	0.009	7	0.014
	and parameters		100	5	0.018	6	0.023
			200	5	0.042	7	0.047
			300	6	0.049	7	0.061

6. Conclusion

This paper discusses on the current complexity results for marginal/partial MAP and full MAP (also called MPE) queries in Bayesian networks (which naturally extend to Markov networks too) and strengthens the theoretical results for them. A known non-approximability result is discussed for MPE, and a previously unpublished result shows that the decision version of MAP is NP-hard in ternary BN trees (and consequently in ternary Markov networks). A correction about a previous pertinence-in-PP proof of the conditional version of MPE (here called MPEE) is discussed and a corrected version yields pertinence in $\text{PP}^{\text{NP}[b]}$, that is, PP with access to an NP-complete problem oracle limited to b calls overall (b means the size of the input). The author is not aware of whether such complexity class equals PP or not. A new proof for the minimisation counterpart of partial

MAP (called MINAP) shows that such query is NP-complete even in Boolean BN trees. An efficient algorithm for MAP in trees is described, which is a specialisation of previous results that yield a fully polynomial-time approximation scheme. The novelty is to exploit this specialised version to implement a highly efficient code for trees with binary variables. Experiments suggest that the time complexity of MINAP grows exponentially in the input size, while MAP's time and memory usages remain considerably flat. At a first glance, this might suggest that the MAP problem is tractable, even though it may simply be a reflection of experimental design and amount of experiments. The question will not be answered without a formal proof in either way, and this remains open.

References

A. M. Abdelbar and S. M. Hedetniemi. Approximating MAPs for belief networks is NP-hard and other theorems. *Artificial Intelligence*, 102(1):21–38, 1998.

H. L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. In *ACM symposium on Theory of computing*, pages 226–234, New York, NY, USA, 1993. ACM.

G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.

A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge Univ. Press, 2009.

C. P. de Campos. New complexity results for MAP in Bayesian networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2100–2106. AAAI Press, 2011.

C. P. de Campos. NP-hardness of MAP in ternary tree Bayesian networks. Technical report, IDSIA, 2013. IDSIA-06-13.

C. P. de Campos and F. G. Cozman. The inferential complexity of Bayesian and credal networks. In *International Joint Conference on Artificial Intelligence*, pages 1313–1318, 2005.

C. P. de Campos and F. G. Cozman. Complexity of inferences in polytree-shaped semi-qualitative probabilistic networks. In *Conference on Artificial Intelligence (AAAI)*, pages 217–223, 2013.

C. P. de Campos, G. Stamoulis, and D. Weyland. A structured view on weighted counting with relations to counting, quantum computation and applications. *Information and Computation*, page 104627, 2020. in press.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

J. Kwisthout. The computational complexity of probabilistic inference. Technical report, Radboud U. Nijmegen, 2011a. ICIS–R11003.

J. Kwisthout. Most probable explanations in Bayesian networks: Complexity and tractability. *International Journal of Approximate Reasoning*, 52(9):1452–1469, 2011b.

J. Kwisthout. Tree-width and the computational complexity of MAP approximations in Bayesian networks. *Journal of Artificial Intelligence Research*, 53:699–720, 2015.

J. Kwisthout. Approximate inference in bayesian networks: Parameterized complexity results. *International Journal of Approximate Reasoning*, 93:119–131, 2018.

J. Kwisthout, H. L. Bodlaender, and L. C. van der Gaag. The Necessity of Bounded Treewidth for Efficient Inference in Bayesian Networks. In *European Conference on Artificial Intelligence (ECAI)*, volume 215, pages 237–242, 2010.

D. D. Mauá and C. P. de Campos. Solving decision problems with limited information. In *Advances in Neural Information Processing Systems 24*, pages 603–611, 2011.

D. D. Mauá and C. P. de Campos. Anytime marginal MAP inference. In *International Conference on Machine Learning (ICML)*, pages 1471–1478, New York, NY, USA, 2012. Omnipress.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. A fully polynomial time approximation scheme for updating credal networks of bounded treewidth and number of variable states. In *International Symposium on Imprecise Probability: Theories and Applications*, pages 277–286. SIPTA, 2011.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. Solving limited memory influence diagrams. *Journal of Artificial Intelligence Research*, 44:97–140, 2012.

D. D. Maua, C. P. de Campos, A. Benavoli, and A. Antonucci. On the complexity of strong and epistemic credal networks. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 391–400, 2013.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. On the complexity of solving polytree-shaped limited memory influence diagrams with binary variables. *Artificial Intelligence*, 205:30–38, 2013.

D. D. Mauá, C. P. de Campos, A. Benavoli, and A. Antonucci. Probabilistic inference in credal networks: New complexity results. *Journal of Artificial Intelligence Research*, 50:603–637, 2014.

D. D. Mauá, C. P. de Campos, and F. G. Cozman. The Complexity of MAP Inference in Bayesian Networks Specified Through Logical Languages. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 889–895, 2015.

J. D. Park. MAP complexity results and approximation methods. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 388–396. Morgan Kaufmann, 2002.

J. D. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research*, 21:101–133, 2004.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, California, 1988.

S. E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, 1994.