Analysis and Methodologies of Synthesis of Solutions in Distributed Expert Systems

Minjie Zhang and Chengqi Zhang
Department of Mathematics, Statistics and Computing Science
University of New England, Armidale, N.S.W. 2351, Australia
{minjie,chengqi }@neumann.une.edu.au

Abstract

In this paper, potential synthesis cases in distributed expert systems (DESs) are identified. Based on these results, necessary conditions of synthesis strategies in different synthesis cases and two methodologies to solve the problem of synthesis of solutions in DESs are proposed. A computational strategy by using analysis methods and a neural network strategy which is an example of inductive methods are introduced in detail for solving synthesis problems in conflict synthesis cases. Both methodologies are evaluated and compared.

Introduction

A distributed expert system (DES) is one of the special configuration of distributed problem solving. It consists of different expert systems (ESs) which are connected by computer networks. In a DES, each expert system (ES) can work individually for solving some specific problems, and can also cooperate with other ESs when dealing with complex problems.

Due to limitation of knowledge, the problem solving ability of single ESs, and the uncertain features of some problems, some tasks (or subtasks) may be solved by more than one ES in order to increase the reliability of the solution. A typical example is when "several doctors diagnose the same patient".

If the same task is allocated to more than one ES, each ES will obtain a solution. For example, two ESs predict an earthquake in a particular area. ES_1 believes that the possibility of the potential earthquake being class 5 is $x_1 = 0.8$, while ES_2 believes that the possibility of the potential earthquake being class 5 is $x_2 = 0.5$. The problem is how to obtain the final uncertainty if more than one uncertainty for the same solution exists. The synthesis strategies are responsible for synthesizing the uncertainties of the solution from different ESs to produce the final uncertainty of the solution.

Consider the following two cases based on the above example.

Case (1): two ESs obtain the uncertainties of the solution (a class 5 earthquake in a particular area) $x_1 = 0.8$ and $x_2 = 0.5$, respectively, based on the same geochemical results. This case demonstrates a belief conflict between ES_1 and ES_2 because they obtained the same solution with different uncertainties given the same evidence. The final uncertainty $S(x_1, x_2)$ should be between x_1 and x_2 (i.e., $\min\{x_1, x_2\} \leq S(x_1, x_2) \leq \max\{x_1, x_2\}$).

Case (2): ES_1 predicts a class 5 earthquake in an area with uncertainty of $y_1 = 0.8$ based on the evidence from a geophysical experiment and ES_2 obtains the same solution with uncertainty of $y_2 = 0.5$ based on the geological structure of this area. The final uncertainty for the solution of a class 5 earthquake in this case should be bigger than any one of y_1 and y_2 (i.e., $S(y_1, y_2) \geq max\{y_1, y_2\}$) because two ESs obtain the same solution from different evidence, so this solution is more reliable than the solution which comes from the same evidence.

The conclusion is, that if two ESs obtain the same solution with two identical uncertainties, such as $x_1 = y_1$, $x_2 = y_2$ (using the above example), the synthesis of x_1 and x_2 represented by $S(x_1, x_2)$ may be different from $S(y_1, y_2)$ if these solutions originate from different evidence.

The above two cases indicate that a right synthesis strategy is not only based on uncertainties, but also based on the relationship between evidence of solutions. If an improper synthesis strategy is chosen in a situation, a wrong solution may result. Therefore, how to classify, design and choose synthesis strategies is one of the critical research issues in a DES field.

Previous strategies for synthesis of solutions in DESs have only considered the solutions, without considering the evidence which result in the solutions. Let us briefly review these strategies. The uncertainty management strategy proposed by Khan (Khan & Jain 1985) in 1985 can solve belief conflicts. A synthesis strategy for a heterogeneous DES introduced by Zhang (Zhang 1992) in 1992 can cope with the synthesis problem for a heterogeneous DES which may use different inexact reasoning models in different ESs. An improved synthesis strategy proposed by Liu (Liu et al. 1992) in 1992 considers both uncertainties of propositions and authorities of ESs. The key limitations of these synthesis strategies are:

(1) The proposers of these strategies did not take into consideration of the evidence which results in the solutions; and

(2)Theses strategies were developed by only mathematical models. They did not work well if the relationship between multiple solutions from different ESs and the desired final solution after synthesis cannot be summarized by a mathematical method.

The purpose of this research is to overcome the above limitations. In this paper, the potential synthesis cases in DESs will be identified thoroughly, two possible methodologies of synthesis of solutions in DESs will be proposed, and examples of two methodologies will be introduced and evaluated.

This paper is organized as follows. In Section 2, the analysis of synthesis problem is described in detail which includes the identification of potential synthesis cases in DESs, and discussion of necessary conditions of choosing or developing synthesis strategies in DESs. In Section 3, the principles of two methodologies for synthesis of solutions are proposed, two different strategies which are developed by two different methodologies are introduced, and compared. Finally, in Section 4, this paper is concluded and further work is outlined.

Analysis of synthesis problem

In this section, the synthesis problem will be described first, then the number of synthesis cases will be identified in DESs, and the necessary conditions of choosing or developing synthesis strategies in DESs will be discussed.

Description of synthesis problem

Let's see an example first. Suppose there are three ESs (e.g. ES_1, ES_2, ES_3) to decide the identity of the organism for a specific patient. ES_1 says that it is pseudomonas with uncertainty 0.36 and proteus with uncertainty -0.9, ES_2 says that it is pseudomonas with uncertainty 0.5 and serratia with uncertainty 0.4, and ES_3 says that it is serratia with uncertainty 0.1 and proteus with uncertainty 0.85. Because ES_1 doesn't mention serratia, we believe that ES_1 has no idea about it. We can represent this unknown by using uncertainty 0 in the EMYCIN model (Melle 1980). Then the above solutions are represented in Table 1.

	Pseudomonas	Serratia	Proteus	Authority
ES_1	0.36	0	-0.9	0.9
ES_2	0.5	0.4	0	0.8
ES_3	0	0.1	0.85	0.9

Table 1: The uncertainties for each attribute value obtained by the ESs.

The purpose of the synthesis of solutions here is to decide the final uncertainty distribution among pseudomonas, serratia, and proteus according to Table 1.

We now formally describe the problems. Suppose there are n ESs in a DES to evaluate the values of an attribute of an object (e.g. in a medical DES, the identity of an organism infecting a specific patient). The solution for an ES_i can be represented as

$$(< object >< attribute > (V_1\ CF_{i1}\ A_i)\ (V_2\ CF_{i2}\ A_i)\ \ldots\ (V_m\ CF_{im}\ A_i)) \qquad 2.1$$

where V_j $(1 \leq j \leq m)$ represents jth possible value, CF_{ij} $(1 \leq i \leq n,\ 1 \leq j \leq m)$ represents the uncertainty for jth value from ES_i, A_i represents the authority of ES_i, and m indicates that there are m possible values for this attribute of the object. For example, there are 6 possible values for the face-up of a die.

From the synthesis point of view, all ESs are concerned with the same attribute of an object. So we will only keep the attribute values, uncertainties, and authorities in the representation. Here is the representation of m possible values with uncertainties from n ESs:

$$\begin{bmatrix} CF_{11}\ CF_{12} \ldots CF_{1m}\ A_1 \\ CF_{21}\ CF_{22} \ldots CF_{2m}\ A_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ CF_{n1}\ CF_{n2} \ldots CF_{nm}\ A_n \end{bmatrix} \qquad 2.2$$

The synthesis strategy is responsible for obtaining final uncertainties ($CF_{*1}\ CF_{*2}\ \ldots\ CF_{*m}$) based on Matrix 2.2 where * indicates the synthesis result from corresponding values with the subscriptions of $1, 2, \ldots, n$ in the same place.

The authority A_i $(1 \leq i \leq n)$ is the confidence level for the solution from ES_i. The value range of the authority is [0, 1]. The higher the authority, the more reliable the solution. It can be assigned for each ES from human experts or generated based on the historical performance of ESs.

Potential Synthesis Cases in DESs

From Section 1, we know, for the same uncertainties, the synthesis results can be different. That is, when we develop synthesis strategies, the first problem is to identify the different synthesis cases in DESs.

In this subsection, we will analyze the synthesis cases in DESs based on the relationship between evidence sets of a solution from different ESs. Informally, we know that there are four relationships between evidence sets of a solution. That is, the evidence sets of a solution from different ESs are (a) identical, (b) inclusion, (c) overlap, and (d) disjoint.

Before we formally define synthesis cases, some preparation work should be done. We use propositions to represent evidence and use conclusions to represent solutions. In order to simplify the explanation, we will fristly work on the assumption that there are two ESs in a DES; then we will extend to any number of ESs in DESs.

Let P be a set of propositions, R be a set of rules, and CF represent the uncertainty of a proposition in an ES.

Definition 1: An *inference network* G in an ES is defined as a directed acyclic graph in which the *nodes* are propositions in P, and the *arcs* are activated rules in R. (Suppose the rule format be $A \longrightarrow B$ in this definition). The *root* of such a network is a proposition in P which is not the premise of any rule in R. In contrast, a *leaf* is a proposition in P which is not the conclusion of any rule in R.

Definition 2: A *rule chain* is defined as any chain from one node A to another node B in an inference network G (1) if there exists a rule α in which A is a premise of the rule α and B is a conclusion of the rule α; or (2) if there exist a sequence of rules in which A is a premise of the rule α and there exists a rule chain from a node in the conclusion of the rule α to node B.

Definition 3: A *general rule chain* is defined as a rule chain from a leaf to the root of an inference network G (Zhang 1992).

Definition 4: The *original evidence set* of a proposition B is represented by $E(B)$, where $E(B)$ is a unique set of leaf propositions which satisfy the condition that there is a rule chain to connect such a leaf to the proposition B.

For example, if there are inference networks G_i in ES_i, where G_i is:
$D \longrightarrow C \longrightarrow A \longrightarrow B$, $F \longrightarrow E \longrightarrow B$, $T \longrightarrow A \longrightarrow C \longrightarrow B$, then $E_i(B) = \{D, F, T\}$.

The next four definitions are our formal definitions of different synthesis cases. In the following definitions, only the evidence in original evidence sets are considered because they are objective.

Definition 5: A *conflict synthesis case* occurs when the original evidence sets of a proposition from different ESs are equivalent, but different ESs produce the same solution with different uncertainties. That is, for a proposition B, if there exist $E_i(B) = E_j(B)$, where $E_i(B)$ is in ES_i, $E_j(B)$ is in ES_j, and $CF_i \neq CF_j$, where CF_i is the uncertainty of the proposition B from ES_i, and CF_j is the uncertainty of the proposition B from ES_j.

For example, if there are inference networks G_i in ES_i and G_j in ES_j, where G_i: $D \longrightarrow C \longrightarrow A \longrightarrow B$, $F \longrightarrow H \longrightarrow B$ and G_j: $D \longrightarrow H \longrightarrow B$, $F \longrightarrow J \longrightarrow B$, then $E_i(B) = \{D, F\}$ and $E_j(B) = \{D, F\}$ are equivalent.

Definition 6: An *inclusion synthesis case* occurs when the original evidence set of a proposition from one ES is a subset of an original evidence set of another ES. Formally, for a proposition B, $E_{B_i} \subset E_{B_j}$, or vice versa, where E_{B_i} is in ES_i, and E_{B_j} is in ES_j.

For example, for a proposition B, there are inference networks G_i in ES_i and G_j in ES_j, where G_i : $A \longrightarrow C \longrightarrow B$, $D \longrightarrow B$, and G_j : $A \longrightarrow K \longrightarrow B$, $D \longrightarrow G \longrightarrow B$, $C \longrightarrow B$, $E \longrightarrow B$. In this example, $E_i(B) = \{A, D\}$, $E_j(B) = \{A, D, C, E\}$, so $E_i(B) \subset E_j(B)$. Note: for ES_j, C is an original evidence while for ES_i, C is a derived evidence, so that C is not in the $E_i(B)$ but in $E_j(B)$.

Definition 7: A *overlap synthesis case* occurs when the original evidence sets of a proposition from different ESs are not equivalent, but the intersection of original evidence sets is not empty. Formally, for a proposition B, $E_i(B) \cap E_j(B) \neq \phi$, $E_i(B) \cap E_j(B) \neq E_i(B)$, and $E_i(B) \cap E_j(B) \neq E_j(B)$ where $E_i(B)$ is in ES_i, and $E_j(B)$ is in ES_j.

For example, for a proposition B, there are inference networks G_i in ES_i, and G_j in ES_j, where $G_i : A \longrightarrow C \longrightarrow B$, $D \longrightarrow B$ and G_j: $A \longrightarrow E \longrightarrow F \longrightarrow B$, $H \longrightarrow G \longrightarrow B$. In this example, $E_i(B) = \{A, D\}$, $E_j(B) = \{A, H\}$, so $E_i(B) \cap E_j(B) = \{A\} \neq \phi$, $E_i(B) \cap E_j(B) \neq E_i(B)$, and $E_i(B) \cap E_j(B) \neq E_j(B)$.

Definition 8: A *disjoint synthesis case* occurs when the intersection of original evidence sets of a proposition from different ESs is empty. Formally, for a proposition B, $E_i(B) \cap E_j(B) = \phi$, where $E_i(B)$ is in ES_i, and $E_j(B)$ is in ES_j.

For instance, for a proposition B, there are inference networks G_i in ES_i and G_j in ES_j where G_i: $A \longrightarrow C \longrightarrow B$, $X \longrightarrow H \longrightarrow B$ and $G_j : F \longrightarrow D \longrightarrow B$. In this example, $E_i(B) = \{A, X\}$, $E_j(B) = \{F\}$, so $E_i(B) \cap E_j(B) = \phi$.

From the above analysis, we know that there are four synthesis cases in DESs based on the relationships between original evidence sets of a solution from different ESs. There are only four synthesis cases because only four relationships of original evidence sets exist in DESs, which are (i) conflict, (ii) inclusion, (iii) overlap, and (iv) disjoint.

The analysis of the above four synthesis cases is based on only two ESs. Now we extend above definitions to n ESs.

Definition 9: A *conflict synthesis case* occurs among n ESs when the original evidence sets of a proposition from n ESs are equivalent, but different ESs produce the same solution with different uncertainties. Formally, for a proposition B, if there exist $E_1(B) = E_2(B) = ... = E_n(B)$, where $E_1(B)$ is in ES_1, $E_2(B)$ is in ES_2, ..., and $E_n(B)$ is in ES_n, and $\exists i, j$, $(1 \leq i, j \leq n, i \neq j)$, $CF_i \neq CF_j$, where CF_i is the uncertainty of the proposition B from ES_i and CF_j is the uncertainty of the proposition B from ES_j.

Definition 10: An *inclusion synthesis case* among n ESs occurs if these exists an original evidence set of a proposition from ES_i ($E_i(B)$) in which $E_i(B)$ strictly includes all other $E_j(B)$. Formally, for a proposition B, $\forall j$ $(1 \le j \le n, j \ne i)$, $E_i(B) \supset E_j(B)$ where $E_1(B)$ is in ES_1, $E_2(B)$ is in ES_2, ..., and $E_n(B)$ is in ES_n.

Definition 11: A *disjoint synthesis case* occurs among n ESs when the intersection of any two original evidence sets of a proposition from n ESs is empty. Formally, for a proposition B, $\forall i, j$, $(1 \le i, j \le n, i \ne j)$, $E_i(B) \cap E_j(B) = \phi$, where $E_1(B)$ is in ES_1, $E_2(B)$ is in ES_2, ..., and $E_n(B)$ is in ES_n.

If the number of ESs in a DES is more than 2, firstly we identify whether the synthesis case belongs to a conflict, inclusion, or disjoint case. If the synthesis case doesn't satisfy the conditions of the above three cases, it is an overlap case.

Necessary Conditions of Synthesis Strategies in DESs

Thus far, literature concerning DESs has not demonstrated any necessary condition of synthesis strategies for each of the synthesis cases. It is very important to find the necessary conditions of synthesis strategies because they can be used to eliminate inappropriate use of synthesis strategies. For example, conflict synthesis strategies are inappropriate to be used in the inclusion, overlap, or disjoint synthesis cases.

In this subsection, we will describe both general necessary conditions of all synthesis strategies and specific necessary conditions for synthesis strategies in each synthesis case. The general conditions are on a more abstract level, while specific conditions are on a more concrete level.

General necessary conditions for all synthesis strategies Let S represent a synthesis strategy of CF_i and CF_j $(S(CF_i, CF_j))$ where CF_i and CF_j represent the uncertainties of a proposition B from ES_i and ES_j, respectively. S could be any of the synthesis functions appropriate to the different situations of conflict, inclusion, overlap, or disjoint.

The following properties are the general necessary conditions which an acceptable synthesis strategy in DESs should satisfy.

(a) Suppose that X is the set of uncertainties of propositions in an inexact reasoning model. If $\forall i, j$, $CF_i \in X$ & $CF_j \in X$, then $S(CF_i, CF_j) \in X$. The reason for this property is that the value of uncertainty after synthesis should be still in the same uncertainty range, otherwise the synthesis result will be meaningless.

(b) The synthesis function S on X must satisfy the associative law. The reason for this property is that in the real world, the final solution of the problem is only based on the evidence which is used to obtain the solution, not on the order of evidence. That is, $S(S(CF_i, CF_j), CF_k) = S(CF_i, S(CF_j, CF_k))$.

(c) The synthesis function S on X must satisfy the commutative law. The reason for this property is the same as (b).

The general necessary conditions are valid only when the synthesis strategies synthesize different uncertainties by accumulative manners (i.e., synthesize two uncertainties at once).

Specific necessary conditions for synthesis strategies in each synthesis case In the *conflict synthesis case*, the necessary condition for the synthesis function should be $min\{CF_i, CF_j\} \le S_{conflict}(CF_i, CF_j) \le max\{CF_i, CF_j\}$, where $S_{conflict}$ is a synthesis function for a conflict synthesis case, because both uncertainties of CF_i and CF_j come from the same original evidence set $(E_i(B) = E_j(B))$. This condition is nothing related to any inexact reasoning model even we use *min* and *max*. In other words, the difference between CF_i and CF_j only comes from the different subjective interpretation of different ESs on the same objective evidence. Since there is no additional evidence for each of the ESs, the opinion from all of them should be considered, and they constrain each other.

In the *disjoint synthesis case*, there is no overlap between E_{B_i} and E_{B_j}. E_{B_i} or E_{B_j} can contribute positively or negatively to the uncertainty of proposition B being true independently. Therefore, if both ESs favor the proposition B being true ($CF_i > 0$ and $CF_j > 0$ under the EMYCIN (Melle 1980) inexact reasoning model), the necessary condition for the synthesis function should be $S_{disjoint}(CF_i, CF_j) > \max\{CF_i, CF_j\}$; where $S_{disjoint}$ represents a synthesis function for the disjoint synthesis case. If both ESs are against the proposition B being true ($CF_i < 0$ and $CF_j < 0$), then $S_{disjoint}(CF_i, CF_j) < \min\{CF_i, CF_j\}$. In all other cases, it should be $\min\{CF_i, CF_j\} \le S_{disjoint}(CFi, CF_j) \le \max\{CF_i, CF_j\}$.

In the *inclusion synthesis case*, if $E_i(B)$ is the subset of $E_j(B)$, the necessary condition of inclusion synthesis case should be $S_{inclusion}(CF_i, CF_j) = CF_j$ where $S_{inclusion}$ is the synthesis function for the inclusion case. The idea behind this is that ES_i gets the solution based on less evidence than ES_j. Evidences used by ES_i are already used by ES_j, so ES_i makes no more contribution to the final solution.

In the *overlap synthesis case*, there is some additional evidence between E_{B_i} and E_{B_j}. The necessary condition for this kind of case should be $S_{overlap}(CF_i, CF_j) \ge S_{conflict}(CF_i, CF_j)$ (if $CF_i \ge 0, CF_j \ge 0$); or $S_{overlap}(CF_i, CF_j) < S_{conflict}(CF_i, CF_j)$ (if $CF_i < 0, CF_i < 0$) where $S_{overlap}$ is the synthesis function for the overlap synthesis case.

Both the conflict synthesis case and the disjoint synthesis case are extreme cases. Both the inclusion synthesis case, and the overlap synthesis case lie between these two extreme cases.

The relationships of the necessary conditions for the synthesis strategies among four synthesis cases should be as follows:

$S_{conflict}(CF_i, CF_j) \leq S_{inclusion}(CF_i, CF_j) \leq S_{disjoint}(CF_i, CF_j)$;

$S_{conflict}(CF_i, CF_j) \leq S_{overlap}(CF_i, CF_j) \leq S_{disjoint}(CF_i, CF_j)$ if $CF_i \geq 0$ and $CF_j \geq 0$

$S_{conflict}(CF_i, CF_j) \geq S_{inclusion}(CF_i, CF_j) \geq S_{disjoint}(CF_i, CF_j)$;

$S_{conflict}(CF_i, CF_j) \geq S_{overlap}(CF_i, CF_j) \geq S_{disjoint}(CF_i, CF_j)$ if $CF_i \leq 0$ and $CF_j \leq 0$

Methodologies of solution synthesis

Description of solution synthesis

If we use X_i to represent a matrix $(n*(m+1))$ of multiple solutions from different ESs (refer to Matrix 2.2), Y_i to represent a vector (m) of the desired final solutions after synthesis of X_i, and f to represent a perfect synthesis function (the relationship between all X_i and Y_i), we can use the following symbols to describe synthesis of solutions in general cases.

$f(X_i) = Y_i \quad$ for $i = 1, 2, 3, ...$

As we described above, we always know all X_i. Questions here are (1) how to define Y_i for any X_i and (2) for how many X_i we know corresponding Y_i. For the first question, it is reasonable to define Y_i as human experts' solution for any X_i. For example, experts in grant agency (e.g. Australian Research Council) normally distribute any grant proposal to 3-5 domain experts to review the proposal, then they make a final decision (score) of the proposal based on the scores and comments from different domain experts. In this example, X_i is the scores of a proposal from different domain experts and Y_i is the final score of the proposal from experts in grant agency. For the second question, if for any X_i, we know Y_i, we have nothing to do about synthesis of solutions. If we don't know Y_i for any X_i, quality of a synthesis function would be difficult to check because we don't know it is good or not. In fact, only for limited X_i, we know corresponding Y_i and f is defined as a psuedo function to map these limited X_i to corresponding Y_i perfectly.

The goal of synthesis of solutions is to find a mapping function f' in which $f'(X_i) = Y_i'$ should be very closed to $f(X_i) = Y_i$ for all X_i.

We now define $\delta(X_i)$ as

$$\delta(X_i) = f(X_i) - f'(X_i) = Y_i - Y_i' = \{\delta_i^1, \delta_i^2, ..., \delta_i^m\}$$

which can be used to measure the difference between the desired final solution Y_i (expert solution) and the real solution Y_i' from the function. m is the number of possible values for an attribute.

If we use $\bar{\delta}_i$ to represent the mean error and δ_i^{max} to represent the maximum error of outputs from f', $\bar{\delta}_i$ and δ_i^{max} are defined as follows, respectively.

$$\bar{\delta}_i = \frac{|\delta_i^1| + |\delta_i^2| + ... + |\delta_i^m|}{m}$$

$$\delta_i^{max} = max\{|\delta_i^1|, |\delta i^2|, ..., |\delta_i^m|\}$$

Generally, there are two methodologies to define f'. One is the method through the analysis of characteristics of the input X_i thoroughly (Zhang 1995) to define f' (analysis methods) and the other is from limited X_i and the corresponding Y_i to define f' (inductive methods).

In the following subsections we will analyse these two methods.

Analysis methods

Basic principle of this methodology is to define a synthesis function f through analysis of characteristics of input X_i. The characteristics may include the relationships among original evidence sets from ESs which derive the input matrix, the factors which affect the desired final solution, and the weights for all factors.

Normally, analysis methods require some preconditions. If synthesis cases satisfy these preconditions, this kind of strategies can work well. In this method, only input has been analysed. The typical examples of analysis methods are: uncertainty management in which the final solution to the task is based on not only the mean value of uncertainties of the proposition but also the uniformity among ESs about the proposition (Khan & Jain 1985); a synthesis strategy for heterogeneous DESs which was developed based on both transformation functions among different inexact reasoning models among heterogeneous ESs and mean values of solutions from DESs (Zhang 1992). The methods of measurement of δ are according to some necessary conditions of synthesis of strategies (Zhang & Zhang 1994), or comparison with other strategies.

Inductive methods

The idea of inductive methods is different from the idea of analysis methods. By using inductive methods, we don't care about the characteristics of the input. However, we must know enough samples. Each sample represents one input matrix X_i and one corresponding output vector Y_i. These samples should cover very wide cases. Based on a number of X_i and corresponding Y_i, we try to induce a synthesis function f to map X_i to Y_i closely.

An example using inductive models is a neural network strategy for synthesis of solutions (Zhang & Zhang 1995). The character of these kind of strategies are based on samples of both inputs and corresponding outputs to find the best mapping functions to match relationships between inputs and corresponding outputs. Once the relationship is found, actual outputs Y_i' comes from the mapping function and X_i.

A synthesis strategy derived by analysis methods

In this subsection, we discuss a computational strategy which was developed by analysis methods.

This computational synthesis strategy is used to solve synthesis problems for conflict cases (Zhang & Zhang 1994). The design principles of this strategy are to (1) analyze the characteristics of input matrix X_i (uncertainties from different ESs) in conflict cases, (2) design a mathematical model to solve inconsistency problems based on the mean value and uniformity of uncertainties from ESs, and (3) use evidential theory to solve contradiction problems. The above procedures are typically main steps of analysis methods.

Different cases of belief conflicts Suppose there are n ESs in a DES to evaluate an attribute with m possible values, the range of uncertainty of ESs is in $[-1, 1]$ and the set of uncertainties given by different ESs is $U_j = \{CF_{ij}\}$, where $j = 1, 2, ..., m$ and $i = 1, 2, ..., n$. U_j represents jth column only of the matrix X_i. Belief conflicts can be divided into two cases:

Case 1: Inconsistency
All of the ESs in a DES believe that the proposition is partially true (or partially false) and the difference is their uncertainties. This situation can be described as: $\forall CF_{ij} \in U_j, CF_{ij} \geq 0$, which means the proposition is partially true, (or $\forall CF_{ij} \in U_j, CF_{ij} \leq 0$, which means the proposition is partially false), where 0 represents the unknown in the EMYCIN inexact reasoning model (Melle 1980).

Case 2: Contradiction
The opinions of all ESs in a DES are contradiction. Some of the ESs believe the proposition is partially true while others believe not. This case can be represented in the following way: $\exists CF_{lj} \in U_j, CF_{lj} < 0$ and $\exists CF_{kj} \in U_j, CF_{kj} > 0$.

The principle of the strategy The key idea of this strategy is to classify conflicts cases into inconsistency and contradiction. For each case, there is a corresponding algorithm to solve the problem.

This strategy includes five steps:

(a) If the range of uncertainties of a proposition is not in [0,1], the uncertainties of the proposition are transformed from that range to the range of [0, 1] by using the heterogeneous transformation functions (Zhang 1992). For example, if an ES uses the EMYCIN model, the range of $[-1, 1]$ should be transformed into the range of [0, 1].

(b) After transformation, the sum of uncertainties may not satisfy the precondition in the Probability model (Duda, Hart, & Nilsson 1976). If such a case happens, the normalization function is used to normalize uncertainties in the Probability model.

(c) If the conflict degree is great, the cluster strategy is used to classify the uncertainties into several subsets. In each subset, the conflict degree should satisfy a certain requirement. At least, it should fall in the case of inconsistency.

(d) For each subset, the *synthesis strategy for inconsistency* is used to obtain the final uncertainty among uncertainty values if there is more than one uncertainty value in a subset of uncertainties. The synthesis strategy for inconsistency calculates one column only from the input matrix (refer to Matrix 2.2) each time.

Here the basic idea of the synthesis strategy for inconsistency is briefly introduced.

Suppose that FCF denotes a final uncertainty value, $MEAN$ is the mean value of all uncertainties (CF_{ij}) in a subset where $i = 1, 2, ..., n$, and $UNIFORMITY$ is the deviation of uncertainty values, the FCF will depend on both $MEAN$ and $UNIFORMITY$ of uncertainties. One method of calculating $MEAN$ and $UNIFORMITY$ is as follows:

$$MEAN = \sum_{i=1}^{k} CF_{ij} * \frac{A_i}{\sum_{j=1}^{k} A_j}$$

$$UNIFORMITY = \sum_{i=1}^{k} |CF_{ij} - MEAN| * \frac{A_i}{\sum_{j=1}^{k} A_j}$$

where A_i is the authority of ES_i.

The formula of calculating FCF is:

$$FCF = \begin{cases} \gamma * MEAN - \beta_1 * UNIFORMITY & \text{if } MEAN \geq p_0 \\ \gamma * MEAN + \beta_2 * UNIFORMITY & \text{if } MEAN < p_0 \end{cases}$$

where γ, β_1, and β_2 are constants. We have derived $\gamma = 1$, $\beta_1 = \beta_2 = \frac{1}{2}$ (Zhang & Zhang 1994).

(e) If there is more than one subset after cluster, the *synthesis strategy for contradiction* is used to obtain the final uncertainty for the whole set of uncertainties. In this case, evidential theory (Shafer 1976) is used to form the synthesis strategy for contradiction.

Suppose S is a finite set, $S = \{s_1, s_2, ..., s_m\}$ where $s_1, s_2, ..., s_m$ are propositions and m is the number of possible values for an attribute. 2^s denotes the set of all subsets of S. Suppose Sp_k and Sp_l are two basic support functions over the same set 2^s. $Sp_k(s_i)$ and $Sp_l(s_i)$ are used to measure the uncertainty of proposition s_i by ES_k and ES_l, respectively where k, $l = 1, 2, ..., n$.

The principle of the synthesis strategy for contradiction is to calculate final uncertainties based on the whole input matrix. The synthesis function is defined as follows:

$$Sp(s_i) = \frac{Sp_k(s_i)Sp_l(s_i) + Sp_k(s_i)Sp_l(s) + Sp_k(s)Sp_l(s_i)}{Sp_k(s)Sp_l(s) + \sum_{i=1}^{m}(Sp_k(s_i)Sp_l(s_i) + Sp_k(s_i)Sp_l(s) + Sp_k(s)Sp_l(s_i))}$$

In the above functions, $Sp_l(s_i)$ and $Sp_l(s)$ are given by:

$$Sp_l(s_i) = \frac{Sp_l'(s_i)}{Sp_l'(s) + \sum_{i=1}^{m} Sp_l'(s_i)}$$

$$Sp_l(s) = \frac{Sp_l'(s)}{Sp_l'(s) + \sum_{i=1}^{m} Sp_l'(s_i)}$$

$$Sp_l'(s_i) = F_1(F_2(CF_{li}) * A_l)$$

$$Sp_l'(s) = F_1(F_2(1 - \sum_{i=1}^{m} CF_{li}) * A_l)$$

where CF_{li} is the uncertainty for the ith possible value from the ES_l (refer to Matrix 2.2), A_l is the authority for ES_l, F_1 is the transformation function from the EMYCIN model to the Probability model, and F_2 is the inverse function of F_1 (Zhang 1992). The same method can be used to define $Sp_k(s_i)$ and $Sp_k(s)$. In the steps (d) and (e), both uncertainties of a proposition from different ESs, and the authorities for each ES are considered.

A synthesis strategy derived by inductive methods

The key idea of inductive methods is to get the best mapping function based on certain number of samples (both input X_i and corresponding Y_i). Normally, we can use mathematical models to find mapping functions such as The Least Square Method. However, for some difficult problems, neural network strategies are very good methods to find the best mapping functions because neural networks can simulate some complicated relationships between inputs (uncertainty matrix from ESs) and outputs (final solution vector after synthesis).

In this subsection, we demonstrate one of inductive methods, neural network strategies. We have proposed three neural network strategies for synthesis of solutions in both conflict and non-conflict cases (Zhang & Zhang 1995). Here we introduce the basic principle of neural network strategies, and give an example of neural network strategies.

The basic principles The basic principles are: the inputs of the neural networks are the matrix of multiple solutions from ESs (refer to Matrix 2.2); the outputs of neural networks should be the desired final solutions after synthesizing multiple solutions. If a neural network can converge for all of patterns after training, this neural network can act as an inductive function.

An example of neural network strategies Now we introduce the structure of a neural network strategy working in conflict cases (Zhang & Zhang 1995). This method is to investigate whether a neural network can be trained to converge for solving belief conflicts if enough patterns are given. The works should be done include to collect enough numbers of patterns from the real world, to set up a neural network architecture, to decide an activation function, and to train this neural network by adjusting the weights of links to accommodate all patterns. Currently, 200 artificial patterns have been created, which cover various possibilities, in which 180 patterns were chosen as training patterns and further 20 patterns were used to test the neural network.

The architecture The input layer has 4 * 6 nodes and output layer has 6 nodes. The hidden layer consists of 49 nodes. The neural network is fully connected. The learning algorithm is backpropagation. Figure 1 shows the architecture of this neural network.

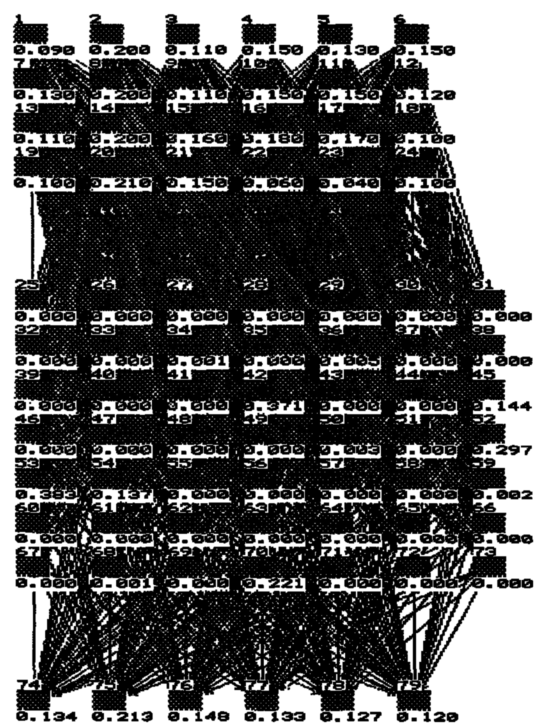

Figure 1: The architecture of the dynamic neural network

Result analysis The neural network was trained by 90,000 cycles.

If we use δi^{max} to represent the maximum error and $\bar{\delta}_i$ to represent the mean error from all testing data,

Figure 2 figures out the distribution of test errors of all test data from test patterns (untrained patterns).

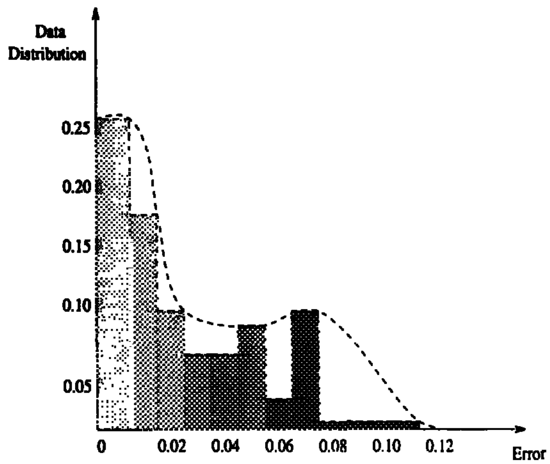

Figure 2: The distribution of the test errors

The mean error $\bar{\delta}_i = 0.036$ and for 93% of test data, $\delta_i^{max} < 0.07$. This result indicates that a neural network synthesis strategy is a good inductive method to solve synthesis problems if enough training patterns are available.

Comparison of two methodologies

Both analysis methods and inductive methods compensate each other in the following ways:

- Computational synthesis strategies are the first choice if the relationship among multiple solutions from different ESs to a subtask and the final solution from a DES to the subtask can be represented by a mathematical model.
- If there are enough patterns available and the neural network converges, the neural network strategy is better than computational strategies because it can simulate human experts quite closely. The neural network strategies cover a wider range of problem solving than computational synthesis strategies, because they can combine different strategies together by training patterns.
- If a neural network does not converge, or it is too hard to get enough training patterns, computational synthesis strategies can reasonably solve synthesis problems.

Conclusion

In this paper, we have identified the potential cases of synthesis in DESs and classified the types of DESs. Based on theses results, necessary conditions of synthesis strategies in different synthesis cases and two methodologies to solve the problem of synthesis of solutions in DESs, which are analysis methods and inductive methods, have been proposed. A computational strategy by using analysis methods and a neural network strategy which is an example of inductive methods have been introduced for solving synthesis problems in conflict synthesis cases. Both methodologies have been compared.

This work gives clear idea of two potential ways for doing research in synthesis of solutions and also offer a guideline for developing and choosing synthesis strategies in DESs by using two different methodologies.

Further work is to apply two different methodologies in real application domains, and investigate how to combine analysis methods with inductive methods.

References

Duda, R., Hart, P., and Nilsson, N. 1976. Subjective Bayesian Method for Rule-based Inference System. *AFIPS* 45: 1075-1082.

Khan, N. and Jain, R. 1985. Uncertainty Management in a Distributed Knowledge Based System. In Proceedings of 9th International Joint Conference on Artificial Intelligence, 318-320.

Liu, D., Zheng, F., Ma, Z., and Shi, Q 1992. Conflict Resolution in Multi-ES cooperation Systems. *Journal of Computer Science and Technology* 7: 144-152.

Melle, W. 1980. A Domain-Independent System That Aids in Constructing Knowledge-Based Consultation Programs. Ph.D. Dissertation, Computer Science Department, Stanford University, CA.

Shafer, G. ed 1976. *A Mathematical Theory of Evidence*, Princeton University Press, Princeton and London.

Zhang, C. 1992. Cooperation Under Uncertainty in Distributed Expert Systems. *Artificial Intelligence* 56: 21-69.

Zhang, M. and Zhang C. 1994. A Comprehensive Strategy for Conflict Resolution in Distributed Expert Systems. *Australia Journal of Intelligent Information Processing Systems* 1(2): 21-28.

Zhang, M. and Zhang, C. 1995. A Dynamic Neural Network Mechanism for Solving Belief Conflict in Distributed Expert Systems. In Proceedings of 6th Australian Conference on Neural Networks, 105-108.

Zhang, M. 1995. Synthesis of Solutions in Distributed Expert Systems. Ph.D. thesis, the University of New England, Australia.