Dempster's Rule of Combination is #P-complete*

Pekka Orponen
Department of Computer Science, University of Helsinki
Teollisuuskatu 23, SF–00510 Helsinki, Finland

Abstract

We consider the complexity of combining bodies of evidence according to the rules of the Dempster–Shafer theory of evidence. We prove that, given as input a set of tables representing basic probability assignments $m_1, \ldots, m_n$ over a frame of discernment Θ, and a set $A \subseteq \Theta$, the problem of computing the combined basic probability value $(m_1 \oplus \ldots \oplus m_n)(A)$ is #P-complete. As a corollary, we obtain that while the simple belief, plausibility, and commonality values $Bel(A)$, $Pl(A)$, and $Q(A)$ can be computed in polynomial time, the problems of computing the combinations $(Bel_1 \oplus \ldots \oplus Bel_n)(A)$, $(Pl_1 \oplus \ldots \oplus Pl_n)(A)$, and $(Q_1 \oplus \ldots \oplus Q_n)(A)$ are #P-complete.

1 Introduction

The Dempster–Shafer theory of evidence [8] has recently been attracting increasing attention as a theoretically well-founded way of dealing with the problem of uncertain information in artificial intelligence systems (cf. [2, 5, 6, 11]). The apparently prohibitive computational complexity of the method has, however, so far rendered it of only limited practical use. The problem of complexity was pointed out already by J. Barnett in his first paper introducing the Dempster–Shafer theory to the wider AI community [1], and since then research has centered on finding efficient implementations of the method in certain restricted situations [1, 4, 9, 10].

The main tool provided by the theory is *Dempster's rule of combination*, which is a formula for combining evidential information provided by different sources. It is generally taken for granted that the complexity of applying this formula grows exponentially in the number of evidential sources (e.g., [4] p.

*Work supported by the Academy of Finland. This research was carried out while the author was visiting the Department of Computer Science, University of Toronto.

324; [9] p. 271), and Barnett is sometimes credited as having proved this fact. However, in a footnote to his paper, Barnett explicitly states: "I have not proved this. However, if the formulae [...] are directly implemented, then the statement stands." ([1], p. 871). Actually, as we shall see, the claim of the exponential complexity of Dempster's rule is slightly inaccurate. We show that the function computed by the rule is *#P-complete* [12, 13][3, pp. 168–169], thus verifying its exponential (or, rather, superpolynomial) complexity — but only assuming the truth of the unproved, though eminently plausible hypothesis that $P \neq \#P$.

The complexity of Dempster–Shafer computations has recently also been studied by Provan, although from a point of view somewhat different from ours [7].

2 Basic Notions

Dempster–Shafer theory concerns itself with the unknown value of some quantity θ, constrained to lie within a *frame of discernment*, or *universe* Θ. A source of evidence concerning the true value of θ is represented as a *basic probability assignment (b.p.a.)* over Θ. This is simply a mapping m associating to each subset A of Θ a real number (or, for computability reasons, a rational) $m(A)$ that represents the strength of evidence in favor of the proposition $\theta \in A$. It should be emphasized that $m(A)$ represents primary evidence *focused* on the set A: the evidence for a set A is not considered to be part of the evidence supporting the proper supersets of A. A b.p.a. $m : \mathcal{P}(\Theta) \rightarrow \mathcal{Q}$ is constrained to satisfy the following axioms:

1. $m(A) \geq 0$ for every $A \subseteq \Theta$;
2. $m(\emptyset) = 0$;
3. $\sum_{A \subseteq \Theta} m(A) = 1$.

A number of functions can be defined for summarizing the primary evidence represented by a basic probability assignment, the most important ones being the *belief*, *plausibility*, and *commonality* functions. The belief value of a set $A \subseteq \Theta$, $\mathrm{Bel}(A)$, represents the total weight of evidence supporting A, and is defined as

$$\mathrm{Bel}(A) = \sum_{B \subseteq A} m(B).$$

The plausibility value of A, $\mathrm{Pl}(A)$, is the total weight of evidence not in contradiction with A, defined as

$$\mathrm{Pl}(A) = \sum_{B \cap A \neq \emptyset} m(B).$$

The commonality value of A, $Q(A)$, represents the weight of evidence equally in support of all the elements of A, i.e., the evidence focused on supersets of A:

$$Q(A) = \sum_{B \supseteq A} m(B).$$

For lack of a better term, we shall refer to the functions Bel, Pl, and Q collectively as *summary* functions.

The information provided by two evidential sources, represented as basic probability assignments m_1 and m_2 over a common universe Θ, may be combined by means of *Dempster's rule of combination* to a joint b.p.a., denoted by $m_1 \oplus m_2$, over the same universe. The basic idea is to assign to a set $A \subseteq \Theta$ the combined weight

$$\sum_{A_1 \cap A_2 = A} m_1(A_1) m_2(A_2).$$

It is easy to verify that this simple scheme indeed yields a b.p.a., except that it may assign a nonzero weight to the empty set. Therefore, the weight of the empty set is explicitly set to zero, and the rest of the weights are normalized by a factor of K^{-1}, where

$$K = \sum_{A_1 \cap A_2 \neq \emptyset} m_1(A_1) m_2(A_2).$$

This scheme is readily generalized to an arbitrary number of b.p.a.'s: given a sequence of b.p.a.'s $m_1, m_2, \ldots, m_n$, their combination $m = \bigoplus_{i=1}^{n} m_i$ is defined as

$$\begin{aligned} m(\emptyset) &= 0; \\ m(A) &= K^{-1} \sum_{\bigcap_i A_i = A} m_1(A_1) m_2(A_2) \cdots m_n(A_n) \end{aligned}$$

for $A \neq \emptyset$, where

$$K = \sum_{\bigcap_i A_i \neq \emptyset} m_1(A_1) m_2(A_2) \cdots m_n(A_n).$$

Dempster's rule is extended to the summary functions in the obvious way; for instance, the combination of two belief functions Bel_1 and Bel_2, determined by b.p.a.'s m_1 and m_2 is defined as

$$(\mathrm{Bel}_1 \oplus \mathrm{Bel}_2)(A) = \sum_{B \subseteq A} (m_1 \oplus m_2)(B).$$

We shall relate Dempster's rule to the class $\#P^1$ of functions [12, 13][3, pp. 168–169]. A function f mapping strings over some alphabet Σ to integers belongs to this class if there is a nondeterministic polynomial time Turing machine

[1] Read "number-P".

M that on each input $x \in \Sigma^*$ has exactly $f(x)$ accepting computation paths; in this case we say that M *counts* f. The class $\#P$ is a functional analogue of the better known class NP of decision problems: to each NP decision problem there corresponds in a natural way a $\#P$ counting function (i.e., the function counting "witnesses" or accepting computations), and vice versa. A function f is *$\#P$-complete* if it belongs to $\#P$, and any other function in $\#P$ can be computed by some deterministic polynomial time Turing machine that is allowed to access values of f at unit cost. The generic $\#P$-complete function is

$\#\mathrm{SAT}(F)$ = the number of satisfying truth assignments to Boolean formula F.

Here the formulas F may be restricted to be in conjunctive normal form. It is easy to see that if $P \neq NP$, then $\#P$-complete functions cannot be computed in polynomial time.

One minor technical issue in proving results about the Dempster–Shafer functions and the class $\#P$ is that the former map into the rationals, not integers. To overcome this problem, let us represent rationals as pairs of integers and fix some suitable one-to-one encoding of such pairs into integers, say

$$\mathrm{code}(p, q) = (p + q)(p + q + 1) + 2q.$$

Then we can say that a function $f : \Sigma^* \to \mathcal{Q}$ is in $\#P$ if there is a function $f^c : \Sigma^* \to \mathcal{N}$ in $\#P$ such that for all x, $f^c(x) = \mathrm{code}(p, q)$, and $f(x) = p/q$.

Lemma 2.1 *Let $f, g : \Sigma^* \to \mathcal{N}$ be two functions in $\#P$. Then also the function $(f/g) : \Sigma^* \to \mathcal{Q}$, where $(f/g)(x) = f(x)/g(x)$ for each $x \in \Sigma^*$, is in $\#P$.*

Proof. By the above encoding, it is sufficient to prove that the class of $\#P$-functions with range $\mathcal{N}$ is closed under addition and multiplication. Let f and g be two such functions, and let M_f and M_g be two nondeterministic Turing machines counting f and g, respectively. The following Turing machine M' then counts $(f + g)$: on input x, M' first makes a nodeterministic move to determine whether to count f or g; in the former case it simulates M_f on x; in the latter case it simulates M_g. Similarly, (fg) can be counted by the following machine M'': on input x, M'' first simulates M_f on input x; if M_f accepts, then M'' simulates M_g on x; otherwise it rejects. □

3 The Complexity of Dempster's Rule

In order to analyze the complexity of the summary functions $\mathrm{Bel}(A)$, $\mathrm{Pl}(A)$, and $Q(A)$, we must decide what is to be counted into the size of the input, viz. how is the underlying basic probability assignment m represented. There are two possibilities: either the input contains the set A for which the value of a summary function is required, *together* with a table listing the non-null values of $m(B)$, or m is presented as a functional oracle, and only A is counted into the

input. The former model, which we consider to be the natural one, and hence use below, corresponds to a situation where an evidential source provides us in advance with a set of at most polynomially many (in the size of the universe Θ) possibilities B such that all B with $m(B) \neq 0$ are included. The other model would correspond to a situation where either the values $m(B)$ come from some database of astronomical size, or the evidential source can be employed to compute all the values $m(B)$ as required, without our knowing in advance a polynomial-size set of possibilities covering all the non-null entries.

If a basic probability assignment m is presented explicitly as a table of the non-null entries, as suggested, then all the simple summary functions $\mathrm{Bel}(A)$, $\mathrm{Pl}(A)$, $Q(A)$ relative to m can easily be computed in polynomial, in fact linear time. The computational complexity gets out of hand only when Dempster's rule is applied.

Theorem 3.1 *Given as input a sequence of b.p.a.'s $m_1, m_2, \ldots, m_n$ over some universe Θ, and a set $A \subseteq \Theta$, it is a #P-complete problem to compute the value $m(A) = (\bigoplus_{i=1}^{n} m_i)(A)$.*

Proof. Let us prove first that the problem is in $\#P$. Recall that

$$m(A) = K^{-1} \sum_{\bigcap_i A_i = A} m_1(A_1) m_2(A_2) \cdots m_n(A_n),$$

where

$$K = \sum_{\bigcap_i A_i \neq \emptyset} m_1(A_1) m_2(A_2) \cdots m_n(A_n).$$

A nondeterministic Turing machine that counts $m(A)$, i.e., that on input $\langle m_1, \ldots, m_n; A \rangle$ has exactly $\mathrm{code}(p, q)$ accepting paths, where $p/q = m(A)$, can be obtained as follows. For each i and B such that $m_i(B) \neq 0$, let $m_i(B)$ be presented in the input as $p_i(B)/q_i(B)$, where $p_i(B)$ and $q_i(B)$ are integers. The machine first deterministically computes, for each i, the least common multiple M_i of the nonzero denominators $q_i(B)$, $B \subseteq \Theta$. Then it replaces the tables for the functions m_i by tables for functions m_i', where $m_i'(B) = M_i m_i(B)$. Now each of the tables m_i' has only integer entries, and

$$\begin{aligned} m(A) &= \frac{\sum_{\bigcap_i A_i = A} m_1(A_1) \cdots m_n(A_n)}{\sum_{\bigcap_i A_i \neq \emptyset} m_1(A_1) \cdots m_n(A_n)} \\ &= \frac{(M_1 \cdots M_n)^{-1} \sum_{\bigcap_i A_i = A} m_1'(A_1) \cdots m_n'(A_n)}{(M_1 \cdots M_n)^{-1} \sum_{\bigcap_i A_i \neq \emptyset} m_1'(A_1) \cdots m_n'(A_n)} \\ &= S'/K', \end{aligned}$$

where

$$\begin{aligned} S' &= \sum_{\bigcap_i A_i = A} m'_1(A_1) \cdots m'_n(A_n), \\ K' &= \sum_{\bigcap_i A_i \neq \emptyset} m'_1(A_1) \cdots m'_n(A_n). \end{aligned}$$

By Lemma 2.1 it is now sufficient to show that both the values S' and K' can be obtained by a $\#P$-computation. To generate exactly S' (resp. K') accepting computations, our machine can guess a sequence of sets $A_1, \ldots, A_n \subseteq \Theta$ and check whether $\bigcap_i A_i = A$ (resp. $\bigcap_i A_i \neq \emptyset$). If this is the case, then the machine branches, by making more nondeterministic moves, into exactly $m'_1(A_1) \cdots m'_n(A_n)$ accepting computations; otherwise it ends this branch in a rejecting state. The total number of accepting computations thus obtained from all the parallel branches is clearly S' (resp. K').

We now proceed to prove that computing $m(A)$ is $\#P$-hard. This we do by showing that the problem of computing #SAT(F) for an arbitrary conjunctive normal form Boolean formula F can be reduced to the problem of computing $m(A)$ for a suitably chosen input $\langle m_1, \ldots, m_n; A \rangle$. Thus, let $F = C_1 \wedge \ldots \wedge C_k$ be a c.n.f. Boolean formula over the variables $x_1, \ldots, x_n$; we may assume that for every variable x_i, some clause C_j contains either one of the literals x_i or $\overline{x}_i$, but no clause contains both. The corresponding sequence of basic probability assignments $m_1, \ldots, m_n$ is constructed as follows. We choose as our universe the set $\Theta = \{1, \ldots, k, *\}$ and consider, for each $i = 1, \ldots, n$, the sets T_i, F_i:

$$\begin{aligned} T_i &= \{*\} \cup \{\, j \,|\text{clause } C_j \text{ does not contain literal } x_i\} \\ F_i &= \{*\} \cup \{\, j \,|\text{clause } C_j \text{ does not contain literal } \overline{x}_i\}. \end{aligned}$$

Because of our assumptions on the literals, the sets T_i and F_i are not equal, and we may define a b.p.a. m_i as:

$$\begin{aligned} &m_i(T_i) = m_i(F_i) = 1/2; \\ &m_i(B) = 0, \text{ for } B \neq T_i, F_i. \end{aligned}$$

We now claim that there is a one-to-one, onto correspondence between the truth assignments

$$h : \{x_1, \ldots, x_n\} \rightarrow \{t, f\}$$

that make formula F true, and sequences of sets $A_1, \ldots, A_n \subseteq \Theta$ such that $\bigcap_i A_i = \{*\}$ and $m_1(A_1) \cdots m_n(A_n) \neq 0$.

Given a truth assignment h, let us define for $i = 1, \ldots, n$:

$$A_i = \begin{cases} T_i & \text{if } h(x_i) = t \\ F_i & \text{if } h(x_i) = f. \end{cases}$$

First, it is clear that because $T_i \neq F_i$ for every i, this mapping is one-to-one everywhere on its domain (which includes satisfying as well as nonsatisfying truth assignments). To see that truth assignments satisfying F map to set sequences with the requisite properties, consider the sequence (A_i) associated with a satisfying assignment h. Denoting $A = \bigcap_i A_i$, it is clear that '$*$' $\in A$, and that $m_1(A_1) \cdots m_n(A_n) = 2^{-n} \neq 0$. To show that in fact '$*$' is the only element in A, observe that since h is a satisfying truth assignment, it is the case for any index j that either there is a literal x_i in C_j such that $h(x_i) = t$, in which case $A_i = T_i$ and $j \notin A_i \supseteq A$; or there is a literal $\overline{x}_i$ in C_j such that $h(x_i) = f$, in which case $A_i = F_i$ and again $j \notin A_i \supseteq A$.

To show that the mapping is onto, let $A_1, \ldots, A_n$ be a sequence of sets such that $A = \bigcap_i A_i = \{*\}$ and $m_1(A_1) \cdots m_n(A_n) \neq 0$. Since $m_i(B) = 0$ for all $B \neq T_i, F_i$, in fact $A_i \in \{T_i, F_i\}$ for each $i = 1, \ldots, n$, and we can define a truth assignment h as follows:

$$h(x_i) = \begin{cases} t & \text{if } A_i = T_i \\ f & \text{if } A_i = F_i. \end{cases}$$

It is clear that h maps to the sequence $A_1, \ldots, A_n$ under the above correspondence. To verify that h satisfies F, assume to the contrary that there is a clause C_j in F that is not true under h. Then it is the case for each variable x_i that if $h(x_i) = t$, then C_j does not contain x_i, and so $j \in T_i = A_i$; similarly, if $h(x_i) = f$, then C_j does not contain $\overline{x}_i$, and so $j \in F_i = A_i$. It follows that $j \in A = \bigcap_i A_i$, and $A \neq \{*\}$.

Having thus established the desired correspondence, we observe further that for any sequence of sets $A_1, \ldots, A_n$, if $m_1(A_1) \cdots m_n(A_n) \neq 0$, then $m_1(A_1) \cdots m_n(A_n) = 2^{-n}$. Also, there are no sequences such that $\bigcap_i A_i = \emptyset$ and $m_1(A_1) \cdots m_n(A_n) \neq 0$, so the normalization constant K is in this case equal to 1. It follows that if we could compute the value $m(\{*\}) = (\bigoplus_i m_i)(\{*\})$, we could also easily obtain the value $\#\mathrm{SAT}(F)$, because

$$\begin{aligned} m(\{*\}) &= K^{-1} \sum_{\bigcap_i A_i = \{*\}} m_1(A_1) \cdots m_n(A_n) \\ &= (\#\text{ satisfying assignments for } F) \cdot 2^{-n} \end{aligned}$$

□

Corollary 3.2 *Given as input a sequence of b.p.a.'s $m_1, m_2, \ldots, m_n$ over some universe Θ, and a set $A \subseteq \Theta$, the problems of computing the values $(\bigoplus_{i=1}^n Bel_i)(A)$, $(\bigoplus_{i=1}^n Pl_i)(A)$, and $(\bigoplus_{i=1}^n Q_i)(A)$ are $\#P$-complete.*

Proof. Let us first verify that the quantities can be computed in $\#P$. On input $\langle m_1, m_2, \ldots, m_n; A\rangle$, $A \neq \emptyset$, a nondeterministic Turing machine counting $(\bigoplus_i \mathrm{Bel}_i)(A)$ (resp., $(\bigoplus_i \mathrm{Pl}_i)(A)$, $(\bigoplus_i Q_i)(A)$) operates as follows: it first

guesses an arbitrary set B and checks that $B \subseteq A$ (resp., $B \cap A \neq \emptyset$, $B \supseteq A$); if this is the case, it proceeds to count the value $(\bigoplus_i m_i)(B)$, otherwise it rejects.

The $\#P$-hardness of the first two problems follows immediately from the proof of Theorem 3.1, because

$$(\bigoplus_i \mathrm{Bel}_i)(\{*\}) = m(\{*\})$$

and

$$(\bigoplus_i \mathrm{Pl}_i)(\{1, \ldots, k\}) = 1 - (\bigoplus_i \mathrm{Bel}_i)(\{*\}) = 1 - m(\{*\}).$$

Superficially, computing $Q(A) = (\bigoplus_i Q_i)(A)$ might seem an easier task, because it can be shown that $Q(A) = K^{-1} Q_1(A) Q_2(A) \cdots Q_n(A)$ [8, p. 61], and each of the values $Q_1(A), \ldots, Q_n(A)$ can be computed in polynomial time. Thus computing $Q(A)$ is only of the same complexity as computing the normalization constant K. But in fact a small modification to the proof of Theorem 3.1 shows that also this problem is $\#P$-complete. If we just remove the element '$*$' from the construction in the proof throughout, then the resulting proof establishes that

$$\begin{aligned} &\textstyle\sum_{\bigcap_i A_i = \emptyset} m_1(A_1) \cdots m_n(A_n) \\ &= 1 - K \\ &= \#\mathrm{SAT}(F) \cdot 2^{-n}. \end{aligned}$$

Thus, given an oracle for computing $Q(A)$, we could easily compute the quantities $K = Q_1(A) \cdots Q_n(A)/Q(A)$ and $\#\mathrm{SAT}(F) = 2^n(1 - K)$ in polynomial time. □

Acknowledgment

I wish to thank Ms. Irene Jäämaa for constructive discussions on the topic of Dempster–Shafer theory.

References

[1] Barnett, J. A., Computational methods for a mathematical theory of evidence, in: *Proceedings, 7th Int. Joint Conf. Artificial Intelligence*, Vancouver, BC (1981) 868–875.

[2] Bhatnagar, R. K. and Kanal, L. N., Handling uncertain information: A review of numeric and non-numeric methods, in: Kanal, L. N. and Lemmer, J. F. (Eds.): *Uncertainty in Artificial Intelligence* (Elsevier – North-Holland, Amsterdam, 1986) 3–26.

[3] Garey, M. R. and Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-completeness* (W. H. Freeman & Co., New York, NY, 1979).

[4] Gordon, J. and Shortliffe, E. H., A method for managing evidential reasoning in a hierarchical hypothesis space, *Artificial Intelligence* **26** (1985) 323–357.

[5] Gordon, J. and Shortliffe, E. H., The Dempster–Shafer theory of evidence, in: Buchanan, B.G. and Shortliffe, E. H. (Eds.): *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project* (Addison-Wesley, Reading, MA, 1985) 272–292.

[6] Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Mateo, CA, 1988).

[7] Provan, G. M., A logic-based analysis of Dempster Shafer theory, *International Journal of Approximate Reasoning* (to appear).

[8] Shafer, G., *A Mathematical Theory of Evidence* (Princeton University Press, Princeton, NJ, 1976).

[9] Shafer, G. and Logan, R., Implementing Dempster's rule for hierarchical evidence, *Artificial Intelligence* **33** (1987) 271–298.

[10] Shenoy, P. P. and Shafer, G., Propagating belief functions with local computations, *IEEE Expert*, **1**(3) (1986) 43–52.

[11] Stephanou, H. E. and Sage, A. P., Perspectives on imperfect information processing, *IEEE Trans. Syst., Man, Cybern.* **SMC-17** (1987) 780–798.

[12] Valiant, L. G., The complexity of computing the permanent, *Theoret. Comput. Sci.* **8** (1979) 189–201.

[13] Valiant, L. G., The complexity of enumeration and reliability problems, *SIAM J. Comput.* **8** (1979) 410–421.