Computational Intelligence, Volume 17, Number 2, 2001

TRANSFERRING CASE KNOWLEDGE TO ADAPTATION KNOWLEDGE: AN APPROACH FOR CASE-BASE MAINTENANCE

SIMON C. K. SHIU, DANIEL S. YEUNG
Department of Computing, Hong Kong Polytechnic University

CAI H. SUN
Business School, Renmin University

XI Z. WANG
Department of Mathematics and Computers, HeBei University

In this article we propose a case-base maintenance methodology based on the idea of transferring knowledge between knowledge containers in a case-based reasoning (CBR) system. A machine-learning technique, fuzzy decision-tree induction, is used to transform the case knowledge to adaptation knowledge. By learning the more sophisticated fuzzy adaptation knowledge, many of the redundant cases can be removed. This approach is particularly useful when the case base consists of a large number of redundant cases and the retrieval efficiency becomes a real concern of the user. The method of maintaining a case base from scratch, as proposed in this article, consists of four steps. First, an approach to learning feature weights automatically is used to evaluate the importance of different features in a given case base. Second, clustering of cases is carried out to identify different concepts in the case base using the acquired feature-weights knowledge. Third, adaptation rules are mined for each concept using fuzzy decision trees. Fourth, a selection strategy based on the concepts of case coverage and reachability is used to select representative cases. In order to demonstrate the effectiveness of this approach as well as to examine the relationship between compactness and performance of a CBR system, experimental testing is carried out using the Traveling and the Rice Taste data sets. The results show that the testing case bases can be reduced by 36 and 39 percent, respectively, if we complement the remaining cases by the adaptation rules discovered using our approach. The overall accuracies of the two smaller case bases are 94 and 90 percent, respectively, of the originals.

Key words: case-base maintenance, knowledge containers, fuzzy decision trees.

1. INTRODUCTION

The growing use of case-based reasoning (CBR) applications has brought with it increased awareness of the importance of case-base maintenance (CBM). According to Leake and Wilson (1998), "Case-base maintenance is the process of refining a CBR system's case base to improve the system's performance." That is, "case base maintenance implements policies for revising the organization or contents (representation, domain content, accounting information, or implementation) of the case base in order to facilitate future reasoning for a particular set of performance objectives." At present, large-scale CBR systems are becoming more prevalent, with case-library sizes ranging from thousands (Kitano and Shimazu 1996; Cheetham and Graf 1997) to millions of cases (Deangdej et al. 1996). Large case-library sizes raise concerns about the utility problem for case retrieval, and various case-deletion policies have been proposed to control case-base growth (Smyth and Keane 1995; Anand et al. 1998).

In the past, researchers have attempted to address various aspects of the CBM problem. To provide maintenance support at the case level, Smyth and Keane (1995) suggested a competence-preserving deletion approach. Competence (or coverage) is the

Correspondence should be addressed to Simon C. K. Shiu at the Department of Computing, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong; e-mail: *csckshiu@comp.polyu.edu.hk.*

range of target problems that a given system can solve and is also a fundamental evaluation criterion of CBR system performance. Smyth and McKenna (1998) also presented a new model of case competence and demonstrated a way in which the proposed model of competence can be used to assist case authors. Anand et al. (1998) proposed to use data-mining techniques in a novel role of a back-end technology for CBR systems, i.e., the acquisition of cases and discovery of adaptation knowledge. Hanney and Keane (1996) presented an inductive learning algorithm to extract adaptation knowledge from the cases in the case base. Their algorithm builds pairs of cases and uses the feature differences of these case pairs to build adaptation rules that are very useful in CBM. The approach of Hanney and Keane is based on the assumption that the differences occurring between cases in the case base are representative of the differences that will occur between future problems and the case base.

Apart from these investigations, Basak et al. (1998) used advanced artificial intelligence (AI) techniques, neural networks, and fuzzy methods to acquire features' importance and eliminate irrelevant features in a given data set. One important thing in the CBR community is to distinguish the salient features from all the features in the data set; feature-selection methods can reduce the task's dimensionality when they eliminate irrelevant features (Wettscherck and Aha 1995). The unsupervised approach in Basak et al. (1998) is very useful in dealing with data sets in a knowledge-poor domain and in helping to reduce the dimension size of the case bases.

Recently, Richter (1995, 1998) proposed the notion of knowledge containers, and it quickly became the standard paradigm for representation of the structural elements in CBR systems. The four knowledge containers are the vocabulary (index) used, the similarity measures, the solution transformation (adaptation knowledge), and the case base. The knowledge in the first three containers is compiled, while the knowledge in the cases is used at run time. According to Richter, each container can carry almost all knowledge available, and the manipulations on one container have little consequences on the others. However, there are no systematic evaluations of the relationship between this compiled and run-time knowledge. In this article, we try to establish a methodology that could be used to transfer case knowledge to adaptation knowledge. We further argue that by shifting knowledge from the case base to another container, we can significantly improve the retrieval efficiency of the system. The methodology, as proposed in this article, integrates identifying salient features, distinguishing different concepts, learning adaptation knowledge, computing case competence, and selecting seed cases together into a framework of CBM. Here we assume that the cases in the case base are a representative sample of the target problems, and the problem space is a regular one; namely, similar problems should have similar solutions. Our methodology focuses on balancing case-retrieval efficiency and competence for a large-size case base. The methodology is mainly based on the idea that a large case library is transformed to a small case library, together with a group of adaptation rules that are generated by fuzzy decision trees. These adaptation rules play the role of complementing the reduction of cases. As a result, a smaller case base is constructed to represent the original.

The details of the maintaining methodology consist of four steps. First, an approach to learning feature weights automatically is used to evaluate the importance of different features in a given case base. Second, clustering of cases is carried out to identify different concepts in the case base using the acquired feature knowledge. Third, adaptation rules are mined for each concept using fuzzy decision trees. Finally, a selection strategy based on the concepts of coverage and reachability is used to select representative cases. In order to demonstrate the effectiveness of this approach as well as to examine the relationship between the compactness and the performance of a CBR system,

experimental testing is carried out using the Traveling and the Rice Taste data sets. The results show that the testing case bases can be reduced by 36 and 39 percent, respectively, if we complement the remaining cases by adaptation rules discovered using our approach. The overall accuracies of the two smaller case bases are 94 and 90 percent, respectively, of the originals. This article is structured as follows: Section 2 describes the methodology in detail. Section 3 explains the experimental analysis and findings. Section 4 discusses the complexity issue of our approach. Finally, the article concludes the contribution of our approach and provides the plan of our future work.

2. METHODOLOGY FOR MAINTAINING THE CASE BASE

Throughout this section we consider a case library in which all features are supposed to take on real-numbered values. We first introduce a weighted distance metric $d_{pq}^{(w)}$ and a similarity measure $SM_{pq}^{(w)}$ that will be used throughout the article.

Let $CL = \{e_1, e_2, \ldots, e_N\}$ denote our discussed case library. Each case in the library can be identified by an index of corresponding features. In addition, each case has an associated action. More formally, we use a collection of features $\{F_j\,(j = 1, \ldots, n)\}$ to index the cases and a variable V to denote the action. The ith case e_i in the library can be represented as an $n+1$-dimensional vector, i.e., $e_i = (x_{i1}, x_{i2}, \ldots, x_{in}, v_i)$, where x_{ij} corresponds to the value of feature $F_j\,(1 \leq j \leq n)$ and v_i corresponds to the action $(i = 1, \ldots, N)$.

Suppose that for each $j\,(1 \leq j \leq n)$, a weight $w_j\,(w_j \in [0, 1])$ has been assigned to the jth feature to indicate the importance of the feature. Then, for any pair of cases e_p and e_q in the library, a weighted distance metric can be defined as

$$d_{pq}^{(w)} = d^{(w)}(e_p, e_q) = \left(\sum_{j=1}^{n} w_j^2 (x_{pj} - x_{qj})^2\right)^{1/2} = \left(\sum_{j=1}^{n} w_j^2 \chi_j^2\right)^{1/2} \tag{1}$$

where $\chi_j^2 = (x_{pj} - x_{qj})^2$. When all the weights are equal to 1, the distance metric defined above degenerates to the Euclidean measure, denoted by $d_{pq}^{(1)}$, in short, denoted by d_{pq}.

Using the weighted distance, a similarity measure between two cases $SM_{pq}^{(w)}$ can be defined as

$$SM_{pq}^{(w)} = \frac{1}{1 + \alpha d_{pq}^{(w)}} \tag{2}$$

where α is a positive parameter. When all the weights take value 1, the similarity measure is denoted by $SM_{pq}^{(1)}$.

It should be noted that the real-value features discussed above could, without difficulty, be extended to the features that take values in a normed vector space. For example, assume that for each feature, a distance measure has been defined already. The distance measure for the jth feature is denoted by ρ_j; i.e., ρ_j is a mapping from $F_j \times F_j$ to $[0, \infty)$ (where F_j denotes the range of the jth feature) with properties

(a) $\rho_j(a, b) = 0$ if and only if $a = b$
(b) $\rho_j(a, b) = \rho_j(b, a)$
(c) $\rho_j(a, b) \leq \rho_j(a, c) + \rho_j(c, b)$.

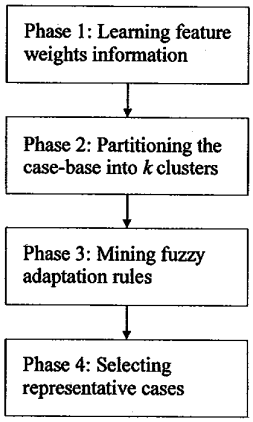

FIGURE 1. Four phases of CBM.

Some typical formula of distance measure such as the following could be used for nonnumerical features:

(a) $\rho_j(a, b) = |a - b|$ if a and b are real numbers
(b) $\rho_j(A, B) = \max_{a \in A, b \in B} |a - b|$ if A and B are intervals
(c) $\rho_j(a, b) = \begin{cases} 1 & \text{if } a \neq b \\ 0 & \text{if } a = b \end{cases}$ if a and b are symbols.

and the distance between two cases x and y can be computed by

$$\text{Distance}(x, y) = \sqrt{\sum_{j=1}^{n} w_j^2 \rho_j^2(x_j, y_j)}.$$

Let us now give the paradigm for maintaining case libraries, with a brief explanation. Our proposed methodology for case maintenance contains four phases, as shown in Figure 1. Phase 1 is the preliminary to phase 2, and its purpose is to assign a weight to each feature. These weights will play an important role in the clustering of the next phase. Phase 2 aims to partition the case library into several clusters using the weighted distance metric with the weights learned in phase 1. To some extent, phase 1 guarantees that the performance of clustering using the weighted distance metric is better than that using the Euclidean metric. Each cluster is considered to have several representative cases, and a nonrepresentative case in the cluster is considered as a perturbation of certain representative cases. The perturbation can be approximated by a group of adaptation rules. Phase 3 aims to mine these adaptation rules, which could be regarded as a kind of adaptation knowledge of the representative cases. This is the most important phase, since the quality of the adaptation knowledge heavily affects the selection of representative cases. Moreover, this phase can be regarded as an integration of the strengths of both the case-based and rule-based methods. We adopt fuzzy decision-tree induction to mine these adaptation rules. Phase 4 is to select several representative cases from each cluster according to the adaptation rules mined in phase 3. Consequently, the representative cases and the adaptation rules, which have the same competence as the original library, become an alternative to the original library.

The proposed methodology is particularly useful when a case base has a lot of redundancy that is not caused simply by repeated cases but rather by the interaction among features. This type of redundancy will seriously affect the quality of the problem-solving ability of a CBR system. By learning the feature weights of the cases, this type of redundancy can be illuminated. For example, two of the following three cases can be regarded as redundant:

Feature weights	1	0	0
Case 1	a	*	*
Case 2	a	*	*
Case 3	a	*	*

where a is a specific value and * is any value.

If a small loss of solution accuracy is acceptable, this approach will be very useful in situations where the user is facing a very large case base and the retrieval efficiency becomes a real concern because of high usage of the system or too many concurrent users.

One of the drawbacks of our approach is the difficulty in determining the balance between solution quality and retrieval efficiency. This is so because we have to define the tolerable range for solution quality before carrying out the knowledge transfer. However, this tolerable range may vary and depend very much on the problem domains and the characteristics of users. Therefore, partial or complete restructuring of the knowledge containers may be necessary if expectation of the solution accuracy has been changed.

2.1. Learning Feature Weights

In this section, a feature-evaluation function (in which the feature weights are regarded as the variables) is defined. The smaller the evaluation value, the better are the corresponding features. Thus we would like to find the weights so that the evaluation function attains its minimum. The task of minimization of the evaluation function with respect to weights is performed using a gradient-descent technique. We formulate this optimization problem as follows.

For a given collection of feature weights w_j ($w_j \in [0, 1]$, $j = 1, \ldots, n$) and a pair of cases e_p and e_q, Equation (1) defines a weighted distance measure $d_{pq}^{(w)}$, and Equation (2) defines a similarity measure $SM_{pq}^{(w)}$. When all weights take value 1, $d_{pq}^{(w)}$ and $SM_{pq}^{(w)}$ degenerate to the Euclidean distance $d_{pq}^{(1)}$ and $SM_{pq}^{(1)}$. A feature-evaluation index E is defined as

$$E(w) = \frac{2\left\{\sum_p \sum_{q(q<p)} \left[SM_{pq}^{(w)}\left(1 - SM_{pq}^{(1)}\right) + SM_{pq}^{(1)}\left(1 - SM_{pq}^{(w)}\right)\right]\right\}}{N(N-1)} \tag{3}$$

where N is the number of cases in the case base.

Noting that the feature-evaluation function $E(w)$ will gradually become zero when $SM_{pq}^{(w)} \to 0$ or 1, we hope to find a collection of weights so that the feature-evaluation function attains its minimum.

To minimize Equation (3), we use a gradient-descent technique. The change in w_j (i.e., Δw_j) is computed as

$$\Delta w_j = -\eta \frac{\partial E}{\partial w_j} \tag{4}$$

for $j = 1, \ldots, n$, where η is the learning rate. For the computation of $\partial E/\partial w_j$, the following expressions are used

$$\frac{\partial E(w)}{\partial w_j} = \frac{2\left[\sum_p \sum_{q(q<p)} \left(1 - 2SM_{pq}^{(1)}\right) \frac{\partial SM_{pq}^{(w)}}{\partial d_{pq}^{(w)}} \frac{\partial d_{pq}^{(w)}}{\partial w_j}\right]}{N(N-1)} \tag{5}$$

$$\frac{\partial SM_{pq}^{(w)}}{\partial d_{pq}^{(w)}} = \frac{-\alpha}{\left(1 + \alpha d_{pq}^{(w)}\right)^2} \tag{6}$$

$$\frac{\partial d_{pq}^{(w)}}{\partial w_j} = \frac{w_j \chi_j^2}{\left(\sum_{j=1}^{n} w_j^2 \chi_j^2\right)^{1/2}} \tag{7}$$

Algorithm 1: Training Algorithm

Step 1. Select the parameter α and the learning rate η.
Step 2. Initialize w_j with random values in [0, 1].
Step 3. Compute Δw_j for each j using Equation (4).
Step 4. Update w_j with $w_j + \Delta w_j$ for each j.
Step 5. Repeat steps 3 and 4 until convergence, i.e., until the value of E becomes less than or equal to a given threshold or until the number of iterations exceeds a certain predefined number.

After training, the function $E(w)$ attains a local minimum. We expect that, on average, the similarity values $\{SM_{pq}^{(w)}, _{p=1,\ldots,N,q<p}\}$ with trained weights are closer to 0 or 1 than those without trained weights, such as $\{SM_{pq}^{(1)}, _{p=1,\ldots,N,q<p}\}$.

2.2. Partitioning the Case Base into Several Clusters

Motivated by the idea that a cluster of cases describing the same concept should have one (or more) representative case(s), we attempt to partition the case library into several clusters using the weighted distance metric with the weights learned in phase 1. Since the considered features are real-valued, many methods, such as C-mean clustering (Bezdek 1981) and Kohonen's self-organized network (Kohonen 1988), can be used to partition the case library. However, we adopt a typical approach to clustering, i.e., the similarity matrix (Fu 1992), which uses only the information of similarity between cases. This approach first transforms the similarity matrix to an equivalent matrix and then considers the cases that are equivalent to each other as one cluster.

Algorithm 2: Clustering Algorithm

Step 1. Give a significant level (threshold) $\beta \in (0, 1]$.

Step 2. Determine the similarity matrix $SM = (SM_{pq}^{(w)})$ according to Equations (2) and (1).

Step 3. Compute $SM1 = SM \circ SM = (s_{pq})$, where $s_{pq} = \max_k [\min(SM_{pk}^{(W)}, SM_{kq}^{(w)})]$.

Step 4. If $SM1 \subset SM$, then go to step 5; else, replace SM with $SM1$ and go to step 3.

Step 5. Determine several clusters based on the rule "case p and case q belong to the same cluster if and only if $s_{pq} \geq \beta$."

It is worth noting that the result of clustering depends strongly on the feature weights that are used in the computation of similarity between two cases. We expect the performance of clustering with the feature weights trained in phase 1 to be better than the performance without trained feature weights. We evaluate the performance of clustering by the following three indexes:

(a) *Intrasimilarity.* For a given cluster L, the intrasimilarity of L is defined as

$$SM_L^{(w)} = \frac{2}{r(r-1)} \sum_{p,q \in L(p<q)} SM_{pq}^{(W)} \tag{8}$$

where r is the number of cases in the cluster L. For a clustering with m clusters $\{L_1, L_2, \ldots, L_m\}$, the intrasimilarity is defined as the average of all its cluster intrasimilarities, i.e.,

$$SM_{\text{int } ra}^{(w)} = \frac{1}{m} \sum_{j=1}^{m} SM_{L_j}^{(w)} \tag{9}$$

It is clear that the value of $SM_{\text{int } ra}^{(w)}$ is in [0, 1]. The bigger the value of $SM_{\text{int } ra}^{(w)}$, the better the performance of the clustering.

(b) *Intersimilarity.* For a pair of clusters L_1 and L_2, the intersimilarity is defined as

$$SM_{L_1, L_2}^{(w)} = \frac{1}{r_1 r_2} \sum_{p \in L_1, q \in L_2} SM_{pq}^{(w)} \tag{10}$$

where r_1 and r_2 are numbers of cases in L_1 and L_2, respectively. For a clustering with m clusters $\{L_1, L_2, \ldots, L_m\}$, the intersimilarity is defined as the average of all pairs of intersimilarities, i.e.,

$$SM_{\text{int } er}^{(w)} = \frac{2}{m(m-1)} \sum_{1 \leq i < j \leq m} SM_{L_i, L_j}^{(w)} \tag{11}$$

Obviously, the value of $SM_{\text{int } er}^{(w)}$ is in [0, 1]. The smaller the value of $SM_{\text{int } er}^{(w)}$, the better the performance of the clustering.

(c) *Number of clusters.* Under an acceptable accuracy, the intrasimilarity of clustering is bigger than or equal to a threshold (>0.5) and the intersimilarity of clustering is smaller than or equal to a threshold (< 0.5).

As a result of phase 2, the original case base is partitioned into m clusters. For a new case whose action (solution) remains to be determined, one cluster to which the new case belongs should be obtained first by retrieval. Since the clustering result of phase 2 is crisp (not fuzzy), we require the clustering with m clusters $\{L_1, L_2, \ldots, L_m\}$ to satisfy the following property:

$$\min_{q \in L_i, q \neq p} SM_{pq}^{(w)} > \max_{i \leq j \leq m, j \neq i} \left(\max_{q \in L_j} SM_{pq}^{(w)} \right) \tag{12}$$

for any $i (1 \leq i \leq m)$ and any case $p \in L_i$.

Equation (12) describes such a situation that the similarity between a case and the cluster to which the case belongs is greater than the similarity between the case and any other cluster. This property can be satisfied if the number of clusters is selected to be appropriate and there exists no noise case that is defined to have all the same feature values as certain existing cases but does not have the same action (solution).

2.3. Mining Adaptation Rules by Fuzzy Decision Trees

After phase 2, the original case base is partitioned into several clusters. In this article, we consider that each cluster has one or more representative cases and that a nonrepresentative case in the cluster is considered as a perturbation of certain representative cases. The perturbation is handled by a group of adaptation rules. In detail, let p be a representative case and q be a nonrepresentative case in some cluster; we expect that the solution of q can be approximately obtained by an appropriate adaptation (adjustment) of the solution of p according to adaptation rules. In other words, the solution adjustment is conducted according to an adaptation rule. For example, if a representative case and a nonrepresentative case are, respectively, $p = (1, 2, 3, 4)$ and $q = (0.9, 2, 3, 4.01)$, in which the first three components are feature values and the last component is the solution, an adaptation rule is "IF the change of first feature is *negatively small*, THEN the adjustment of the solution is *very positively small*," and then q's solution can be considered to be obtained by p's solution with an adjustment based on the adaptation rule. This phase investigates how to find these adaptation rules.

A machine-learning technique, fuzzy decision-tree induction, is used to mine the adaptation knowledge. One popular and powerful heuristic algorithm for generating crisp decision trees is called *ID3*. The earlier version of ID3, which is based on minimum information entropy to select expanded attributes, was proposed by Quinlan (1986). Subsequently, a fuzzy version of ID3 based on minimum fuzzy entropy was suggested by several authors (Umanol et al. 1994; Ichihashi 1996; Jeng 1997). Due to its many advantages, such as robustness and comprehensibility, we adopt this technique.

For each cluster $L = \{e_1, e_2, \ldots, e_m\}$ obtained from phase 2, we denote each case in the form of $e_i = (x_{i1}, x_{i2}, \ldots, x_{in}, v_i)$, where x_{ij} corresponds to the value of feature $F_j (1 \leq j \leq n)$ and v_i corresponds to the action $(i = 1, \ldots, m)$. Arbitrarily taking a case $e_k (1 \leq k \leq m)$in the cluster L, a set of vectors, namely, $\{f_i | f_i \in R^{n+1},\ i = 1, 2, \ldots, m\}$, can be given in the following way:

$$f_i = e_i - e_k = (x_{i1} - x_{k1}, x_{i2} - x_{k2}, \ldots, x_{in} - x_{kn}, v_i - v_k) = \{y_{i1}, y_{i2}, \ldots, y_{in}, u_i\}$$

We attempt to find several adaptation rules with respect to the case $e_k (1 \leq k \leq m)$ from the set of vectors $\{f_i | f_i \in R^{n+1}, i = 1, 2, \ldots, m\}$ by fuzzy decision tree.

Consider a problem of learning from examples in which there are $n + 1$ numerical attributes $\{A^{(1)}, A^{(2)}, \ldots, A^{(n)}, A^{(n+1)}\}$($A^{(n+1)}$ is the classification attribute). Then

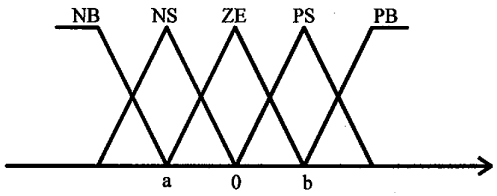

FIGURE 2. Five membership functions.

$\{f_i | i = 1, 2, \ldots, m\}$ can be regarded as m examples described by the $n+1$ attributes. We first fuzzify these $n+1$ numerical attributes into linguistic terms.

The number of linguistic terms for each attribute is assumed to be five (which can be enlarged or reduced if needed in a real problem). These five linguistic terms are *Negative Big, Negative Small, Zero, Positive Small,* and *Positive Big,* in short, *NB, NS, ZE, PS,* and *PB,* respectively. Their membership functions are supposed to have triangular form and are shown in Figure 2. For each attribute (the kth attribute $A^{(k)}$, $1 \leq k \leq n+1$) with the attribute values $Range(A^{(k)}) = \{y_{1k}, y_{2k}, \ldots, y_{mk}\}$, the two parameters in Figure 2, a and b, are defined by

$$a = \sum_{y \in N} y/Card(N) \qquad \text{and} \qquad b = \sum_{y \in P} y/Card(P) \tag{13}$$

in which $N = \{y | y \in Range(A^{(k)}), y < 0\}$, $P = Range(A^{(k)}) - N$, and $Card(E)$ denotes the cardinality of a crisp set E.

After this fuzzification, each example can be regarded as a $5 \times (n+1)$-dimensional vector of membership degree. By putting the m vectors together, a matrix with m rows and $5 \times (n+1)$ columns is formed. According to this matrix, we propose our fuzzy decision-tree generation procedure—fuzzy ID3. In comparison with the existing versions of fuzzy ID3, our proposed version is founded on the viewpoint that each linguistic term of attributes and each node in the tree are considered to be a fuzzy set defined on the example-label space $\{1, 2, \ldots, m\}$.

To avoid confusion of notations, we denote the five linguistic terms of the kth attribute by $L_k = \{A_1^{(k)}, \ldots, A_5^{(k)}\}$ for each k $(1 \leq k \leq n)$ and $L_{n+1} = \{C_1, \ldots, C_5\}$. Let N be an arbitrary node of a given fuzzy decision tree. The relative frequency of the node N with respect to the cluster $C_l \in L_{n+1}$ is defined as

$$f_l(N) = \frac{M(N \cap C_1)}{M(N)} \tag{14}$$

where $M(A)$ denotes the sigma count (the sum of all membership degrees) of a fuzzy set A. Usually, $f_l(N)$ is regarded as the subsethood of N in C_l and is interpreted as the degree of truth for the fuzzy rule **IF** N **THEN** C_l. The fuzzy entropy of the node N with respect to the clusters C_l $(l = 1, 2, \ldots, 5)$ is defined as

$$FE(N) = \sum_{l=1}^{5} f_l(N)[1 - f_l(N)] \tag{15}$$

Consider a nonleaf node S and n attributes $A^{(1)}, \ldots, A^{(n)}$ to be selected. For each k $(1 \leq k \leq n)$, the attribute $A^{(k)}$ takes five values of the fuzzy subsets $A_1^{(k)}, \ldots, A_5^{(k)}$.

Hence, for the attribute $A^{(k)}$, five son nodes of S, $S \cap A_1^{(k)}, \ldots, S \cap A_5^{(k)}$, will result. The information gain of the attribute $A^{(k)}$ at the node S is defined as

$$Gain(A^{(k)}, S) = FE(S) - \sum_{i=1}^{5} \frac{M(S \cap A_i^{(k)})}{\sum_{j=1}^{5} M(S \cap A_j^{(k)})} FE(S \cap A_i^{(k)}) \tag{16}$$

Algorithm 3: Fuzzy ID3 Heuristic Algorithm. Consider the whole training set as the first candidate node. Given a leaf standard of frequency γ, **WHILE** there exist candidate nodes, **DO**

Step 1. Randomly choose one candidate node S with n attributes $A^{(1)}, \ldots, A^{(n)}$ to be selected.
Step 2. If the frequency of some cluster exceeds γ at the node S, then regard the node S as a leaf and go to step 6.
Step 3. Compute $Gain(A^{(k)}, S)$ $(k = 1, 2, \ldots, n)$.
Step 4. Select k_0 such that $Gain_{k_0}(S) = \max_{1 \le k \le n} Gain(A^{(k)}, S)$.
Step 5. If $Gain_{k_0}(S) \le 0$, then regard the node S as a leaf. If $Gain_{k_0}(S) > 0$, then select the k_0th attribute as the expanded attribute, generate the son nodes of S, and regard these son nodes as new candidate nodes.
Step 6. Label node S, which is no longer a candidate node.

The key points of the fuzzy ID3 heuristic are that (1) the nonpositive *Gain* is regarded as a leaf standard and (2) the positive maximum *Gain* is the expanded attribute standard.

After generating the fuzzy decision tree, a set of adaptation rules can be extracted from the tree. The extraction is straightforward; i.e., each path from the root to a leaf is converted into an adaptation rule (fuzzy production rule). Thus the number of leaves is just the number of adaptation rules. With respect to the extracted adaptation rules, we need a reasoning mechanism to predict the amount of adjustment for the solution of nonrepresentative cases. We propose our fuzzy reasoning mechanism as follows:

Suppose that there have been L extracted adaptation rules, denoted by $\{P_i, i = 1, 2, \ldots, L\}$. The ith adaptation rule $P_i (1 \le i \le L)$ is represented in the following form:

$$\textbf{IF } [A^{(1)} = V_1^i] \textbf{ AND } [A^{(2)} = V_2^i] \textbf{ AND} \cdots \textbf{ AND } [A^{(n)} = V_n^i] \textbf{ THEN } [U = V_{n+1}^i] \tag{17}$$

in which $A^{(j)} (j = 1, \ldots, n)$ and U are variables (attributes), and their value V_j^i can be one of the five fuzzy sets defined in Figure 2 or empty. It should be noted that $V_j^i (1 \le j \le n)$ being empty means that the proposition $[A^{(j)} = V_j^i]$ does not appear in rule (17).

Let $e = \{y_1, y_2, \ldots, y_n, u\}$ be an example remaining to be tested; i.e., the attribute values $y_j (1 \le j \le n)$ are known, but the adjustment u is unknown. The value of u is determined by the following procedure:

1. For each production rule $P_i (1 \le i \le L)$ and its every proposition $[A^{(j)} = V_j^i]$ with $V_j^i \ne$ empty $(1 \le j \le n)$, compute $MD_j^{(i)}$, which denotes the membership degree of $y_j \in V_j$
2. Compute the overall similarity $SM^{(i)}$ by

$$SM^{(i)} = \min_j (MD_j^{(i)}) \tag{18}$$

3. Compute x_k $(k = 1, 2, \ldots, 5)$ by

$$x_k = \max_i \{SM^{(i)} | V_{n+1}^i = CLASS_k\} \tag{19}$$

where $CLASS_k (k = 1, 2, 3, 4, 5)$ are the five fuzzy sets *NB*, *NS*, *ZE*, *PS*, and *PB* defined in Figure 2. This results in a fuzzy set $\{x_1, x_2, \ldots, x_5\}$ defined on $\{NB, NS, ZE, PS, PB\}$.

4. The adjustment amount u is obtained by the following defuzzification formula:

$$u = \frac{2ax_1 + ax_2 + bx_4 + 2bx_5}{x_1 + x_2 + \cdots + x_5} \tag{20}$$

where parameters a and b are given by Equation (13).

It is worth noting that the operations *min* and *max* used in Equations (18) and (19) can be extended, respectively, to T-norm and S-norm represented by

$$T = 1 - \left[(1-x)^p + (1-y)^p + (1-x)^p(1-y)^p\right]^{1/p} \quad \textit{and}$$
$$S = (x^p + y^p + x^p y^p)^{1/p} \quad (p \geq 0) \tag{21}$$

where p is a parameter.

As a result of this phase, for each case of a considered cluster, a set of adaptation rules (fuzzy production rules) is generated, and a reasoning mechanism for this set of fuzzy rules is given.

2.4. Selecting Representative Cases

This phase aims to select representative cases from each cluster according to the adaptation rules obtained in phase 3. Our selection strategy is based on a ε-coverage concept. Before introducing this concept, we review some related works for case deletion.

Smyth and Keane (1995) described a technique for measuring the local coverage of individual cases with respect to a system's retrieval and adaptation characteristics. They also suggested deleting cases based on their coverage and reachability. They defined the coverage of a case as "the set of target problems that it can be used to solve," and the reachability of a target problem "is the set of cases that can be used to provide a solution for the target." Based on these measures, Smyth and Keane classified cases within the case base into four groups: Pivotal (if its reachability is a singleton consisting of itself), Auxiliary (if its coverage is subsumed by the coverage of a case to which it is reachable), Spanning (if its coverage space links regions covered by other cases), and Support (groups of cases having the same coverage). The deletion policy (footprint policy) suggested by Smyth and Keane (1995) was to delete auxiliary cases first, then support cases, then spanning cases, and finally pivotal cases. If more than one case is a candidate for deletion, substrategies are formulated when deciding on which case to delete. The disadvantage of the footprint deletion policy is that the coverage and reachability of a case depend on the adaptation knowledge available.

Instead of the deletion, we propose a selection strategy that makes use of concepts proposed by Smyth and Keane with our modification (called *ε-coverage* and *ε-reachability*, respectively). Compared with the preceding deletion strategy, the meaning of our proposed selection strategy is very clear.

Let L be a cluster in which each case e is accompanied with a set of adaptation rules $AR(e)$, ε be a small positive number, and $e_p = (x_{p1}, x_{p2}, \ldots, x_{pn}, v_p)$ and $e_q = (x_{q1}, x_{q2}, \ldots, x_{pn}, v_q)$ and $e_q = (x_{q1}, x_{q2}, \ldots, x_{pn}, v_q)$ be two cases in the cluster L. According to the reasoning mechanism established in phase 3, an adjustment amount Δ of the solution for case e_q can be obtained by matching $(x_{q1} - x_{p1}, \ldots, x_{qn} - x_{pn})$ against $AR(e_p)$. If $v_q + \Delta \in (v_p - \varepsilon, v_p + \varepsilon)$, then e_p is said to ε-cover with e_q. The ε-coverage and ε-reachability of the case e_p are defined by

$$\text{Coverage}(e_p) = \{e | e \in L,\ e \text{ is } \varepsilon\text{-covered by } e_p\} \tag{22}$$

and

$$\text{Reachability}(e_p) = \{e | e \in L,\ e\ \varepsilon\text{-covers with } e_p\} \tag{23}$$

respectively.

The ε-coverage of a case e represents the generalization capability of this case. The bigger the number of cases in the ε-coverage, the more representative is the selected case e. As a kind of rule extraction, it is a commonsense application of Occam's razor. On the other hand, the ε-reachability of a case e represents the degree to which e can be replaced by another case. The smaller the number of cases in the ε-reachability, the more important is the selected case e. As an index of evaluation of selected cases, to a great extent it reflects the difference between the rule-based and case-based approaches. Our proposed selection procedure integrates these two approaches.

For the cluster L and its one subset S, S is said to have the same competence as the cluster L under an error standard ε if each case in the cluster L can be ε-covered by a certain case in S. Usually, subset S is composed of several representative cases of L. One key point of our selection strategy is to guarantee that the selected cases have the same competence as the original case library.

Another key point of our selection strategy is to consider the number of selected cases. There is always a tradeoff between the number of cases to be stored in the case library of a case-based expert system and the performance of retrieval efficiency. The larger the case library, the more the problem space is covered; however, it also would downgrade the system performance if the number of cases grows to an unacceptably high level. Naturally, we expect the number of selected representatives to be as small as possible.

Considering the preceding key points, we formulate our selection strategy as an optimization problem, as described below:

For a given cluster obtained in phase 3, find its one subset so that the subset has the same competence as the original cluster and the number of cases in the subset is minimum. Noting that $e_p \in \text{Coverage}(e_p)$ holds for each e_p in the cluster, one can easily see that the preceding optimization problem is equivalent to the following optimal set cover (OSC) problem:

OSC Problem. Let T be a finite set and $F = \{S_1, S_2, \ldots, S_p\}$ be a family of subsets of T. We say F is a *cover* of T if $\cup_{i=1}^{p} S_i \supset T$. We say $|F^*|$ is an *optimal cover* of T if F^* is a cover of T and $|F^*| \leq |F|$ for an arbitrary cover of T, F, where $||$ denotes the number of elements of a set. The problem of finding F^* is called the *OSC problem*.

Unfortunately, the OSC problem was proven to be NP-hard by Johnson (1973). Thus, finding the exact solution for our proposed optimization problem of selecting representative cases is not realistic. An intuitive and powerful heuristic algorithm is designed below to find the approximate solution.

Algorithm 4: Case-Selection Algorithm. Given a cluster L and error-standard ε, R is initialized to be an empty set.

Step 1. For each case e in L, determine $Coverage(e)$ by a set of adaptation rules associated with the case e, $AR(e)$.

Step 2. Find case e^* such that $|Coverage(e^*)| = \max_{e \in L} |Coverage(e)|$. If there exists more than one case, so that the maximum is reached, select a case e^{**} from them so that $|Reachability(e^{**})| = \min |Reachability(e^*)|$. If there exists more than one case, so that the minimum is reached, select one randomly as case e^*.

Step 3. Put $R = R \cup \{e^*\}$ and $L = L - Coverage(e^*)$, if $|L| = 0$, then stop; else, go to step 2.

Consequently, the set R is regarded as an approximate solution of the optimization problem.

3. EXPERIMENTAL ANALYSIS

Our CBM methodology has been described. In this section we try to demonstrate the effectiveness of this approach as well as to examine the relationship between the compactness and the performance of a CBR system. Two standard data sets are used in this study. The first one is a case base of 1024 cases from the Travel domain. Each case consists of 11 attributes such as type of vacation, length of stay, holiday type, hotel, etc. This case base is available from the URL *http//www.ai-cbr.org.* The second data set is adopted from the Nozaki et al. (1997) Rice Taste problem. Each case consists of five inputs and a single output whose values are associated with subjective evaluations of the flavor, appearance, taste, stickiness, toughness, and overall evaluation of 105 different kinds of rice. The experiments are carried out using a Pentium III machine, and the programs are written in Microsoft Visual C++.

3.1. Travel Case Base

Size and Accuracy of the Transformed Case Base. Table 1 shows a sample record of the Travel case base. We use "Holiday Type," "Number of Persons," "Region," "Transportation," "Duration," "Season," "Accommodation," and "Hotel" as the problem features, and "Price" as the solution feature.

After applying the learning feature-weights algorithm mentioned in Section 2.1 to these cases, the feature-weight results shown in Table 2 are obtained (learning iterations = 100 cycles, $\alpha = 0.157$, and $\eta = 10$).

The clustering of cases is carried out both with and without the feature-weights information. The results are shown in Table 3. When feature-weights information is used to guide the clustering of cases, much better results are obtained. The user can specify a particular significant level β for further learning of adaptation rules.

In our experiment we have chosen $\beta = 0.96$ as the significant level. As a result, the cases are partitioned into 55 classes. Some of these classes are shown in Table 4. We label classes with less than 10 records as Odd classes and the rest Not-odd classes. The learning of fuzzy adaptation rules is carried out on the Not-odd classes.

In the mining of fuzzy adaptation rules, we fuzzify the numeric features (see Table 1) into five linguistic variables, i.e., *Negative Big, Negative Small, Zero, Positive Small,* and *Positive Big.* The symbolic features are used directly in the mining process.

TABLE 1. A Sample Record of the Travel Case Base

Name	Data Type	Example
Case	Symbolic	Journey1021
Journey Code	Numeric	1021
Holiday Type	Symbolic	Recreation
Number Of Persons	Numeric	6
Region	Symbolic	France
Transportation	Symbolic	Car
Duration	Numeric	14
Season	Symbolic	January
Accommodation	Symbolic	Holiday Flat
Hotel	Symbolic	H. Flat Cheval Blanc, France
Price	Numeric	1728

The general form of a fuzzy adaptation rule generated from the fuzzy decision tree is as follows:

IF the change of X_1 is [*Small* | *Medium* | *Big* | *Symbolic Values*]

[AND the change of X_2 is [*Small* | *Medium* | *Big* | *Symbolic Values*]
[AND the change of X_3 is [*Small* | *Medium* | *Big* | *Symbolic Values*]]]

THEN the change of Price is [*Negative Big* | *Negative Small* | *Zero* | *Positive Small* | *Positive Big*].

where X = {Holiday Type, Number of Persons, Region, Transportation, Duration, Season, Accommodation, and Hotel}. The maximum number of antecedents of each fuzzy rule is limited to two in this experiment. For example, in cluster 7, which consists of 13 cases, one of the adaptation rules is

Rule1: IF Holiday Type is changed from "Education" to "City"

THEN the change of Price is Positive Small.
The rule's confidence is 0.89.

According to the case-selecting strategy defined in Section 2.4, we select cases {2, 10, 9, 12, 1} as the representative cases in this cluster 7 (see Table 5). As a result of this selection, a total of 25 fuzzy adaptation rules are also selected (i.e., each case has 5 adaptation rules on average). A specific ε is selected for each Not-odd cluster by controlling the relative solution error to be less than 15 percent. The overall selection results of the Not-odd clusters are shown in Table 6.

TABLE 2. Feature Weights of the Problem Features

Holiday Type	Number of Persons	Region	Transportation	Duration	Season	Accommodation	Hotel
0.1374	0.0891	0.0662	0.3691	1.0000	0.0440	0.3443	0.0503

TABLE 3. Clustering of Cases With and Without Feature Weights

	Without Feature-Weight Information			With Feature-Weight Information		
Significant Level (β)	No. of Classes	No. of Classes Which Contain Only One Case Record	Cases in the First Five Largest Classes (%)	No. Of Classes	No. of Classes Which Contain Only One Case Record	Cases in the First Five Largest Classes (%)
0.92	1022	1020	0.68	9	2	99.12
0.93	1022	1020	0.68	10	2	98.83
0.94	1022	1020	0.68	12	3	98.63
0.95	1022	1020	0.68	54	13	55.37
0.96	1022	1020	0.68	55	14	55.37

Therefore, the total number of deleted cases is 365. After removing these deleted cases, we want to evaluate how much knowledge we have lost by transferring them to adaptation knowledge (i.e., the fuzzy adaptation rules). Therefore, we use them to test the solution accuracy of the smaller case base with the adaptation rules. The average relative error of the solution in each cluster is shown in the last column of Table 6. We defined relative error = (real value − adapted value)/real value * 100%. The adapted value is generated from the smaller case base together with the fuzzy adaptation rules.

TABLE 4. Clusters of the Travel Case Base

Cluster No.	Number of Cases	Odd or Not-odd Class
1	40	Not-odd
2	10	Not-odd
3	31	Not-odd
4	76	Not-odd
5	53	Not-odd
6	18	Not-odd
7	13	Not-odd
8	2	Odd
9	32	Not-odd
10	69	Not-odd
11	78	Not-odd
12	116	Not-odd
13	228	Not-odd
⋮	⋮	⋮
53	3	Odd
54	1	Odd
55	2	Odd

TABLE 5. Reachability and Coverage of Each Case in Cluster 7 of the Travel Case Base

Case Number (x)	Number of Cases Which Are Covered by Case(x)	The Actual Cases Which Are Covered by Case(x)	No. of Adaptation Rules
1	3	5, 8, 13	2
2	8	3, 4, 5, 6, 7, 8, 11, 13	6
3	4	1, 4, 6, 7	4
4	4	1, 3, 5, 7	7
5	2	1, 7	3
6	2	1, 5	4
7	2	3, 6	5
8	5	1, 3, 4, 5, 6	5
9	5	3, 4, 6, 7, 13	4
10	6	3, 4, 5, 6, 11, 13	6
11	1	1	4
12	5	3, 4, 5, 7, 13	7
13	5	1, 3, 4, 5, 7	7

The result shows that the Travel case-base size can be reduced by 36 percent if we complement the remaining cases by the adaptation rules discovered using our approach. The overall accuracy of the smaller case base is 94 percent of the original.

We randomly select 80 percent of the cases (820 cases) from Travel Agent case base as the training data set and the other 20 percent cases (204) as the testing data set. $\alpha = 0.155$ and $\eta = 10$, $\beta = 0.96$.

Problem-Solving Ability of the Transformed Case Base. In order to evaluate the problem-solving ability of the transformed case base to new cases, we divided the testing Travel case base into training cases (80 percent) and testing cases (20 percent), respectively. After applying Algorithm 1 in Section 2.1 to the 820 training cases, the feature-weight results shown in Table 7 are obtained (learning iterations = 100 cycles, $\alpha = 0.155$, and $\eta = 10$).

Using $\beta = 0.96$, the cases are divided into 8 Not-odd clusters (with 792 cases) and 10 Odd clusters (with 28 cases). After learning the fuzzy adaptation rules and carrying out the selection process, 254 cases are deleted, representing a 31 percent reduction. The average number of adaptation rules for each remaining representative case is 4, and the maximum number of antecedents of each fuzzy rule is 2. We then apply the 204 new cases to test this transformed case base's problem-solving ability. This ability is calculated using the relative error concept, i.e., relative error = (real value − adapted value)/real value $*$ 100%. We compare our approach with the traditional k-nearest neighbors approach. The result is shown in Table 8. It can be seen that the retrieval efficiency has been much improved using our approach.

3.2. Rice Taste Case Base

Using an experimental approach similar to the one described earlier, the Rice Taste data set is partitioned successfully into 14 clusters, with 3 clusters having more than 12 cases. Learning of fuzzy adaptation rules is carried out to these three clusters, and a

TABLE 6. Selection of Representative Cases in All the Not-odd Clusters and the Relative Errors

Cluster No.	Number of Cases	No. of Representative Cases	No. of Deleted Cases	Average Relative Error of Deleted Cases
1	40	18	22	5.91%
2	10	4	6	1.85%
3	31	18	13	6.41%
4	76	47	29	5.87%
5	53	31	22	6.22%
6	18	14	4	3.15%
7	13	5	8	2.35%
9	32	18	14	4.79%
10	69	53	16	5.73%
11	78	57	21	6.70%
12	116	70	46	6.72%
13	228	140	88	8.48%
17	30	12	18	5.59%
19	12	4	8	6.81%
20	12	7	5	1.73%
23	24	15	9	4.70%
25	41	29	12	5.99%
28	26	18	8	3.46%
31	13	5	8	2.25%
33	11	3	8	3.80%
	Total 933	Total 659	Total 365	Overall Average 6.22%

TABLE 7. Feature Weights of the Problem Features

Htype	Npersons	Region	Transport	Duration	Season	Accommodation	Hotel
0.0384	0	0.0341	0.4614	1	0.0148	0.1292	0

representative case-selection strategy is used with $\varepsilon = 0.15$, $\varepsilon = 0.1$, and $\varepsilon = 0.15$, respectively, for these three Not-odd clusters. The result shows that the Rice Taste case-base size can be reduced by 39 percent and that the overall accuracy of the smaller case base is 90 percent of the original.

4. COMPLEXITY ISSUES

In our approach, the main idea is to transfer case knowledge to adaptation knowledge for the benefit of better retrieval performance. Therefore, the time and space complexity of our approach is solely dependent on the complexity of generating the fuzzy adaptation rules and selecting the representative cases (see Section 2). First, we

TABLE 8. Relative Error and Solution Seeking Times Comparison

	Our Approach	k-Nearest Neighbors
Average relative error	38.51%	46.35%
Average solution seeking times	194	820

discuss the time complexity of our approach as follows:

The time complexity is equal to the sum of the "multiplication," "division," "max," and "min" operations required. In the first phase (i.e., learning feature weights; see Section 2.1), the mining function (Equation 3) is differentiable (smooth) and the search technique is gradient descent, and it can be guaranteed that the training algorithm (Algorithm 1; see Section 2.1) is convergent if the learning rate is appropriately small. Our experiments show that it converges even when we use a large learning rate (we use 10 as the learning rate in the Travel case base). By using a larger learning rate, we can reduce the number of epochs to a fixed value (e.g., 50 to 100). From Equations (1) to (7), we can easily see that the time complexity of Algorithm 1 for one epoch is equal to $\mathbf{O(N^2)}$, where $\mathbf{N}$ is the number of cases in the case base. Therefore, the total time complexity of our training algorithm is $\mathbf{O(N^2)}$.

In the second phase (i.e., partitioning cases into clusters; see Section 2.2), the partitioning algorithm (Algorithm 2) mainly involves the multiplication of two similarity matrices (i.e., step 3 of Algorithm 2); therefore, the time complexity is equal to $\mathbf{O(N^2)}$.

In the third phase (i.e., fuzzy decision-tree generation; see Section 2.3), the time complexity of this phase is equal to the generation of the fuzzy decision trees using the fuzzy ID3 heuristic algorithm (Algorithm 3). From Equation (15), the time complexity for generating one fuzzy decision tree is $\mathbf{O(N)}$. Therefore, the time complexity of generating the decision tree for each case in all the Not-odd clusters is less than $\mathbf{O(N^2)}$.

In the fourth phase (i.e., selecting representative cases; see Section 2.4), Algorithm 4 requires two major computations. The first step involves computing the case coverage in each cluster; on average, this requires $(N/k)(N/k-1)rp$ operations, where N is the number of cases, k is the number of clusters, r is the average number of fuzzy adaptation rules for each case, and p is the average number of antecedents in each rule. Since r and p are very small compared with N, the complexity of step 1 in Algorithm 4 is $\mathbf{O(N^2)}$. Steps 2 and 3 involve sorting the cases according to their coverage ability, and the computation complexity will not exceed $\mathbf{O(N^2)}$. Therefore, the overall computational complexity of Algorithm 4 is also bounded by $\mathbf{O(N^2)}$.

After the analysis of the time complexity of our approach, the space complexity is much simpler. The space complexity is determined by the size of the case base and the temporary storage required in each of the four phases described in Section 2. Among the four phases, the multiplication of the two matrices in phase 2 requires the largest amount of memory (i.e., $\mathbf{N^2}$). Since hardware and memory cost has been reduced significantly recently, we believe that memory space is not a critical concern when using our approach. However, we would like to carry out more study on this in our future work.

5. SUMMARY AND FUTURE WORK

In this article we have proposed a CBM methodology that was based on the idea of transferring knowledge between knowledge containers in a CBR system. The main idea is to transform a large case library to a small case library together with a group of adaptation rules that are generated by fuzzy decision trees. These adaptation rules play the role of complementing the reduction of cases. Our approach has four major steps, i.e., learning feature weights, clustering cases, mining adaptation rules, and representative case selection. Experimental testing is carried out using the Traveling and Rice Taste data sets. The experimental analysis of our method showed promising results. Our future work includes (1) developing different selection policies based on ideas such as subsumption, conflict, and ambiguity that exist among cases, (2) investigating the implementation issues of our approach for on-line or periodic updates, (3) evaluating what will be the effects if features are not independent, and (4) exploring the effects of other techniques such as rough sets theory on the CBM work.

ACKNOWLEDGMENTS

We are grateful to four anonymous reviewers for their constructive comments, which greatly helped in improving the quality of the paper. This project was supported by Hong Kong Polytechnic University Grant PA25 and Research Fellowship Grant G-YY12.

REFERENCES

ANAND, S. S., D. PATTERSON, J. G. HUGHES, and D. A. BELL. 1998. Discovering case knowledge using data mining. *In* Proceedings of the 2nd Pacific Asia Conference on Knowledge Discovery and Data Mining: Current Issues and New Applications, PAKDD-98. Springer-Verlag, Berlin, pp. 25–35.

BASAK, J., R. K. DE, and S. K. PAL, 1998. Unsupervised feature selection using a neuro-fuzzy approach. Pattern Recognition Letters, **19**:998–1006.

BEZDEK, J. C. 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

CHEETHAM, W., and J. GRAF. 1997. Case-based reasoning in color matching. *In* Proceedings of the Second International Conference on Case-Based Reasoning, ICCBR-97. Springer-Verlag, Berlin, pp. 1–12.

DEANGDEJ, J., D. LUKOSE, E. TSHUI, P. BEINAT, and L. PROPHET. 1996. Dynamically creating indices for two million cases: A real world problem. *In* Proceedings of the 3rd European Workshop of Case-Base Reasoning, EWCBR-96. Springer-Verlag, Berlin, pp. 105–119.

FU, G. 1992. An algorithm for computing the transitive closure of a similarity matrix. Fuzzy Sets and Systems, **51**:189–194.

HANNEY, K., and M. T. KEANE. 1996. Learning adaptation rules from a case-base. *In* Proceedings of the 3rd European Workshop of Case-Base Reasoning, EWCBR-96. Springer-Verlag, Berlin, pp. 179–192.

ICHIHASHI, H., T. SHIRAI, T. NAGASAKA, and T. MIYOSHI. 1996. Neuro-fuzzy ID3: A method of inducing fuzzy decision trees with linear programming for maximizing entropy and an algebraic method for incremental learning. Fuzzy Sets and Systems, **81**:157–167.

JENG, B. C., Y. M. JENG, and T. P. LIANG. 1997. FILM: A fuzzy inductive learning method for automated knowledge acquisition. Decision Support Systems, **21**:61–73.

JOHNSON, D. S. 1973. Approximate algorithm for combinational problems. Journal of Computer and Systems Science, **9**(3):256–278.

KITANO, H., and H. SHIMAZU. 1996. The experience sharing architecture: A case study in corporate-wide case-based software quality control. *In* Case-Based Reasoning: Experiences, Lessons, and Future Directions. *Edited by* D. Leake. AAAI Press, Menlo Park, CA, pp. 235–268.

KOHONEN, T. 1998. Self-Organization and Associate Memory. Springer-Verlag, Berlin.

LEAKE, D. B., and D. C. WILSON. 1998. Categorizing case-base maintenance: Dimensions and directions. *In* Proceedings of the 4th European Workshop of Case-Base Reasoning, EWCBR-98. Springer-Verlag, Berlin, pp. 196–207.

NOZAKI, K., H. ISHIBUCHI, and H. TANAKA. 1997. A simple but powerful heuristic method for generating fuzzy rules from numerical data. Fuzzy Sets and Systems, **86**:251–270.

QUINLAN, J. R. 1986. Induction of decision trees. Machine Learning, **1**:81–106.

RICHTER, M. M. 1995. The knowledge contained in similarity measures. Invited talk at ICCBR-95. *Available at* http://wwwagr.informatil.uni-kl.de/~lsa/CBR/ Richtericcbr95remarks.html.

RICHTER, M. M. 1998. Introduction. *In* Case-Based Reasoning Technology: From Foundations to Applications. *Edited by* M. Lena, B. Bartsch-Sporl, H. D. Burkhard, and S. Wess. Springer-Verlag, Berlin, pp. 1–15.

SMYTH, B., and M. T. KEANE. 1995. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. *In* Proceedings of the 14th International Joint Conference on Artificial Intelligence, IJCAI-95. Morgan Kaufmann, San Mateo, CA, pp. 377–382.

SMYTH, B., and E. MCKENNA. 1998. Modeling the competence of case-bases. *In* Proceedings of the 4th European Workshop of Case-Base Reasoning, EWCBR-98. Morgan Kaufmann, San Mateo, CA, pp. 23–25.

UMANOL, M., H. OKAMOTO, I. HATONO, H. TAMURA, F. KAWACHI, S. UMEDZU, and J. KINOSHITA. 1994. Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems. *In* Proceedings of the 3rd IEEE Conference on Fuzzy Systems, Piscataway, NJ, Vol. 3. pp. 2113–2118.

WETTSCHERCK, D., and D. W. AHA. 1995. Weighting features. *In* Proceedings of the 1st International Conference of Case-Base Reasoning, ICCBR-95. Springer-Verlag, Berlin, pp. 347–358.

LIST OF SYMBOLS

e_p, e_q	two cases, p and q
$d_{pq}^{(w)}$	weighted distance metric
$d_{pq}^{(1)}$	weighted distance metric with all weights taking value 1
$SM_{pq}^{(w)}$	similarity measure
$SM_{pq}^{(1)}$	similarity measure with all weights taking value 1
α	positive parameter
η	learning rate
$E(w)$	feature evaluation index
w_j	weight assigned to the jth feature of a case
Δw_j	change in weight w_j
β	threshold value
$SM_{\text{int}\_ra}^{(w)}$	intrasimilarity
$SM_{\text{int}\_er}^{(w)}$	intersimilarity
$FE(N)$	fuzzy entropy of the node N
ε	error standard