효율성 제고를 위한 근사적 증거병합 방법

이 계 성[†]

요 약

Dempster-Shafer 증거병합 방법의 가장 큰 문제는 계산 복잡도가 지수적인 증가를 갖는다는 점이다.이는 가설 집단을 이루는 원소의 개수가 각 가설을 이루는 속성 값들의 모든 부분 집합으로 focal 요소로 구성되기 때문이다. 이 문제를 피하기 위해 본 논문에서는 근사적 증거 병합 방법을 제안한다. 이 방법은 간단한 응용에 적용하여 그 성능을 조사하고 다른 증거 병합 방법의 하나인 VBS의 결과와 비교해 본다. 근사적 증거 병합방법은 계산 속도를 크게 개선하였고 전문가가 허용하는 편차 수준에서 신뢰 값을 갖는 것으로 평가되었다.

An Approximate Evidence Combination Scheme for Increased Efficiency

Gyesung Lee[†]

ABSTRACT

A major impediment in using the Dempster-Shafer evidence combination scheme is its computational complexity, which in general is exponential since DS scheme allows any subsets over the frame of discernment as focal elements. To avoid this problem, we propose a method called approximate evidence combination scheme. This scheme is applied to a few sample applications and the experiment results are compared with those of VBS. The results show that the approximation scheme achieves a great amount of computational speedup and produces belief values within the range of deviation that the expert allows.

키워드 : evidence combination, uncertaint, Dempster-Shafer scheme

1. Introduction

A major impediment in using the Dempster-Shafer (DS) evidence combination scheme is its computational complexity, which in general is exponential since DS scheme allows any subsets over the frame of discernment as focal elements[2]. For example, if the size of the frame of discernment is p, then the maximum possible number of focal elements is 2^p. Therefore, when two belief functions, each with one variable (attribute), are combined, the total number of possible set intersections can be 2^{2p}. If each variable can have multiple values, then the complexity order for evidence combination of n variables becomes 2^r, where r is $a_1 \cdot a_2 \cdots a_n$ and a_i represents the number of possible values for a variable v_i [3].

To avoid possible exponential explosion of the focal elements, various methods have been suggested. Some methodologies restrict the problem to singleton hypotheses where the belief functions allow only singleton hypothesis. Other methodologies allow multiple hypotheses but those must map into a node in the hierarchy [2].

Another simple solution was partitioning technique, where the overall inferencing structure is divided into partitions, these partitions form a their own problem subspace, and thus reduce the problem size. This technique is useful especially when the inferencing network is very large and complex, as can happen in real-world applications. However, this technique introduced other disadvantages, e.g., duplicate and therefore repetitive evidence seek that makes the overall reasoning longer.

Zarley [4] and Shenoy [2] have developed efficient computational algorithms to work on general network structures. The main idea of these methods is to reduce the size of frame of discernment from the entire set to a subset that is relevant among variables (joint variable) and then to propagate intermediate results to neighboring joint variables. This method

※ 본 연구는 단국대학교 대학연구비의 지원으로 연구되었음.
† 종신회원 : 단국대학교 전자계산학과 교수
논문접수 : 2001년 7월 12일, 심사완료 : 2002년 1월 3일

is called local computation and propagation.

We focus our efficiency issue on an inferencing structure in which the knowledge is represented by a network or hierarchical model or both. The hierarchical network is comprised of nodes representing primary attributes or characteristics. These primary characteristics are related each other, i.e., a characteristic is derived by other supporting characteristics. The links in the knowledge structure represent the direction in which the node is derived. On the other hand, the methods developed by [2] and [4] use more general form of knowledge structure. In their structure, they allow bi-directional links between two related nodes. When new evidence is observed for a node, the belief in the node itself is first updated and the change is propagated to others adjacent nodes. In some applications, the two relations, 'support' and 'supported by' are defined. In this paper, a few methods are compared on how belief is computed, updated, and propagated in the structure where hypothesis nodes are hierarchically related. We review in Section 2 two efficient methodologies for evidence combination schemes, Zarley's DELIEF and Shenoy's VBS (valuation-based system), working on network structures. Our focus of this paper is the development of an efficient computational algorithm. In section 3, we introduce our approximation scheme that greatly simplifies the evidence combination and thus is able to produce a great deal of efficiency gain. We then compare the performance of the scheme with that of two other methodologies. In addition to the approximation scheme we developed an belief combination scheme that works efficiently in hierarchical structure. This is discussed in Section 4. Finally, Section 5 concludes the paper.

2. DELIEF and VBS

Zarley's DELIEF transforms a general network model into a Markov tree for local computations and belief propagation. Nodes in the network represent singleton or set hypotheses, and links represent relations between hypotheses. For example, if presence of A ⇒ presence of B, the network model established a pair of links between nodes A and (A, B), and B and (A, B), where (A, B) represents a joint node of A and B. The basic idea of using Markov tree is that size of the nodes in the tree is balanced and that the size of joint variables at nodes of the tree is smaller than the size of the overall frame. Therefore, complexity of the computations can be reduced to 2^k, where k is size of the maximum node (in terms of the number of variables) in the tree.

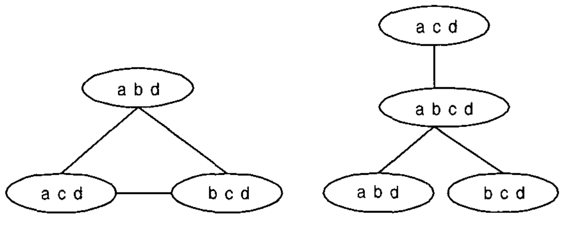
Neighboring nodes in Markov network are repetitively merged to create a Markov tree. The resultant nodes contain more components than the original nodes but the tree conversion algorithm is made to keep the size of merged node as small as possible by choosing two nodes which share maximum number of components. The order of the computational complexity is determined by the size of the maximum node. Therefore, assuming 'True' and 'False' for each variable, belief computation in Markov tree can reduce the computational complexity from 2^n to 2^k, where n is the size of entire frame, k is the maximum size of node in Markov tree, and k is usually less than or equal to n. The saving factor is defined as the ratio of computational complexities. The saving factor in this case is $2^n/2^k = 2^{(n-k)}$.

The advantage of this scheme is validated when the saving factor is large enough to compensate the overhead resulting from the conversion process and local propagation. When the degree of linkage among nodes in the network is very high, then the Markov tree conversion may not reduce the size of maximum node sufficiently to achieve significant saving in the computation.

As a simple example, a frame of discernment is defined by four variables, a, b, c, and d, and rules expressing their relationships are given in (Figure 1)(a) and the corresponding Markov network and Markov tree are illustrated in (Figure 1)(b) and (c), respectively. In this case the maximum node size is the same as the frame of discernment, therefore, there is no computational gain on a Markov tree. Overhead in the tree generation and putting together local computations even makes this scheme computationally more expensive than a straight DS computation.

if a & b then d
if a & c then d
if a & c then d

(a) rules



(b) Markov network (c) Markov tree

(Fig. 1) A simple Markov tree generation

Shenoy's valuation-based system (VBS) develops a more efficient mechanism for computation. Like DELIEF [4], he achieves a great amount of computational saving by utilizing the local computations and propagation over the general inferencing network. Furthermore, his system made unnecessary the conversion to other form of structures, e.g., Markov tree. Two major operations, combination and marginalization, are performed for computing the belief values for each variable in the inferencing network. Combination is used for evaluating belief functions for joint variables, and marginalization is used for coarsening the belief functions to a focused subset of variables by deleting irrelevant variables. These two operations are components of a fusion algorithm for belief function computation. The basic idea of this algorithm is to successively delete variables until a final goal variable is left. When a variable is deleted, relevant belief functions are combined by using DS computation over the variables which are union of joint variables and then the generated belief function is coarsened with respect to target variable(s), where target variables are those that are affected by combined belief functions, except for the deleted variable. Though VBS can always produce efficient performance, the exponential growth of computations over the frame of discernment cannot be avoided in the worst case.

3. Approximation method

Two general methodologies discussed above still face the exponential computational complexity even though the overall performance is greatly enhanced. In an inferencing structure, supporting and concluding nodes which are directly linked form a frame of discernment and thus the size of frame of discernment is reduced to the one of locally relevant attributes. However, if the size of the inferencing structure is large and complex, the local computation could be a bottleneck in overall performance. The approximation scheme separates the LHS (evidence) and RHS (conclusions) of rules for belief function computation. Instead of applying DS computation scheme to the hypothesis space consisting of set of evidences and conclusions that make up a rule (as would be done in [2] and [4]), we create the belief function for a rule in such a way that belief values of the conclusion hypotheses are computed by multiplying minimum basic probability assignment (bpa) of the LHS evidences with the belief function associated with the rule.

We performed an experiment with approximation scheme using a set of test problems, and compared the result with that of Shenoy's VBS. We were able to show that the computational gain using approximation is up to a hundred times over VBS. We implemented two programs in Lisp, one for our approximation scheme and the other for VBS. We made up a variety of test cases that cover a number of possible situations. The efficiency is determined in terms of computation speed using different test cases and the computation speed is obtained by measuring the time that the entire computation takes. We used Lisp built-in time checking function for measuring the computing time.

The experimental cases and results are summarized in <Table 1> and <Table 2>, respectively. Case1 and Case2 contain 4 and 5 variables, respectively, and Case3 and Case4 contain 8 variables. Links in Case3 are much more populated than in Case4. From <Table 2>, we verify that the gain of the approximate scheme increases exponentially as the number of variables increases and that there is no difference with respect to gain between densely linked network (Case3) and sparse network (Case4). In cases where there are a number of attributes and rules, applying regular DS scheme based on Shenoy's VBS to the entire rule base still requires a tremendous computational cost.

We obtained a great amount of computational saving from approximate scheme but in exchange of a loss of accuracy. We tested two schemes with randomly generated belief values of evidences (conditions) and belief values of conclusions, and performed a total of 300 different belief combinations for each test problem. <Table 3> summarizes

〈Table 1〉 Four sample problems

Case 1	Case 2
if a & b then h with b1 if a & c then h with b2 if b & c then h with b3	if a,b & c then h with b1 if b & d then h with b2 if a,c and d then h with b3
case 3	Case 4
if a,b,c,d & e then h with b1 if b,c,d,e & f then h with b2 if c,d,e,f,& f then h with b3 if a,b,d,e & f then h with b4	if a,b & d then h with b1 if b,c & e then h with b2 if c,f & g hen h with b3 if a & f then h with b4

〈Table 2〉 Efficiency Comparison

	VBS	Approx.	Gain
Case1 (4 vars)	1.26 sec	.14 sec	9.0
Case2 (5 vars)	2.43 sec	.14 sec	17.4
Case3 (8 vars)	17.0 sec	.17 sec	100.0
Case4 (8 vars)	15.5 sec	.16 sec	96.9

the occurrences of differences of two results, each from two schemes. In most cases the values of approximation scheme are greater than those of VBS scheme. This shows that when the number of variables is small then the difference in belief value between two schemes is insignificant. But as the size of the problem increases the difference becomes significant. We also notice that the difference distribution spreads out over a higher band of difference values. In Case1, 92% of belief combinations show that the difference is less than .1 and 81% of cases are in range of .05-.14 in Case2. 75% of cases fall into a wider range of .30-.49 and the average difference is .392. The average difference implies that the approximate scheme generates belief values .392 higher than the regular DS scheme.

From the experiment, we observed of VBS that belief values of goal variables after belief combination drastically drop even though belief values of the conditions (variables) are relatively high. This creates a big concern in some application domain, where most of evidences don't have high degree of belief and thus applying regular DS based on VBS to this domain may result in unacceptably low belief values for the conclusions. From the experience we can conclude that the exact belief values of the hypotheses are not important. Rather the ranks of those are more meaningful in analyzing the conclusions.

〈Table 3〉 Difference distribution

	Case1	Case2	Case3	Case4
.0 -.09	277	188	1	6
.10 - .19	23	112	2	135
.20 - .29	0	0	36	138
.30 - .39	0	0	112	21
.40 - .49	0	0	113	0
.50 - .74	0	0	36	0
.75 - .99	0	0	0	0

Even in the approximation scheme, a large number of rule firing and a large size of frame of discernment of goal attributes still need a large amount of computational resources for DS computations. In the following section we describe an efficient DS belief computation algorithm used in conjunction with approximation scheme.

4. An efficient DS belief combination algorithm

In our revised evidence combination scheme, we generalize the singleton hypothesis assumption by allowing multiple hypotheses which, unlike Gordon's or Shenoy's definition, could be any subset of hypotheses. In [1], the hypothesis set was constrained to any subset of hypotheses under a parent hypothesis in the hierarchy. Note that set of hypotheses for evidence combination cannot come from the different parent groups or different levels. The new scheme is to keep the number of focal elements small by collecting relevant focal elements between two belief functions, where relevant focal elements are those that share common hypothesis in their focal elements. From now on one of two belief functions is called 'existing belief function' and the other 'new belief function'.

The number of focal elements in the existing belief function is determined by the contents of the new belief function and domain knowledge structure. In our scheme, any subset of child hypotheses under a parent node can be asserted from a rule. Subset nodes are not represented in the hierarchy, but whenever a basic probability assignment (bpa) is assigned to it, it is stored in its parent node.

Let Bel_1 be the belief function that represents the existing belief function, and Bel_2 is a new belief function that represents the belief values from the rule to be fired. m_1 and m_2 represent their corresponding basic probability assignments. The goal of the DS evidence combination scheme defines a new bpa, denoted $m_1 \oplus m_2$ and their corresponding belief function, denoted $Bel_1 \oplus Bel_2$.

Definition : If two focal elements each from Bel_1 and Bel_2 satisfy either subset or superset relation, then they are called *inclusive*, otherwise, they are called *exclusive*.

From the regular DS scheme we recognized a fact that, after evidence combination, the new bpa's and subsequent belief functions for exclusive focal elements can be computed by simply multiplying a certain constant factor. To derive this constant factor, we bring up a few notational definitions.

Without losing generality, we can define Bel_2 with two focal elements : n_k and n_l (One can easily extend to any number of elements.). Focal elements in Bel_1, which is accumulated up to now are divided into two parts : (i) Part A : inclusive focal element (IFE) group and (ii) Part B : exclusive focal element (EFE) group as shown in <Table 4>.

For Part A, the regular DS scheme is applied to get a new bpa. Let EFE's in Part B define $\bar{a} = \{ b_1, b_2, b_3, \cdots b_z\}$, where b_i can be a singleton hypothesis or any subset of hypotheses under the same parent and at the same level. Since they are

〈Table 4〉 Efficiency Comparison

	Part A	Part B	
	IFE's of n_k and n_l	EFEs of n_k and n_l	θ_1
		$b_1, b_2, b_3, \cdots$	
n_k n_l	Regular DS	ϕ	n_k n_l
θ_2	IFE's of n_k and n_l	$b_1, b_2, b_3, \cdots$	$\theta_{\neg}$

all exclusive, their set intersection with n_k and n_l results in ϕ. Core is defined as the union of all the focal elements of a belief function. In formalizing the algorithm, we try to follow the terminologies that were introduced in [5]. Let

Core of Bel_1 = $\{a_1, a_2, \cdots a_x, b_1, b_2, \cdots b_z\}$
Core of Bel_2 = $\{ n_k, n_l \}$
where $b_i \cap n_p$ = for $1 < i < z$ and p = k, l. a_i's and b_j's are IFE's and EFE's of Bel_1, respectively.

Here we are going to show an algorithm by which we can compute $m(b_k)$ without calculating the individual set intersection and cross productions among components in . In part A of the table, the regular DS combination scheme is performed. Let's take a look at Part B.

$$m(b_k) = m_1(b_k) \cdot m_2(\theta_2) / K$$

where $1 < k < z$, $K = 1 - \Sigma_{a \cap b = \varphi}\, m_1(a) \cdot m_2(b)$, and $m(b_k)$ is the resultant probability mass, and

$$m(\bar{a}) = m(b1) + m(b2) + \cdots$$
$$m_1(\bar{a}) = m_1(b_1) + m_1(b_{12}) + \cdots$$

$$m(\bar{a})_{diff} = m(\bar{a}) - m_1(\bar{a})$$
$$= \sum_{k=1,z} \{m(b_k) - m_1(b_k)\}$$
$$= \sum_{k=1,z} \{m_1(b_k) * m(\theta_2)/m_1(b_k)\}$$
$$= m(\theta_2)/K-1) \sum_{k=1,z} m_1(b_k)$$

Thus,

$$\frac{m(\theta_2)}{K} = 1 + m(\bar{a})_{diff} / \Sigma_{k=1,z}\, m_1(b_k)$$
$$= 1 + m(\bar{a})_{diff} / m_1(\bar{a})$$
$$= m(\bar{a}) / m_1(\bar{a})$$

This is defined as an irrelevancy factor, IRF. Therefore, new basic probability value for b_k for $1 < k < z$ is :

$$m(b_k) = m_1(b_k) \cdot IRF$$

Note that IRF contains only basic probability numbers of $\bar{a}$.

We showed that some computational gain can be achieved by treating EFE's as a focal element. We can easily collect the IFE's and EFE's from the given hierarchy. If a node for a focal element of Bel_2 is located in the hierarchy, its supersets (to the root) and its subset (to the leaves) can easily be picked up. The frame of discernment then consists of these IFE's and $\bar{a}$ for the rest.

5. Conclusion

This paper has discussed an efficient scheme of belief combination. This includes an approximation method over the inferencing structure and an efficient belief combination algorithm to work on a hypothesis structure. We have demonstrated a great amount of efficiency gain from approximation method. This was possible because the approximation method reduced the number of evidences involved in evidence combination by selecting evidence with minimum belief. A disadvantage of this scheme is that the minimum belief evidence overrides other evidences with higher belief and their effects are ignored. To avoid this, other alternative is suggested to average belief values of evidences or multiply them. These alternatives are being investigated now.

Additional efficiency gain in our belief combination algorithm is achieved when the size of frame of discernment is large and two belief functions to be combined include less common hypotheses. This algorithm has been useful in an application, Playmaker [1], where the size of frame of discernment is large, e.g., a hypothesis space (17 hypotheses for a 'facies' attribute), and a set of evidence indicate at most 3 or 4 hypotheses.

Even though a number of experiments have been performed on various sets of sample cases, it is required to establish formal analysis on the performance of the scheme in terms of computational complexity other than experimental results. This issue needs to be further studied in the future.

References

[1] Lee, G, "Rule models for the integrated design of knowledge acquisition, reasoning, and knowledge refinement," The transactions of the Korea Information Processing Society, Vol.3, No.7, pp.908-914, 1996.
[2] Shenoy, P. P., "Valuation network representation and solution of asymmetric decision problems," European journal of operation research, Vol.121, No.3, 2000.

[3] Wang, S. and Valtorata, M., "On the exponential growth rateof Dempster-Shafer belief functions," Applications of AI X, pp.15-24, 1994.

[4] Zarley, D., "An evidential reasoning system," Master thesis, Dept. of Computer Science, The univ. of Kansas, 1988.

[5] Shafer, G., "A mathematical theory of evidence," Princeton University Press, 1976.

[6] Gordon, J. and Shortliffe, E. H., "A method for managing evidential reasoning in a hierarchical hypothesis space," AI 26, pp.323-357, 1985.

이 계 성

e-mail : gslee@anseo.dankook.ac.kr

1980년 서강대학교 전자공학과(학사)

1982년 한국과학기술원 전자계산학과(석사)

1994년 미국 Vanderbilt Univ. 전자계산학과 (박사)

1982년~1985년 경제기획원 전산처리관

1994년~1996년 대구대학교 전산정보학과 전임강사

1996년~현재 단국대학교 전자계산학과 부교수

관심분야 : 기계학습, Data mining, Intelligent Agents